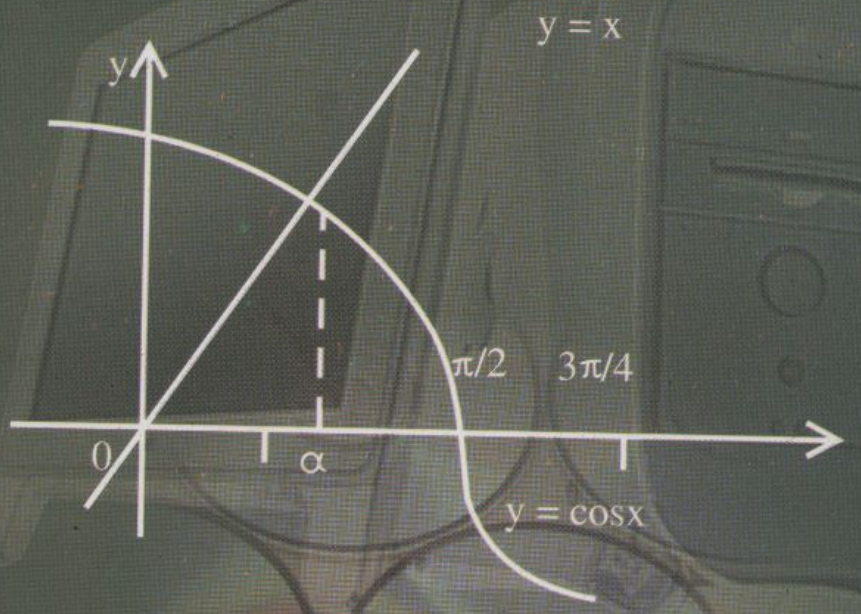
Prof.Dr. S A Bhatti
Mr. N A Bhatti

A First Course in NUMERICAL ANALYSIS With C++



Fifth Edition

Numerical Analysis is a Science

– computation is an art.

Fifth Enlarged Edition

A First Course in NUMERICAL ANALYSIS With C++

Prof. Saeed Akhter Bhatti
Mr. Naeem Akhtar Bhatti

SHAHARYAR PUBLISHERS
AL-FAZAL MARKET, URDU BAZAR, LAHORE.

CAN BE HAD FROM:

A-ONE PUBLISHERS
AL-FAZAL MARKET, URDU BAZAR, LAHORE
Phone No. 37232276 - 37357177 - 37224655
Email: aonepub@hotmail.com / info @aonepublishers.com
Website: aonepublishers.com

©Copyright 2013 Prof. S. A. Bhatti

All rights reserved. No part of this book may be reproduced, stored in a retrieval system,, or transmitted, in any form or by any means, electronic, mechanical, photocopying, recording, or otherwise, without the prior permission of Author or Publisher.

Book: A First Course in Numerical Analysis with C++

Author: Prof. Saeed Akhter Bhatti
Mr. Naeem Akhtar Bhatti

First Edition: May, 1990
Second Edition: Mar, 1996
Third Edition: Aug, 1999
Fourth Edition: May, 2002
Fifth Edition: Jan, 2008
Reprint: 2013

Quantity: 1100

Price: **Rs. 400/= (Rupees Four Hundred Only)**
Library Edition: **Rs. 500/= (Rupees Five Hundred Only)**

Published by:
Muhammad Azhar
(Shaharyar Publishers)

Printed at:
Ali Ejaz Printers,
Rattigun Road, Lahore

ISBN: **969-8105-01-8**

In the loving memory of our parents,

Mr and Begum Sana Ullah Sufi

Preface To The Fifth Edition

The Goal of this enlarged edition of our book on **Numerical Analysis** remains the same as for the previous editions: to give a comprehensive and state-of-the-art treatment of all the important aspects of the subject. In this, we have made modifications in all the first eight chapters and added extra problems at the end of each chapter. A new chapter, Chapter 9, based on eigenvalues and eigenvectors has been included. We have tried to cover all the basic and important procedures to compute eigenvalues and eigenvectors of a matrix. This chapter has been written especially on the request of users of the subject in various engineering universities.

We gratefully thank the users and the reviewers of the previous editions who provided valuable suggestions and ideas for the improvement of this book. Their feedback is valuable in our efforts for continuous improving this book. We are also thankful to our various teaching assistants both at BIIT and FUIMCS who checked the references and exercises in many chapters.

The authors would also like to thank Professor Akram Javed, Faculty of Science, University of Engineering and Technology, Taxila, for his many useful comments. We are thankful to Prof. Aftab Ahmad, Director, Institute of Management and Computer Sciences, Foundation University, Rawalpindi, for providing us the necessary infrastructure to complete this project. Thank you all.

The Bhattis
Islamabad,

Preface To The Fourth Edition

The Fourth Edition of this book on numerical analysis is in your hands now. It is geared specifically to the needs and background of our students. During this period, we received several comments from the users. In reviewing their comments, we have made modifications in some chapters of the book to sharpen the reader's understanding of the material presented. The plan of presentation of all chapters has been that of step by step. We start with an elementary method and then proceed to develop this or alternative, more sophisticated methods. The presentation just given is, of course, much over-simplified. In practice, a combination of conventional mathematical analysis and numerical analysis is likely to be used. Proofs of formulas are given where these are reasonably easy to follow but have been omitted in the more difficult cases.

A major change has been made in computer programs that implement the use of numerical methods presented in the book for solving problems. This edition contains computer programs written in C++. They have deliberately been kept as straightforward as possible so that the reader should understand the precise function of every step in each program. While the programs are intended primarily for educational-purposes, they can, of course, be used for solving some simple practical problems. However, for more complex practical problems, they do not offer any guarantee regarding the accuracy, adequacy or completeness of any information herein. Therefore, the user should make use of the excellent software packages now available. Hopefully the reader will appreciate this edition. We recommend them to learn and make more substantial use of their computers. We have benefited much by sitting at the feet of the wise, and we hope that, through this book, it may be possible to transmit a spark from their fire to all our readers. Good luck!

We would like to thank the users and reviewers of the previous editions whose comments and suggestions have enormously proved to be valuable in updating the material of the book. However, comments and suggestions for further improvements to the book and supporting software are welcome and can be communicated to us through the publisher. The authors would also like to express their gratitude to Prof. Akram laved, Dean, Faculty of Science, UET, Taxila, for his many useful comments received to improve the quality of this book and particularly to Dr. Jamil Sarwar, Director, BIIT, Rawalpindi, for providing necessary facilities to accomplish this reviewing exercise.

In closing, we are also grateful to our families for their continued patience and understanding during the review effort.

The Bhattis
Islamabad
May, 2002

Preface To The First Edition

The importance of Numerical Analysis to the scientists and engineers is now widely acknowledged. In the book world, there is no dearth of good books on numerical analysis written by foreign authors but the majority of these books are not available in this country. I have written this book to meet the long-felt need of indigenous students.

The main feature of the present text is to introduce numerical methods – covering the syllabi of various universities, colleges and other institutes, where this subject is being taught as a first course. In writing such an elementary book, I have inevitably been confronted by the problem of selection of material, which covers to a great extent the syllabi of the concerned institutes. Naturally, some will disagree with me over this choice of selection. I respect their prerogative. However, I shall be relieved if it is felt that the topics included do provide a reasonably solid background to the student's training and one from which he can easily proceed to further advanced courses in the subject.

The book is designed for a one-semester course in numerical analysis and consists of eight chapters. Each chapter includes a large number of thoroughly explained examples and problems of various complexity. These problems are very necessary and the students should work them out carefully. Each question has been designed to test the student's understanding of a particular formula. The answers of these problems are given at the end of the book. Proofs of formulas are included only where these are reasonably easy to follow, but the formulas are mentioned without proofs in the more difficult cases. It has been tried to keep the explanation straightforward and practically-oriented. The minimum prerequisite for using this book is elementary calculus (including some exposure to series and partial derivatives), linear algebra (determinant an matrices) and differential equations. It is also assumed that the student has taken a programming course in one of the computer languages. Fortran 77, which continues to be an excellent computer language for a wide variety of mathematical problems, is used in this book. Computer programs are given at appropriate places in the text.

No book emerges fully formed from an author's forehead. I would like to acknowledge the inspiration and encouragement I received from my colleagues and the help of many students who worked with early versions of the manuscript and checked exercise solutions and text examples.

The responsibility for any errors, omissions or lack of clarity naturally remains with me. I would appreciate having any such omissions, oversights or needed corrections called to my attention so that they can be implemented for improving the quality of this book. I would also like to thank Mr. Ghulam Shabir Qureshi and Syed Akbar Shah for their help in turning rough drafts into a beautifully prepared final manuscript.

I would like to express my gratitude to the National Book Council of Pakistan for accepting the manuscript of this book under the Creative Writer's Scheme. I also wish to thank the anonymous referees who reviewed the manuscript.

Above all, I wish to thank my family, without whose encouragement, patience and sacrifice this book would not have been completed.

Saeed A. Bhatt
Islamabad
May, 1990

Contents

Chapter 1 Error Analysis

1.1 INTRODUCTION TO NUMERICAL ANALYSIS 1

1.2 DEFINITION OF AN ERROR 2

1.3 SOURCES OF ERRORS 3

1.3.1 Gross Errors 3

1.3.2 Rounding Errors 4

1.3.3 Truncation Errors 5

1.4 SOME DEFINITIONS 6

1.4.1 Significant Digits 6

1.4.2 Precision and Accuracy 7

1.4.3 Absolute, Relative and Percentage Errors 7

1.5 EFFECT OF ROUNDING ERRORS IN ARITHMETIC OPERATIONS .. 8

1.5.1 Error Accumulation in Addition 9

1.5.2 Error Accumulation in Subtraction 9

1.5.3 Error Accumulation in Multiplication 10

1.5.4 Error Accumulation in Division 12

1.5.5 Errors of Powers and Roots 14

1.5.6 Error in Function Evaluation 16

1.6 NUMERICAL CANCELLATION 17

1.7 EVALUATION OF FUNCTIONS BY SERIES EXPANSION AND ESTIMATION OF ERRORS 19

PROBLEMS 22

Chapter 2 Finite Differences

2.1 DIFFERENCE TABLE 27

2.2 DETECTION AND CORRECTION OF ERRORS IN A DIFFERENCE TABLE 30

2.3 DIFFERENCE OPERATIONS 35

2.3.1 Forward Difference Operator 35

2.3.2 Backward Difference Operator 40
2.3.3 Central Difference Operator 41
2.3.4 Shift Operator 42
2.3.5 Mean Operator 43
2.4 RELATIONSHIPS BETWEEN OPERATORS 43
PROBLEMS 44

Chapter 3 Interpolation
3.1 INTRODUCTION 51
3.1.1 Choice of a Suitable Interpolation Formula 51
3.1.2 Checking the Interpolated Value 52
3.1.3 Type of Interpolation Formulas for Equally-Spaced Data Points 52
3.1.4 Type of Interpolation Formulas for Unequally-Spaced Data Points 52
3.2 NEWTON'S FORWARD DIFFERENCE INTERPOLATION FORMULA 52
3.3 NEWTON'S BACKWARD DIFFERENCE INTERPOLATION FORMULA 56
3.4 INTERPOLATION WITH CENTRAL DIFFERENCE FORMULAS 62
3.4.1 Stirling's Interpolation Formula 62
3.4.2 Bessel's Interpolation Formula 64
3.4.3 Everett's Interpolation Formula 65
3.4.4 Gaussian Interpolation Formula 65
3.5 LAGRANGE'S FORMULA 69
3.6 ITERATIVE INTERPOLATION METHOD 73
3.7 ERROR ESTIMATION IN INTERPOLATION 80
3.7.1 Error in Newton's Forward Difference Formula 81
3.7.2 Error in Newton's Backward Difference Formula 83
PROBLEMS 85

Chapter 4 Numerical Differentiation
4.1 INTRODUCTION 93
4.2 DERIVATION OF DIFFERENTIATION FORMULAS 93
4.3 RELATIONSHIP BETWEEN OPERATORS E AND D 94
4.4 DERIVATIVES USING NEWTON'S FORWARD DIFFERENCE INTERPOLATION FORMULA 95

4.5 DERIVATIVES USING NEWTON'S BACKWARD DIFFERENCE INTERPOLATION FORMULA 103

4.6 DERIVATIVES USING CENTRAL DIFFERENCE INTERPOLATION FORMULAS 108

4.6.1 Derivatives Using Stirling's Interpolation Formula 109

4.6.2 Derivatives Using Bessel's Interpolation Formula 111

4.6.3 Derivatives Using Everett's Interpolation Formula 113

4.6.4 Derivatives Using Gauss Interpolation Formula 114

PROBLEMS 120

Chapter 5 Numerical Integration

5.1 INTRODUCTION 125

5.2 DERIVATION OF INTEGRATION FORMULA BASED ON NEWTON'S FORWARD DIFFERENCES 126

5.3 THE NEWTON-COTES FORMULAS 127

5.3.1 Trapezoidal Rule 127

5.3.2 Simpson's $\frac{1}{3}$rd rule 129

5.3.3 Combination of Trapezoidal and Simpson's Rules 130

5.3.4 Simpson's $\frac{3}{8}$th Rule 131

5.3.5 Boole's Rule 132

5.3.6 Weddle's Rule 132

5.4 ESTIMATION OF ERRORS IN SOME NEWTON-COTES FORMULAS 135

5.4.1 Error in Trapezoidal Rule 135

5.4.2 Error in Simpson's $\frac{1}{3}$rd Rule 136

5.5 AUTOMATIC SUBDIVISION OF INTERVALS 149

5.5.1 Repeated Use of Trapezoidal Rule 149

5.5.2 Romberg Integration 152

PROBLEMS 158

Chapter 6 Ordinary Differential Equations

6.1 INTRODUCTION 165

6.1.1 Classification of Differential Equations 165

6.1.2 Categories of ODEs 166
6.2 METHODS TO SOLVE ODEs 167
6.3 NUMERICAL METHOD TO SOLVE ODEs 168
6.4 PICARD'S METHOD 169
6.5 TAYLOR SERIES METHOD 172
6.6 EULER'S METHOD AND ITS VARIATIONS 175
6.7 RUNGE-KUTTA METHODS 177
6.8 PREDICTOR-CORRECTOR METHODS 184
6.8.1 Milne-Simpson Predictor-Corrector Method 186
6.8.2 Adams-Bashforth Predictor-Corrector Method 189
6.8.3 Adams-Moulton Method 194
6.9 SOLUTION OF SIMULTANEOUS AND HIGHER-ORDER ORDINARY DIFFERENTIAL EQUATIONS 203
6.9.1 Solution of First-Order Simultaneous Differential Equations . . . 203
6.9.2 Solution of Nth-Order Differential Equations 203
PROBLEMS 208

Chapter 7 Non-Linear Equations

7.1 INTRODUCTION 217
7.2 METHODS TO SOLVE NON-LINEAR EQUATIONS 218
7.3 SIMPLE ITERATIVE METHOD 218
7.3.1 Termination of an Iterative Procedure 219
7.3.2 Flowchart for a Simple Iterative Procedure 220
7.3.3 Graphical Representation of Convergence 221
7.3.4 Localization (Approximation) of Roots 222
7.3.5 Convergence 225
7.4 ACCELERATION OF CONVERGENCE 230
7.5 NEWTON-RAPHSON METHOD 233
7.5.1 Geometrical Interpretation 233
7.5.2 Order of Newton-Raphson Method 234
7.5.3 Special Cases of Newton-Raphson Method 237
7.6 THE BISECTION METHOD 242
7.7 THE SECANT METHOD 246

7.8 METHOD OF FALSE POSITION AND ITS MODIFIED FORM 249
7.9 DETERMINATION OF MULTIPLE ROOTS 254
7.10 ZEROS OF POLYNOMIALS 256
7.10.1 Evaluation of a Polynomial (Birga-Vieta Method) 256
7.10.2 Evaluation of Derivatives of Polynomials 257
PROBLEMS 261

Chapter 8 Linear Systems of Equations
8.1 BASIC CONCEPTS 267
8.2 METHODS TO SOLVE A SYSTEM OF LINEAR EQUATIONS 268
8.3 CRAMER'S RULE AND ITS MODIFIED FORM 269
8.4 GAUSSIAN ELIMINATION METHODS 272
8.4.1 Pivot Strategy 280
8.4.2 Partial Pivoting Scheme 281
8.4.3 Complete Pivoting Scheme 282
8.5 TRIANGULAR DECOMPOSITION (FACTORIZATION) METHOD 284
8.5.1 Solution of Systems of Equations 284
8.5.2 Inverse of a Matrix A using L and U 285
8.6 TRIANGULAR DECOMPOSITION FOR SYMMETRIC MATRICES 290
8.7 SOLUTION OF TRIDIAGONAL SYSTEMS OF EQUATIONS 293
8.8 ITERATIVE METHODS 298
8.8.1 Jacobi's Method 299
8.8.2 Gauss-Seidel Method 305
PROBLEMS 310

Chapter 9 Eigenvalues and Eigenvectors
9.1 INTRODUCTION 323
9.2 METHODS TO SOLVE EIGENVALUE PROBLEMS 324
9.2.1 General Method 324
9.2.2 Leverrier-Faddeev Method 331
9.2.3 Power Method 335
9.3 MATRIX DEFLATION 344
9.3.1 Hotelling's Deflation 345

9.3.2 Hotelling's Deflation for Symmetric Matrices 348
9.4 PROPERTIES OF EIGENVALUES AND EIGENVECTORS 350
9.5 GERSHGORIN'S THEOREM 351
PROBLEMS 356
Bibliography 364
Index 365
Problems 371

Chapter 1

Error Analysis

1.1 INTRODUCTION TO NUMERICAL ANALYSIS

When a mathematical problem can be solved analytically, its solution may be exact, but more frequently, there may not be a known method of obtaining its solution. For example, it is rather difficult to solve the following integral analytically:

$$\int_0^t \frac{e^{-x^2}\,dx}{\sqrt{1-x^2}}\,;\ -1 \le t \le 1.$$

Many more such examples can be cited for which solutions by analytical means are either impossible or may be so complex that they are quite unsuitable for practical purposes. In this situation, the only way of obtaining an idea of the behaviour of a solution is to approximate the problem in such a manner that the number representing the solution can be produced. The process of obtaining a solution is to reduce the original problem to a repetition of the same step or series of steps so that the computations become automatic. Such a process is called a **numerical method** and the derivation and analysis of such methods lie within the discipline of **numerical analysis**. Thus, the subject of numerical analysis is concerned with the derivation, analysis and implementation of methods for obtaining reliable numerical answers to complex mathematical problems. In other words, numerical analysis is the subject concerned with the construction, analysis, and use of algorithms for the numerical solution of mathematical problems to given degree of numerical accuracy.

Numerical methods provide estimates that are very close to the exact analytical solutions; obviously, an error is introduced into the computation. It is important to understand that an error here does not mean a human error, such as a blunder or mistake or oversight but rather a discrepancy between the exact and approximate (computed) values. Such errors are likely to arise in all methods described in this book. In fact, numerical analysis is a vehicle to study errors in computations. It is not a static discipline. The continuous change in this field is to devise algorithms, which are both fast and accurate. These algorithms may become obsolete and may be replaced by more powerful algorithms as computer capability increases or as new techniques are developed. It is necessary to point out from personal experience that the best test of whether one understands a method is not to carry out a hand calculation (although this can be useful in early stages of attempting to understand the logic), but to program the method in a specific programming language, like BASIC, FORTRAN, PASCAL, C, C++ and JAVA

and run it on a computer. We all know that computers are ideally suited to handle tedious computations with high speed, accuracy and without ever making mistakes. Hence, the use of numerical method for the analysis, simulation and design of scientific and engineering processes and systems has been increasing at a rapid rate. This course is introduced to better prepare future scientists and engineers in understanding the fundamentals of numerical methods, especially their applications, limitations and potentials.

Although good computer programming skills can enhance the study of numerical analysis, actually writing programs are not always necessary. Numerical analysis is so important that extensive commercial software packages are available. For example, IMSL (International Mathematical and Statistical Library). It has several routines for numerical methods written in FORTRAN and C++. Some other packages are LAPACK (Linear Algebra Package) written in FORTRAN 77, LINPACK, EISPACK, Mathematica, Derive, Maple, MathCad, MathLab, MacSyma NUMERICOMP, etc. In addition a set of books, **Numerical Recipes**, lists and discusses numerical analysis programs in a variety of computer languages. However, one special feature of most of these programs is their ability to carry out many operations with exact arithmetic; an interesting example is to see the value of π displayed to 100 dp.

1.2 DEFINITION OF AN ERROR

The knowledge we have of the physical world is obtained by doing experiments and making measurements. It is important to understand how to express such data and how to analyze and draw meaningful conclusions from it. In doing this it is crucial to understand that all measurements of physical quantities are subject to uncertainties. It is never possible to measure anything exactly. It is good, of course, to make the error as small as possible but it is always there. And in order to draw valid conclusions the error must be indicated and dealt with properly. Take the measurement of a person's height as an example. Assuming that his height has been determined to be 5′ 8″, how accurate is our result?

Well, the height of a person depends on how straight he stands, whether he just got up (most people are slightly taller when getting up from a long rest in horizontal position), whether he has his shoes on, and how long his hair is and how it is made up. These inaccuracies could all be called **errors of definition**. A quantity such as height is not exactly defined without specifying many other circumstances. Even if you could precisely specify the "circumstances", your result would still have an error associated with it. The scale you are using is of limited accuracy; when you read the scale, you may have to estimate a fraction between the marks on the scale, etc. If the result of a measurement is to have meaning it cannot consist of the measured value alone. An indication of how accurate the result is must be included also. Indeed, typically more effort is required to determine the error or uncertainty in a measurement than to perform the measurement itself. Error, then, has to do with uncertainty in measurements that nothing can be done about. If a measurement is repeated, the values obtained will differ and none of the results can be preferred over the others. Although it is not possible to do anything about such error, it can be characterized. For instance, the repeated

measurements may cluster tightly together or they may spread widely. This pattern can be analyzed systematically.

All measurements, however, carefully and scientifically performed are subject to errors. Errors once committed contaminate subsequent results. **Errors analysis** is the study and evaluation of these errors; its main functions are to estimate the errors and suggest ways to eliminate or minimize them. Investigations of error propagation are, of course, particularly important in connection with iterative processes and computations where each value depends on its predecessors. Examples of such problems are in linear systems of equations, ordinary and partial differential equations. Since the study of errors is central to numerical analysis, we shall discuss it at length.

An error in a numerical computation is the difference between the actual value of a quantity and its computed (approximate) value. If x represents the computed value of a quantity, the actual value for which is x*, then the difference,

$$E = x^* - x \qquad \ldots \qquad (1.1)$$

is called the **error of approximation**.

1.3 SOURCES OF ERRORS

A numerical method for solving a given problem will, in general, involve an error of one or several types. Although different sources initiate the error, they all cause the same effect: **diversion from the exact answer**. Some errors are small and may be neglected, while others may be devastating if overlooked. In all cases, error analysis must accompany the computational scheme, whenever possible.

The main sources of error are as follow:

- Gross errors,
- Round errors,
- Truncation errors.

1.3.1 Gross Errors

Although gross errors are not directly concerned with most of the numerical methods discussed in this book, they can sometimes have great impact on the success of modeling efforts. Thus, they always be kept in mind when applying numerical techniques in context of real-world problems.

The gross errors are either caused by human mistakes or by the computer. Such mistakes are trivial, with better or no effect on the accuracy of the calculation, or they may be so serious as to render the calculated results quite wrong. A few examples of these errors are as follows:

i) Misreading or misquoting the figures, particularly in the interchanges of adjacent digits,

ii) Use of inaccurate mathematical formula (algorithm) to solve a particular problem, and

iii) Use of inaccurate data.

These errors are not very serious and can be avoided, if enough care is takem in using proper numerical analysis techniques. We shall primarily concern ourselves with the latter types of errors.

1.3.2 Rounding Errors

When a numerical method is actually run on a digital computer after transcription to computer program form a kind of error called **round-off error** is introduced.

The error introduced by rounding-off numbers to a limited number of decimal places is called the **rounding error**. In simple words, the error in the result that is caused by rounding is called round-off error. For example, it would be impracticable to mention the distance between two points on the earth as 15.2967 metres. It would be more reasonable if it were to be round to the nearest whole number, i.e., 15 metres. Thus, the error introduced by rounding is 0.2967 metres. Another example is the value of π = 3.1415926353 and may be meaningfully rounded-off to 3.1416 or 3.142.

Rounding-off errors play an important role in numerical analysis. In order to obtain a smaller error as a result of rounding-off, we may apply the following rules when performing manual calculations (these rules are not normally applied when performing extensive computer calculations).

Suppose we are given a number and we want to round it to the first decimal place. We discard all digits after the first decimal place and proceed as follows:

(a) If the first discarded digit is less than 5, the previous digit is unchanged. For example, the number 56.44, when rounded to the first decimal place, then it becomes 56.4.

(b) If the discarded digit is greater that 5, the previous digit is increased by 1. For example, the number 56.46, when rounded to first decimal, it becomes 56.5.

(c) If the discarded digit is exactly 5, the previous digit is unchanged, if it is even and is increased by 1, if it is odd. For example, the number 56.45, becomes 56.4 and 56.75 becomes 56.8.

However, the most commonly used rule (we are familiar with) for rounding-off the numbers is: **"if the discarded digit exceeds or equals 5, we add 1 to the last retained digit"**.

Analysis of the round-off error present in the final result of a numerical computation, usually termed the **accumulated** rounded-off error, is difficult, particularly when the algorithm used is of some complexity. Except in very simple cases, the accumulated error is not simply the sum of the **local** round-off error, that is, errors resulting from individual rounding or truncating operations. The local error at any stage of the calculation is propagated throughout the remaining part of the computation. In order to establish a round-off error bound, we must assume the worst possible outcome for the result of each arithmetic operation and follow the preparation of all such errors throughout the remaining calculations.

1.3.3 Truncation Errors

Truncation is defined as the replacement of one infinite series (or iterative process) by another with fewer terms. The error arising from this approximation is called the **truncation errors.** We shall devote considerable attention to truncation errors associated with the numerical methods discussed in this book. Because when different numerical methods are compared, we usually consider the truncation errors first.

In analyzing errors arising from the truncation of series, several types of series expansions can be considered. These include (but are not limited to) the following:

- Binomial expansion,
- Infinite geometric progression,
- Taylor/MacLaurin series.

In order to understand better the properties of truncation error, we turn to a mathematical formulation that is used commonly in numerical analysis for expressing functions in an approximate fashion – **the Taylor series.**

For example, the Taylor series expansion of f(x) about some chosen point x is defined by

$$f(x) = f(x_0) + (x - x_0)f'(x_0) + \frac{(x - x_0)^2}{2!}f''(x_0) + \frac{(x - x_0)^3}{3!}f'''(x_0)$$

$$+ \ldots + \frac{(x - x_0)^i}{i!}f^{(i)}(x_0) + \ldots + \frac{(x - x_0)^n}{n!}f^{n}(x_0) + R_n \qquad \ldots (1.2)$$

where R_n is the **remainder term** (error caused by truncating terms) that is included to account for all terms (n + 1) to infinity and is given by:

$$R_n = \frac{(x - x_0)^{n+1}}{(n+1)!}f^{(n+1)}(Z) \qquad \ldots (1.3)$$

where the subscript n connotes that this is the remainder for the nth-order approximation and Z is some value of x that lies somewhere between x_0 and x, i.e., $x_0 \le Z \le x$.

It is often convenient to simplify Taylor series by defining a step size $h = x - x_0$ and expressing (1.2) and (1.3) as,

$$f(x) = f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + \frac{h^3}{3!}f'''(x_0) + \ldots +$$

$$\frac{h^n}{n!}f^{(n)}(x_0) + R_n \qquad \ldots (1.4)$$

where $$R_n = \frac{h^{n+1}}{(n+1)!}f^{(n+1)}(Z) \qquad \ldots (1.5)$$

At this stage, it is sufficient to say that the remainder term provides an estimate of the maximum absolute error. We have devoted Section 1.7 to the computation of such errors.

1.4 SOME DEFINITIONS

Before proceeding further, let us define the following terms:

- Significant digits (figures),
- Precision and accuracy,
- Absolute, relative and percentage errors.

1.4.1 Significant Digits

In considering rounding errors, it is necessary to be precise in the usage of approximate digits. A **significant digit** in an approximate-number is a digit, which gives reliable information about the size of the number. In other words, a **significant digit** is used to express accuracy, that is, how many digits in the number have meaning. Whenever we use a number in a computation, we must have awareness that it can be used with confidence. To be significant, the last digit contained should be accurate within half a unit in the last decimal place. For example, if an approximate number α is equal to 1.23 and the value of α lies in the interval $1.225 \leq \alpha \leq 1.235$, then α is said to have three significant digits.

In considering significant digits, the following rules are generally used for number written in the conventional form:

a) Leading zeros are not significant.

b) Following zeros that appear after the decimal point are significant.

c) Following zeros that appear before the decimal point may or may not be significant, as more information is required to decide.

d) The significant digits in a number do not depend on the position of the decimal point in the number.

The above rules are illustrated by the following examples:

i) The number .0002025 has only four significant digits. The leading zeros are not significant.

ii) The number .00202570 has six significant digits. The following zeros should not be written unless it is significant.

iii) The number 2025000 may have four, five, six or seven significant digits depending upon the situation. The conventional form of writing number is somewhat ambiguous in this instance.

iv) The number 12546 and .12546 both contain five significant digits.

Note: The simplest way of reducing the number of significant digits in the representation of a number is merely to ignore the unwanted digits. This procedure,

known as **chopping**, was used by many early computers. A more common and better procedure is **rounding**, which involves adding 5 to the first unwanted digit and then chopping. For example, π chopped to four decimal places, is 3.1415, but it is 3.1416 when rounded; the representation of 3.1416 is correct to five significant digits (5S). The error involved in the reduction of the number of digits is called **round-off error**. Since π is 3.14159..., we note that chopping has introduced much more round-off error than rounding.

1.4.2 Precision and Accuracy

The meaning of the terms: precision and accuracy are often confused.

Precision is the number of digits in which a number is expressed or an answer given irrespective of the correctness of these digits. For example, if we are using a four-figure logarithmic table to perform calculations, our final answer will seldom be correct to four figures because of the accumulation of round-off errors.

Accuracy, on the other hand, is the number of digits to which an answer is correct. Accuracy can be quoted in either of the following ways:

i) to a given number of decimal places (abbreviated to **dp** throughout this book), or,

ii) to a given number of significant figures (abbreviated to **sf**).

Suppose, the result of a calculation is obtained as 65.5432, then the answer has a precision of 4 dp. If we know that the last two digits are unreliable, then the result may be rounded to 65.54 to achieve an accuracy of 2 dp or 4 sf. When statements about precision are made, the units involved need to be expressed. Thus, the quantity 6.474 kg is accurate to 4 sf, but precise to the nearest .001 kg; also the quantity is precise to the nearest .01 metre, but accurate to 1 sf.

Numerical methods should be sufficiently accurate (or unbiased) to meet the requirements of a particular scientific problem and they also should be precise enough. We now discuss the errors in performing numerical computations.

1.4.3 Absolute, Relative and Percentage Errors

The accuracy of any computation is always of great importance. There are two common ways to express (measure) the size or error in a computed result by **absolute error** and **relative error**. Let us define them one by one.

Absolute Error

We use the term **absolute error** (abbreviated to AE) to denote the actual value of a quantity less its rounded (approximate) value. If x and x* are respective by the rounded and actual values of a quantity, then the absolute error is defined by,

$$AE = |x^* - x| \qquad \ldots (1.6)$$

For example, if $x^* = 4.83$ and $x = 4.832$, then,

$$AE = |4.83 - 4.832| = .002$$

Generally, if a number is correct to n dp, it has a rounding error:

$$AE \le \frac{1}{2} \times 10^{-n}.$$

Relative Error

Relative error (abbreviated to RE) is the ratio of the absolute error to the absolute actual value of a quantity.

Thus, $$RE = \frac{AE}{|x^*|}; \; x^* \neq 0. \qquad \ldots (1.7)$$

If the actual value is not known, the relative error is defined by,

$$RE = \frac{AE}{|x|}; \; x \neq 0. \qquad \ldots (1.8)$$

As a measure of accuracy, relative error is more precise and meaningful than the absolute error, this is particularly so when the actual value is either very small or very large. The size of AE depends on the units used, whereas RE is a dimensionless quantity.

From the above example, $RE = \frac{.002}{4.83} = .00041.$

A decimal number correct to n significant-digits has:

$$RE \le 5 \times 10^{-n}.$$

Percentage Error

Relative error expressed in percentage is called the **percentage error** (abbreviated by PE) and is defined by,

$$PE = 100 \times RE \qquad \ldots (1.9)$$

From the above example, $PE = 100 \times .00041 = .041\%$.

It is also called **probable error.**

In order to investigate the effect of total error in a method, we often compute an **error bound** which is a limit on how large and small the error can be.

1.5 EFFECT OF ROUNDING ERRORS IN ARITHMETIC OPERATIONS

In this section, we shall derive formulas for AE, and RE, for each of the fundamental operations of arithmetic, namely, addition, subtraction, multiplication and division, etc. Idea of error bound will also be introduced.

1.51 Error Accumulation in Addition

Let x_1 and x_2 be two approximate numbers and z be their sum. Then,

$$z = x_1 + x_2 \qquad \ldots (1.10)$$

Let e_1, e_2 and e_z be the errors in x_1, x_2 and z respectively.

Thus, we may add (or subtract) the errors from respective number:

$$z - e_z = (x_1 - e_1) + (x_2 - e_2)$$
$$= (x_1 + x_2) - (e_1 + e_2)$$

From (1.10), we have,

$$e_z = e_1 + e_2$$

Thus, the error simply add. So, the absolute error of two approximate numbers is given below:

$$AE = |e_z| \le |e_1| + |e_2| \qquad \ldots (1.11)$$

The above proof can be extended to the sum of any given number of factors, i.e.,

$$AE = |e_z| \le |e_1| + |e_2| + \ldots + |e_n| \qquad \ldots (1.12)$$

Hence, the absolute error of the sum of n approximate numbers does not exceed the sum of the absolute errors of the numbers.

The relative error is calculated using the following relation:

$$RE = \frac{\text{Absolute Error}}{\text{Absolute sum of the given number}}$$

$$= \frac{AE}{|z|} \qquad \ldots (1.13)$$

1.5.2 Error Accumulation in Subtraction

Let $z = x_1 - x_2$, where $x_1 > x_2$. $\ldots (1.14)$

As before, $z - e_z = (x_1 - e_1) - (x_2 - e_2)$

$$= (x_1 - x_2) - (e_1 - e_2)$$

From (1.14), we have,

$$e_z = e_1 - e_2$$

$$AE = |e_z| \le |e_1| + |e_2| \qquad \ldots (1.15)$$

which is same as (1.11).

Hence, the absolute error of a difference between two numbers is the sum of the absolute errors of the given numbers. This formula can also be extended to any number of factors. Thus the formula for the addition of numbers and subtraction of numbers are the same.

Example 1 If the numbers 0.3062 – 0.25026 + 2.51392 are rounded, estimate the maximum absolute and relative errors. Find also the range in which the true answer lies.

Solution Let $x_1 = 0.3062$, $x_2 = 0.25026$ and $x_3 = 2.51392$.

Thus, $z = x_1 - x_2 + x_3 = 2.56986$.

Let e_1, e_2 and e_3 be the errors in x_1, x_2 and x_3, respectively. Thus, the absolute errors in the respective numbers are as follows:

$$|e_1| \le \frac{1}{2}\times 10^{-4}$$

$$|e_2| \le \frac{1}{2}\times 10^{-5}$$

$$|e_3| \le \frac{1}{2}\times 10^{-5}$$

$$AE = |e_1| + |e_2| + |e_3|$$

$$\le \frac{1}{2}\times 10^{-4} + \frac{1}{2}\times 10^{-5} + \frac{1}{2}\times 10^{-5} = 0.6 \times 10^{-4}$$

$$RE = \frac{AE}{z} = \frac{0.6\times 10^{-4}}{2.56986} = 0.2335 \times 10^{-4}.$$

The result lies in the range $z \pm AE$:

$$2.56986 \pm 0.6 \times 10^{-4}$$

or $$2.56980 \le z \le 2.56992.$$

The answer may be rounded meaningfully to 2.57, which is correct to 3 sf (2 dp).

1.5.3 Error Accumulation in Multiplication

Suppose, we want to multiply two approximate numbers, x_1 and x_2.

Let $z = x_1 \cdot x_2$. ... (1.16)

As before, $z - e_z = (x_1 - e_1)(x_2 - e_2)$

$$= x_1 x_2 - x_1 e_2 - x_2 e_1 + e_1 e_2$$

Since e_1 and e_2 are small quantities, their product is still smaller and hence may be neglected. Thus,

$$z - e_z = x_1 x_2 - x_1 e_2 - x_2 e_1$$

From (1.16), we have,

$$e_z = x_1 e_2 + x_2 e_1 \quad \text{... (1.17)}$$

Dividing (1.17) by z, we get

$$RE = \left|\frac{e_z}{z}\right| \le \left|\frac{e_1}{x_1}\right| + \left|\frac{e_2}{x_2}\right| \qquad \ldots (1.18)$$

Hence, the relative error modulus of the product of two numbers does not exceed the sum of the relative error moduli of the given numbers.

Example 2 If the given numbers are rounded, estimate the relative and absolute errors of the product, $4.0643 \times .37487$. Find also the range in which the product lies.

Solution Let $x_1 = 4.0643$ and $x_2 = .37487$.

$$z = x_1 \cdot x_2 = 1.5236.$$

$$|e_1| \le \frac{1}{2}\times 10^{-4}; \quad |e_2| \le \frac{1}{2}\times 10^{-5}$$

Relative error, $RE = \left|\frac{e_1}{x_1}\right| + \left|\frac{e_2}{x_2}\right|$

$$< \frac{\frac{1}{2}\times 10^{-4}}{4.0643} + \frac{\frac{1}{2}\times 10^{-5}}{.37487}$$

$$< .2564 \times 10^{-4}$$

Absolute Error, $AE = RE \times z$

$$= .2564 \times 10^{-4} \times 1.5263 = .39 \times 10^{-4}.$$

Thus, the product leis in the range : $z \pm AE$

$$1.5263 \pm .39 \times 10^{-4}$$

or $1.523561 \le z \le 1.523639$

or 1.524 correct to 4 sf (or 3 dp).

Example 3 The values of x_1 and x_2 have been estimated as follows:

$$x_1 = 4.57 + e_1 \text{ and } x_2 = 8.48 + e_2$$

where $|e_1| < .35$ and $|e_2| < .82$. Find the range in which the product of x_1 and x_2 lies.

Solution

Upper Limit:

$$x_1 x_2 = (4.57 + e_1)(8.48 + e_2)$$
$$\leq (4.57 + |e_1|)(8.48 + |e_2|)$$
$$< (4.57 + .35)(8.48 + .82)$$
$$< (4.92)(9.30) = 45.76.$$

Lower Limit:

$$x_1 \cdot x_2 = (4.57 + e_1)(8.48 + e_2)$$
$$\geq (4.57 - |e_1|)(8.48 - |e_2|)$$
$$> (4.57 - .35)(8.48 - .82)$$
$$> (4.22)(7.66) = 32.33.$$

So, the product lies in the range, 32.33 to 45.76.

1.5.4 Error Accumulation in Division

Given two rounded numbers, x_1 and x_2.

Then $z = \dfrac{x_1}{x_2}$; $x_2 \neq 0$. ... (1.19)

As before, $z - e_z = \dfrac{x_1 - e_1}{x_2 - e_2}$

$$= \frac{x_1\left(1 - \dfrac{e_1}{x_1}\right)}{x_2\left(1 - \dfrac{e_2}{x_2}\right)}$$

$$= \frac{x_1}{x_2}\left(1 - \frac{e_1}{x_1}\right)\left(1 - \frac{e_2}{x_2}\right)^{-1}$$

Expanding with the help of binomial theorem and ignoring the product of errors being small, we have,

$$z - e_z = \frac{x_1}{x_2}\left(1 - \frac{e_1}{x_1}\right)\left(1 - \frac{e_2}{x_2}\right)$$

$$= \left(\frac{x_1}{x_2} - \frac{e_1}{x_2}\right)\left(1 + \frac{e_2}{x_2}\right)$$

$$= \frac{x_1}{x_2} - \frac{e_1}{x_2} + \frac{x_1 e_2}{x_2^2}$$

$$e_z = \frac{e_1}{x_2} - \frac{x_1 e_2}{x_2^2} \qquad \ldots (1.20)$$

Dividing (1.20) by z, we get,

$$\frac{e_z}{z} = \left(\frac{\frac{e_1}{x_2} - \frac{x_1 . e_2}{x_2^2}}{\frac{x_1}{x_2}}\right)$$

$$= \frac{x_2}{x_1}\left(\frac{e_1}{x_2} - \frac{x_1 . e_2}{x_2^2}\right)$$

$$= \frac{e_1}{x_1} - \frac{e_2}{x_2}$$

$$RE = \left|\frac{e_z}{z}\right| \le \left|\frac{e_1}{x_1}\right| + \left|\frac{e_2}{x_2}\right| \qquad \ldots (1.21)$$

Thus, the relative error of a quotient of two terms is equivalent to the sum of the relative error moduli of the dividend and divisor.

Example 4 Given the data, $\frac{4.0643}{37.487}$, estimate the following quantities:

a) the relative error,

b) the maximum absolute error, and

c) the range in which the quotient lies.

Solution Let $x_1 = 4.0643$ and $x_2 = 37.487$.

$$z = \frac{x_1}{x_2} = 0.1084$$

$$|e_1| \le \frac{1}{2} \times 10^{-4}$$

$$|e_2| \le \frac{1}{2} \times 10^{-3}$$

a) $$RE = \left|\frac{e_1}{x_1}\right| + \left|\frac{e_2}{x_2}\right|$$

$$\le \frac{\frac{1}{2} \times 10^{-4}}{4.0643} + \frac{\frac{1}{2} \times 10^{-3}}{37.487}$$

$$= .2564 \times 10^{-4}$$

b) $$AE = RE \times z$$

$$= .2564 \times 10^{-4} \times 0.1084 = 0.0287 \times 10^{-4}$$

e) The quotient lies in range, $z \pm AE$

$$0.1084 \pm 0.0287 \times 10^{-4}$$

or 0.1083972 to 0.1084028

or 0.108 correct to 3 dp.

1.5.5 Errors of Powers and Roots

Let $z = x^n$, where n is the power and denotes an integral or a fractional quantity.

As before, $z - e_z = (x - e_1)^n = x^n\left(1 - \frac{e_1}{x}\right)^n$.

Expanding the right side by the binomial theorem, and neglecting higher powers of $\frac{e_1}{x}$, we get

$$z - e_z = x^n\left(1 - n\frac{e_1}{x}\right)$$

$$= x^n - n\,e_1\,x^{n-1}$$

Therefore, $e_z = n e_1 x^{n-1}$. ... (1.22)

Dividing (1.22) by z, we get

$$\frac{e_z}{z} = n\, e_1 \frac{x^{n-1}}{x^n}$$

$$= \frac{n\, e_1}{x}$$

$$RE = \left|\frac{e_z}{z}\right| \le |n| \cdot \left|\frac{e_1}{x}\right| \qquad \ldots (1.23)$$

Thus, the relative error modulus of a factor raised to a power is the product of the modulus of power and the relative error of the factor.

Example 5 Given $\sqrt{48.424}$, determine the maximum absolute error, relative error and the range in which the answer lies.

Solution Let $x = 48.425;\ n = \frac{1}{2}$.

Let $z = \sqrt{x} = \sqrt{48.424} = 6.959$

Therefore, $|e_1| \le \frac{1}{2} \times 10^{-3}$.

$$RE = \frac{1}{2} \times \frac{10^{-3}}{2} \times \frac{1}{48.425}$$

$$= .005 \times 10^{-3}$$

$$AE = RE \times z = 0.005 \times 10^{-3} \times 6.959 = .035 \times 10^{-3}$$

The correct value of z lies in the range, $z \pm AE$, i.e.,

$$6.959 \pm .035 \times 10^{-3}.$$

Example 6 Evaluate $\sqrt{6.2343 \times \frac{.82135}{2.7268}}$, and find the minimum transmitted error if the given numbers are rounded.

Solution Let $x_1 = 6.2343$, $x_2 = .82137$ and $x_3 = 2.7268$.

$$z = \sqrt{\frac{x_1 x_2}{x_3}} = 1.37035$$

$$n = \frac{1}{2};\ |e_1| = |e_3| \le \frac{1}{2}\times 10^{-4};\ \ |e_2| \le \frac{1}{2}\times 10^{-5}$$

$$RE \le \frac{1}{2}\left\{\frac{\frac{1}{2}\times 10^{-4}}{6.2343} + \frac{\frac{1}{2}\times 10^{-5}}{.82137} + \frac{\frac{1}{2}\times 10^{-4}}{2.7268}\right\}$$

$$= .16222 \times 10^{-4}$$

$$AE = RE \times z = .16222 \times 10^{-4} \times 1.37035 = 2.223 \times 10^{-5}$$

So, the answer lies in the range $z \pm AE$

$$1.37035 \pm 2.223 \times 10^{-5}$$

or $\quad$ 1.37033 to 1.37037

Thus, the answer, correct to 3 sf, is 1.37.

1.5.6 Error in Function Evaluation

Let $z = f(x)$.

As before, $z + e_z = f(x + e_1)$.

Using Taylor series expansion and neglecting higher powers of e_1, being small, we have,

$$Z + e_z = f(x) + e_1 f'(x)$$

or $\quad e_z \hat{=} e_1 f'(x)$

Therefore, $AE = |e_z| \le |e_1 f'(x)|$. $\quad$ ... (1.24)

Dividing (1.24) by z, we get,

$$RE = \left|\frac{e_z}{z}\right| \le \left|e_1 \frac{f'(x)}{f(x)}\right| \qquad \text{... (1.25)}$$

The formula can be extended to any given number of factors, for example, $z = f(x_1, x_2, \ldots, x_n)$:

$$RE = \left|\frac{e_z}{z}\right| \le \left|e_1 \frac{\delta f}{\delta x_1}\right| + \left|e_1 \cdot \frac{\delta f}{\delta x_2}\right| + \ldots + \left|e_n \frac{\delta f}{\delta x_n}\right|$$

$$\le \sum_{i=1}^{n} \left|e_i \frac{\delta f}{\delta x_i}\right| \qquad \text{... (1.26)}$$

where $\frac{\delta f}{\delta x_i}$ are partial derivatives with respect to x_i, for i = 1, 2, ..., n.

Example 7 Estimate the absolute and relative errors if (i) $f(x) = e^x$ and (ii) $f(x) = \sin x$, for $x = 0.2345$, where x is rounded.

Solution

(i) $f(x) = e^x = e^{.2345} = 1.2643$

$f'(x) = e^x = 1.2643$

$$|e_1| \le \frac{1}{2} \times 10^{-4}$$

$$RE \le \left| e_1 \frac{f'(x)}{f(x)} \right| \le \frac{1}{2} \times 10^{-4} \times \frac{1.2643}{1.2643} = \frac{1}{2} \times 10^{-4}$$

$$AE = RE \times f(x) = \frac{1}{2} \times 10^{-4} \times 1.2643 = 0.00006$$

(ii) $f(x) = \sin(x) = \sin(0.2345) = .0041$

$f'(x) = \cos(x) = \cos(0.2345) = .9999$

$$|e_1| \le \frac{1}{2} \times 10^{-4}$$

$$RE \le \frac{1}{2} \times 10^{-4} \times \frac{.9999}{.0041} = 121.94 \times 10^{-4}$$

$$AE = RE \times z = 121.94 \times 10^{-4} \times 0.0041 = 4.9995 \times 10^{-5}$$

Note: The given angle in the trigonometric should be reported in radians. If the given angle, say θ, is in degrees, it should be converted to radians as:

$$\theta \text{ in degrees} = \frac{\theta \pi}{180} = \frac{\theta}{57.3} = \theta \times 0.0174 \text{ radians.}$$

1.6 NUMERICAL CANCELLATION

Accuracy may result in loss when two nearly equal numbers are subtracted. For example, the two numbers 9.4157233 and 9.4157227 are each accurate to 8 sf, yet their difference (0.0000006) is accurate to only 1 sf. Thus, care should be taken to avoid such subtractions where possible. This phenomenon is also called **subtractive cancellation.**

Case 1: We take first an example of evaluating roots of the quadratic equation, $ax^2 + bx + c = 0$, where $a \neq 0$. The roots are given by the formula:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

If 4ac is very very small as compared with b^2, the two quantities b and $\sqrt{b^2 - 4ac}$ will be nearly equal and one of the roots will be subject to a large error, thus resulting in a considerable loss of significance. It may lead to uncertainty in deciding whether the roots are real or complex. To avoid this situation we modify the formula in the following way:

If $b > 0$, the bigger root will be computed as, $x_1 = \dfrac{-b + \sqrt{b^2 - 4ac}}{2a}$ and the smaller root will be computed without loss of significance, suing the following:

$$x_1 x_2 = \frac{c}{a}; \quad x_2 = \frac{c}{a x_1}$$

Determining the smaller root by this method is far superior from the point of view of numerical analysis.

We shall illustrate the method by means of the following example.

Example 8 Find the roots of the equation, $x^2 - 40.12x + 1.3 = 0$, correct to 4 sf (The coefficients are exact).

Solution Let the roots be x_1 and x_2. Using the usual quadratic formula, the roots are: $x_1 = 40.087571$, $x_2 = 0.032429$. The larger root x_1 is given to 8 sf, whereas the smaller root x_2 to 5 sf. Thus, there is a loss of 3 sf. The second root is comparatively inaccurate. If the larger root, $x_1 = 40.09$ to 4 sf, then the smaller root, $x_2 = \dfrac{c}{a x_1} = 0.03243$ to 4 sf. Thus, a comparable number of significant figures can be given here as for the larger toot.

Another way to improve the quadratic formula is to calculate the roots with the following formulas:

i) $$x_1 = \frac{-2c}{b + \sqrt{b^2 - 4ac}} \text{ and}$$

ii) $$x_2 = \frac{-2c}{b - \sqrt{b^2 - 4ac}}.$$

In the cases when $|b| \approx \sqrt{b^2 - 4ac}$, we should proceed with caution to avoid loss of precision due to catastrophic cancellation. If $b > 0$, then x_1 should be computed with the formula (ii) and x_2 should be computed with formula (i). However, if $b < 0$, then x_1 should be computed using formula (i) and x_2 should be computed using formula (ii).

Case 2: Another example to illustrate the avoidance of loss of significance is as follows:

Example 9 Compare the results of computing f(500) and g(500) using six digits and rounding. The functions are as follows:

i) $f(x) = x\left[\sqrt{x+1} - \sqrt{x}\right]$ and

ii) $g(x) = \dfrac{x}{\sqrt{x+1}+\sqrt{x}}$.

Solution

i) $f(x) = x\left[\sqrt{x+1} - \sqrt{x}\right]$

$$f(500) = 500\left[\sqrt{500+1} - \sqrt{500}\right]$$

$$= 500\,[22.3830 - 22.3607]$$

$$= 500 \times 0.0223 = 11.1500$$

ii) $g(x) = \dfrac{x}{\sqrt{x+1}+\sqrt{x}}$

$$g(500) = \frac{500}{\sqrt{500+1}+\sqrt{500}}$$

$$= \frac{500}{44.7437} = 11.1748$$

The function g(x) is algebraically equivalent to f(x) as shown below:

$$f(x) = \left[x\sqrt{x+1} - \sqrt{x}\right] \times \frac{\sqrt{x+1}+\sqrt{x}}{\sqrt{x+1}+\sqrt{x}}$$

$$= \frac{x}{\sqrt{x+1}+\sqrt{x}}$$

The answer g(500) = 11.1748 involves less error and is the same as that obtained by rounding the true answer 11.174753 ... to six digits.

1.7 EVALUATION OF FUNCTIONS BY SERIES EXPANSION AND ESTIMATION OF ERRORS

This section deals with the problems of finding values of trigonometric, logarithmic, exponential and other functions by means of series expansion and also estmating errors, which arise when the series are truncated.

We confine our attention to the **Taylor series**, which is considered to be the foundation of numerical analysis. The series is commonly used in deriving several numerical methods.

Let f(x) be a function that is infinitely differentiable on an interval I containing the numbers x_0 and x. Then, for each positive integer n, the value of f(x) at x is given by:

$$f(x) = f(x_0) + (x - x_0) f'(x_0) + \frac{(x - x_0)^2}{2!} f''(x_0) + \frac{(x - x_0)^3}{3!} f'''(x_0)$$

$$+ \ldots + \frac{(x - x_0)^n}{n!} f^n(x_0) + R_n(x, x_0) \qquad \ldots (1.27)$$

where $R_n(x, x_0)$ is the is the remainder term and is included to account for all terms from (n + 1) to infinity.

$$R_n(x, x_0) = \int_0^x \frac{(x - t)^n}{n!} f^{(n+1)}(t)dt$$

$$= \frac{(x - x_0)^{n+1}}{(n+1)!} f^{(n+1)}(Z) \qquad \ldots (1.28)$$

for some unknown number Z which lies between x and x_0.

For a convergent series, $R_n(x, x_0)$ tends to zero as $n \to \infty$, i.e.,

$$\lim_{n \to \infty} R_n(x, x_0) = 0,$$

It follows that $f(x) = \sum_{n=0}^{\infty} \left(\frac{(x - x_0)^n}{n!} f^{(n)}(x_0) \right)$ ... (1.29)

The right hand side of (1.29) is called a **Taylor series representation** for f(x). It is a power of $(x - x_0)$ because the coefficients $\frac{f^{(n)}(x_0)}{n!}$ are constant — that is, they do not depend on x. The quantity Z in the remainder term is unknown and is difficult to calculate it. Nevertheless, we know the range in which Z lies. If we approximate f(x) in (1.27) or (1.29) by the first n term of the series, then the maximum error introduced in this series is given by the remainder term (1.28). Conversely, if the accuracy required is known before hand, then it would be possible to find the number of terms n such that the finite series give the required accuracy.

MacLaurin's Series

When $x_0 = 0$, in the Taylor series, we get MacLaurin's series and MacLaurin's polynomials. From (1.27), we get,

$$f(x) = f(0, x_0) = f(0) + f'(0) + \frac{x^2}{2!} f''(0) + \frac{x^3}{3!} f'''(0)$$

$$+ \ldots + \frac{x^n}{n!} f^{(n)}(0) + R_n(x) \qquad \ldots (1.30)$$

where $R_n(x) = \dfrac{x^{n+1}}{(n+1)!} f^{(n+1)}(Z)$ and $0 \le Z \le x$. ... (1.31)

Further examples of error analyses will be introduced in later chapters.

Example 10 Obtain a second degree polynomial approximation to the function $f(x) = \sqrt{x+1}$, using Taylor series about $x_0 = 0$. Calculate the truncation error for $x = 0.4$.

Solution

$$f(x) = (1+x)^{\frac{1}{2}}; \qquad f(0) = 1$$

$$f'(x) = \frac{1}{2}(1+x)^{-\frac{1}{2}}; \qquad f'(0) = \frac{1}{2}$$

$$f''(x) = -\frac{1}{4}(1+x)^{-\frac{3}{2}}; \qquad f''(0) = -\frac{1}{4}$$

$$f'''(x) = \frac{3}{8}(1+x)^{-\frac{5}{2}}; \qquad f'''(0) = \frac{3}{8}$$

Fron (1.29), we have,

$$f(x) = f(x_0) + (x - x_0) f'(0) + \frac{(x-x_0)^2}{2!} f''(x_0)$$

$$= f(0) + x f'(0) + \frac{x^2}{2} f''(0)$$

$$= 1 + \frac{x}{2} + \frac{x^2}{2} \times -\frac{1}{4}$$

$$= 1 + \frac{x}{2} - \frac{x^2}{8}$$

The truncation error (alternately, absolute error) can be calculated from the remainder term,

$$R(x) \le \frac{x^2}{3!} f'''(0) = \frac{x^3}{8} \times \frac{3}{8} = \frac{x^3}{16}$$

Therefore, $R(.4) \le \dfrac{0.4^3}{16} = 0.0042.$

Example 11 MacLaurin's series for e^x is given by,

$$e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \ldots + \frac{x^n e^z}{n!}$$

where $0 < Z < x$. Determine the number of terms of this series such that their sum giwes the value of e^x correct to 8 dp where $x = 1$.

Solution The remainder term is given by,

$$R(x) \le \left| \frac{x^n}{n!} e^z \right|, \text{ at } Z = x.$$

The maximum relative error,

$$\boxed{RE = \frac{\text{Absolute error}}{\text{Actual value}} = \left| \frac{x^n}{n!} e^z \times \frac{1}{e^z} \right| = \frac{x^n}{n!}}$$

$$= \frac{1}{n!} \text{ at } x = 1.$$

For an accuracy of 8 dp, we have to add n terms such that

$$RE = \frac{1}{n!} < \frac{1}{2} \times 10^{-8}$$

or $\quad n! = 2 \times 10^8 = 200000000$

If we take $n = 11$, $n! = 11! = 39916800$ and for $n = 12$, $12! = 479001600$. It is clear from above that about 12 terms of the series will be required to get an accuracy of 8 dp.

PROBLEMS

1. Find the absolute and relative errors in each of the following cases (all numbers are rounded).

   (a) $187.2 + 93.5$

   (b) 0.281×3.7148

   (c) $\sqrt{28.315}$

   (d) $\sqrt{\dfrac{6.2342 \times .82137}{27.268}}$

   (e) $2.3\,(4.18 - 3.24)$

   (f) $\dfrac{1.3384 - 2.038}{4.577}$

   (g) Evaluate the following as accurately as possible, assuming all values to be rounded.

      i) $8.24 + 5.33$

      ii) $124.53 - 124.52$

      iii) 4.27×3.13

iv) $9.48 \times 0.513 - 6.72$

v) $0.25 \times 2.84/0.64$

vi) $1.73 - 2.16 + 0.08 + 1.00 - 223 - 0.97 + 3.02$

2. If $x = 1.0$ and $y = 2.5$ round-up numbers, find the maximum absolute error involved in evaluating:

(a) $x + y$; (b) $\frac{x}{y}$; (c) $x^2 + xy + y^2$.

3. If two numbers x and y are in error by 1.0 and 0.5 respectively and the value of x is 10 and that of y is 6, state the intervals within exact values of x, y, $x - y$ and $x + y$ lie.

4. (a) Two parameters u and v have been estimated as follows:

$$u = 2.5 + e_1$$

$$v = 4.5 + e_2$$

where $|e_1| < 0.2$ and $|e_2| < 0.4$. Find bounds on the values of the product and quotient of u and v.

(b) The length and breadth of a rectangle are given by 2.52 cm and 1.78 cm respectively. Find the range in which its area lies, giving the answer to as many dp as are meaningful.

(c) Determine the largest relative error in a calculation of the cross-sectional area of a wire from a measurement on its diameter D, where $D = 0.825 \pm 0.002$ cm.

$$(\text{Area} = \pi\frac{D^2}{4})$$

5. (a) Suppose 1.414 is considered to be an approximation of $\sqrt{2}$. Find the absolute and relative errors due to this choice.

(b) If $u = 0.1$ and $v = 0.01$ are rounded numbers, calculate the maximum absolute error in $\frac{u}{v}$.

(c) Determine the maximum relative error where p_1 is calculated from the relation: $p_1u_2^n = p_2u_2^n$; where $n = 1.4$. The maximum relative errors of u_1, u_2 and p_2 are 0.75%, 0.75% and 2.0% respectively.

(d) Obtain the range of values within which lies the exact value of

$2.7654 + 3.8006 - \frac{15.178}{0.9876}$, if all numbers are rounded off.

6. Obtain correctly rounded off answers for each of the following (all quantities are assumed rounded to the number of digits shown):

(i) cos 18°, (ii) sin 0.18, (iii) e^x for $x = 7.765$, (iv) ln x for $x = 1.377$.

(b) Rearranging the series speeds up the convergence:

$$\frac{\pi}{8} = \frac{1}{1\times3} + \frac{1}{5\times7} + \frac{1}{9\times11}\cdots \qquad \ldots (1)$$

Write a computer program in C++ to compute π using this series instead.

(c) Use the Taylor series;

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!}\cdots$$

to write a program to compute cos x, correct to 4 dp, (x being in radians). See how many terms are required to achieve 4-figure agreement with the library function cos ( ).

13. (a) For small x, show that

i) $x + \frac{x^2}{2} + \frac{x^3}{6} + \ldots$ is better than $e^x - 1$.

ii) $\frac{x^3}{6} - \frac{x^5}{120} + \ldots$ is better than $x - \sin x$.

iii) $\frac{x}{2} - \frac{x^2}{8} + \ldots$ is better than $1 - \sqrt{1-x}$.

(b) For value of v in the neighbourhood of $\frac{\pi}{2}$, show that $2\sin^2(\frac{\pi}{2} - v)\,z$ is better than $1 - \sin v$.

14. Use Taylor's theorem to estimate the truncation error in each of the following approximation formulas, when the step size h is small:

a) $$f'\left(x + \frac{h}{2}\right) = \frac{f(x+h) - f(x)}{h}$$

b) $$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

c) $$f'(x) = \frac{-f(x+2h) + 8f(x+h) - 8f(x-h) + (x-2h)}{12h}$$

15. Derive the Taylor series approximate

$$\ln(1+x) = x - \frac{1}{2}x^2 + \frac{1}{3}x^3 - \frac{1}{4}x^4 + \ldots + \frac{(-1)^{n-1}}{n}x^n$$

stating clearly the form of the error term. How might it be bounded?

Chapter 2

Finite Differences

2.1 DIFFERENCE TABLE

Suppose we have a function f(x) which is tabulated over a range of values (called **tabular points**) of the independent variable x. Let us denote the uniform difference (constant spacing or step-size) between any two successive values by h so that,

$$x_1 - x_0 = h = x_2 - x_1 = \ldots = x_n - x_{n-1}$$

or $x_1 = x_0 + h$

$$x_2 = x_1 + h = x_0 + 2h$$

$$\ldots$$

$$x_p = x_0 + ph$$

$$\ldots$$

$$x_n = x_0 + nh$$

and $f(x_p) = f_p = f(x_0 + ph)$

In many numerical processes concerned with tabulated functions certain quantities called **finite differences** are important. A finite difference is a mathematical expression of the form $f(x + b) - f(x + a)$. The procedure to compute differences is explained below.

To build up the difference table, we first write down the values of x_i 's as well as the corresponding values of f_i 's as shown below:

x_i	f_i	1st	2nd	3rd
x_0	f_0			
		$f_1 - f_0$		
x_1	f_1		$f_2 - 2f_1 + f_0$	
		$f_2 - f_1$		$f_3 - 3f_2 + 3f_1 - f_0$
x_2	f_2		$f_3 - 2f_2 + f_1$	
		$f_3 - f_2$		$f_4 - 3f_3 + 3f_2 - f_1$
x_3	f_3		$f_4 - 2f_3 + f_2$	
		$f_4 - f_3$		
x_4	f_4			

The first-order differences are obtained from the second column by subtracting each value from the next below and placing the differences to the right but halfway between the two values from which they have been obtained. In this way, the column containing all the first-order differences is formed, but each difference column contains one entry less than its predecessor column.

We are now in a position to produce a column of second-order differences from the column of the first-order differences in a similar way. In computing differences, great care should be exercised to avoid arithmetic errors in the subtractions – the fact that we subtract the upper value from the lower causes a real source of confusion. The sign of the differences is important and shows whether the function is increasing or decreasing in the range of the values obtained.

There are several uses of a difference table; a few of which are as follows:

i) A difference table provides a convenient way for examining at a glance how a particular function behaves. It is particularly applicable in determining the behaviour of the derivatives of a given function.

ii) If there are some errors in the data, the differences will also contain errors. By inspecting the difference table, often the error (or errors) can be detected and corrected.

iii) It helps in filling missing values.

iv) It helps in extending the list of values.

The word **finite** refers to the finite-size of the interval (increment) used in the table as opposed to the infinitesimal interval, which are met in infinitesimal calculus. For this reason, the theory and application of finite differences is sometimes referred to as **Finite Calculation**. It plays an important role in interpolation, numerical differentiation, numerical integration, numerical solutions of difference, ordinary and partial differential equations and time series analysis.

A numerical example at this stage should help clarify some basic concepts for constructing a difference table.

Example 1 Construct the difference table for the function $f(x) = x^4$ for $x = -2$ to $x = 4$, at the interval of 1. [Usually written as $x = -2(1)4$; the figure in brackets being the constant increment.]

Solution The values of f_i and the differences are shown in the table below:

x_i	$f_i = x_i^4$	1st	2nd	3rd	4th	5th
–2	16					
		–15				
–1	1		14			
		–1		–12		
0	0		2		24	
		1		12		0
1	1		14		24	
		15		36		0
2	16		50		24	
		65		60		
3	81		110			
		175				
2	256					

An examination of the difference table reveals that all fourth-order differences are constant and thus the fifth and all higher-order differences would be zero, which is the peculiar property of an exact polynomial (i.e., when all entries in the table are exact and not rounded).

Some obvious results

a) The nth-differences of an exact polynomial of degree n are constant.

b) The (n + 1)st differences of that polynomial are zero.

c) The above values are only true of polynomials when they are tabulated at equal intervals.

If the function does not represent an exact polynomial, the above results will not hold. In practice, we always deal with rounded numbers, where we seldom come across a column with all its differences zeros. The differences of rounded numbers are irregular and thus give rise to the irregular part of the table. In that case, the nth-order differences due to the rounding errors oscillate between $\pm 2^{n-1}$.

The reason for this is that when the tabulated values are rounded, each value has an error usually lying in the range $\pm \frac{1}{2}$, if we work in units of the last place. These errors will build-up in the differences just as do mistakes, and eventually, if the true values have convergent differences, they will become greater than the true differences. In the worst case, the rounded-off errors will be alternately $+\frac{1}{2}$ and $-\frac{1}{2}$ and their contribution to any

nth difference will be,

$$\pm\frac{1}{2}\cdot\left\{\binom{n}{0}+\binom{n}{1}+\binom{n}{2}+\ldots+\binom{n}{n}\right\}=\pm 2^{n-1}$$

Example 2 Construct the difference table for the function $f(x)=\sqrt{x^2+x+1}$, rounded to 4 dp, for x = 10 (1)16.

x	$f(x)=\sqrt{x^2+x+1}$	1st	2nd	3rd	4th
10	10.5375				
		.9969			
11	11.5326		5		
		.9974		–2	
12	12.5300		3		3
		.9977		1	
13	13.5275		4		–2
		.9981		–1	
14	14.5258		3		–1
		.9984		–2	
15	15.5242		1		
		.9985			
16	16.5227				

Since the function is tabulated at 4 dp, each difference is also to 4 dp. Because of this, the decimal point and the leading zeros may be omitted in the formation of a difference table and they may then be written as integers. This makes the table easier to construct and much neater too. For instance, the first entry in the column of fourth differences is an abbreviation of 0.0003. The table shows that the fourth-order differences oscillate and are all within the range $\pm 2^{4-1}=\pm 8$.

2.2 DETECTION AND CORRECTION OF ERRORS IN A DIFFERENCE TABLE

It is likely that an error (errors) may show up while constructing differences. We observe a very peculiar kind of error propagation, which we shall illustrate in this section. An error caused by reversing the order of a pair of digits in a number is commonly made in copying down the number from the given data. It affects the other differences in the table. We may denote the error in a single entry in the difference table by the symbol, ε, which can be negative, positive, small or large. Its effect on the differences spreads out fan-wise as shown in the table below:

x	f	1st	2nd	3rd	4th
0	0				
		0			
1	0		0		ε
		0		ε	
2	0		ε		–4ε
		ε		–3ε	
3	ε		–2ε		+6ε
		–ε		3ε	
4	0		ε		–4ε
		0		–ε	
5	0		0		ε
		0			
6	0				

This fan-wise (triangular patterns) propagation of ε in the difference table grows quickly and makes it possible in certain cases to locate an error and also to find its numerical value, thus enabling us to rectify it with the help of tabular values. A glance at the table reveals that the coefficients of ε in the nth-order differences are binomial coefficients of x, which occur in the expansion of $(1 - x)^n$. For example, the coefficients in the third-order difference column are 1, –3, 3, –1, which occur in the expansion of $(1 - x)^3$ in the increasing powers of x, i.e., $1 - 3x + 4x^2 - x^3$. The corresponding coefficients for the fourth-order differences of $(1 - x)^3$ are 1, –4, 6, –4, 1. The binomial coefficients in the fifth and sixth difference columns are 1, –5, 10, –10, 5, –1 and 1, –6, 15, –20, 15, –6, 1, respectively. The table shows that the higher-order differences are very sensitive to slight changes in any of the ordinates or lower-order differences. Relatively small input changes generate relatively large output changes. For the identification of gross errors, the above picture should be kept in mind.

We illustrate the procedure by means of the following example.

Example 3 The following table contains an incorrect value of f(x). Locate the error, suggest a possible cause and a suitable correction:

x	1	2	3	4	5	6	7	8	9	10
f(x)	37	74	135	226	353	531	739	1010	1341	1738

Solution **Difference Table**

x	f	1st	2nd	3rd	4th
1	37				
		37			
2	74		24		
		61		6	
3	135		30		0
		91		6	
4	226		36		9
		127		15	
5	353		51		–36
		178		–21	
6	531		30		54
		208		33	
7	739		63		–36
		271		–3	
8	1010		60		9
		331		6	
9	1341		66		
		397			
10	1738				

In the above table, f(x) seems to represent an exact polynomial; thus all fourth-order differences should be zero. The error seems to have appeared in the fourth-order differences with coefficients: 1, – 4, 6, – 4, 1. The incorrect difference may be written as:

$$1(9), -4(9), 6(9), -4(9), 1(9)$$

This indicates that the error is 9. The next step is to locate the incorrect functional value. This can be moving backward to the second column. It shows that the term in error is 531 and the correct value is 531 – 9 = 522. The likely cause of the error may be due to wrongly copying the digits. The result can be checked by correcting the wrongly-placed entry and reconstructing the difference table. If the function is known analytically, it would be preferable to recalculate it at x = 6, so that the correction can be made with certainty rather just estimated.

In the above example, the functional values are exact and it was fairly easy to locate and correct the error with certainty, but this is not always the case especially when the values of f(x) have been rounded, since the errors will not then be exact multiples of the binomial coefficients. In such a case, we can only make an estimate of the error. Moreover, in a difference table in which there are two or more errors, their fans will eventually overlap, making it more difficult to discover the errors. Some more care is necessary to find out a reasonable pattern to locate the error(s) in such cases.

A table in which two errors have been made is more difficult to analyze since the binomial coefficients overlap. The following pattern shows a possible example.

Solution **Difference Table**

f	1st	2nd	3rd	4th
0				
	0			
0	→	0		
	0	→	ε_1	
0	→	ε_1	→	$-4\varepsilon_1$
	ε_1	→	$-3\varepsilon_1$	
ε_1	→	$-2\varepsilon_1$	→	$6\varepsilon_1$
	$-\varepsilon_1$	→	$3\varepsilon_1$	
0	→	ε_1	→	$\varepsilon_2-4\varepsilon_1$
	0	→	$\varepsilon_2-4\varepsilon_1$	
0	→	ε_2	→	$-4\varepsilon_1+4\varepsilon_1$
	ε_2	→	$-3\varepsilon_2$	
ε_2	→	$-2\varepsilon_2$		$6\varepsilon_2$
	$-\varepsilon_2$	→	$3\varepsilon_2$	
0	→	ε_2		
	0			
0				

It may be possible to identify the error pattern in the third-order differences column but the confusion in the fourth-order difference would probably be too great to give an opportunity to detect the error. We, therefore, concentrate on the problems where only one mistake is made.

Example 4 It is suspected that the following table contains an error. By differencing, locate any probable error and correct it. Check by re-differencing if any correction is made. The values of f(x) are rounded to 3 dp.

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7
f(x)	0.905	0.819	0.741	0.677	0.607	0.549	0.497

Solution **Difference Table**

x	f (x)	1st	2nd	3rd	4th
0.1	0.905				
		– 86			
0.2	0.819		8		
		– 78		6	
0.3	0.741		14		– 26
		– 64		– 20	
0.4	0.677		– 6		38
		– 70		18	
0.5	0.607		12		– 24
		– 58		– 6	
0.6	0.549		6		
		– 52			
0.7	0.497				

Comparing the third-differences with the coefficients of x, we get

1	– 3	3	– 1
↓	↓	↓	↓
6	– 20	18	– 6

We may deduce that $\varepsilon = 6$, i.e., 0.006 and the corrected value of f(0.4) = 0.677 – 0.006 = 0.671. We can check by reconstructing the difference table, but we are not 100% sure that it is the correct value, we just estimated. However, it may prove to be a reasonable estimate. If the entries are more in the data and the above mentioned options fail to give a reasonable clue to pick-up the error, we should use of the fifth or sixth difference column and then try again.

If there is no obvious pattern for locating an error in the difference table, we use the following formula for finding the error.

$$\text{Error} = \frac{\text{Largest value in a column}}{\text{Corresponding coefficient of } \varepsilon \text{ in that column}}$$

In this section, we have studied how to locate and correct a single error in a difference table. If there are two or more errors in the entries, it is usually not easy to separate their overlapping effects and thus locations and corrections of such errors become extremely difficult. In some cases, irregular behaviour of the differences may be caused not by errors but by irregularities in the functions.

2.3 DIFFERENCE OPERATIONS

To refer to specific entries in a difference table we use some operators, called **difference operators**. An operator is not a number but it is an operation, which when applied to a function changes it to some other function. The operator technique proves to be a most useful tool when we wish to construct formulas for interpolation, numerical differentiation, numerical integration, etc. One of the biggest advantages is that we can fix the type of formula desired in advance and then proceed directly toward the goal.

The following operators are commonly used:

Δ Forward-difference operator (usually read as delta)

∇ Backward-difference operator (usually read as del or nebla)

δ Central difference operator (read as sigma)

μ Average (mean) operator (read as mu)

E Shift operator

Let us define these operators one by one. It must be emphasized that these operators assume equally-spaced data points.

2.3.1 Forward Difference Operator

The forward difference operator Δ is defined by the following relation:

$$\Delta f_r = f_{r+1} - f_r$$

where r is an integer, and $\Delta f_r = \Delta f(x_r)$.

Also, $\Delta f_{r+1} = \Delta f(x_r + h)$ and $\Delta f_{r+\frac{1}{2}} = \Delta f\left(x_r + \frac{h}{2}\right)$.

In words, when Δ operates on a function, we first shift r by r + 1 and then subtract the original function from the shifted function. This produces the difference function Δf_r.

Thus, $\Delta f_0 = f_1 - f_0$

$$\Delta f_1 = f_2 - f_1$$

$$\vdots$$

$$\Delta f_{-1} = f_0 - f_{-1}, \text{ etc.}$$

$\Delta f_{-1}, \Delta f_0, \Delta f_1$, are called **first-order forward differences**. The differences of the first-order differences are called **second-order differences** and are computed as follows:

Thus, $\Delta^2 f_r = \Delta(\Delta f_r)$

$$= \Delta(f_{r+1} - f_r)$$

$$= \Delta f_{r+1} - \Delta f_r$$

$$= (f_{r+2} - f_{r+1}) - (f_{r+1} - f_r)$$

$$= f_{r+2} - 2f_{r+1} + f_r$$

The higher-order differences are obtained in the same way.

Thus, $\Delta^3 f_r = \Delta(\Delta^2 f_r)$

$$= \Delta\{f_{r+2} - 2f_{r+1} + f_r\}$$

$$= \Delta f_{r+2} - 2\Delta f_{r+1} + \Delta f_r$$

$$= (f_{r+3} - f_{r+2}) - 2(f_{r+2} - f_{r+1}) + (f_{r+1} - f_r)$$

$$= f_{r+3} - 3f_{r+2} + 3f_{r+1} - f_r$$

$$\Delta^4 f_r = f_{r+4} - 4f_{r+3} + 6f_{r+2} - 4f_{r+1} + f_r$$

In general, nth-order differences are given by:

$$\Delta^n f_r = \Delta^{n-1} f_{r+1} - \Delta^{n-1} f_r$$

where $\Delta^n f_r \neq (\Delta f_r)^n$, and $n \geq 1$.

The following difference table shows how the forward differences of all orders can be formed:

x	f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x_0	f_0				
		Δf_0			
x_1	f_1		$\Delta^2 f_0$		
		Δf_1		$\Delta^3 f_0$	
x_2	f_2		$\Delta^2 f_1$		$\Delta^4 f_0$
		Δf_2		$\Delta^3 f_1$	
x_3	f_3		$\Delta^2 f_2$		
		Δf_3			
x_4	f_4				

We observe from the above table that differences with the same subscripts all lie on a downward sloping diagonal.

While experimenting with differences, we observe that if x^n is a polynomial of degree n, then Δx^n is a polynomial of degree $(n - 1)$. In other words, differencing behaves like differentiation in the sense of reducing the degree of a polynomial.

Thus, $\Delta x^n = (x+1)^n - x^n$

$$= nx^{n-1} + n(n-1)x^{n-2} + \ldots$$

If the above process is continued for n times, the polynomial x^n is reduced to degree zero, i.e., constant. This is exactly what was shown by Example 1, that the nth-order differences of a polynomial of degree n are constant and all higher-order differences are zero.

Algorithm for Generating Differences Using Forward Scheme

In general, for a function tabulated at n points, the corresponding forward difference table can be represented by a matrix of size $(n - 1) * (n - 1)$. Note that only the elements in the columns from 1 to $n - i$, where the row $i = 1, 3, \ldots, n - 1$, are of interest.

The algorithm to generate forward differences table may look like the following:

Steps

```
For   J = 1 TO n - 1 by 1 DO
      FOR   I = 1 TO n - J by 1 DO
            IF (J = 1) THEN
      SET   D_IJ = F(X_IJ) - D_I,J-1
            ELSE
      SET   D_IJ = D_I+1,J-1 - D_I,J-1
      PRINT "all differences, D_I,J"
```

This algorithm will compute the forward differences of all orders that can be computed from the given function table. The data with equi-spaced abscissas are initialized in the program.

Example 5 Computerize the algorithm for generating forward differences. Use the following test data:

x	1.2	1.4	1.6	1.8	2.0
y	5.64642	6.44218	7.17356	7.83327	8.41471

Solution

Program No. 1: Difference Table

```
# include<iostream.h>
# include<stdio.h>
# include<conio.h>

void main(void)
  {
    clrser( );
    float interval, array[20][20]={0.0};
    int no, col, x, y;

    cout<<"\tDIFFERENCE TABLE";
    cout<<"\n\n\tENTER THE FIRST VALUE : "; cin>>array[0][0];
    cout<<"\n\tENTER THE INTERVAL : "; cin>>interval;
    cout<<"\n\tENTER TOTAL NO. OF X : "; cin>>no;

    for(int i=1; i<no; i++)
        array[i][0]=array[i-1][0]+interval;

    cout<<"\n\n\tENTER FUNCTIONAL VALUES : \n";
    for(i=0;i<no;i++)

        {
          cout<<"\tX("<<i<<") = ";cin>>array[i][1];
        }

    cout<<"\n\tHOW MANY COLUMNS ARE REQUIRED : "; cin>>col;
    for(i=1; i<=(col+2);i++)
        array[j][i]=array[j+1][i-1]-array[j][i-1];

    clrscr( );
    cout<<"\t\tDIFFERENCE TABLE\n\n";
    cout<<" X          F(X)";
    for(i=1;i<col;i++)
        cout<<"     col     "<<;

    cout<<"\n\n";

    for(i=0;i<no;i++)
        cout<<""<<array[i][0]<<"\n\n";
```

```
x=8; y=5;
for(i=1;i<=(col+1);i++)
  {
    gotoxy(x,y);
    for(int j=0;j<=(no-i);j++)
      {
        cout<array[j][i];
        y+=2;
        gotoxy(x,y);
      }
    x+=9; y=i+5;
  }
}
```

DIFFERENCE TABLE

ENTER THE FIRST VALUE : 1.2

ENTER THE INTERVAL : 0.2

ENTER TOTAL NO. OF X : 5

ENTER FUNCTIONAL VALUES:

X(0) = 5.64642

X(1) = 6.44218

X(2) = 7.17356

X(3) = 7.83327

X(4) = 8.41471

HOW MANY COLUMNS ARE REQUIRED : 4

Computer Program

DIFFERENCE TABLE

X	F(X)	col 1	col 2	col 3	col 4
1.2	5.64642				
		0.79576			
1.4	644218		– 0.06438		
		0.73138		– 0.07167	
1.6	7.17356		– 0.07167		0.00069
		0.65971		– 0.00660	
1.8	7.83327		– 0.07827		
		0.58144			
2.0	8.41471				

2.3.2 Backward Difference Operator

The backward difference operator ∇ is defined by the following relation:

$$\nabla f_r = f_r - f_{r-1}$$

Hence, we shift r backward by one step, the function becomes f_{r-1} and subtract this function from the original f_r.

Thus, $\nabla f_1 = f_1 - f_0$

$$\nabla f_0 = f_0 - f_{-1}$$

$$\nabla f_2 = f_2 - f_1$$

The above differences are called first-order backward differences. In a similar manner, we can define backward differences of higher-orders. Thus, we obtain:

$$\begin{aligned}\nabla^2 f_r &= \nabla(\nabla f_r)\\ &= \nabla(f_r - f_{r-1})\\ &= \nabla f_r - \nabla f_{r-1}\\ &= (f_r - f_{r-1}) - (f_{r-1} - f_{r-2})\\ &= f_r - 2f_{r-1} + f_{r-2}\end{aligned}$$

Similarly, $\nabla^3 f_r = f_r - 3f_{r-1} + 3f_{r-2} - f_{r-3}$.

In general, nth-order differences are given by:

$$\nabla^n f_r = \nabla^{n-1} f_r - \nabla^{n-1} f_{r-1}\,;\ n \geq 1.$$

With the help of this operator, we can construct the table for backward differences:

x	f	∇f	$\nabla^2 f$	$\nabla^3 f$	$\nabla^4 f$
x_0	f_0				
		∇f_1			
x_1	f_1		$\nabla^2 f_2$		
		∇f_2		$\nabla^3 f_3$	
x_2	f_2		$\nabla^2 f_3$		$\nabla^4 f_4$
		∇f_3		$\nabla^3 f_4$	
x_3	f_3		$\nabla^2 f_4$		
		∇f_4			
x_4	f_4				

We observe from the above table differences with the same subscripts all lie on an upward sloping diagonal.

Algorithm to Generate Differences Using Backward Scheme

In general, for a function tabulated at n points, the corresponding backward difference table can be represented by a matrix of size $n \times n$. Note that only the elements in the columns from 1 to $n - i$, where the row $i = 2, 3, \ldots, n$, are of interest.

The algorithm to generate backward differences table may look like the following:

Steps

```
For     J = 1 TO n – 1 by 1 DO
        FOR     I = J + 1 TO n by 1 DO
                IF (J = 1) THEN
        SET     D_IJ = F(X_I) – F(XD_I-1, J-1)
                ELSE
        SET     D_IJ = D_I,J-1 – D_I-1,J-1
        PRINT "all differences, D_I,J"
```

23.3 Central Difference Operator

The central difference operator δ is defined as:

$$\delta f_r = f_{r+\frac{1}{2}} - f_{r-\frac{1}{2}}$$

Thus, $\delta f_{r+\frac{1}{2}} = f_{\left(r+\frac{1}{2}\right)+\frac{1}{2}} - f_{\left(r-\frac{1}{2}\right)-\frac{1}{2}} = f_{r+1} - f_r$

Similarly, $\delta^2 f_r = \delta(\delta f_r)$

$$= \delta\left(f_{r+\frac{1}{2}} - f_{r-\frac{1}{2}}\right)$$

$$= \delta f_{r+\frac{1}{2}} - \delta f_{r-\frac{1}{2}}$$

$$= (f_{r+1} - f_r) - (f_r - f_{r-1})$$

$$= f_{r+1} - 2f_r + f_{r-1}$$

In general, nth-order differences are given by:

$$\delta^n f_r = \delta^{n-1} f_{r+\frac{1}{2}} - \delta^{n-1} f_{r-\frac{1}{2}}$$

The difference table for δ is given below:

x	f	δf	$\delta^2 f$	$\delta^3 f$	$\delta^4 f$
x_0	f_0				
		$\delta f_{\frac{1}{2}}$			
x_1	f_1		$\delta^2 f_1$		
		$\delta f_{\frac{3}{2}}$		$\delta^2 f_{\frac{3}{2}}$	
x_2	f_2		$\delta^2 f_2$		$\delta^4 f_4$
		$\delta f_{\frac{5}{2}}$		$\delta^3 f_{\frac{5}{2}}$	
x_3	f_3		$\delta^2 f_3$		
		$\delta f_{\frac{7}{2}}$			
x_4	f_4				

We note that all differences with the same subjects lie on the same horizontal line and all even-order differences have integer subscripts. The central difference notation is preferable for many purposes but has the disadvantage of requiring fractional suffixes.

It is to be kept in mind that whatever notation we use, there is only one difference table and hence each entry in the table has one of the three names, for instance,

$$f_{r+1} - f_r = \Delta f_r = \nabla f_{r+1} = \delta f_{r+\frac{1}{2}}$$

Also, $\Delta f_0 = \nabla f_1 = \delta f_{\frac{1}{2}}$

$$\Delta^2 f_0 = \nabla^2 f_2 = \delta^2 f_1$$

$$\Delta^3 f_2 = \nabla^3 f_5 = \delta^3 f_{\frac{7}{2}}$$

$$\Delta^4 f_{-2} = \nabla^4 f_2 = \delta^4 f_0, \text{ etc.}$$

2.3.4 Shift Operator

The shift operator (also called the **step operator**) E is defined by,

$$E f_r = f_{r+1}$$

$$E^{-1} f_r = f_{r-1}$$

$$E^2 f_r = E(f_r) = E f_{r+1} = f_{r+2}$$

In general, $E^n f_r = f_{r+n}$.

2.3.5 Mean Operator

The mean (or average) operator μ is defined by,

$$\mu f_r = \frac{1}{2}\left(f_{r+\frac{1}{2}} + f_{r-\frac{1}{2}}\right)$$

Thus, $\mu f_{r+\frac{1}{2}} = \frac{1}{2}\left\{f_{r+\frac{1}{2}+\frac{1}{2}} + f_{r+\frac{1}{2}-\frac{1}{2}}\right\} = \frac{1}{2}(f_{r+1} + f_r)$.

2.4 RELATIONSHIPS BETWEEN OPERATORS

Various relationships exist between operators. For example,

$$\Delta f_r = f_{r+1} - f_r$$

$$\Delta f_r = Ef_r - f_r = (E-1)\, f_r$$

or, $\Delta = E - 1$

or, $E = E - \Delta$

Similarly, $\nabla f_r = f_r - f_{r-1}$

$$= f_r - E^{-1} f_r$$

or, $\nabla = 1 - E^{-1}$

and $E = (1-\nabla)^{-1}$

$$\delta f_r = f_{r+\frac{1}{2}} - f_{r-\frac{1}{2}} = E^{\frac{1}{2}} f_r - E^{-\frac{1}{2}} f_r$$

$$= \left(E^{\frac{1}{2}} - E^{-\frac{1}{2}}\right) f_r$$

or, $\delta = E^{\frac{1}{2}} - E^{-\frac{1}{2}}$

Also, $\mu f_r = \frac{1}{2}\left(f_{r+\frac{1}{2}} + f_{r-\frac{1}{2}}\right)$

$$= \frac{1}{2}\left(E^{\frac{1}{2}} + E^{-\frac{1}{2}} f_r\right)$$

$$= \frac{1}{2}\left(E^{\frac{1}{2}} + E^{-\frac{1}{2}}\right) f_r$$

or $\mu = \frac{1}{2}\left(E^{\frac{1}{2}} + E^{-\frac{1}{2}}\right)$

The relationships between various operators are given in the following table:

	E	Δ	∇	δ
E	E	$1+\Delta$	$(1-\nabla)^{-1}$	$1+\frac{1}{2}\delta^2+\delta\sqrt{1+\frac{\delta^2}{4}}$
Δ	$E-1$	Δ	$\nabla(1-\nabla)^{-1}$	$\frac{\delta^2}{2}+\delta\sqrt{1+\frac{\delta^2}{4}}$
∇	$1-E^{-1}$	$\Delta(1+\Delta)^{-1}$	∇	$-\frac{\delta^2}{2}+\delta\sqrt{1+\frac{\delta^2}{4}}$
δ	$E^{\frac{1}{2}}-E^{-\frac{1}{2}}$	$\Delta(1+\Delta)^{-\frac{1}{2}}$	$\nabla(1-\nabla)^{-\frac{1}{2}}$	δ
μ	$\frac{1}{2}\left(E^{\frac{1}{2}}+E^{-\frac{1}{2}}\right)$	$\left(1+\frac{\Delta}{2}\right)(1+\Delta)^{-\frac{1}{2}}$	$\left(1-\frac{\nabla}{2}\right)(1-\nabla)^{-\frac{1}{2}}$	$\sqrt{1+\frac{\delta^2}{4}}$

The above relationships can easily be proved and we leave this as an exercise to the student to fill in the details of the above results.

PROBLEMS

1. Construct the difference tables for the following functions:

   a) $f(x) = x^4 - x - 1$, over the range, $x = -3(1)5$.

   b) $f(x) = 2x^3 + 2x^2 + 2x - 1$, over the range, $x = -1(1)7$.

   c) $f(x) = 2x^3 + 2x^2 - 3x + 4$, over the range, $x = -1(.5)1$.

   d) $f(x) = 2^x$ for $x = 0(1)6$. Will there ever be a column of constant differences in this case?

   e) $f(x) = \sin x$ for $x = 1.0(0.1)1.6$.

   f) $f(x) = 2x^3 + 3x + 1$ for $x = 0.1(0.1)0.5$. What can you say about fourth-order difference column? What is the reason for your observation?

   g) $f(x) = 3x^3 + 4x^2 + 1$ and $f(x) = x^3$ for values $x = 0(1)5$. What do you conclude from the third-order differences column of the difference tables based on the above functions?

2. (a) It is suspected that there is an error in one of the values of f(x) in the following table:

x	1.0	1.52	2.0	2.5	3.0	3.5	4.0	4.5	5.0
f(x)	–38	–46	–59	–76	–92	–118	–140	–161	–180

Construct the differences-table, detect and correct the error.

(b) Consider the following table of values:

x	1	2	3	4	5	6	7	8	9	10	11
f(x)	7	10	17	33	63	121	185	287	423	598	817

It is suspected that one of the values may have been recorded in error. Assuming that the data follow a polynomial, determine which one, if any, of the functional values is in error and what it should be?

(c) Locate the error and estimate the correct value for the following table:

x	0	1	2	3	4	5	6	7	8
f(x)	1.0000	1.1002	1.2013	1.3045	1.4105	1.5210	1.6366	1.7586	1.8881

Construct the differences-table, detect and correct the error.

3. Locate and correct mistakes in each of the following tables:

x	1	2	3	4	5	6	7	8	9	10
f(x)	7	12	21	34	51	70	97	126	159	196
z(x)	.500	.520	.540	.560	.579	.589	.618	.637	.655	.674

Construct the differences-table, detect and correct the error.

4. The table of values for two quadratic polynomials y(x) and z(x) are given to 3 sf as follows:

x	1.0	1.1	1.2	1.3	1.4	1.5	1.6
y(x)	1.00	1.13	1.35	1.76	2.10	2.69	3.46
z(x)	4.00	4.87	5.91	7.15	8.60	10.3	12.3

Locate and correct the errors (other than those attributed to rounding off) in each table.

5. Compute the missing entries in the following tables:

a.

f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x				
	x			
x		−4		
	3		−1	
9		−5		x
	x		x	
x		0		
	x			
x				

b.

f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x				
	x			
x		−3		
	2		x	
7		−3		x
	x		3	
x		0		
	x			
x				

c.

f	Δf	$\Delta^2 f$	$\Delta^3 f$	$\Delta^4 f$
x				
	x			
1		12		
	x		x	
x		x		24
	60		60	
x		108		
	x			
241				

6. (a) The following table of values contains an error. Locate the incorrect value and find an estimate of correct value:

x	−1	0	1	2	3	4	5	6
f(x)	1.51	1.17	1.51	2.35	4.23	6.61	9.67	13.41

Reconstruct the difference table with the correct value. Comment on the nature of the function f(x).

(b) The table below contains an error. Locate the incorrect the error:

x	3.60	3.61	3.62	3.63	3.64	3.5	3.66	3.67	3.68
f(x)	.112046	.120204	.128350	.136462	.144600	.152702	.60788	.168857	.17690

(c) Use the difference table method to locate and correct the error in the following table of values:

x	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
f(x)	10.30	10.70	11.04	11.26	11.30	11.01	10.60	9.74	8.46	6.70

7. (a) Form the difference table for the function given below. Find the values of a, b, and c, so that $\Delta^4 f(a) = \nabla^4 f(b) = \delta^4 f(c) = -0.0428$.

x	0	1	2	3	4	5	6
f(x)	.3679	.7358	.9197	.9810	.9963	.9994	.9999

(b) Tabulate the function $f(x) = x(x-1)(x-2)$ for $x = -0.2(0.1)0.2$, correct to 3 dp. What do you say on the value of $\delta^4 f$?

(c) Prove that the sum of the numbers in any column of a difference table is equal to the difference between the last and first numbers in the preceding column.

Set up a table showing the first and second differences for the following data to check the arithmetical work:

0.0000, – 0.0104, – 0.0206, – 0.0307, – 0.0404, – 0.0496.

(d) (i) Construct the difference table for the following functional values:

x	−2	−1	0	1	2	3	4
f(x)	15	1	1	3	19	85	261

If the origin $x_0 = 1$, determine the values of Δf_0, ∇f_{-1}, $\delta f_{\frac{3}{2}}$, $\delta^2 f_1$, $\Delta^3 f_0$, $\nabla^3 f_2$, $\Delta^2 f_1$ and $\delta^4 f_0$.

(ii) Given the set of values:

x	10	15	20	25	30	35
y	19.97	21.51	22.47	23.52	24.65	25.89

Construct the difference table and report the values of Δy_{20}, $\Delta^2 y_{10}$, $\Delta^3 y_{15}$ and $\Delta^5 y_{10}$.

(e) Given the difference table:

f	f(x)	1st	2nd	3rd	4th
−1	−14				
		14			
0	0		0		
		14		96	
2	14		96		0
		110		96	
4	124		192		
		302			
6					

If the origin $x_0 = 2$, express using forward, backward and central differences in the entries, 110, 302 and 192.

8. (a) The values of y shown in the following table are alleged to be derived from a fourth entries degree polynomial. Test this and correct the value, where necessary.

x	0	1	2	3	4	5	6	7	8	9	10
f(x)	0	2	20	90	272	605	1332	2450	4160	6642	10100

(b) Suggest appropriate correction for the following table of values:

x	10	11	12	13	14	15	16	17	18
f(x)	2.1544	2.2240	2.2894	2.3513	2.4121	2.4662	2.5198	2.5713	2.6207

(c) The following table contains an error. Identify the error and estimate the correct value of the function:

x	1	2	3	4	5	6	7	8	9
f(x)	10	12	15	21	32	50	79	115	166

(d) Locate the incorrect entry in the following tables and estimate the correct value of each function:

(i)

x	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
f(x)	−9.800	−9.061	−8.341	−7.594	−6.671	−5.776	−4.530	−2.945	−0.899	1.736	5.100

(ii)

x	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
f(x)	0.905	0.819	0.741	0.677	0.607	0.549	0.497	0.449	0.407

(e) Locate the incorrect entry in the following table and find its correct value:

x	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7
f(x)	0.000	0.012	0.072	0.252	0.672	1.500	2.952	5.922

0.8	0.9	1.0	1.1	1.2
8.832	13.932	21.000	30.492	42.912

9. Prove the following relationships:

(a) $\Delta = \sqrt{E} \cdot \delta$ (b) $\Delta = E\nabla$ (c) $\delta^2 = \Delta - \nabla = \Delta\nabla - \nabla\Delta$

(d) $E = 1 + \mu\delta + \frac{\delta^2}{2}$ (e) $\mu^2 = 1 + \frac{\delta^2}{4}$ (f) $E = 1 + \delta\sqrt{E}$

(g) $\nabla = E^{-1}\Delta$ (h) $2\mu\delta = \Delta + \nabla$ (i) $\mu\delta = \frac{1}{2}(\Delta + \nabla)$

(j) $E^{\frac{1}{2}} = \mu + \frac{1}{2}\delta$ (k) $E^{-\frac{1}{2}} = \mu - \frac{1}{2}\delta$ (l) $\Delta + \nabla = \frac{\Delta}{\Delta} - \frac{\nabla}{\nabla}$

10.(a) Find Δy_n, $\Delta^2 y_n$ and $\Delta^3 y_n$ in the following cases:

(i) $y_n = n^2$

(ii) $y_n = n^3 + 3n^2$

(iii) $y_n = n^3 - n^2 + 17 - 1$

(iv) $y_n = n(n-1)(n-2)(n-3)(n-4)$

(b) Prove that $y_n = 3^n(A + Bn)$ satisfies the equation,

$$y_{n+2} - 6y_{n+1} + 9y_n = 0$$

(c) If $f(x) = \sin(\pi x)$, prove that $\Delta f = -2f$.

(d) If $f(x) = x^3$, compute the following:

(i) $\left(\frac{\Delta^2}{E^2}\right) f(x)$ (ii) $\frac{\Delta^2 f(x)}{E^2 f(x)}$

(e) Obtain the following results:

(i) $\Delta\left(\frac{f_n}{g_n}\right)$ (ii) $\Delta\left(\frac{1}{g_n}\right)$ (iii) $\Delta(\log f_n)$

(iv) $\Delta(f_n \cdot g_n) = f_n \Delta g_n + g_{n+1} \Delta f_n$ (v) $\Delta\sqrt{f_r} = \frac{\Delta f_r}{\sqrt{f_r} + \sqrt{f_{r+1}}}$

(f) Find $\Delta^2 x^4$.

(g) Find $x\Delta(x\Delta - 1)x^2$.

11.(a) Show that,

(i) $\Delta f_i = \nabla f_{i+1} = \delta f_{i+\frac{1}{2}}$ (ii) $\Delta^2 f_i = \nabla^2 f_{i+2} = \delta^2 f_{i+1}$

(iii) If $f(x) = x^4$, then $\Delta^2 f(x) = 12x^2 + 24x + 14$ and $\Delta^4 f(x) = 24$

(iv) If $f(x) = 2^x$, then $\Delta f(x) = f(x)$.

(b) Show that

(i) $\nabla^3 f_i = \Delta^3 f_{i-3}$ (ii) $\Delta^4 f_i = \nabla^4 f_{i+4}$ (iii) $\Delta^3 \nabla f_i = \Delta^4 f_{i-1}$

(iv) $\delta^2 f_i = \Delta^2 f_{i-1}$

Chapter 3

Interpolation

3.1 INTRODUCTION

Suppose we are given a table based on certain values of x and the corresponding values of a function f(x):

x	0	1	2	3	...	100
f(x)	10	85	90	98	...	125

The values in the table can be obtained by an experiment or generated if we know the function f(x). The process of computing an approximation value of the function at some point within the range (0, 100), but not in the table of data, is called **interpolation**. If the value of x lies outside the range, the process of estimating the value is called **extrapolation**.

Error of extrapolation increases as the point of interest goes farther from the data points. If a higher order interpolation is used for extrapolation without theoretical ba errors may increase rapidly as the order of polynomial increases. Application of extrapolation may be seen in various sections of this book: for instance, see the Newton-Cotes open integration formulas, the Romberg's integration method and the predictor-corrector methods.

In most of this chapter, we limit the interpolating function to be a **polynomial**. Interpolation has many applications in approximation theory, numerical differentiation, numerical integration, numerical solutions of ordinary and partial differential equations, and for making computer drawn curves to pass through specified points.

We are now going to describe several methods, in each case some kind of advice is given as to the circumstances under which the method should be applied.

3.1.1 Choice of a Suitable Interpolation Formula

The following are considered in choosing a method for interpolating polynomials:

a) Whether the given points x_i are equally spaced.

b) Whether the interpolation is desired towards the beginning, centre or end of a difference table.

3.1.2 Checking the Interpolated Value

The next is the question of checking the interpolated value. A single interpolation is not easy to check. One possibility is to repeat the interpolation using a different formula, but this will be more than double the labour, since the first-interpolation is usually done by the easiest formula. When possible a functional relationship such as $e^{-x} = \frac{1}{e^{x}}$ is a better check. This still requires two-interpolations but since they involve different tables, the formula may be used for both.

3.1.3 Type of Interpolation Formulas for Equally-Spaced Data Points

The following three types of interpolation formulas are used for equally-spaced data points:

(a) **Newton's forward difference interpolation formula**. It uses differences near the beginning of the difference table.

(b) **Newton's backward difference interpolation formula**. It uses differences near the end of the difference table.

(c) **Central difference interpolation formulas**. These formulas employ differences in the centre of the difference table. The following central difference formulas are commonly used:

i) Stirling's formula
ii) Bessel's formula
iii) Everett's formula
iv) Gauss forward and backward formulas

3.1.4 Type of Interpolation Formulas for Unequally-Spaced Data Points

The following formulas may be used for unequally-spaced data points:

i) Newton's divided difference interpolation formula;
ii) Lagrange's formula;
iii) Aitken's formula; and
iv) Hermite's formula

We shall describe only Lagrange and Aitken formulas, because they are suitable for both, equally and unequally-spaced data points. The above formulas can also be employed for extrapolation; however, the error may increase rapidly the farther we extrapolate from the given values. With the widespread use of computers tabular interpolation has lost much of its importance. The methods under the present category have been widely used.

3.2 NEWTON'S FORWARD DIFFERENCE INTERPOLATION FORMULA

The most basic formula for interpolation with equidistant points is **Newton's forward difference interpolation** (sometimes also called the **Gregory-Newton**) **formula**.

Given a set of n pairs of values:

$$(x_0, f_0), (x_1, f_1), (x_2, f_2), \ldots, (x_n, f_n).$$

We shall derive this formula with the help of two difference operators, E and Δ.

The function to be estimated is written as:

$$f_p = E^p f_0 = (1+\Delta)^p f_0 \qquad \ldots (3.1)$$

Expanding $(1+\Delta)^p$, we have

$$f_p = \left\{1 + p\Delta + \frac{1}{2!}p(p-1)\Delta^2 + \frac{1}{3!}p(p-1)(p-2)\Delta^3 + \ldots + \frac{1}{n!}p(p-1)(p-2)\cdots(p-n+1)\Delta^n\right\} f_0$$

$$= f_0 + p\Delta f_0 + \frac{1}{2!}p(p-1)\Delta^2 f_0 + \frac{1}{3!}p(p-1)(p-2)\Delta^3 f_0 + \ldots + \frac{1}{n!}p(p-1)(p-2)\cdots(p-n+1)\Delta^n f_0 \qquad \ldots (3.2)$$

where $p = \dfrac{(x_p - x_0)}{h}$, obtained from $x_p = x_0 + ph$.

The coefficient of $\Delta^n f_0$ will contain p^n which is an nth degree polynomial.

Remarks

i) This formula is used for interpolation near the beginning of a difference table, but in odd cases, it may also be applied in other parts of the table by suitably shifting the origin. Shifting the origin does not affect the result, but on the other hand it may result in a simpler formula, which is less prone to error.

ii) This formula is usually applicable for $0 < p < 1$. When working with differences, we can select any value of x in the table to be labeled as x_0. This is mostly done to keep p within the range.

Example 1 a) Compute the difference table for the following set of data-points:

x	.00	.25	.50	.75	1.00
f(x)	.0000	.2763	.5205	.7112	.8427

b) Use Newton's forward difference formula to pass a fourth degree polynomial through the above data.

c) Use the above polynomial to interpolate for f (0.125).

Solution a) The forward differences are computed as follows:

x	f(x)	Δ	Δ^2	Δ^3	Δ^4
0.00	.0000				
		2763			
0.25	.2763		–321		
		2442		–214	
0.50	.5205		–535		157
		1907		–57	
0.75	.7112		–593		
		1315			
1.00	.8427				

b) $h = x_1 - x_0 = .25 - .00 = .25$

$$x_p = x_0 + ph = 0.125$$

$$p = \frac{(x_p - x_0)}{h} = \frac{(.125 - .00)}{.25} = 0.5$$

Since the calculated value of p lies in the range (0, 1), it makes the forward difference formula applicable. The values to be used in formula (3.2) are underlined in the above table.

$$f_p = f_0 + p\Delta f_0 + \frac{p(p-1)}{2!}\Delta^2 f_0 + \frac{p(p-1)(p-2)}{3!}\Delta^3 f_0 + \frac{p(p-1)(p-2)(p-3)}{4!}\Delta^4 f_0$$

$$= .000 + p \times 0.2763 + \frac{(p^2 - p)}{2} \times -0.0321 + \frac{(p^3 - 3p^2 - p)}{6} \times -0.0321$$

$$+ \frac{(p^4 - 6p^3 + 11p^2 - 6p)}{6} \times 0.0157$$

$$= .2763\,p - .0125\,p^2 + .0008\,p^3 - .0006\,p^4$$

c) Inserting $p = 0.50$ in the above polynomial, we get

$$f_p = .2763 \times .50 - .0125 \times (.50)^2 + .0008 \times (.50)^3 - .0006 \times (.50)^4$$

$$= .13815 - .00313 + .0001 - .00004 = 0.1351$$

The students should be careful not to think of the answer 0.1351 as the correct answer. It is an estimate of the correct answer based on the assumption that f(x) is a fourth-degree polynomial.

Example 2 Use Newton's forward difference formula to interpolate the value for f(1.75) from the following data:

(0.5, 0.000), (1.0, 1.357), (1.5, 2.000), (2.0, 2.625), and (2.5, 4.000).

Solution The difference table is as follows:

x	f(x)	Δ	Δ^2	Δ^3	Δ^4
0.5	0.000				
		1375			
1.0	1.357		–750		
		625		750	
1.5	2.000		0		0
		625		750	
2.0	2.625		750		
		1375			
2.5	4.000				

$x_p = 1.75;\ x_0 = 0.5;\ x_1 = 1.0;$

$h = x_1 - x_0 = 0.5$

$$p = \frac{(x_p - x_0)}{h} = \frac{(1.75 - 0.5)}{0.5} = 2.5$$

As p (= 2.5) does not lie between 0 and 1, we cannot use the origin to be $x_0 = 0.5$. Let us shift the origin to 1.0. Then, $p = \frac{(1.75-1)}{0.5} = 1.5$. We cannot use $x_1 = 1$ as the origin because still $p > 1$. Let us shift the origin to $x_0 = 1.5$. $p = \frac{(1.75-1.5)}{0.5} = \frac{0.25}{0.5} = 0.5$. So, we can use $x_0 = 1.5$ as the origin because the calculated value of $p < 1$.

The entries used in this case are underlined in the difference table. The reduced form of Newton's formula is as follows:

$$f_p = f_0 + p\Delta f_0 + \frac{p(p-1)}{2}\Delta^2 f_0$$

Inserting the values in the above reduced formula, we get,

$$f_p = 2.000 + 0.5 \times 0.625 + \frac{0.5(0.5-1)}{2} \times 0.750$$

$$= 2.000 + 0.313 - 0.094 = 2.219$$

Example 3 Write a computer program to implement Newton's forward difference interpolation formula. Use the following data for testing:

x	2	4	6	8	10	12	14
f(x)	23	93	259	569	1071	1813	2843

Estimate f (2.58).

Solution For computer program, see Example 4. However, the computer output for this example is given below:

Compute Output:

X	F (X)	1ST	2ND	3RD	4TH
2.00	23.00				
		70.00			
4.00	93.00		96.00		
		166.00		48.00	
6.00	259.00		144.00		.00
		310.00		48.00	
8.00	569.00		192.00		.00
		502.00		48.00	
10.00	1071.00		240.00		.00
		742.00		48.00	
12.00	1813.00		288.00		
		1030.00			
14.00	2843.00				

ANSWER = 36.23

3.3 NEWTON'S BACKWARD DIFFERENCE INTERPOLATION FORMULA

We shall derive Newton's backward difference formula using the difference operators E and ∇.

We know that, $f_p = E^p f_0 = (1-\nabla)^{-p} f_0$ ... (3.3)

Expanding $(1-\nabla)^{-p}$, we obtain,

$$f_p = \left\{1 + p\nabla + \frac{p(p+1)}{2!}\nabla^2 + \frac{p(p+1)(p+2)}{3!}\nabla^3 + \cdots + \frac{p(p+1)\cdots(p+n-1)}{n!}\nabla^n\right\} f_0$$

$$= f_0 + p\nabla f_0 + \frac{p(p+1)}{2!}\nabla^2 f_0 + \frac{p(p+1)(p+2)}{3!}\nabla^3 f_0 + \cdots$$

$$+ \frac{p(p+1)\cdots(p+n-1)}{n!}\nabla^n f_0 \qquad \ldots (3.4$$

This is called **Newton's backward difference interpolation** (also called the **Gregory-Newton) formula.**

Remarks

i) This formula is used toward the end of the difference table but can also be applied in other parts of the table by suitably shifting the origin. This situation occurs whenever a table is being extended, for example, when the solution to a differential equation is being obtained by a step-by-step method.

ii) The formula is valid for $0 < p < 1$.

Example 4 (a) Using Newton's backward difference formula, compute f (11.8) from the following data:

x	2	4	6	8	10	12	14
f(x)	23	93	259	569	1071	1813	2843

(b) Write a computer program to implement Newton's backward difference interpolation formula.

Solution (a) The backward differences are computed in the following table:

x	f(x)	∇	∇^2	∇^3	∇^4
2	23				
		70			
4	93		96		
		166		48	
6	256		144		0
		310		48	
8	569		192		0
		502		48	
10	1071		240		0
		442		48	
12	1813		288		
		1030			
14	4843				

$x_p = 11.8$

Taking $x_0 = 14$, $p = \frac{(11.8-14)}{2} = -1.1$. As the calculated value is outside its acceptable range, we cannot accept the origin to be at $x_0 = 14$. The suitable origin may be $x_0 = 10$, which gives $p = \frac{11.8-10}{2} = \frac{1.8}{2} = 0.9$. The entries used for the backward difference formula are underlined in the above difference table. Substituting these values in formula (3.4), we get:

$$f_p = 1071 + 0.9 \times 502 + \frac{0.9(0.9+1)}{2} \times 192 + \frac{0.9(0.9+1)(0.9+2)}{6} \times 48$$

$$= 1071 + 451.8 + 164.16 + 39.67 = 1727$$

(b) This program can be used for Newton's forward and backward interpolation formulas. It is done via a main menu. Menu Choice 1 is for the forward difference formula, while Menu Choice 2 is for the backward difference formula.

Computer Program:

```
# include<iostream.h>
# include<conio.h>
# include<process.h>

float interval, x0, p, array [20][20] = {0.0};
int no, col, x,y;
void difftable( )
  {
    cout<<"\tDIFFERENETTABLE";
    cout<<"\n\n\tENTER THE FIRST VALUE : "; cin>>array[0][0];
    cout<<"\n\tENTER THE INTERVAL : "; cin>>interval;
    cout<<"\n\tENTER TOTAL NO. OF X : "; cin>>no;

    for(int i=1; i<no; i++)
        {
          array[i][0]=array[i-1][0]+interval;
        }

    cout<<"\n\tENTER FUNCTIONAL VALUES : \n";
    for(i=0;i<no;i++)
        {
          cout<<"\tX("<<i<<") = ";cin>>array[i][1];
        }
```

```
cout<<"\n\tHOW MANY COLUMNS ARE REQUIRED : "; cin>>col;
for(i=2; i<=(col+2); i++)
    {
      for(int j=0; j<=(no-i); j++)
        {
          array[j][i]=array[j+1][i-1]-array[j][i-1];
        }
    }

clrscr( );
cout<<"\t\tDIFFERENCE TABLE\n";
cout<<" X          F(X)  ";
for(j=1;i<=col;i++)
    {
      cout<<"     col     "<<i;
    }

cout<<"\n";

for(i=0;i<no;i++)
    {
      cout<<"     "<<array[i][0]<<"\n\n";
    }

x=8; y=3;
for(i=1;i<=(col+1);i++)
   {
     gotoxy(x,y);
     for(int j=0; j<=(no-i); j++)
       {
         cout<<array[j][i];
         y+=2;
         gotoxy(x,y);
       }
     x+=9; y=i+3;
   }
}

void findx( )
 {
  float xp;
  cout<<"\n\t XP FOR WHICH VALUE OF F(X) IS REQUIRED :     "; cin>>xp;
  int i=0;
```

```
    while(((xp-array[i][0])/interval>1)&&(I<no))
      {
        i++;
      }
    x0=i;
    p=(xp-array[x0][0])/interval;
  }

void nford( )
  {
    findx( );

    cout<<"\n\ntanswer = ";
    cout<<(array[x0][1]+(p*array[x0][2]+(p*(p-1) /2 * array[x0][3])
    +p*(p-1)*(p-2) /6 * array[x0][4])+(p*(p-1)*(p-2)*(p-3) /24 * array[x0][5]);
  }

void nback( )
  {
    findx( );
    cout<<"\n\n\tanswer = ";
    cout<<(array[x0][1]+(p*array[x0-1][2]+(p*(p+1) /2 * array[x0-2][3])
    +p*(p+1)*(p+2) /6 * array[x0-3][4])+(p*(p+1)*(p+2)*(p+3) /24 * array[x0-4][5]);
  }

void main (void)
  {
    clrscr ( ); difftable ( ); getch ( );

    int choice;
    while (1)
      {
        clrscr ( );
        cout<<"\n\n\t\tMAIN MENU";
        cout<<"\n\n\tFORWARD DIFFERENCE INTERPOLATION FORMULA --- 1";
        cout<<"\n\n\tBACKWARD DIFFERENCE INTERPOLATION FORMULA --- 2";
        cout<<"\n\n\tTO EXIT ----------------------------------------------------------";
        cout<<"\n\n\n\tENTER YOUR CHOICE : ";
        cin>>choice;
        switch(choice)
          {
            case 1:clrscr ( );nford( );getch( );break;
            case 2:clrscr ( );nback( );getch( );break;
            case 3:exit(0)
          }
      }
  }
```

Computer Output

DIFFERENCE TABLE

ENTER THE FIRST VALUE : 2

ENTER THE INTERVAL : 2

ENTER TOTAL NO. OF X : 7

ENTER FUNCTIONAL VALUES:

X(0) = 23

X(1) = 93

X(2) = 259

X(3) = 569

X(4) = 1071

X(5) = 1813

X(6) = 2843

HOW MANY COLUMNS ARE REQUIRED : 4

DIFFERENCE TABLE

X	F(X)	col 1	col 2	col 3	col 4
2	23				
		70			
4	93		96		
		166		48	
6	259		144		0
		310		48	
8	569		192		0
		502		48	
10	1071		240		0
		742		48	
12	1813		288		
		1030			
14	2843				

MAIN MENU

FORWARD DIFFERENCE INTERPOLATION FORMULA --- 1

BACKWARD DIFFERENCE INTERPOLATION FORMULA --- 2

TO EXIT ---------- 3

ENTER YOUR CHOIC : 1

XP, FOR WHICH VALUE OF F(X) IS REQUIRED : 2.58

A N S W E R = 36.233509

ENTER YOUR CHOICE : 2

XP, FOR WHICH VALUE OF F(X) IS REQUIRED : 11.8

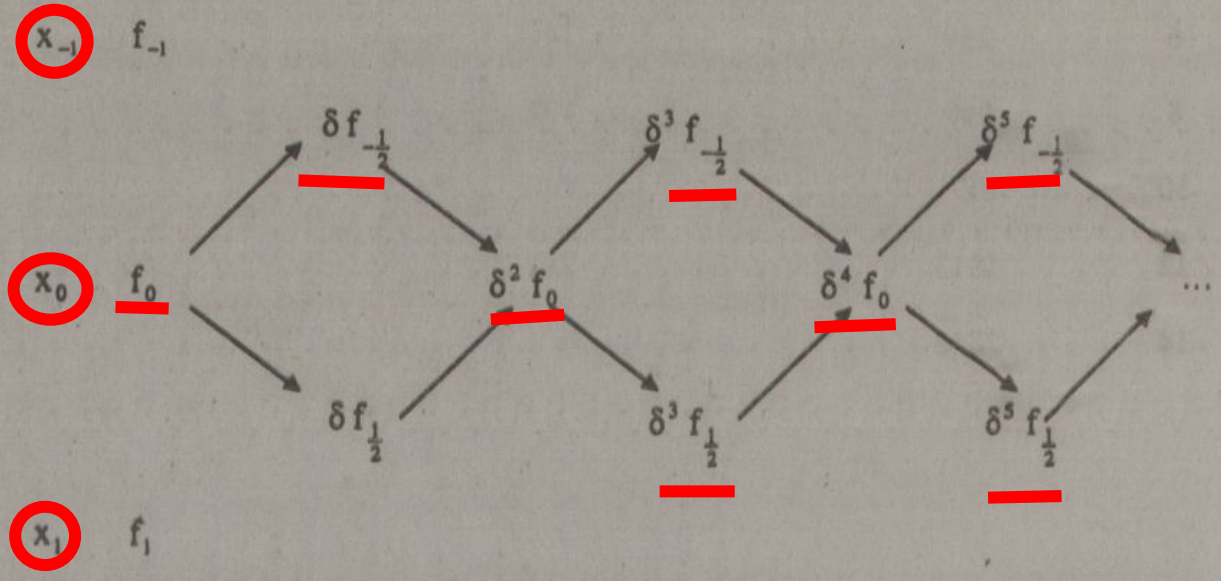
A N S W E R = 1726.63208

3.4 INTERPOLATION WITH CENTRAL DIFFERENCE FORMULAS

The two formulas by Newton are used occasionally and almost exclusively at the beginning or at the end of a table. More important are formulas which make use of central differences, a whole series of such formulas with slightly different properties can be constructed. In this section, we shall mention without proofs some well-known central difference formulas. The structure of all these formulas can easily be demonstrated by sketching a difference scheme where different quantities are represented by points. If the column to the left stands for the function values, then we have the first differences and so on.

3.4.1 Stirling's Interpolation Formula

Stirling's formula follows the path through the difference table given below:



It is expressed as follows:

$$f_p = f_0 + \frac{1}{2}p\left(\delta f_{-\frac{1}{2}} + \delta f_{\frac{1}{2}}\right) + \frac{1}{2!}p^2\delta^2 f_0 + \frac{p(p^2-1)}{2\times 3!}\left(\delta^3 f_{-\frac{1}{2}} + \delta^3 f_{\frac{1}{2}}\right)$$

$$+ \frac{p^2(p^2-1)}{4!}\delta^4 f_0 + \frac{p(p^2-1)(p^2-4)}{2\times 5!}\left(\delta^5 f_{-\frac{1}{2}} + \delta^5 f_{\frac{1}{2}}\right)$$

$$+ \frac{p^2(p^2-1)(p^2-4)}{6!}\delta^6 f_0 + \cdots \qquad \ldots \quad (3.5(a))$$

It can also be written in another form as:

$$f_p = f_0 + p\mu\delta f_0 + \frac{1}{2!}p^2\delta^2 f_0 + \frac{1}{3!}p(p^2-1)\mu\delta^3 f_0 + \frac{1}{4!}p^2(p^2-1)\delta^4 f_0$$

$$+ \frac{1}{5!}p(p^2-1)(p^2-4)\mu\delta^5 f_0 + \frac{1}{6!}p^2(p^2-1)(p^2-4)\delta^6 f_0 + \cdots$$

$$\ldots \quad (3.5(b))$$

Example 5 Use Stirling's Interpolation formula to find f(1.62) from the following table:

x	1.2	1.4	1.6	1.8	2.0
f(x)	5.6464	6.44218	7.17356	7.83327	8.41471

Solution: The difference table for is as follows:

x	f (x)	δ	δ²	δ³	δ⁴
1.2	5.6464				
		79576			
1.4	6.44218		– 6438		
		73138		– 729	
x_0 = 1.6	7.17356		– 7167		69
		65971		– 660	
1.8	7.83327		– 7827		
		58144			
2.0	8.41471				

Taking $x_0 = 1.6$, $h = x_1 - x_0 = 1.8 - 1.6 = .2$

$$P = \frac{(1.62 - 1.6)}{0.2} = \frac{0.02}{0.2} = 0.1$$

Inserting the values in Stirling's formula 3.5(a), we get,

$$f_p = 7.17356 + \frac{1}{2} \times 0.1\,(.73138 + .65971) + \frac{1}{2} \times 0.1 \times 0.1 \times -.7167$$

$$+ \frac{0.1(0.1 \times 0.1 - 1)}{12} \times (-.00729 - .00660)$$

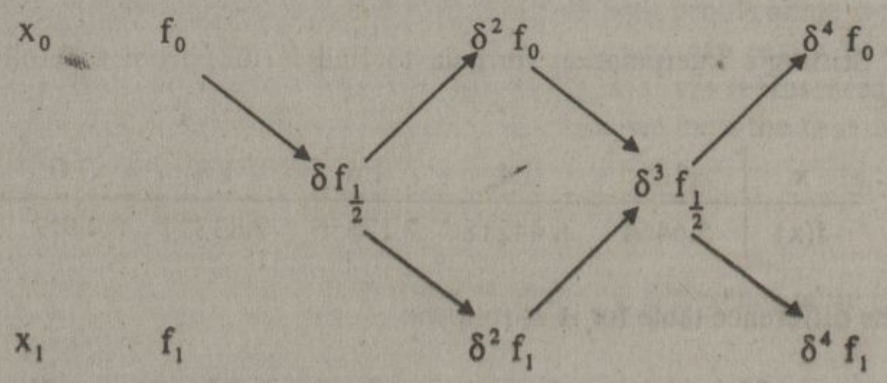
$$+ \frac{0.1 \times 0.1(0.1 \times 0.1 - 1)}{24} \times .00069$$

$$= 7.17356 + 0.06955 - 0.00036 + 0.00011 - 0.00000$$

$$= 7.24286$$

3.4.2 Bessel's Interpolation Formula

Bessel's formula follows the path through the difference table:



Bessel's formula is expressed as follows:

$$\boxed{f_p = f_0 + p\delta f_{\frac{1}{2}} + \frac{p(p-1)}{2.2!}\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{p(p-1)\left(p-\frac{1}{2}\right)}{3!}\delta^3 f_{\frac{1}{2}} + \frac{(p+1)p(p-1)(p-2)}{2.4!}\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots} \quad \ldots \quad (3.6(a))$$

It can also be written in another form as:

$$f_p = f_0 + p\,\delta f_{\frac{1}{2}} + \frac{1}{2!}p(p-1)\mu\delta^2 f_{\frac{1}{2}} + \frac{1}{3!}p(p-1)\left(p-\frac{1}{2}\right)\mu\,\delta^3 f_{\frac{1}{2}}$$

$$+ \frac{1}{4!}(p+1)p(p-1)(p-2)\mu\,\delta^4 f_{\frac{1}{2}} + \cdots \quad \ldots \quad (3.6(b))$$

3.4.3 Everett's Interpolation Formula

Everett's formula follows the path through the difference table:

x_0	f_0	———	$\delta^2 f_0$	———	$\delta^4 f_0$	...
x_1	f_1	———	$\delta^2 f_1$	———	$\delta^4 f_1$	...

Everett's formula is expressed as follows:

$$f_p = q f_0 + \frac{q(q^2-1)}{3!}\delta^2 f_0 + \frac{q(q^2-1)(q^2-4)}{5!}\delta^4 f_0 + \cdots$$

$$+ p f_1 + \frac{p(p^2-1)}{3!}\delta^2 f_1 + \frac{p(p^2-1)(p^2-4)}{5!}\delta^4 f_1 + \cdots \quad \ldots \quad (3.7)$$

where $q = 1 - p$.

3.4.4 Gaussian Interpolation Formula

There are the following two such formulas:

a) Gauss Forward Interpolation Formula

b) Gauss Backward Interpolation Formula

Let us discuss them one by one.

a) Gauss Forward Interpolation Formula

This formula follows the zigzag path as indicated in the difference table:

$$\begin{array}{ccccccccc}
x_0 & f_0 & & & \delta^2 f_0 & & \delta^4 f_0 & & \\
 & & \searrow & & & \searrow & & \searrow & \\
 & & & \delta f_{\frac{1}{2}} & & \delta^3 f_{\frac{1}{2}} & & & \delta^5 f_{\frac{1}{2}} \\
 & & \nearrow & & \nearrow & & \nearrow & & \\
x_1 & f_1 & & & & & & &
\end{array}$$

Gauss forward formula is expressed as follows:

$$f_p = f_0 + p\delta f_{\frac{1}{2}} + \frac{p(p-1)}{2!}\delta^2 f_0 + \frac{p(p+1)(p-1)}{3!}\delta^3 f_{\frac{1}{2}}$$

$$+ \frac{(p+1)p(p-1)(p-2)}{4!}\delta^4 f_0 + \cdots \quad \ldots \quad (3.8)$$

b) Gauss Backward Interpolation Formula

This formula follows the zigzag path as indicated in the following difference table:

$$\begin{array}{ccccccccc}
x_{-1} & f_{-1} & & & & & & & \\
 & & \delta f_{-\frac{1}{2}} & & \delta^3 f_{-\frac{1}{2}} & & \delta^5 f_{-\frac{1}{2}} \\
x_0 & f_0 & & \delta^2 f_0 & & \delta^4 f_0 &
\end{array}$$

Gauss backward formula is expressed as follows:

$$f_p = f_0 + p\delta f_{-\frac{1}{2}} + \frac{p(p+1)}{2!}\delta^2 f_0 + \frac{p(p+1)(p-1)}{3!}\delta^3 f_{-\frac{1}{2}}$$

$$+ \frac{(p+1)p(p-1)(p-2)}{4!}\delta^4 f_0 + \cdots \qquad \ldots \quad (3.9)$$

Note that if we take the mean of Gauss forward and backward formulas, we get Stirling's interpolation formulas.

As mentioned earlier that for most purposes formulas using central differences are to be preferred. However, some remarks about their use are in order. The two Gaussian interpolation formulas are of interest almost exclusively from theoretical standpoint. Stirling's formula is suitable for values of small values of p, for example, $-\frac{1}{4} \le p \le +\frac{1}{4}$, and Bessel's formula which is probably the most used of all interpolation formulas, is suitable for values of p not too far from $\frac{1}{2}$, for example, $-\frac{1}{4} \le p \le \frac{1}{4}$. Everett's formula which is simple and fast is perhaps the one which is most generally useful and further because even differences are used together with the function values.

Example 6 Given the following table of values:

x	2.2	2.6	3.0	3.4	3.8	4.2	4.6
f(x)	.374607	.438371	.500000	.559193	.615661	.669131	.719340

a) Construct the difference table including differences up to 4th order only.

b) Interpolate f(3.64) using the following formulas centred at x = 3.4:

(i) Stirling (ii) Bessel (iii) Everett and (iv) Gaussian forward and backward.

Solution (a) Difference Table

x	f (x)	δ	δ^2	δ^3	δ^4
2.2	.374607				
		63764			
2.6	.438371		– 2135		
		61629		– 301	
3.0	.500000		– 2436		12
		59193		– 289	
$x_0 = 3.4$	.559193		– 2725		16
		56468		– 273	
3.8	.615661		– 2998		10
		53470		– 263	
4.2	.669131		– 3261		
		50209			
4.6	.719340				

b) (i) Stirling's Formula

$$\boxed{x_p = 3.64, \quad x_0 = 3.4, \quad h = 0.4}$$

$$P = \frac{(x_p - x_0)}{h} = \frac{(3.64 - 3.4)}{0.4} = 0.6$$

Substituting values in formula (3.5(a)), we get,

$$f_p = .559193 + \frac{6}{2}(.059193 + .056468) + \frac{.6 \times .6}{2} \times -.002725$$

$$+ \frac{0.6(0.6 \times 0.6 - 1)}{12}(-.000289 - .000273)$$

$$+ \frac{0.6 \times 0.6(0.6 \times 0.6 - 1)}{24} \times .000016$$

$$= .559193 + .034698 - .000491 + .000018 + .000000$$

$$= 0.593418$$

(ii) Bessel's Formula

Substituting values in formula (3.6(a)), we get,

$$f_p = .559193 + 0.6 \times .056468 + \frac{.6(.6-1)}{4}(-.002725 - .002998)$$

$$+ \frac{0.6(0.6-1)\left(0.6-\frac{1}{2}\right)}{6} \times -0.000273$$

$$+ \frac{(0.6+1)0.6(0.6-1)(0.6-2)}{48} \times (.000016 + .000010)$$

$$= .559193 + .033881 + .000343 + .000001 + .000000$$

$$= 0.593418$$

(iii) Everett's Formula

$$q = 1 - .6 = .4$$

Substituting values in formula (3.7), we get,

$$f_p = 0.4 \times .559193 \frac{.4(.4 \times .4 - 1)}{6} \times -0.002725$$

$$+ \frac{.4(.4 \times .4 - 1)(.4 \times .4 - 4)}{120} \times 0.000016$$

$$+ .6 \times 0.615661 + \frac{0.6(0.6 \times .6 - 1)}{6} \times -0.002998$$

$$+ \frac{0.6(0.6 \times .6 - 1)(0.6 \times .6 - 4)}{120} \times -0.000010$$

$$= .223677 + .000153 + .000000 + .369397 + .000192 + .000000$$

$$= 0.593419$$

(iv) (a) Gauss Forward Formula

Substituting values in formula (3.8), we get,

$$f_p = .559193 + 0.6 \times .056468 + \frac{.6(0.6-1)}{2} \times -.002725$$

$$+ \frac{0.6(0.6+1)(0.6-1)}{6} \times -.000273$$

$$+ \frac{(.6+1) \times 0.6(.6-1)(0.6-2)}{24} \times .000016$$

$$= .559193 + .033881 + .000327 + .000017 + .00000$$

$$= 0.593418$$

(b) Gauss Backward Formula

Substituting values in formula (3.9), we get,

$$f_p = .559193 + 0.6 \times .059193 + \frac{.6(0.6+1)}{2} \times -.002725$$

$$+ \frac{0.6(0.6+1)(0.6-1)}{6} \times -.000289$$

$$+ \frac{0.6(.6+1)(.6-1)(0.6-2)}{24} \times .000016$$

$$= .559193 + .035516 + .001308 + .000018 + .000000$$

$$= 0.593419$$

3.5 LAGRANGE'S FORMULA

It was mentioned in the previous sections that difference table could be used for interpolation, but this was restricted to the case of function values at equidistant intervals.

To introduce the basic idea behind the Lagrange's formula, consider the following:

Given the data points $x_0, x_1, \ldots, x_n$ (may or may not be equidistant), the problem is to find an nth degree polynomial f(x) using Lagrange's formula.

Lagrange's formula can be derived by writing:

$$\begin{aligned} f(x) = \; & A_0(x-x_1)(x-x_2)\ldots(x-x_n) \\ & + A_1(x-x_0)(x-x_2)\ldots(x-x_n) \\ & + A_2(x-x_0)(x-x_1)\ldots(x-x_n) \\ & \vdots \\ & + A_i(x-x_0)\ldots(x-x_{i-1})(x-x_{i-1})\ldots(x-x_n) \\ & \vdots \\ & + A_n(x-x_0)(x-x_1)\ldots(x-x_{n-1}) \end{aligned} \qquad \ldots (3.10)$$

where $A_0, A_1, \ldots, A_n$ are unknown constants.

If we substitute $x = x_0$ in (3.10), we get,

$$f(x_0) = A_0(x_0 - x_1)(x_0 - x_2)\ldots(x_0 - x_n)$$

$$A_0 = \frac{f(x_0)}{(x_0 - x_1)(x_0 - x_2)\cdots(x_0 - x_n)}$$

Similarly, substituting $x = x_1, x = x_2, \ldots, x = x_n$ respectively in (3.10), we get,

$$A_1 = \frac{f(x_1)}{(x_1 - x_0)(x_1 - x_2)\cdots(x_1 - x_n)}$$

$$A_2 = \frac{f(x_2)}{(x_2 - x_0)(x_2 - x_1)\cdots(x_2 - x_n)}$$

$$A_n = \frac{f(x_n)}{(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-1})}$$

Substituting the values of $A_0, A_1, \ldots, A_n$ in (3.10), we get the following formula due to Lagrange:

$$\begin{aligned} f(x) = & \frac{(x - x_1)(x - x_2)\cdots(x - x_n)}{(x_0 - x_1)(x_0 - x_2)\cdots(x_0 - x_n)} f(x_0) \\ & + \frac{(x - x_0)(x - x_2)\cdots(x - x_n)}{(x_1 - x_0)(x_1 - x_2)\cdots(x_1 - x_n)} f(x_1) \\ & + \frac{(x - x_0)(x - x_1)\cdots(x - x_n)}{(x_2 - x_0)(x_2 - x_1)\cdots(x_2 - x_n)} f(x_2) \\ & + \ldots \\ & + \frac{(x - x_0)(x - x_1)\cdots(x - x_{n-1})}{(x_n - x_0)(x_n - x_1)\cdots(x_n - x_{n-1})} f(x_n) \qquad \ldots (3.11) \end{aligned}$$

It is obvious that (3.11) is a polynomial of degree n and can be written as:

$$f(x) = L_0(x) f(x_0) + L_1(x) f(x_1) + L_2(x) f(x_2) + \ldots + L_n(x) f(x_n)$$

It can be concisely represented as:

$$f(x) = \sum_{i=0}^{n} L_i(x) f(x_i) \qquad \ldots (3.11(a))$$

where $L_i(x) = \prod_{\substack{i=0 \\ j=0 \\ j \neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right)$

Another form of this formula is:

$$f(x) = \sum_{i=0}^{n} \left(\prod_{\substack{j=0 \\ j \neq i}}^{n} \left(\frac{x - x_j}{x_i - x_j} \right) \right) f(x_i) \qquad \ldots (3.11(b))$$

The basic formula, apparently due to Waring, is associated with the name of Lagrange. This is one of the more practical and simpler method to be used on computer; but difficult for hand calculations if data points are more. Evaluation of error is also not easy. It is meant for equispaced or unequispaced data.

Example 7 (a) Fit a polynomial using Lagrange's formula to the following data:

(1, 4), (3, 7), (4, 8) and (6, 11).

(b) Use the polynomial to estimate a value for x = 5.

(c) Write a computer program to implement Lagrange's formula.

Solution (a) The data-points are:

x	1	3	4	6
f(x)	4	7	8	11

Inserting values in Lagrange's formula, we get,

$$f(x) = \frac{(x-3)(x-4)(x-6)}{(1-3)(1-4)(1-6)} \times 4 + \frac{(x-1)(x-4)(x-6)}{(3-1)(3-4)(3-6)} \times 4$$

$$+ \frac{(x-1)(x-3)(x-6)}{(4-1)(4-3)(4-6)} \times 8 + \frac{(x-1)(x-3)(x-4)}{(6-1)(6-3)(6-4)} \times 11$$

$$= \frac{-2}{15}\left(x^3 - 13x^2 + 54x - 72\right) + \frac{7}{6}\left(x^3 - 11x^2 + 34x - 24\right)$$

$$+ \frac{-8}{6}\left(x^3 - 10x^2 + 27x - 18\right) + \frac{11}{30}\left(x^3 - 8x^2 + 19x - 12\right)$$

$$= \frac{1}{30}\left(2x^3 - 21x^2 + 103x + 36\right)$$

(b) The interpolated value at $x = 5$ is as follows:

$$f(5) = \frac{1}{30}\left(2\times5^3 - 21\times5^2 + 103\times5 + 36\right)$$

$$= \frac{1}{30}\times 276 = 9$$

(c) **Program No. 3** **LAGRANGE'S FORMULA**

```
# include<iostream.h>
# include<iostream.h>

void main (void)
  {
    float table[10][2], xp,temp,ans=0.0;
    int no, y=0,a=7,i,j;

    cout<<"How Many Values Of X   :   ";
    cin>>no;
    cout<<"\nEnter The Values Of X and f(x)\n";
    cout<<"\n\t    x              |              f(x)";
    cout<<"\n\t-----------------------------------";

    for(i=0;i<no;i++)                                    // Input of X & Fx
      {
        gotoxy(11,a);
        cin>>table[i][y];
        gotoxy(21,a);
        cin>>table[i][y+1];
        a++;
      }

    cout<<"\nEnter The Value Of X   :   ";
    cin>>xp;

    for(j=0;j<no;j++)                                    // calculation of formula
      {
        temp=1;
        for(i=0;i<no;i++)
          if(i!=j)
```

```
            temp*=((xp-table[i][0] / (table[j][0]-[i][0]));

        ans+=temp*table[j][1];
    }
  cout<<"\nA N S W E R =          : "<<ans;        //output
}
```

Computer Output

How Many Values of X : 4

Enter the Values of x and f(x)

x	f(x)
1	4
3	7
4	8
6	11

Enter The Value of X : 5

A N S W E R : 9.2

3.6 ITERATIVE INTERPOLATION METHOD

Like Lagrange's method, this formula is also more suitable for computer application and its use is also not limited to only uniformly spaced data. The iterative interpolation process is based on the repeated application of simple (linear) interpolation method. This method is due to **Aitken**.

Consider the following data points (equally or unequally spaced):

x	x_0	x_1	x_2	x_3	...	x_n
f(x)	f_0	f_1	f_2	f_3	...	f_n

In order to estimate the value of the function f corresponding to any value of x, we proceed as follows:

Let $f_0 = f(x_0)$

$f_1 = f(x_1)$

$\vdots$

$$f_k = f(x_k)$$

$$\vdots$$

$$f_n = f(x_n)$$

also let $f(x \mid x_0, x_1, \ldots, x_n)$ denote the unique polynomial of degree n coinciding with $f(x)$ at $x_0, x_1, \ldots, x_n$.

Hence,
$$f(x \mid x_0) = f(x_0)$$
$$f(x \mid x_1) = f(x_1)$$
$$\vdots$$
$$f(x \mid x_n) = f(x_n)$$

Also,
$$f(x \mid x_0, x_1) = \frac{1}{(x_1 - x_0)}\begin{vmatrix} x - x_0 & f(x \mid x_0) \\ x - x_1 & f(x \mid x_1) \end{vmatrix}$$

$$= \frac{1}{x_1 - x_0}\begin{vmatrix} x - x_0 & f_0 \\ x - x_1 & f_1 \end{vmatrix}$$

$$= \frac{1}{(x_1 - x_0)}\{(x - x_0)f_1 - (x - x_1)f_0\}$$

$$f(x \mid x_0, x_2) = \frac{1}{x_2 - x_0}\begin{vmatrix} x - x_0 & f_0 \\ x - x_2 & f_2 \end{vmatrix}$$

$$= \frac{1}{(x_2 - x_0)}\{(x - x_0)f_2 - (x - x_2)f_0\}, \text{ etc.}$$

Similarly, $f(x \mid x_0, x_1, x_2) = \dfrac{1}{(x_2 - x_1)}\begin{vmatrix} x - x_1 & f(x \mid x_0, x_1) \\ x - x_2 & f(x \mid x_0, x_2) \end{vmatrix}$

$$f(x \mid x_0, x_1, x_3) = \frac{1}{(x_3 - x_1)}\begin{vmatrix} x - x_1 & f(x \mid x_0, x_1) \\ x - x_3 & f(x \mid x_0, x_3) \end{vmatrix}$$

and $\quad f(x \mid x_0, x_1, x_4) = \dfrac{1}{(x_4 - x_1)}\begin{vmatrix} x - x_1 & f(x \mid x_0, x_1) \\ x - x_4 & f(x \mid x_0, x_4) \end{vmatrix}$

denote polynomials of degree ≤ 2 that pass through the four points $(x_0, f_0), (x_1, f_1), (x_2, f_2)$; $(x_0, f_0), (x_1, f_1), (x_3, f_3)$; and $(x_0, f_0), (x_1, f_1), (x_4, f_4)$, respectively, whereas

$$f(x \mid x_0, x_1, x_2, x_3) = \frac{1}{x_3 - x_2}\begin{vmatrix} x - x_2 & f(x \mid x_0, x_1, x_2) \\ x - x_3 & f(x \mid x_0, x_1, x_3) \end{vmatrix}$$

denotes polynomial of degree ≤ 3 and so on.

Continuing the above process, we can develop the interpolating polynomials to any degree we want:

$$f(x \mid x_0, x_1, x_2, x_3, x_4) = \frac{1}{(x_4 - x_3)}\begin{vmatrix} x - x_3 & f(x \mid x_0, x_1, x_2, x_3) \\ x - x_4 & f(x \mid x_0, x_1, x_2, x_4) \end{vmatrix}$$

$$f(x \mid x_0, x_1, x_2, x_3, x_5) = \frac{1}{(x_5 - x_3)}\begin{vmatrix} x - x_3 & f(x \mid x_0, x_1, x_2, x_3) \\ x - x_5 & f(x \mid x_0, x_1, x_2, x_5) \end{vmatrix}$$

The following table illustrates the arrangement of the work needed to construct $f(x \mid x_0, x_1, \ldots, x_n)$:

x_0	$x - x_0$	$f(x \mid x_0)$				
x_1	$x - x_1$	$f(x \mid x_1)$	$f(x \mid x_0, x_1)$			
x_2	$x - x_2$	$f(x \mid x_2)$	$f(x \mid x_0, x_2)$	$f(x \mid x_0, x_1, x_2)$		
x_3	$x - x_3$	$f(x \mid x_3)$	$f(x \mid x_0, x_3)$	$f(x \mid x_0, x_1, x_3)$	$f(x \mid x_0, x_1, x_2, x_3)$	
x_4	$x - x_4$	$f(x \mid x_4)$	$f(x \mid x_0, x_4)$	$f(x \mid x_0, x_1, x_4)$	$f(x \mid x_0, x_1, x_2, x_4)$	$\cdots$
	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$

The tabular values are generated row-wise (or column-wise). Since the current value are generated from the previous values that is why this method is often called the **iterative interpolation method** and also named as **Neville's formula**. The rightmost value in the table is the required value of interpolation.

Example 8 (a) Using Aitken's iterative scheme, find the value of log 4.5 from the following values:

x	4.0	4.2	4.4	4.6
f(x)	0.60206	0.62325	0.64345	0.66276

(b) Write a computer program to implement Aitken's method.

Solution (a) $x = 4.5$

Aitken's table is as follows:

$x_0 = 4.0$	$x - x_0 = .5$	0.60206			
$x_1 = 4.2$	$x - x_1 = .3$	0.62325	0.65504		
$x_2 = 4.4$	$x - x_2 = .1$	0.64345	0.65380	0.65318	
$x_3 = 4.6$	$x - x_3 = -.1$	0.66276	0.65264	0.65324	0.65321

$$f(x \mid x_0, x_1) = \frac{1}{(x_1 - x_0)} \begin{vmatrix} x - x_0 & f(x \mid x_0) \\ x - x_1 & f(x \mid x_1) \end{vmatrix}$$

$$= \frac{1}{(4.2 - 4.0)} \begin{vmatrix} .5 & .60206 \\ .3 & .62325 \end{vmatrix}$$

$$= \frac{(.5 \times .62325 - .3 \times .60206)}{0.2}$$

$$= \frac{(.311625 - .180618)}{0.2} = 0.65504$$

$$f(x \mid x_0, x_2) = \frac{1}{(x_2 - x_0)} \begin{vmatrix} x - x_0 & f(x \mid x_0) \\ x - x_2 & f(x \mid x_2) \end{vmatrix}$$

$$= \frac{1}{(4.4 - 4.0)} \begin{vmatrix} .5 & .60206 \\ .1 & .64345 \end{vmatrix}$$

$$= \frac{(.321725 - .060206)}{0.4} = 0.65380$$

$$f(x \mid x_0, x_3) = \frac{1}{(x_3 - x_0)} \begin{vmatrix} x - x_0 & f(x \mid x_0) \\ x - x_3 & f(x \mid x_3) \end{vmatrix}$$

$$= \frac{1}{(4.6 - 4.0)} \begin{vmatrix} .5 & .60206 \\ -.1 & .66276 \end{vmatrix}$$

$$= \frac{(.5 \times .66276 + .1 \times .60206)}{0.6}$$

$$= \frac{(.33138 - .060206)}{0.6} = 0.65264$$

$$f(x \mid x_0, x_1, x_2) = \frac{1}{(x_2 - x_1)} \begin{vmatrix} x - x_1 & f(x \mid x_0, x_1) \\ x - x_2 & f(x \mid x_0, x_2) \end{vmatrix}$$

$$= \frac{1}{(4.4 - 4.2)} \begin{vmatrix} .3 & .65504 \\ -.1 & .65380 \end{vmatrix}$$

$$= \frac{(.3 \times .65380 - .1 \times .65504)}{0.2}$$

$$= \frac{(.19614 - .065504)}{0.2} = 0.65318$$

$$f(x \mid x_0, x_1, x_3) = \frac{1}{(x_3 - x_1)} \begin{vmatrix} x - x_1 & f(x \mid x_0, x_1) \\ x - x_3 & f(x \mid x_0, x_3) \end{vmatrix}$$

$$= \frac{1}{(0.4)} \begin{vmatrix} .3 & .65504 \\ -.1 & .65264 \end{vmatrix}$$

$$= \frac{(.195792 - .065504)}{0.4} = 0.65324$$

$$f(x \mid x_0, x_1, x_2, x_3) = \frac{1}{(x_3 - x_2)} \begin{vmatrix} x - x_2 & f(x \mid x_0, x_1, x_2) \\ x - x_3 & f(x \mid x_0, x_1, x_3) \end{vmatrix}$$

$$= \frac{1}{(0.2)} \begin{vmatrix} .1 & .65318 \\ -.1 & .65324 \end{vmatrix}$$

$$= \frac{(.195324 - .065318)}{0.2} = 0.65321$$

The rightmost entry in each row in the table gives,

$f(4.5 \mid x_0, x_1) = 0.65504$

$f(4.5 \mid x_0, x_1, x_2) = 0.65318$

$f(4.5 \mid x_0, x_1, x_2, x_3) = 0.65321$

It is seen that log 4.5 = 0.65321, which is the anticipated answer.

(b) **Program No. 4** **Aitken's Method**

```
# include<conio.h>
# include<iostream.h>
# include<complex.h>
# include<stdio.h>

void main ( )
  {
    clrscr( );
    float x[10],f[10],r[10][10],diff[10],xp;
    int i,j,l,m,n,p,k,y,z;
    double term1,term2,term3;
    cout<<"n\t\t\t Aitken Method\n\n";
    cout<<"Enter the number of  X data   :     ";
    cin>>n;
    cout<<"Enter value of  xp   :     ";
    cin>>xp;
    for(i=0;i<n;i++)
      {
        cout<<"Enter value of  X["<<i<<"]\t";
        cin>>x[i];
        diff[i] = xp – x[i];
      }

  cout<<"\n\n\nGiven the values of function\n\n\n";
  for(i=0;i<=n-1;i++)
  {
      cout<<"Enter value of  F("<<i<<")\t";
      cin>>f[i];
  }
  for(i=0;i<=n-1;i++)
      r[i][0] = f[i];
  for(i=0;i<=n-1;i++)
  {
      for(j=0;j<n-1;j++)
      {
        term1 = diff[i]*r[j+1][i];
        term2 = diff[j+1]*r[i][i];
        term3 = x[j+1] – x[i];
```

```
        if(term3 !=0)
                r[j+1][i+1] = (term1 – term2)/term3;
    }
}
y = 13;
clscr( );
gotoxy(3,5);
cout<<"                    Implementation of Aitken's Method\n"
for(i=0;i<=n-1;i++)        //loop to print the value of x differences
  {
    y = i + 13;
    gotoxy(5,y);

    cout<<setiosflags(ios::fixed)<<setiosflags(ios::showpoint)<<setprecision(5)<<x[1];
    gotoxy(15,y);
    cout<<"\t"<<setiosflags(ios::fixed)<<setprecision(5)<<diff[i];
  }
  p=0;
  m=25;
  k=13;
  for(i=0;i<=n-1;i++)
    {
      k = i + 13;
      for(j=p;j<=n-1;j++)
        {
          gotoxy(m,k);
          cout<<setiosflags(ios::fixed)<<setprecision(5)<<setw(10)<<r[j][i];
          k = k + 1;
        }
      p = p + 1;
      m = m + 11;
    }
    z = 0;

    for(y=0;y<n-1;y++)
            z = z + 1;
    cout<<"\n\n\n\tAtXp="<<setw(15)<<setiosflags(ios::fixed)<<setprecision(3)<<xp"
    function value is\t"<<setiosflags(ios::fixed)<<setprecision(5)<<setw(15)<<r[y][z];
    getch( );

}
```

Computer Output

```
        Aitken Method
Enter the number of X data  : 4
Enter value of xp : 4.5
Enter value of X[0]     4.0
Enter value of X[1]     4.2
Enter value of X[2]     4.4
Enter value of X[3]     4.6

Given the values of function

Enter value of F[0]     .60206
Enter value of F[1]     .62325
Enter value of F[2]     .64345
Enter value of F[3]     .66276

            Implementation of Aitken's Method
 4.00000    0.50000    0.60206
 4.20000    0.30000    0.62325    0.65504
 4.40000    0.10000    0.64345    0.65380    0.65318
 4.60000   -0.10000    0.66276    0.65264    0.65324    0.65321

    At Xp = 4.500          Function value is  0.65321
```

3.7 ERROR ESTIMATION IN INTERPOLATION

So far, we have studied several formulas for interpolation. The basic principle in all these formulas is the approximation of a polynomial so that this polynomial passes through the set of points in a given table.

The error in an interpolation process is introduced by several sources:

(a) The truncation error due to terminating the series at the term in, say, the nth differences.

(b) The round-off errors in the function values and resulting errors in the differences causing oscillation in the differences.

(c) The round-off errors in the individual terms of the formula and their sum.

(d) Inaccuracy, usually due to rounding-off, in the given value of p.

We can estimate errors in any of the interpolation formulas from the first neglected term. However, we conclude this section by computing the error estimates in Newton's forward and backward difference formulas.

3.7.1 Error in Newton's Forward Difference Formula

If the function $y = f(x)$ is known explicitly, the remainder term in case of the nth-order forward difference formula is as follows:

$$E = \frac{h^{n+1}}{(n+1)!} p(p-1)\cdots(p-n) f^{(n+1)}(Z) \qquad \ldots (3.12)$$

where $x_0 < Z < x_n$.

If the function is specified by tabular values, the error is given by the following relation:

$$E = \frac{p(p-1)\cdots(p-n)}{(n+1)!} \Delta^{n+1} f_0 \qquad \ldots (3.13)$$

What can be done if the next term (i.e., (n+1)st) is not available? In this case, check if an additional point is available on the other side, namely f_{-1}. If it is available, $\Delta^{n+1} f_{-1}$ can be computed and used as an approximation for $\Delta^{n+1} f_0$.

Example 9 Given $f(x) = e^x$, for $x = 0(0.1)0.5$ correct to 4 dp.

i) Make a difference table and interpolate f(.175) using Newton's forward difference formula.

ii) Calculate the actual value of e^x for $x = .175$. Find the error in both the results.

iii) Use the formula (3.12) and estimate the error.

iv) What discretization size should be used if the entries are given to 6 dp?

Solution i) **Difference Table**

x	$f(x) = e^x$	Δ	Δ^2	Δ^3	Δ^4
0.0	1.0000				
		1052			
0.1	1.1052		110		
		1162		13	
0.2	1.2214		123		−2
		1285		11	
0.3	1.3499		134		5
		1419		16	
0.4	1.4918		150		
		1569			
0.5	1.6487				

$$x_p = 0.175;\ h = 0.1;\ x_0 = 0.1$$

$$p = \frac{(0.175 - 0.1)}{0.1} = 0.75$$

Using Newton's forward difference formula (3.2), we get,

$$f_p = 1.1052 + 0.75 \times .1162 + \frac{0.75(.75-1)}{2} \times .0123$$

$$+ \frac{0.75(0.75-1)(0.75-2)}{6} \times .0011$$

$$+ \frac{0.75(0.75-1)(0.75-2)(0.75-3)}{24} \times .0005$$

$$= 1.1052 + 0.08715 - 0.0012 + 0.00004 - 0.00001$$

$$= 1.1912$$

ii) True value, $e^x = e^{.175} = 1.1912$

Error = True value – Interpolated value

$$= 1.1912 - 1.1912 = 0$$

iii) $x_0 = 0.1;\ p = 0.75;\ h = 0.1$

$x_5 = 0.5,\ f(x) = e^x$

The fifth derivative is $f^{(v)}(x) = e^x$.

The maximum value, $f^{(v)}(x) = e^x = 1.64872$

$$E = \frac{h^5}{5!} p(p-1)(p-2)(p-3)(p-4)\, f^{(v)}(x)$$

$$= \frac{0.75(0.75-1)(0.75-2)(0.75-3)(0.75-4)}{120} \times (.1)^5 \times 1.64872$$

$$= 0.0138411 \times 0.00001 \times 1.64872 = 0.0000002$$

iv) To keep the accuracy less than $\frac{1}{2} \times 10^{-6}$, h should be:

$$h = \left[\frac{E \times 120}{p(p-1)(p-2)(p-3)(p-4)} \times \frac{1}{1.64872}\right]^{\frac{1}{5}}$$

$$= \left[\frac{0.0000005 \times 120}{1.64872 \times 1.7139}\right]^{\frac{1}{5}} = \left[\frac{0.00006}{2.8257412}\right]^{\frac{1}{5}}$$

$$= \left(2.12333 \times 10^{-5}\right)^{\frac{1}{5}} = 0.1163$$

3.7.2 Error in Newton's Backward Difference Formula

If the function $y = f(x)$ is known explicitly, the remainder term in case of the nth-order backward difference formula is as follows:

$$E = \frac{h^{n+1}}{(n+1)!} p(p+1)(p+2)\cdots(p+n) f^{(n+1)}(Z) \qquad \ldots (3.14)$$

where $x_0 < Z < x_n$.

If the function $y = f(x)$ is not known but is specified only by tabular values, the error is given by the following relation:

$$E = \frac{p(p+1)(p+2)\cdots(p+n)}{(n+1)!} \nabla^{n+1} f_0 \qquad \ldots (3.15)$$

Let us illustrate this method with an example.

Example 10 Given $f(x) = \sin x$, compute the values of $f(x)$ for $x = 0.1(0.1)0.8$ correct to 4 dp.

a) Construct the difference table and interpolate $f(.75)$ using Newton's backward difference formula.

b) Calculate the exact value of $\sin x$ for $x = 0.75$. Find the error in both the results.

c) Use the formula (3.14) and estimate the error.

d) What discretization size should be used if the entries are given to 6 dp.

Solution $f(x) = \sin x$; $n = 0(0.1)0.8$ radians.

a) **Difference Table**

x	f (x)	∇	∇^2	∇^3	∇^4	∇^5
0.1	0.0998					
		989				
0.2	0.1987		−21			
		968		−8		
0.3	0.2955		−29		1	
		939		−7		−12
0.4	0.3894		−36		−11	
		903		−18		−8
0.5	0.4797		−54		−19	
		849		1		7
0.6	0.5646		−53		−12	
		796		−11		
$x_0 = 0.7$	0.6442		−64			
		732				
0.8	0.7174					

$$x_p = 0.75;\ h = 0.1;\ x_0 = 0.7$$

$$p = \frac{x_p - x_0}{h} = \frac{0.75 - 0.7}{0.1} = 0.5$$

Using Newton's backward difference formula (3.4), we get,

$$f_p = 0.6442 + 0.5 \times 0.0796 + \frac{1}{2} \times 0.5\,(0.5 + 1) \times -0.0053$$

$$+ \frac{1}{6} \times 0.5\,(0.5 + 1)\,(0.5 + 2) \times 0.0001$$

$$+ \frac{1}{24} \times 0.5\,(0.5 + 1)\,(0.5 + 2)\,(0.5 + 3) \times -0.0019$$

$$= 0.6442 + 0.0398 - 0.00199 + 0.00003 - 0.000052$$

$$= 0.68199$$

(b) $f(x) = \text{Sin } x$

$= \text{Sin } 0.75 = 0.68164$

Error $= f_p - f(x)$

$= 0.68199 - 0.68164 = 0.0035$

(c) $f^{(iv)}(x) = \sin x$

$f^{(v)}(x) = \cos x$

Maximum, $f^{(v)}(x) = f^{(v)}(0.1) = 0.9950$

$$E = \frac{h^{n+1}}{(n+1)!} p(p+1)(p+2)(p+3)(p+4)\, f^{(v)}(x)$$

$$= \frac{(.1)^5}{5!} \times 0.5(0.5+1)(0.5+2)(0.5+3)(0.5+4) \times 0.9950$$

$$= \frac{0.00001}{120} \times (0.5)(1.5)(2.5)(3.5)(4.5) \times 0.9950$$

$$= 0.00000245$$

d) $E = \frac{1}{2} \times 10^{-6} = 0.0000005$

From the error formula, we get,

$$h = \left[\frac{E \times 120}{p(p+1)(p+2)(p+3)(p+4) \times f^{(v)}(x)} \right]^{\frac{1}{5}}$$

$$= \left[\frac{0.0000005 \times 120}{0.5(1.5)(2.5)(3.5)(4.5) \times 0.9950} \right]^{\frac{1}{5}}$$

$$= (0.000002041)^{\frac{1}{5}} = 0.073$$

PROBLEMS

1. (a) Show that a curve $y = f(x)$, where $f(x)$ is of the fourth degree, can be drawn through the points given by:

x	–1	0	1	2	3	4	5
f(x)	23	13	3	1	34	148	408

Use Newton's forward difference formula to find y exactly when $x = 1.2$.

(b) Given the following data:

x	– 4	– 2	0	2	4	6
f(x)	180	0	4	0	40	504

Use Newton's forward difference formula to find f(1.75).

(c) Consider the following table of values:

x	0.2	0.3	0.4	0.5	0.6
f(x)	0.2304	0.2788	0.3222	0.3617	0.3979

Find f(0.36) using Newton's forward difference formula.

(d) Prepare the difference table for the following data:

x	– 1	0	1	2	3
f	10	2	0	10	62

Using Newton's forward difference formula, interpolate the value for f(–.05).

(e) Prepare the difference table for the following data:

x	0	0.2	0.4	0.6	0.8
f	0.12	0.46	0.74	0.9	1.2

Find the value for f(0.1) using Newton's forward difference formula.

(f) Generate the difference table for the following data:

x	2.0	2.2	2.4	2.6	2.8	3
f	0.301	0.342	0.380	0.415	0.447	0.477

Hence estimate f(2.15) using Newton's forward difference formula.

2. (i) Use Newton's backward difference interpolation formula to estimate the value of f(1.45) from the following data:

x	1.0	1.1	1.2	1.3	1.4	1.5
f(x)	2.0	2.1	2.3	2.7	3.5	4.5

(ii) Use Newton's forward difference formula to find f(1.05) from the above data.

(iii) Consider the following data:

x	−1	−0.75	−0.50	−0.25	0	0.25
f(x)	−0.4401	0.0447	0.4311	0.6694	0.7652	0.7522

0.50	0.75	1
0.6714	0.5587	0.4401

(a) Use Newton's forward difference interpolation formula to estimate f(−0.33).

(b) Use Newton's backward difference interpolation formula to estimate f(0.62).

(iv) The following table of values represents a polynomial of degree $n \leq 3$. It is given that there is an error in one of the tabular values of f(x) near the end of the table.

x	0	0.1	0.2	0.3	0.4
f(x)	2.00	2.11	2.28	2.39	2.56

(a) Locate the error and correct the value.

(b) Reconstruct the difference table and estimate f(0.35) using Newton's backward difference formula.

3. (a) The values of a low degree polynomial are given in the table below. It is suspected that there is a transposition error in one of the values. By differencing, locate and correct the error and find f(2.5):

x	2	3	4	5	6	7	8	9	10
f(x)	15	40	85	165	259	400	585	820	1111

(b) One of the functional values in the following table contains an error:

x	2.0	2.1	2.2	2.3	2.4	2.5	2.6	2.7
f(x)	1.4142	1.4491	1.4832	1.5160	1.5492	1.5811	1.6125	1.6432

i) Detect and correct the erroneous term and then reconstruct the difference with the corrected functional value.

ii) Find f(2.05) using Newton's forward difference formula.

iii) Find f(2.65) using Newton's backward difference formula.

4. (i) Using Stirling's interpolation formula, find f(3.25) from the following data:

x	1	2	3	4	5
f(x)	0.0000	0.6931	1.0986	1.3863	1.6094

(ii) The following table gives the value of p_x of a polynomial of the fourth degree for certain values of p_x:

x	5	6	7	8	9
p_x	6.195	5.919	5.630	5.326	5.006

Estimate the polynomial using Stirling's formula when x = 7.5.

5. (a) Given the following table:

x	f(x)
0.01	98.4342
0.02	48.4392
0.03	31.7775
0.04	23.4492
0.05	18.4542

Estimate the value of f(x) corresponding to x = 0.0341 using the following formulas:

(i) Stirling (ii) Bessel (iii) Everett and (iv) Gauss forward and backward.

(b) Consider the following table of values:

x	1.6	1.8	2.0	2.2	2.4	2.6	2.8	3.0
f(x)	.0495	.0605	.0739	.0903	.1102	.1346	.1644	.2009

Find the values of f(2.35) and f(2.2) using all the central difference formulas you have studied.

(c) Use the following table of current i against deflection, θ:

θ	.40	.45	.50	.55	.60	.65
i	1.268	1.449	1.639	1.839	2.052	2.281

to find i, when $\theta = .536$, from the Stirling and Everett formulas. Check the answer using Newton's forward difference formula.

(d) Consider the following data:

x	1.0	1.1	1.2	1.3	1.4	1.5	1.6
f(x)	1.54308	1.66852	1.81066	1.97091	2.15090	2.35241	2.57746

Find the value of f(1.35) using:

(i) Stirling, (ii) Bessel, (iii) Everett, and (iv) Gauss both formulas.

(e) Consider the following data:

x	1.0	1.2	1.4	1.6	1.8	2.0
f(x)	0.367879	0.301194	0.246597	0.201897	0.165299	0.135335

Find the value of f(1.675) using:

(i) Stirling, (ii) Bessel, (iii) Everett, and (iv) Gauss both formulas.

(f) Kinematic viscosity of water, v, is related to temperature in the following manner:

T(°F)	40	50	60	70	80
v	1.66	1.41	1.22	1.06	0.93

Use a suitable interpolation formula to predict v at T = 62 °F.

(g) You measure the voltage drop v, across for a number of different values of current i. The results are:

i	0.25	0.75	1.25	1.75	2.25
v	−0.23	−0.33	0.70	1.88	6.00

Use a suitable interpolation formula to estimate the voltage drop for i = 0.9.

6. (a) Find f(x) at x = 1, from the following table using Lagrange's formula:

x	−1	0	2	3
f(x)	6	10	12	19

Use a suitable interpolation formula to estimate the voltage drop for i = 0.9.

(b) Fit a polynomial to the following data:

(−4, 180), (−2, 0), (0, 4), (1, 0), (3, 40) and (5, 504).

Use the polynomial to find a value for f(2.4).

(c) Fit a polynomial for the function $f(x) = \frac{2^x}{x}$, for x = 2, 4 and 8. Use this polynomial to estimate f(6).

(d) Given the following table of values:

x	14	17	31	35
y	68.7	64.0	44.0	39.1

What is y(27)?

(e) Use Lagrange's interpolation formula to obtain a polynomial of least degree that assumes the following values:

x	1	2	3	4
y	7	11	28	63

Use the polynomial obtained to estimate f(4.5). Check the answer using Newton's backward difference formula.

(f) The function y = f(x) is given in the points (7, 3), (8, 1), (9, 1) and (10, 9). Find the value of y for x = 9.5 using Lagrange's interpolation formula.

(g) Use Lagrange's formula to estimate f(2.0) and f(4.5) from the following data:

x	1.6	2.9	3.7	4.8
f(x)	0.6250	0.3448	0.2703	0.2083

(h) Given the following data:

x	1	2	4	8
f(x)	1	3	7	11

Find f(7) using Lagrange's formula.

(i) Let $f(x) = \frac{8x}{2^x}$. Fit a polynomial for the function when x = 0(1)3. Estimate the value of f(1.5).

(j) Given the points:

(x_i, f_i) : (0, 1), (1, .765198), (2, .223891), (3, – .260052).

(a) Find the Lagrange's interpolation polynomial $p_3(x)$ for the given data set. Use the polynomial to approximate f(1.5).

(b) Find the Newton's interpolation polynomial $p_3(x)$ for the given data set. Use the polynomial in (a) above to approximate f(1.5).

(k) Given the points:

$$(x_i, f_i): (0, 1), (.25, .9689), (.5, .8776), (.75, -.5403).$$

(a) Find the Lagrange's interpolation polynomial $p_3(x)$ for the given data set. Use the polynomial obtained to approximate f(.6).

(b) Find the Newton's interpolation polynomial $p_3(x)$ for the given data set. Use the polynomial obtained in (a) above to approximate f(.6).

(l) Find the Lagrange's interpolation polynomial to fit the following data:

x	0	1	2	3
f(x)	0	1.7183	6.3891	19.0855

Use the polynomial to estimate its value at x = 1.5.

(m) Applying Lagrange's formula, find a cubic polynomial which approximate the following values:

$$y(1) = -3,\ y(3) = 9,\ y(4) = 30 \text{ and } y(6) = 132.$$

Find also the value of y(4.5).

7. (a) Estimate the interpolation polynomial for $f(x) = x^2 + \sin \pi x$ through (0, 0), (1, 1) and (2, 4), using Newton's forward difference formula at x = 0.5.

(b) What is the exact error when x = 0.5?

(c) What is the maximum error in (a) above?

(d) Find the largest value of h that will ensure 4 dp accuracy in the value of f(x), assuming quadratic interpolation is used.

8. Consider the following table of values:

x	1	2	3	4	5
f(x)	1.0000	1.4142	1.7320	2.0000	2.2361

(a) Use Newton's forward difference formula to estimate f(1.5). Using third-order interpolation, estimate also the maximum error.

(b) If we are known that $f(x) = \sqrt{x}$, what is the error in this case? Find also the maximum error.

(c) Find the largest value of h that will give 6 dp accuracy in the value of x, assuming third-order interpolation is used.

9. (a) Use Aitken's formula to estimate f(0.2) as accurately as possible from the following rounded values of f(x):

x	.17520	.25386	.33565	.42078	.50946
f(x)	.84147	.86742	.89121	.91276	.93204

(b) Use Aitken's formula to estimate f(1.4) correct to 4 dp from the following data:

x	1.20	1.25	1.30	1.35	1.45	1.50
f(x)	0.1823	0.2231	0.2624	0.3365	0.3716	0.4055

(c) Use Aitken's formula to estimate f(5) correct to 2 dp from the following data:

x	1	4	7	9
f(x)	2	13	122	504

(d) Use Aitken's method to evaluate log 3.63 from the following table:

x	3.50	3.60	3.70	3.80
log x	1.2527632	1.28093	1.30833	1.33500

(e) Consider the function, $f(x) = \dfrac{1}{1+20x^2}$. Calculate the values of the function correct to 4 dp, for x = 0.2(0.2)1.0. Estimate the value of f(0.55) using Aitken's method.

10.(a) Consider the following table of values:

x	– 0.1	0.1	0.3	0.5	0.7	0.9
f(x)	.7196	.8075	.8812	.9385	.9776	.9975

i) Use Newton's forward difference interpolation formula to estimate f(0.25) using upto fourth-order differences.

ii) Find the maximum error.

(b) Consider the following table:

x	.2	.4	.6	.8	1
f(x)	.19951	.39646	.58813	.72210	.94608

i) Find the value of f(0.3) using Lagrange and Aitken's formula.

ii) Use Newton's forward difference formula to estimate f(0.3) using upto third-order differences. Find the maximum error.

11.(a) Consider the following part of a difference table:

x	f(x)	δ	δ^2	δ^3	δ^4
6	1296				
	→	2800	→	1344	
8	4096	→	3104	→	384
	→	5904	→	1728	
10	10000				

Compute f(9) using Stirling's formula for interpolation.

(b) Given the following part of a difference table:

x^0	Sin x^0	δ	δ^2	δ^3	δ^4
25	0.422618	→	− 3216	→	23
	→	77382	→	590	
30	0.500000	→	− 3806	→	384

Estimate f(26.5) using Bessel's formula for interpolation.

12. Using table values of Q.5(d) above, do the following:

(a) Estimate the value of f(1.425) using Newton's backward difference formula with fourth-order differences.

(b) Compute the maximum error.

13. Using tabular values of Q.5(e) above, do the following:

(a) Use Newton's backward difference formula to estimate f(1.90) with fourth-order differences.

(b) Compute the maximum error if the tabular values are based on the function $f(x) = e^{-x}$.

(c) Compute the largest value of h that will give 7 dp accuracy in the value of x, assuming the same order of interpolation as used in (a) above.

Chapter 4

Numerical Differentiation

4.1 INTRODUCTION

Numerical differentiation is useful in estimating the derivatives of a function f(x) when either f(x) is very complicated and is difficult to differentiate easily, or, it is not known as explicit expression in x, but the values of the function are given in a tabular form. We use numerical differentiation only when there is no better alternative method available to compute derivatives analytically or when the analytical solution is rather complicated. Generally, it is considered that numerical differentiation is basically an **unstable process** which means that small values of h can lead to greatly magnified errors in the final result. In fact, we may not always expect reasonable results even when the original data are known to be accurate. In actual practice this operation is avoided altogether if possible because it tends to enhance the effects of rounding errors present in the tabular values. This is particularly true when the $f(x_1)$ values are themselves subject to more error, as they would probably be if determined experimentally. If derivative values are computed in such cases, particularly when the results are to be used in subsequent calculations, it is usually better to consider curve fitting, using least-squares technique and differentiate the formula for the curve.

In this chapter, we shall derive some formulas for estimating derivatives. In spite of some inherent shortcomings, numerical differentiation is useful to derive formulas for solving integrals, ordinary and partial differential equations. Standard examples of numerical differentiation often use known functions so that the numerical approximation can be compared with the exact answer.

4.2 DERIVATION OF DIFFERENTIATION FORMULAS

Formulas for numerical differentiation may easily be obtained by differentiating interpolation polynomials.

In order to derive a differentiation formula, we differentiate a suitable interpolation formula with respect to p.

We shall write, $x = x_0 + ph$. Differentiating this w.r.t.p., we get,

$$\frac{dx}{dp} = h$$

or, $$\frac{dx}{dp} = \frac{1}{h} \qquad \ldots (4.1)$$

Also, $f_p = f(x) = f(x_0 + ph)$ ... (4.2)

Differentiating (4.2) w.r.t.x., we get,

$$\frac{df_p}{dx} = \frac{d}{dx} f(x_0 + ph)$$

$$= \frac{d}{dp} f(x_0 + ph) \frac{dx}{dp} \qquad \text{(Note the step.)}$$

$$= \frac{1}{h} \frac{d}{dp} f(x_0 + ph)$$

$$= \frac{1}{h} \frac{df_p}{dp}$$

Denoting $\frac{df_p}{dp}$ by f'_p, we get,

$$f'_p = \frac{1}{h} \frac{df_p}{dp} \qquad \ldots (4.3)$$

4.3 RELATIONSHIP BETWEEN OPERATORS E AND D

Before proceeding further, let us define one more operator D, called the **differential operator,**

$$Df_r = f'(x_r) = f'_r$$

Taylor series may also be written in the following manner:

$$f(r+1) = f(r) + hf'(r) + \frac{h^2}{2!} f''(r) + \frac{h^3}{3!} f'''(r) + \ldots$$

or, $$f_{r+1} = f_r + hf'_r + \frac{h^2}{2!} f''_r + \frac{h^3}{3!} f'''_r + \ldots$$

$$= f_r + hDf_r + \frac{h^2}{2!} D^2 f_r + \frac{h^3}{3!} D^3 f_r + \ldots$$

$$Ef_r = (1 + hD + \frac{h^2}{2!} D^2 + \frac{h^3}{3!} D^3 + \ldots) f_r$$

$$= e^{hD} f_r$$

or, simply, $E = e^{hD}$ ... (4.4)

4.4 DERIVATIVES USING NEWTON'S FORWARD DIFFERENCE INTERPOLATION FORMULA

(a) Elementary Approach (Using Interpolation Formula)

First-order Derivative

Newton's forward difference formula (3.2) is written as follows:

$$f_p = f_0 + p\Delta f_0 + \frac{1}{2}\left(p^2 - p\right)\Delta^2 f_0 + \frac{1}{6}\left(p^3 - 3p^2 + 2p\right)\Delta^3 f_0$$
$$+ \frac{1}{24}\left(p^4 - 6p^3 + 11p^2 - 6p\right)\Delta^4 f_0 + \cdots$$

Differentiating this formula with respect to p, we get,

$$\frac{d f_p}{d p} = \Delta f_0 + \frac{1}{2}(2p - 1)\Delta^2 f_0 + \frac{1}{6}\left(3p^2 - 6p + 2\right)\Delta^3 f_0$$
$$+ \frac{1}{24}\left(4p^3 - 18p^2 + 22p - 6\right)\Delta^4 f_0 + \ldots$$

Since, $f'_p = \frac{1}{h}\frac{d f_p}{d p}$, we get the first derivative as follows:

$$f'_p = \frac{1}{h}\left\{\Delta f_0 + \frac{1}{2}(2p - 1)\Delta^2 f_0 + \frac{1}{6}\left(3p^2 - 6p + 2\right)\Delta^3 f_0 + \frac{1}{12}\left(2p^3 - 9p^2 + 11p - 3\right)\Delta^4 f_0 + \cdots\right\} \quad \ldots (4.5)$$

Higher-order Derivatives

The method can be extended to find the higher-order derivatives. Differentiating (4.5) w.r.t.p., we get,

$$f''_p = \frac{1}{h}\frac{d f'_p}{d p}$$
$$= \frac{1}{h^2}\left\{\Delta^2 f_0 + \frac{1}{6}(6p - 6)\Delta^3 f_0 + \frac{1}{12}\left(6p^2 - 18p + 11\right)\Delta^4 f_0 + \cdots\right\}$$
$$= \frac{1}{h^2}\left\{\Delta^2 f_0 + (p - 1)\Delta^3 f_0 + \frac{1}{12}\left(6p^2 - 18p + 11\right)\Delta^4 f_0 + \cdots\right\} \quad \ldots (4.6)$$

Similarly, $f_p''' = \frac{1}{h}\frac{d f_p''}{d p}$

$$= \frac{1}{h^3}\left\{\Delta^3 f_0 + \frac{1}{2}\left(2p^2 - 3\right)\Delta^4 f_0 + \cdots\right\} \qquad \ldots (4.7)$$

The same procedure can be repeated to calculate the derivatives of any order.

Special Cases

Formulas for derivatives when

(i) $p = 0$

$$f_0' = \frac{1}{h}\left\{\Delta f_0 - \frac{1}{2}\Delta^2 f_0 + \frac{1}{3}\Delta^3 f_0 - \frac{1}{4}\Delta^4 f_0 + \cdots\right\}$$

$$f_0'' = \frac{1}{h^2}\left\{\Delta^2 f_0 - \Delta^3 f_0 + \frac{11}{12}\Delta^4 f_0 - \cdots\right\}$$

$$f_0''' = \frac{1}{h^3}\left\{\Delta^3 f_0 - \frac{3}{2}\Delta^4 f_0 - \cdots\right\}$$

(ii) $p = \frac{1}{2}$

$$f_{\frac{1}{2}}' = \frac{1}{h}\left\{\Delta f_0 - \frac{1}{24}\Delta^3 f_0 + \frac{1}{24}\Delta^4 f_0 - \cdots\right\}$$

$$f_{\frac{1}{2}}'' = \frac{1}{h^2}\left\{\Delta^2 f_0 - \frac{1}{2}\Delta^3 f_0 + \frac{7}{24}\Delta^4 f_0 - \cdots\right\}$$

$$f_{\frac{1}{2}}''' = \frac{1}{h^3}\left\{\Delta^3 f_0 - \Delta^4 f_0 - \cdots\right\}$$

(b) Derivatives Using Difference Operators

First-order Derivative

We can also derive the above formulas using difference operators.

Since, $e^{hD} = E = 1 + \Delta$, taking logarithm of both sides, we get,

$$hD = \log(1 + \Delta).$$

Expanding the right hand side, we get,

$$hD = \Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \dots$$

$$D = \frac{1}{h}\left\{\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \cdots\right\}$$

or, $$Df_0 = \frac{1}{h}\left\{\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \cdots\right\} f_0$$

Since, $Df_0 = f_0'$, we can write the above as follows:

$$f_0' = \frac{1}{h}\left\{\Delta f_0 - \frac{1}{2}\Delta^2 f_0 + \frac{1}{3}\Delta^3 f_0 - \frac{1}{4}\Delta^4 f_0 + \cdots\right\} \qquad \dots (4.8)$$

Higher-order Derivatives

$$f_0'' = Df_0 \times Df_0$$

$$= \frac{1}{h^2}\left\{\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \cdots\right\}^2 f_0$$

$$= \frac{1}{h^2}\left\{\Delta^2 f_0 - \Delta^3 f_0 + \frac{11}{12}\Delta^4 f_0 + \cdots\right\} \qquad \dots (4.9)$$

$$f_0''' = Df_0 \times Df_0 \times Df_0$$

$$= \frac{1}{h^3}\left\{\Delta - \frac{\Delta^2}{2} + \frac{\Delta^3}{3} - \frac{\Delta^4}{4} + \cdots\right\}^3 f_0$$

$$= \frac{1}{h^3}\left\{\Delta^3 f_0 - \frac{3}{2}\Delta^4 f_0 + \cdots\right\} \qquad \dots (4.10)$$

The rest of the formulas in this chapter will be derived using the elementary approach (i.e., using interpolation formulas).

It must be recognized that numerical differentiation is subject to considerable error. It should also be noted that all these formulas involve division of a combination of differences (which are prone to loss of significance or cancellation errors, especially if h is small) by a positive power of h. Consequently, if we want to reduce the round-off errors, we should use a large value of h. In brief, large errors may occur in numerical differentiation, based on direct polynomial approximation, so that an error check is always advisable. There are alternative methods, based on polynomials, which use more sophisticated procedures such as least-squares or mini-max, and other alternatives

involving other basis functions (for example, trigonometric functions). However, the best policy is probably to use numerical differentiation only when it cannot be avoided!

Example 1 (a) Use the following table of values,

x	0.0	0.5	1.0	1.5	2.0
f(x)	2.0286	2.4043	2.7637	3.1072	3.4350

to compute, $f'(.25)$, $f''(.25)$ and $f'''(.25)$.

(b) Write a computer program to implement the method for computing the first two derivatives.

Solution (a) $x_p = .25,\ h = 0.5,\ x_0 = 0.0$

$$p = \frac{(x_p - x_0)}{h} = \frac{(.25 - .00)}{.5} = 0.5$$

Difference Table

x	f(x)	Δ	Δ^2	Δ^3	Δ^4
$x_0 = 0.0$	2.0286				
		3757			
0.5	2.4043		– 163		
		3594		4	
1.0	2.7637		– 159		– 2
		3435		2	
1.5	3.1072		– 157		
		3278			
2.0	3.4350				

Substituting values of p and the required differences in (4.5), we get,

$$f'(.25) = f'_{.5} = \frac{1}{.5}\left\{.3757 + \frac{1}{2}(2\times 0.5 - 1)(-.0163) + \frac{1}{6}(3\times .5^2 - 6\times .5 + 2)\times .0004 + \frac{1}{12}(2\times .5^3 - 9\times .5^2 + 11\times .5 - 3)\times -.002\right\}$$

$$= \frac{1}{.5}\{.3757 - .000 - .0000 - .0000\} = 0.7514$$

Substituting values of p and the required differences in (4.6), we get,

$$f''(.25) = f''_{.5} = \frac{1}{.5^2}\left\{-.0163 + (.5-1)\times .0004 + \frac{1}{12}(6\times .5\times .5 - 18\times .5 + 11)\times -.0002\right\}$$

$$= \frac{1}{.25}\{.0163 - .0002 - .0001\}$$

$$= \frac{1}{.25} \times (-.0166) = -0.0664$$

Similarly, substituting values of p and the required differences in (4.7), we get,

$$f'''(.25) = f'''_{.5} = \frac{1}{.5^3}\left\{.0004 + \frac{1}{2}(2 \times .5 - 3) \times -.0002\right\}$$

$$= \frac{1}{.125}\{.0004 + (-.0002)\} = 0.0048$$

It is worthwhile to remember that the derivatives at non-tabular points can be obtained by extrapolation.

(b) Computer Program No. 5: Numerical Differences

```
# include<iostream.h>
# include<conio.h>

class NewForwDiff
  {
    public:
            NewForwDiff( );
            void input( );
            void result( );
            void get_1st_der( );
            void get_2nd_der( );
    private:
            int degree,values,actual_ degree;
            float delta[10],xp,p,x[10],fx[10],h;
  }

    NewForwDiff::NewForwDiff( )
    {
        clscr( );
        cout<<"\n\t\tFROM NEWTON'S FORWARD DIFFERENCE FORMULA\n\n";
        degree=values=xp=actual_degree=0;
        p= -2;
        for(int i=0; i<10; i++)
                delta[i]=x[i]=fx[i]=0.0;
    }
```

```
void NewForwDiff::input( )
{
        cout<<"How many values you want for x?\t";
        cin>>values;
        cout<<"Upto what power of Delta:\t";
        cin>>degree;
//      degree=degree>4 ||degree<1 ? 4 : degree;
        cout<<"\n Value of Xp:\t";
        cin>>xp;
        for(int i=0;i!=values;i++)
        {
            cout<<"\nEnter  X"<<i+1<<":\t";
            cin>>x[i];
            cout<<"Enter  F("<<i+1<<"):\t";
            cin>>fx[i];
        }
}

void NewForwDiff::result( )
{
        clscr( );
        cout<<"\nX\t";
        for(int i=0;i!=values;i++)
              cout<<"\t"<<x[i];
        cout<<endl;
        cout<<"\nF(x)\t";
        for(i=0;i!=values;i++)
             cout<<"\t"<<fx[i];
        cout<<endl;
        h=x[1]-x[0];
        for(int j=0;temp= -1;j<values && (p<0 || p>1); temp=j, j++)
              p=(xp-x[j])/h;

        cout<<"\nValue of P is :\t"<<p<<"\n";
        for(actual_degree=1, j=values; actual_degree<=degree&&j>1,actual_
            degree++, j--)
        {
              cout<<"\n\nDelta power "<<actual_degree<<":\t";
              for(int k=0;k<j-1;k++)
              {
                  fx[k]=fx[k+1]-fx[k];
                  cout<<fx[k]<<"\t";
              }
```

```
            delta[actual_degree-1]=fx[temp];
            cout<<delta[actual_degree-1];
        }
        get_1st_der( );
        get_2nd_der( );
}

void NewForwDiff::get_1st_der( );
{
        float parray[]={1,2*p-1,3*p*p-6p+2,4*p*p+18*p*p+22*p-6},
                        div[]={1,2,6,24},
                        ans=0;

        for(int i=0;i<actual_degree;i++)
                ans+=delta[i]*parray[i]/div[i];
        cout<<"\n\n\nf '("<<xp<<"):\t"<<ans/h;
}

    void NewForwDiff::get_2nd_der( );
    {
        float parray[]={1,p-1,6*p*p-18*p+11},
                        div[]={1,1,12},
                        ans=0;
        for(int i=1;i<actual_degree;i++)
                ans+=delta[i]*parray[i-1]/div[i-1];

        cout<<"\n\n\nf "("<<xp<<"):\t"<<ans/(h*h);
    }
void main (void)
{
    NewForwDiff obj;
    obj.input( );
    obj.result( );
    getch( );
}
```

Computer Output

FROM NEWTON'S FORWARD DIFFERENCE FORMULA

```
How many values you want for X? 5
Upto what power of Delta:      4

Value of Xp:   .25

Enter X1:      0
Enter F(1):    2.0286

Enter X2:      .5
Enter F(2):    2.4043

Enter X3:      1
Enter F(3):    2.7637

Enter X4:      1.5
Enter F(4):    3.1072

Enter X5:      2
Enter F(5):    3.435
```

X	0	0.5	1	1.5	2
F(X)	2.0286	2.4043	2.7637	3.1072	3.435

Value of P is : 0.5

Delta Power 1:	0.3757	0.3594	0.3435	0.3278	0.375
Delta Power 2:	– 0.0163	– 0.0159	– 0.0157	0.0163	
Delta Power 3:	0.0004	0.0002	0.0004		
Delta Power 4:	– 0.0002	– 0.0002			
f ′ (0.25):	0.7512				
f ″(0.25):	– 0.066232				

4.5 DERIVATIVES USING NEWTON'S BACKWARD DIFFERENCE INTERPOLATION FORMULA

Differentiating Newton's backward difference formula (3.3) w.r.t.p., we have,

First-order

$$f'_p = \frac{1}{h}\frac{df_p}{dp}$$

$$= \frac{1}{h}\left\{\nabla f_0 + \frac{1}{2}(2p+1)\nabla^2 f_0 + \frac{1}{6}(3p^2+6p+2)\nabla^3 f_0 + \frac{1}{12}(2p^3+9p^2+11p+3)\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.11)$$

Second-order

$$f''_p = \frac{1}{h}\frac{df'_p}{dp}$$

$$= \frac{1}{h^2}\left\{\nabla^2 f_0 + (p+1)\nabla^3 f_0 + \frac{1}{12}(6p^2+18p+11)\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.12)$$

Similarly, other higher-order derivatives can be obtained.

Special Cases Formulas for derivatives when

(i) $p = 0$

$$f'_0 = \frac{1}{h}\left\{\nabla f_0 + \frac{1}{2}\nabla^2 f_0 + \frac{1}{3}\nabla^3 f_0 + \frac{1}{4}\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.13)$$

$$f''_0 = \frac{1}{h^2}\left\{\nabla^2 f_0 + \nabla^3 f_0 + \frac{11}{12}\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.14)$$

(ii) $p = \frac{1}{2}$

$$f'_{\frac{1}{2}} = \frac{1}{h}\left\{\nabla f_0 + \nabla^2 f_0 + \frac{23}{24}\Delta^3 f_0 + \frac{11}{24}\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.15)$$

$$f''_{\frac{1}{2}} = \frac{1}{h^2}\left\{\nabla^2 f_0 + \frac{3}{2}\Delta^3 f_0 + \frac{43}{24}\nabla^4 f_0 + \cdots\right\} \quad \ldots (4.16)$$

Example 2 (a) The deflection f(x) measured at various distances x from one end of a cantilever is given in the following table:

x	0.0	0.2	0.4	0.6	0.8	1.0
f(x)	0.0000	0.0456	0.1278	0.3494	0.4027	0.4825

Evaluate f ′(0.85) and f ″(1.0) based on Newton's backward difference interpolation formula.

(b) Write a computer program to implement the method for computing the first two derivatives.

Solution (a) The difference table is as follows:

x	f(x)	∇	∇^2	∇^3	∇^4
0.0	0.0000				
		455			
0.2	0.0456		368		
		823		1025	
0.4	0.1278		1393		−4101
		2216		−3076	
0.6	0.3494		−1683		5024
		533		1948	
$x_0 = 0.8$	0.4027		265		
		798			
1.0	0.4825				

(i) $x_0 = 0.8;\ x_p = 0.85,\ h = 0.2.$

$$p = \frac{(x_p - x_0)}{h} = \frac{(0.85 - .08)}{0.2} = 0.25$$

Substituting values of p and the required differences in (4.11), we get,

$$f'(.85) = \frac{1}{0.2}\left\{.0533 + \frac{1}{2}(0.25\times 2+1)\times(-.01683)\right.$$
$$+\frac{1}{6}(3\times .25^2 + 6\times .25 + 2)\times(-.3076)$$
$$\left.+\frac{1}{12}(2\times .25^3 + 9\times .25^2 + 11\times .25 + 3)\times(-.4101)\right\}$$
$$= \frac{1}{.5}\{0.0533 - 0.1262 - 0.1891 - 0.2170\}$$
$$= -2.3950$$

(ii) $x_0 = 1;\ p = 0.$

Substituting values of p and the required differences in (4.14), we get,

$$f_0'' = \frac{1}{0.2^2}\left\{0.0265 + 0.1948 + \frac{11}{12}\times 0.5025\right\}$$

$$= \frac{1}{.04}\times 0.6819 = 17.0457$$

(b) Computer Program No. 6: Numerical Differentiation

```
# include<iostream.h>
# include<conio.h>

class NewBackDiff
  {
    public:
            NewBackDiff( );
            void input( );
            void result( );
            void get_1st_der( );
            void get_2nd_der( );

    private:
            int degree,values,actual_ degree;
            float nebla[10],xp,p,x[10],fx[10],h;
  };

    NewBackDiff::NewBackDiff( )
    {
        clscr( );
        cout<<"\n\tNEWTON'S BACKWARD DIFFERENCE FORMULA\n\n";
        degree=values=xp=actual_degree=0;
        p= -2;
        for(int i=0; i<10; i++)
                nebla[i]=x[i]=fx[i]=0.0;
    }

    void NewBackDiff::input( )
    {
            cout<<"How many values you want for X?\t";
            cin>>values;
            cout<<"Upto what power of Delta:\t";
```

```
        cin>>degree;
//      degree=degree>4 || degree<1 ? 4 : degree;
        cout<<"\n value of Xp:\t";
        cin>>xp;
        for(int i=0;i!=values;i++)
        {
            cout<<"\nEnter X"<<i+1<<":\t";
            cin>>x[i];
            cout<<"Enter F("<<i+1<<"):\t";
            cin>>fx[i];
        }
    }

void NewBackDiff::result( )
{
        clscr( );
        cout<<"\nX\t";
        for(int i=0;i!=values;i++)
            cout<<"\t"<<x[i];

        cout<<endl;
        cout<<"\nF(x)\t";
        for(i=0;i!=values;i++)
            cout<<"\t"<<fx[i];

        cout<<endl;
        h=x[1]-x[0];
        for(int j=values-1,temp= -1;j>=0&&(p<0 || p>1); temp=j,j--)
            p=(xp-x[j])/h;

        cout<<"\nValue of P is :\t"<<p<<"\n";
        for(actual_degree=1, j=values; actual_degree<=degree&&j>1,actual_
            degree++, j--)
        {
            cout<<"\n\nNebla power "<<actual_degree<<":\t";
            for(int k=0;k<j-1;k++)
            {
                fx[k]=fx[k+1]-fx[k];
                cout<<fx[k]<<"\t";
            }
            nebla[actual_degree-1]=fx[temp-actual_degree];
        }
        get_1st_der( );
```

```
        get_2nd_der( );
}

void NewBackDiff::get_1st_der( );
{
        float parray[]={1,2*p+1,3*p*p+6*p+2,2*p*p*p+9*p*p+11p+3},
                        div[]={1,2,6,12},
                        ans=0;

        for(int i=0;i<actual_degree;i++)
                ans+=nebla[i]*parray[i]/div[i];
        cout<<"\n\n\n\nf '("<<xp<<"):\t"<<ans/h;
}

    void NewBackDiff::get_2nd_der( );
    {
        float parray[]={1,p+1,6*p*p+18*p+11},
                        div[]={1,1,12},
                        ans=0;

        for(int i=1;i<actual_degree;i++)
                ans+=nebla[i]*parray[i-1]/div[i-1];

        cout<<"\n\n\n\nf "("<<xp<<"):\t"<<ans/(h*h);
    }

void main (void)
{
    NewBackDiff obj;
    obj.input( );
    obj.result( );
    getch( );
}
```

Computer Output

FROM NEWTON'S BACKWARD DIFFERENCE FORMULA

```
How many values you want for X? 6
Upto what power of Nebla:      4

Value of Xp:    .85
```

```
Enter X1:      0
Enter F(1):    0

Enter X2:      .2
Enter F(2):    .0455

Enter X3:      .4
Enter F(3):    .1278

Enter X4:      .6
Enter F(4):    .3494

Enter X5:      .8
Enter F(5):    .4027

Enter X6:      1
Enter F(6):    .4825
```

X	0	0.2	0.4	0.6	0.8	1
F(X)	0	0.0455	0.1278	0.3494	0.4027	0.4825

```
Value of P is : 0.25

Nebla Power 1:     0.0455    0.0823    0.2216    0.0533    0.0798

Nebla Power 2:     0.0368    0.1393   -0.1683    0.0265

Nebla Power 3:     0.1025   -0.3076    0.1948

Nebla Power 4:    -0.4101    0.5024

f'(0.85):        -2.393843

f''(0.85):       -27.383205
```

4.6 DERIVATIVES USING CENTRAL DIFFERENCE INTERPOLATION FORMULAS

The formulas derived in Sections 4.4 and 4.5 are not very accurate. A relatively higher accuracy can be achieved if we use one of the central difference formulas. Consequently, the numerical differentiation is more accurate if the derivatives of an interpolation formula at the centre of the data-points are used.

4.6.1 Derivatives Using Stirling's Interpolation Formula

Considering Stirling's formula (3.5(b)) in a bit different form:

$$f_p = f_0 + p\mu\delta f_0 + \frac{1}{2}p^2\delta^2 f_0 + \frac{1}{6!}(p^3 - p)\mu\delta^3 f_0 + \frac{1}{24}(p^4 - p^2)\delta^4 f_0 + \cdots$$

The derivatives are computed as follows:

$$f_p' = \frac{1}{h}\frac{df_p}{dp}$$

$$= \frac{1}{h}\left\{\mu\delta f_0 + p\delta^2 f_0 + \frac{1}{6}(3p^2 - 1)p\delta^3 f_0 + \frac{1}{6}(2p^3 - p)\delta^4 f_0 + \cdots\right\} \quad \ldots (4.17)$$

$$f_p'' = \frac{1}{h^2}\left\{\delta^2 f_0 + p\mu\delta^3 f_0 + \frac{1}{12}(6p^2 - 1)\delta^4 f_0 + \cdots\right\} \quad \ldots (4.18)$$

Special Cases

Substituting $p = 0$ in (4.17) and (4.18) respectively, we get,

$$f_0' = \frac{1}{h}\left\{\mu\delta f_0 - \frac{1}{6}\mu\delta^3 f_0 + \frac{1}{30}\mu\delta^5 f_0 - \cdots\right\} \quad \ldots (4.19)$$

$$f_0'' = \frac{1}{h^2}\left\{\delta^2 f_0 - \frac{1}{12}\delta^4 f_0 + \frac{1}{90}\delta^6 f_0 - \cdots\right\} \quad \ldots (4.20)$$

Example 3 The following table gives the coordinates of points on a certain polynomial curve:

x	0.0	0.2	0.4	0.6	0.8	1.0	1.2
y	0.710	1.175	1.811	2.666	3.801	3.801	3.801

Calculate the radius of curvature ρ using the following formula:

$$\rho = \frac{\left(1 + (y')^2\right)^{1.5}}{y''}, \text{ at the point } x = 0.6.$$

Solution: Difference Table

x	y	δ	δ^2	δ^3	δ^4	δ^5
0.0	0.710					
		465				
0.2	1.175		171			
		636		48		
0.4	1.811		219		13	
		855		61		2
$x_0 = 0.6$	2.666		280		15	
		1135		76		2
0.8	3.801		356		17	
		1491		93		
1.0	5.292		449			
		1940				
2.0	7.232					

$$x_0 = 0.6;\ h = 0.2;\ x_p = 0.6$$

$$\rho = \frac{(x_p - x_0)}{h} = \frac{(0.6 - 0.6)}{0.2} = 0$$

Substituting the required values in (4.19), we get,

$$f_0' = \frac{1}{0.2}\left\{\left(\frac{0.855 + 1.135}{2}\right) - \frac{1}{6}\left(\frac{0.016 + 0.076}{2}\right) + \frac{1}{30}\left(\frac{0.002 + 0.002}{2}\right)\right\}$$

$$= \frac{1}{0.2}\{0.995 - 0.011 + 0.000\}$$

$$= \frac{1}{0.2} \times 0.984 = \boxed{4.92}$$

Substituting the required values in (4.20), we have

$$f_0'' = \frac{1}{0.2^2}\left\{0.280 - \frac{1}{12} \times 0.15\right\}$$

$$= \frac{1}{0.04} \times 0.27875 = 6.969$$

$$\rho = \frac{\left(1+(f_0')^2\right)^{1.5}}{f_0''}$$

$$= \frac{\left(1+(4.92)^2\right)^{1.5}}{6.969} = \boxed{18}$$

4.6.2 Derivatives Using Bessel's Interpolation Formula

Bessel's interpolation formula:

$$f_p = f_0 + p\delta f_{\frac{1}{2}} + \frac{p(p-1)}{2.2!}\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{p(p-1)\left(p-\frac{1}{2}\right)}{3!}\delta^3 f_{\frac{1}{2}}$$

$$+ \frac{(p+1)p(p-1)(p-2)}{2.4!}\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots$$

The above formula can be written in the following simplified form:

$$f_p = f_0 + p\,\delta f_{\frac{1}{2}} + \frac{1}{4}\left(p^2 - p\right)\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{1}{6}\left(p^3 - \frac{3}{2}p^2 + \frac{1}{2}p\right)\delta^3 f_{\frac{1}{2}}$$

$$+ \frac{1}{48}\left(p^4 - 2p^3 - p^2 + 2p\right)\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots$$

The derivatives are computed as follows:

$$f_p' = \frac{1}{h}\frac{d f_p}{d p}$$

$$= \frac{1}{h}\left\{\delta f_{\frac{1}{2}} + \frac{1}{4}(2p-1)\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{1}{6}\left(3p^2 - 3p + \frac{1}{2}\right)\delta^3 f_{\frac{1}{2}}\right.$$

$$\left. + \frac{1}{24}\left(2p^3 - 3p^2 - p + 1\right)\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots\right\} \qquad \ldots (4.21)$$

$$f_p'' = \frac{1}{h^2}\frac{d f_p'}{d p}$$

$$= \frac{1}{h^2}\left\{\frac{1}{2}\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{1}{2}(2p-1)\delta^3 f_{\frac{1}{2}} + \frac{1}{24}\left(6p^2 - 6p - 1\right)\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots\right\}$$

$$\ldots (4.22)$$

Example 4 Consider the following table of values:

x	0.0	0.2	0.4	0.6	0.8	1.0	1.2	1.4
f(x)	0.0000	0.0096	0.0896	0.3456	0.9216	2.0000	3.8016	6.5856

Compute the values of f_p' and f_p'' obtained from Bessel's formula with $x = 0.63$.

Solution: **Difference Table**

x	f(x)	δ	δ^2	δ^3	δ^4
0.0	0.0000				
		96			
0.2	0.0096		704		
		800		1056	
0.4	0.0896		1760		384
		2560		1440	
$x_0 = 0.6$	0.3456		3200		384
				1824	
0.8	0.9216		5024		384
		10784		2208	
1.0	2.0000		7232		384
		18016		2592	
1.2	3.8016		9824		
		27840			
1.4	6.5856				

$$x_p = 0.63; \quad x_0 = 0.60; \quad h = 0.2$$

$$\rho = \frac{(x_p - x_0)}{h} = \frac{(0.63 - 0.60)}{0.2} = 0.15$$

$$\therefore f_p' = f'(0.63) = \frac{1}{0.2}\Big\{0.5760 + \frac{1}{4}(2\times 0.15 - 1)(0.3200 + 0.5024)$$
$$+ \frac{1}{6}\left(3\times 0.15^2 - 3\times 0.15 + \frac{1}{2}\right)(0.1824)$$
$$+ \frac{1}{24}(2\times 0.15^3 - 3\times 0.15^2 + 2)(0.0384 + 0.0384)\Big\}$$
$$= \frac{1}{0.2}\Big\{0.5760 - 0.1750 * 0.8224 + \frac{1}{6}(0.0675 - 0.2250)(0.1824)$$
$$+ \frac{1}{24}(0.0068 - 0.06750 - 0.15 + 2)(0.0768)\Big\}$$

$$= \frac{1}{0.2}\{0.5760 - 0.1439 - 0.00479 + 0.0062\}$$

$$= \frac{1}{0.2} \times 0.4335 = 2.1676$$

$$\therefore f''_p = f''(0.63) = \frac{1}{0.2 \times 0.2}\left\{\frac{1}{2}(0.3200 + 0.5024) + \frac{1}{2}(2 \times 0.15 - 1)(0.1824)\right.$$

$$\left. + \frac{1}{24}(2 \times 0.15^2 - 6 \times 0.15 - 1)(0.0384 + 0.0384)\right\}$$

$$= \frac{1}{0.4}\left\{\frac{1}{2} \times 0.8224 - 0.35 \times 0.1824 + \frac{1}{24}(0.1350 - 0.9000 - 1)(0.0768)\right\}$$

$$= \frac{1}{0.4}\{0.4112 - 0.0638 - 0.0057\} = 7.2600$$

4.6.3 Derivatives Using Everett's Interpolation Formula

Everett's formula is expressed as follows:

$$f_p = q f_0 + \frac{1}{6} q(q^2 - 1)\delta^2 f_0 + \frac{1}{120} q(q^2 - 1)(q^2 - 4)\delta^4 f_0 + \cdots$$

$$+ p f_1 + \frac{1}{6} p(p^2 - 1)\delta^2 f_1 + \frac{1}{120} p(p^2 - 1)(p^2 - 4)\delta^4 f_1 + \cdots$$

Substituting $(1 - p)$ for q in the above, we get,

$$f_p = (1-p) f_0 + \frac{1}{6}(1-p)\{(1-p)^2 - 1\}\delta^2 f_0 + \frac{1}{120}(1-p)\{(1-p)^2 - 1\}\{(1-p)^2 - 4\}\delta^4 f_0$$

$$+ \cdots + p f_1 + \frac{1}{6}(p^3 - p)\delta^2 f_1 + \frac{1}{120}(p^5 - p^3 + 4p)\delta^4 f_1 + \cdots$$

$$= (1-p) f_0 + \frac{1}{6}(-p^3 + 3p^2 - 2p)\delta^2 f_0 + \frac{1}{120}(-p^5 + 5p^4 - 5p^3 - 5p^2 + 6p)\delta^4 f_0$$

$$+ \cdots + p f_1 + \frac{1}{6}(p^3 - p)\delta^2 f_1 + \frac{1}{120}(p^5 - 5p^3 + 4p)\delta^4 f_1 + \cdots$$

Differentiating with respect to p, we can get the respective derivatives:

$$f'_p = \frac{1}{h}\frac{d}{dp} f_p$$

$$= \frac{1}{h}\left\{-f_0 + \frac{1}{6}(-3p^2 + 6p - 2)\delta^2 f_0 + \frac{1}{120}(-5p^4 + 20p^3 - 15p^2 - 10p + 6)\delta^4 f_0\right.$$

$$\left. + \cdots + f_1 + \frac{1}{6}(3p^2 - 1)\delta^2 f_1 + \frac{1}{120}(5p^4 - 15p^2 + 4)\delta^4 f_1 + \cdots\right\} \quad \ldots (4.23)$$

$$f''_p = \frac{1}{h^2}\frac{d}{dp}f'_p$$

$$= \frac{1}{h^2}\left\{\frac{1}{6}(-6p+6)\delta^2 f_0 + \frac{1}{120}(-20p^3+60p^2-30p-10)\delta^4 f_0 + \cdots + \frac{1}{6}(6p)\delta^2 f_1 + \frac{1}{120}(20p^3-30p)\delta^4 f_1 + \cdots\right\}$$

$$= \frac{1}{h^2}\left\{(1-p)\delta^2 f_0 - \frac{1}{12}(2p^3-6p^2-3p+1)\delta^4 f_0 + \cdots + p\,\delta^2 f_1 + \frac{1}{12}(2p^3-3p)\delta^4 f_1 + \cdots\right\} \quad \ldots (4.24)$$

4.6.4 Derivatives Using Gauss Interpolation Formula

Let us consider the Gauss forward and backward formulas one by one.

a) From Gauss Forward Difference Formula

$$f_p = f_0 + p\delta f_{\frac{1}{2}} + \frac{1}{2}(p^2-p)\delta^2 f_0 + \frac{1}{6}(p^3-p)\delta^3 f_{\frac{1}{2}} + \frac{1}{24}(p^4-2p^3-p^2+2p)\delta^4 f_0 + \cdots$$

Differentiating with respect to p, we can get the respective derivatives:

$$f'_p = \frac{1}{h}\frac{d}{dp}f_p$$

$$= \frac{1}{h}\left\{\delta f_{\frac{1}{2}} + \frac{1}{2}(2p-1)\delta^2 f_0 + \frac{1}{6}(3p^2-1)\delta^3 f_{\frac{1}{2}} + \frac{1}{24}(4p^3-6p^2-2p+2)\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{h}\left\{\delta f_{\frac{1}{2}} + \frac{1}{2}(2p-1)\delta^2 f_0 + \frac{1}{6}(3p^2-1)\delta^3 f_{\frac{1}{2}} + \frac{1}{12}(2p^3-3p^2-p+1)\delta^4 f_0 + \cdots\right\}$$

$$\ldots (4.25)$$

$$f''_p = \frac{1}{h^2}\frac{d}{dp}f'_p$$

$$= \frac{1}{h^2}\left\{\delta^2 f_0 + p\,\delta^3 f_{\frac{1}{2}} + \frac{1}{12}\left(6p^2 - 6p - 1\right)\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{h^2}\left\{\delta^2 f_0 + p\,\delta^3 f_{\frac{1}{2}} + \frac{1}{12}\left(6p^2 - 6p - 1\right)\delta^4 f_0 + \cdots\right\} \qquad \ldots (4.26)$$

b) From Gauss Backward Difference Interpolation Formula

$$f_p = f_0 + p\delta f_{-\frac{1}{2}} + \frac{1}{2}\left(p^2 + p\right)\delta^2 f_0 + \frac{1}{6}\left(p^3 - p\right)\delta^3 f_{-\frac{1}{2}}$$

$$+ \frac{1}{24}\left(p^4 - 2p^3 - p^2 + 2p\right)\delta^4 f_0 + \cdots$$

Differentiating with respect to p, we can get the respective derivatives:

$$f'_p = \frac{1}{h}\frac{d}{dp}f_p$$

$$= \frac{1}{h}\left\{\delta f_{-\frac{1}{2}} + \frac{1}{2}(2p+1)\delta^2 f_0 + \frac{1}{6}\left(3p^2 - 1\right)\delta^3 f_{-\frac{1}{2}} + \frac{1}{24}\left(4p^3 - 6p^2 - 2p + 2\right)\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{h}\left\{\delta f_{-\frac{1}{2}} + \frac{1}{2}(2p+1)\delta^2 f_0 + \frac{1}{6}\left(3p^2 - 1\right)\delta^3 f_{-\frac{1}{2}} + \frac{1}{12}\left(2p^3 - 3p^2 - p + 1\right)\delta^4 f_0 + \cdots\right\}$$

$$\ldots (4.27)$$

$$f''_p = \frac{1}{h^2}\frac{d}{dp}f'_p$$

$$= \frac{1}{h^2}\left\{\delta^2 f_0 + p\,\delta^3 f_{-\frac{1}{2}} + \frac{1}{12}\left(6p^2 - 6p - 1\right)\delta^4 f_0 + \cdots\right\} \qquad \ldots (4.28)$$

Example 5 Consider the following table of values obtained from the function,

$$f(x) = \frac{1}{1+x^2}:$$

x	0.44	0.46	0.48	0.50	0.52	0.54	0.56
f(x)	0.83780	0.82535	0.81274	0.80000	0.78715	0.77423	0.76127

a) Obtain the first two derivatives, $f'(0.5)$ and $f''(0.5)$, based on Stirling, Bessel, Everett and Gauss forward and backward formulas.

b) Compute the values of the derivatives using analytical method. Compare the two results and comment on your observation.

Solution: **Difference Table**

x	f(x)	δ	δ^2	δ^3	δ^4
0.44	0.83780				
		1245			
0.46	0.82535		−16		
		−1261		3	
0.48	0.81274		−13		−1
		−1274		2	
$x_0 = 0.50$	0.80000		−11		2
		−1285		4	
0.52	0.78715		−7		−1
		−1292		3	
0.54	0.77423		−4		
		−1296			
0.56	0.76127				

a) $x_p = 0.5;\ x_0 = 0.5;\ h = 0.02$

$$\rho = \frac{(x_p - x_0)}{h} = \frac{(0.5 - 0.5)}{0.02} = 0$$

(i) Using relation (4.19) due to Stirling:

$$f_0' = \frac{1}{h}\left\{\frac{1}{2}\left(\delta f_{\frac{1}{2}} + \delta f_{\frac{1}{2}}\right) - \frac{1}{12}\left(\delta^3 f_{-\frac{1}{2}} + \delta^3 f_{\frac{1}{2}}\right) + \cdots\right\}$$

$$= \frac{1}{0.02}\left\{\frac{1}{2}(-0.01274 - 0.01285) - \frac{1}{12}(0.00002 + 0.00004)\right\}$$

$$= \frac{1}{0.02}\{-0.012795 - 0.000005\}$$

$$= -0.64$$

Substituting values in (4.20), we get:

$$f_0'' = \frac{1}{h^2}\left\{\delta^2 f_0 - \frac{1}{12}\delta^4 f_0\right\}$$

$$= \frac{1}{0.02\times 0.02}\left\{-0.00011 - \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.0004}\{-0.00011 - 0.0000017\}$$

$$= \frac{1}{0.0004}\times -0.00011167 = -0.2792$$

(ii) Using relations (4.21) and (4.22) due to Bessel with p = 0, we get:

$$f_0' = \frac{1}{h}\left\{\delta f_{\frac{1}{2}} - \frac{1}{4}\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{1}{12}\delta^3 f_{\frac{1}{2}} + \frac{1}{12}\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots\right\}$$

$$= \frac{1}{0.02}\left\{-0.01285 - \frac{1}{4}(-0.00011 - 0.00007) + \frac{1}{\ } \times 0.00004\right.$$

$$\left. + \frac{1}{12}(+0.00002 - 0.00001)\right\}$$

$$= \frac{1}{0.02}\{-0.01285 + 0.000045 + 0.0000033 + 0.0000008\}$$

$$= \frac{1}{0.02}\times -0.01280$$

$$= -0.64$$

$$f_0'' = \frac{1}{h^2}\left\{\frac{1}{2}\left(\delta^2 f_0 + \delta^2 f_1\right) + \frac{1}{2}\delta^3 f_{\frac{1}{2}} - \frac{1}{24}\left(\delta^4 f_0 + \delta^4 f_1\right) + \cdots\right\}$$

$$= \frac{1}{0.02\times 0.02}\left\{\frac{1}{2}(-0.00011 - 0.00007) + \frac{1}{2}\times 0.00004 - \frac{1}{24}(0.00002 - 0.00001)\right\}$$

$$= \frac{1}{0.0004}\{-0.00009 - 0.00002 - 0.00000042\}$$

$$= \frac{1}{0.0004}\times -0.00011042$$

$$= -0.27604$$

(iii) Using relations (4.23) and (4.24) due to Everett, with p = 0, we get:

$$f_0' = \frac{1}{h}\left\{-f_0 - \frac{1}{3}\delta^2 f_0 + \frac{1}{20}\delta^4 f_0 + f_1 + \frac{1}{6}\delta^2 f_1 + \frac{1}{30}\delta^4 f_1\right\}$$

$$= \frac{1}{0.02}\left\{-0.80000 - \frac{1}{3}\times -0.00011 + \frac{1}{20}\times 0.00002 + 0.78715 - \frac{1}{6}\times -0.00007 + \frac{1}{30}\times -0.00001\right\}$$

$$= \frac{1}{0.02}\{-0.80000 + 0.000037 + 0.000001 + 0.78715 + 0.000012 - 0.00000033\}$$

$$= \frac{1}{0.02}\times -0.0128 = -0.64$$

$$f_0'' = \frac{1}{h^2}\left\{\delta^2 f_0 - \frac{1}{12}\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{0.02\times 0.02}\left\{-0.00011 - \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.0004}\{-0.00011 - 0.0000017\}$$

$$= -0.27925$$

(iv) Using relations (4.25) and (4.26) due to Gauss forward with p = 0, we get:

$$f_0' = \frac{1}{h}\left\{\delta f_{\frac{1}{2}} - \frac{1}{2}\delta^2 f_0 - \frac{1}{6}\delta^3 f_{\frac{1}{2}} + \frac{1}{12}\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{0.02}\left\{-0.01285 - \frac{1}{2}\times -0.00011 - \frac{1}{6}\times 0.0004 + \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.02}\{-0.01285 + 0.000055 - 0.0000067 + 0.0000017\}$$

$$= \frac{1}{0.02}\times -0.01280$$

$$= -0.640$$

$$f_0'' = \frac{1}{h^2}\left\{\delta^2 f_0 - \frac{1}{12}\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{0.02\times 0.02}\left\{-0.00011 - \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.0004}\{-0.00011 - 0.0000017\}$$

$$= \frac{1}{0.0004}\times -0.0001117$$

$$= -0.2792$$

(v) Using relations (4.27) and (4.28) due to Gauss Forward with $p = 0$, we get:

$$f_0' = \frac{1}{h}\left\{\delta f_{-\frac{1}{2}} - \frac{1}{2}\delta^2 f_0 - \frac{1}{6}\delta^3 f_{-\frac{1}{2}} + \frac{1}{12}\delta^4 f_0 + \cdots\right\}$$

$$= \frac{1}{0.02}\left\{-0.01274 + \frac{1}{2}\times -0.00011 - \frac{1}{6}\times 0.0002 + \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.02}\{-0.01274 - 0.00[illegible] \quad [illegible] - 0.0000017\}$$

$$= \frac{1}{0.02}\times -0.012797$$

$$= -0.64$$

$$f_0'' = \frac{1}{h^2}\left\{\delta^2 f_0 - \frac{1}{12}\delta^4 f_0\right\}$$

$$= \frac{1}{0.02\times 0.02}\left\{-0.00011 - \frac{1}{12}\times 0.00002\right\}$$

$$= \frac{1}{0.0004}\{-0.00011 - 0.0000017\}$$

$$= \frac{1}{0.0004}\times -0.0001117$$

$$= -0.2792$$

(b) **Analytical Solution**

$$f(x) = \frac{1}{1+x^2} = (1+x^2)^{-1}$$

$$f'(x) = -1\times 2x\,(1+x^2)^{-2} = -2x\,(1+x^2)^{-2}$$

$$f'(0.5) = \frac{-2\times 0.5}{[1+(0.5)^2]} = \frac{-1}{1.5625} = -0.64$$

$$f''(x) = -2(1+x^2)^{-2} + 8x^2(1+x^2)^{-3}$$

$$= \frac{-2}{(1+x^2)^2} + \frac{8\times(0.5)^2}{[1+(0.5)^2]^3}$$

$$= \frac{-2}{1.5625} + \frac{2}{1.953125}$$

$$= -1.28 + 1.024 = -0.256$$

The analytical solution for the first derivative is the same but the second derivative is not correct. It may be due to the rounded numbers used in the data and hence the second derivative failed to produce the reliable answer.

PROBLEMS

1. (a) (i) How are formulas for the derivatives of a function obtained from interpolation formulae?

(ii) Why is the accuracy of the usual numerical differentiation process not necessarily increased if the argument interval is reduced?

(iii) When should numerical differentiation be used?

(iv) What are the shortcomings of numerical differentiation?

(b) Given the following table of values:

x	2.3	2.5	2.7	2.9	3.1	3.3
f(x)	9.97	12.18	14.88	18.17	22.20	27.11

Calculate the following derivatives based on Newton's forward difference interpolation formula:

f_0', f_0'', $f'(2.4)$ and $f''(2.4)$.

2. (a) Evaluate the first derivative of the function based on Newton's forward difference formula using the following table of values at the point x_0:

x	f(x)
x_{-2}	0.84805
x_{-1}	0.85717
x_0	0.86603
x_1	0.87462
x_2	0.88295

The above table is a part of a table of sin x at 1° intervals and x = 60°. Check by analytical consideration the result. What is the error? Take 1° = 0.01745 radians.

(b) Find the value of f ′(0.15) based on the forward difference interpolation formula using the following data:

x	0.1	0.2	0.3	0.4	0.5	0.6
f(x)	0.425	0.475	0.400	0.450	0.525	0.575

3. If y = f(x) is a cubic polynomial given by:

x	1	2	3	4	5	6	7	8
y	2.105	2.808	3.614	4.604	5.857	7.451	9.467	11.985

Find y′(4.75) and y″(4.75) based on Newton's backward difference formula.

4. (a) A function is represented by the following table:

x	1.0	1.2	1.4	1.6	1.8	2.0
y	0.000	–0.112	–0.016	0.336	0.992	2.000

Find, correct to 3 dp, the values of y, y′ and y″, when x = 1.45, based on Stirling's formula for interpolation.

(b) Obtain the first and second derivatives at x = 7 of the function tabulated below:

x	5	6	7	8	9	10
f(x)	196	394	686	1090	1624	2306

Use derivatives computed from the Stirling's formula.

5. (a) Evaluate f ′(0.25) and f ″(0.25) based on Bessel's interpolation formula from the following tabular values:

x	0.0	0.1	0.2	0.3	0.4	0.5
f(x)	1.1445	1.0983	1.0575	1.0210	0.9881	0.9582

(b) Using derivatives from Bessel's interpolation formula, calculate $f'(0.8)$, $f''(0.8)$, $f'(0.85)$ and $f''(0.85)$ from the following table:

x	f(x)	δ	δ^2	δ^3	δ^4
0.7	0.87342		653		15
		7621		106	
0.8	0.94963		759		19
		8380		125	
0.9	1.03343		884		25

6. (a) Starting from Everett's interpolation formula, derive the expressions for the first two derivatives. Estimate from the following table, the values of f_p, f'_p, and f''_p when $x = 1.45$.

x	1.0	1.2	1.4	1.6	1.8	2.0
f(x)	0.000	–0.112	–0.016	0.336	0.992	2.000

(b) Given the following values of J(x), estimate using Everett's formula, the values of J(x), J′(x) and J″(x) at x = 1.055:

x	.8	.9	1.0	1.1	1.2	1.3
J(x)	.368842	.405950	.440051	.477092	.498289	.522023

(c) Consider the following table of values of f(x) = tan x:

x	f(x)
1.36	4.67344
1.38	5.17744
1.40	5.79788
1.42	6.58112
1.44	7.60183

Use Everett's formula to compute $f'(1.4)$. Compute also the exact value and the error thus created.

7. Consider the following tables:

(a)

x	0	1	2	3
f(x)	1	0	–1	7

Calculate $f(1.5)$, $f'(1.5)$ and $f''(1.5)$, using Lagrange's interpolation formula.

(b)

x	0	0.5	1.0	1.5
$f(x) = \sqrt{x}$	0.0000	0.7071	1.0000	1.2247

Calculate f(.55), f ′(.55) and f ″(.55), using Lagrange's interpolation formula. Find also the error in each case.

(c) Let $f(x) = 2^x \sin x$. Compute f ′(1.05) and f ″(1.05), using Lagrange's interpolation formula:

x	1.0	1.04	1.06	1.10
f(x)	1.6829	1.7733	1.8188	1.9103

8. Considering a uniform beam of 1 m long simply supported at both ends, the bending moment is given by the following relation:

$$y'' = \frac{M(x)}{EI}$$

where y(x) is the deflection, M(x) is the bending moment and EI is the flexural rigidity.

Calculate the bending moment at each grid-point including the two end points, assuming that the deflection distribution is among the following:

x (in m)	0.0	0.2	0.4	0.6	0.8	1.0
y(x) (in cm)	0.0	7.78	10.68	8.37	3.97	0.0

Assume, $EI = 1.2 \times 10^7\ Nm^2$.

Estimate the values of the bending moment, using the following formulas:

i) Forward difference for $x_p = 0.25$.

ii) Backward difference for $x_p = 0.90$.

iii) Central difference (all formulas) for $x_p = 0.55$.

9.(a) Consider the following table of values:

x	1.0	1.2	1.4	1.6	1.8
f(x)	2.7183	3.3201	4.0552	4.9530	6.0496

Using Stirling's formula, find f ′(1.4) and f ″(1.4).

(b) The following table gives the distance traveled d from rest by a car at various times t:

t	0	.5	1.0	1.5	2.0	2.5	3.0	3.5
d	0.00	0.07	0.53	1.60	3.61	6.61	10.62	15.62

4.0	4.5	5.0
21.54	28.09	35

What was the acceleration of the car, when t = 0, 2.4 and 4.6? Work as far as 4th differences.

10. (a) The distance D = D(t) traveled by an object is given in the table below:

t	8	9	10	11	12
D(t)	17.453	21.460	25.752	30.301	35.084

(i) Find the velocity v(10) by Stirling's formula.

(ii) Compare your answer with $D(t) = -70 + 7t + 70e^{-\frac{1}{10}}$.

(b) From the table below, calculate f′(0.4), f″(0.4) and f‴(0.4):

x	f(x)	δ	δ^2	δ^3	δ^4
0.3	0.17835		104		– 18
		1477		255	
0.4	0.19312		359		– 23
		1836		232	
0.5	0.21148		591		– 30

Use derivatives based on Stirling's formula.

11. Consider the following table of values:

x	0.0	0.2	0.4	0.6	0.8	1.0	1.2	1.4	1.6
f(x)	1.000	1.102	1.207	1.330	1.486	1.689	1.955	2.301	2.727

(a) i) Estimate the first two derivatives based on Newton's forward difference formula when $x_p = 0.2$.

ii) Estimate the first two derivatives based on Newton's backward difference formula when $x_p = 1.6$.

(b) i) Estimate the first two derivatives based on each of the following formulas taking $x_p = 0.6$:

(i) Stirling, (ii) Bessel, (iii) Everett, and (iv) Gauss forward and backward formulas.

ii) Estimate the first two derivatives based on Newton's backward difference formula when $x_p = 1.6$.

(c) Consider the following table based on the function $f(x) = \sqrt{x}$ to 5 dp:

x	1.00	1.05	1.10	1.15	1.20	1.25	1.30
f(x)	1.00000	1.02470	1.04881	1.07238	1.09545	1.11803	1.14018

i) Estimate the values of f′(1.00) and f″(1.00) using Newton's forward difference formula.

ii) Estimate the values of f′(1.30) and f″(1.30) using Newton's backward difference formula.

Chapter 5

Numerical Integration

5.1 INTRODUCTION

Integration is the process of measuring the area under a function plotted on a graph. Why would we want to do so? Among the most common examples are finding the velocity of a body from acceleration functions, and displacement of a body from velocity data. Throughout the engineering fields, there are (what sometimes seems like) countless applications for integral calculus. Sometimes, the evaluation of expressions involving these integrals can become daunting, if not indeterminate. For this reason, a wide variety of numerical methods have been developed to find the integral.

The purpose of this chapter is to develop the basic principles of **numerical integration**, which are used to obtain approximate results for some definite integrals. We restrict ourselves to define integrals of the form:

$$I = \int_a^b f(x)dx \qquad \text{.... (5.1)}$$

where a and b are finite and f(x) is a continuous function of x for $a \le x \le b$. Some examples of definite integrals are,

$$\int_2^4 x\,dx\,,\quad \int_{-1}^{\frac{1}{2}} x^{-x^2}\,dx\,,\quad \int_0^{\frac{\pi}{2}} \frac{dx}{1+\sin^2 x}\,,\quad \int_0^2 \frac{e^{2x}}{1+x^2}dx\,,\ \text{etc.}$$

The indefinite integrals are included among the solutions of ordinary differential equations discussed in Chapter 6. The value of I is interpreted as an area bounded by the curve $y = f(x)$, the x-axis, and the two ordinates at $x = a$ and $x = b$; I represents a number which is interpreted as an **area**. The numerical integration is often referred to as **quadrature** (also **mechanical quadrature**) which simply means working out an area.

The use of numerical integration becomes necessary when either the function f(x) cannot be integrated analytically or the analytical solution of the integral presents such difficulties that it is faster to find a numerical solution or when the values of functions are available only in a tabular form but no information is available about the function itself.

There are several methods available in the literature for numerical integration, but the most commonly used methods may be classified into the following two groups:

(a) **Newton-Cotes formulas** that employ functional values at equally-spaced data-points, and

(b) The **Gaussian quadrature formulas** that employ functional values at equally-spaced data-points determined by certain properties of orthogonal polynomials.

We shall mostly confine ourselves to the Newton-Cotes formulas, which can be derived by integrating one of the interpolation formulas.

We now approach the object of numerical integration: the goal is to approximate the definite integral of f(x) over the interval [a, b] by evaluating f(x) at a definite number of sample points.

Since integration is the inverse of differentiation, we use the following relation for evaluating integrals:

$$\int_{x_0}^{x_n} f(x)\,dx = h\int_0^p f_p\,dp \qquad \ldots (5.2)$$

Integration formulas are used to derive the predictor-corrector formulas for solving differential equations (see Chapter 6).

5.2 DERIVATION OF INTEGRATION FORMULA BASED ON NEWTON'S FORWARD DIFFERENCES

Integrating Newton's forward difference formula (3.2), we get:

$$\int_{x_0}^{x_n} f(x)\,dx = h\int_0^p f_p\,dp$$

$$= h\int_0^p\left[f_0 + p\Delta f_0 + \frac{1}{2}(p^2-p)\Delta f_0 + \frac{1}{6}(p^3-3p^2+2p)\Delta^3 f_0 + \frac{1}{24}(p^4-6p^3+11p^2-6p)\Delta^4 f_0 + \cdots\right]$$

$$= h\left\{pf_0 + \frac{1}{2}p^2\Delta f_0 + \frac{1}{2}\left[\frac{p^3}{3}-\frac{p^2}{2}\right]\Delta^2 f_0 + \frac{1}{6}\left[\frac{p^4}{4}-p^3+\frac{p^2}{2}\right]\Delta^3 f_0 + \frac{1}{24}\left[\frac{p^5}{5}-\frac{6p^4}{4}+\frac{11p^3}{3}-3p^2\right]\Delta^4 f_0 + \cdots\right\} \qquad \ldots (5.3)$$

From (5.3), we can derive several other well-known formulas. For example, imposing the limits (0, 1), we get the formula due to **Laplace**:

$$\int_{x_0}^{x_n} f(x)\,dx = \int_{x_0}^{x_1} f(x)\,dx$$

$$= h\int_0^1 f_p\,dp$$

$$= h\left\{f_0 + \frac{1}{2}\Delta f_0 - \frac{1}{12}\Delta^2 f_0 - \frac{1}{24}\Delta^3 f_0 - \frac{19}{720}\Delta^4 f_0 + \cdots\right\} \qquad \ldots (5.4)$$

5.3 THE NEWTON-COTES FORMULAS

The Newton-Cotes formulas can be derived from the relations (5.3) and (5.4). The following formulas are worth studying:

(a) Trapezoidal rule

(b) Simpson's $\frac{1}{3}$rd rule

(c) Simpson's $\frac{3}{8}$th rule

(d) Boole's rule

(e) Weddle's rule.

The above formulas are simple and some of them are widely used in practice. The use of a particular formula depends upon the nature of the problem to be tackled and also to some extent upon the accuracy desired in the final answers. These rules basically replace f(x) to approximate polynomials, which are then integrated analytically. If the degree of a polynomial is too high, errors due to round-off and local irregularities can cause problems. That is why it is only the lower-degree Newton-Cotes formulas that are often used.

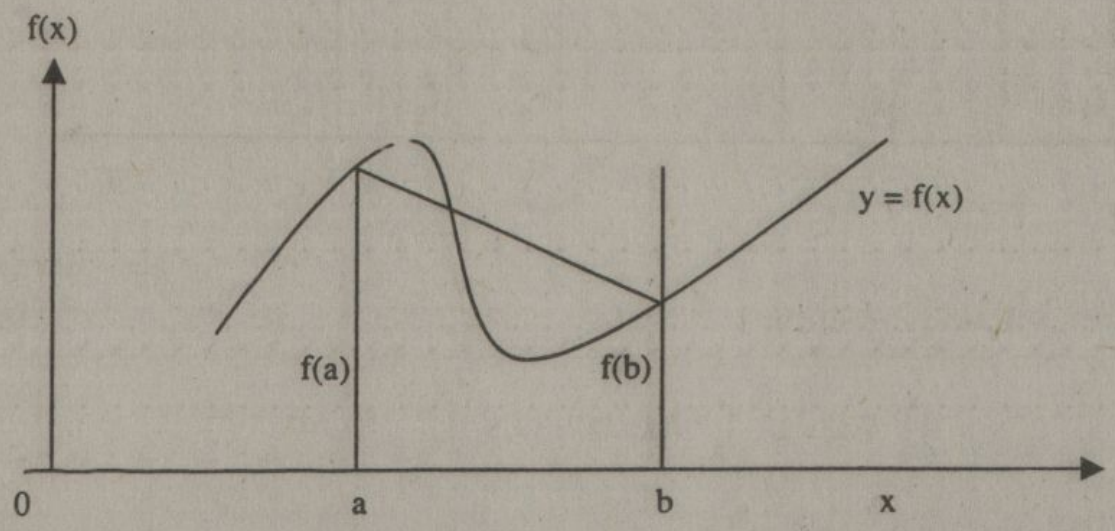
Let us describe the above mentioned formulas one by one.

5.3.1 Trapezoidal Rule

Truncating (5.3) after the first-order differences, we get,

$$I_T = \int_{x_0}^{x_1} f(x)\,dx = h\left[pf_0 + \frac{p^2}{2}\Delta f_0\right]_0^1$$

For fitting a straight line, we use the limits (0, 1), in other words, the integration is over one interval (or two ordinates or two terms):



$$I_T = \int_{x_0}^{x_1} f(x)\,dx$$

$$= h\left[pf_0 + \frac{p^2}{2}\Delta f_0\right]_0^1$$

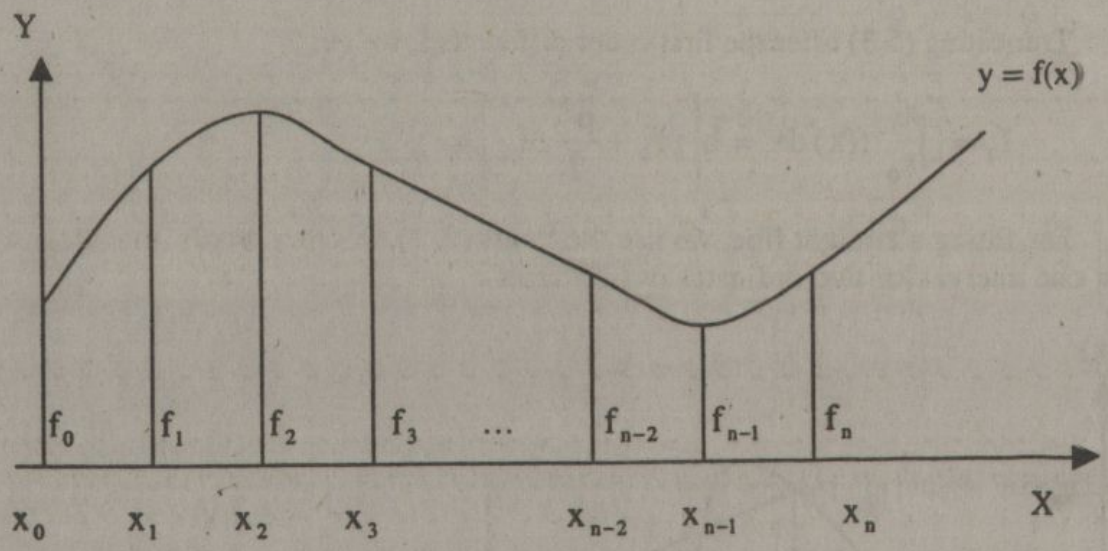
$$= h\left[f_0 + \frac{1}{2}\Delta f_0\right]$$

$$= h\left[f_0 + \frac{1}{2}(f_1 - f_0)\right]$$

$$= \frac{h}{2}[f_0 + f_1] \qquad \ldots (5.5)$$

This is called the **trapezoidal** (or **trapezium**) **rule**. Between x_0 and x_1, the function f(x) is approximated as straight line and the area under the curve representing f(x) is considered to be the area under the straight line.

If n intervals are used, the formula (5.5) is extended as follows to calculate total area between $x_0 = x$ and $x = x_n$.



$$I_T = \int_{x_0}^{x_n} f(x)\,dx$$

$$= \int_{x_0}^{x_1} f(x)\,dx + \int_{x_1}^{x_2} f(x)\,dx + \ldots + \int_{x_{n-1}}^{x_n} f(x)\,dx$$

$$= \frac{h}{2}[f_0 + f_1] + \frac{h}{2}[f_1 + f_2] + \ldots + \frac{h}{2}[f_{n-1} + f_n]$$

$$= \frac{h}{2}[f_0 + 2(f_1 + f_2 + f_3 + \ldots + f_{n-1}) + f_n]$$

$$= \frac{h}{2}\left[(f_0 + f_n) + 2\sum_{i=1}^{n-1} f_i\right] \qquad \ldots (5.6)$$

The above formula is the **trapezoidal rule** for n intervals. It is also called **multiple-segment** or **composite** trapezoidal rule. Note that all functional values except the first and the last are multiplied by 2. The total area under the curve can, therefore, be approximated by the sum of areas of n trapezia.

Trapezoidal rule is not so accurate, but it is simple and moreover can be used for any number of intervals. Approximations to the integrals can be improved to some extent making the step size h, smaller and smaller (in other words, by increasing the number of intervals). One of the most difficult problems in quadrature is to decide how large n should be taken to achieve the desired accuracy. It is sufficient to say at this point that the error tends to be zero as n tends to infinity.

5.3.2 Simpson's $\frac{1}{3}$rd rule

If we truncate the expression in (5.3) after the second-order differences and impose limits (0, 2), we have,

$$I_S = \int_{x_0}^{x_2} f(x)\,dx$$

$$= h\left\{pf_0 + \frac{p^2}{2}\Delta f_0 + \frac{1}{2}\left[\frac{p^3}{3} - \frac{p^2}{2}\right]\Delta^2 f_0\right\}_0^2$$

$$= h\left[2f_0 + 2\Delta f_0 + \frac{1}{3}\Delta^2 f_0\right]$$

$$= h\left[2f_0 + 2(f_1 - f_0) + \frac{1}{3}(f_2 - 2f_1 + f_0)\right]$$

$$= \frac{h}{3}[f_0 + 4f_1 + f_2] \qquad \ldots (5.7)$$

The above relation is called **Simpson's $\frac{1}{3}$rd rule** or simply **Simpson's rule**. If n intervals (should be even in numbers) are to be used, we have the following general expression for composite **Simpson's rule**:

$$I_S = \int_{x_0}^{x_n} f(x)\,dx$$

$$= \int_{x_0}^{x_2} f(x)\,dx + \int_{x_2}^{x_4} f(x)\,dx + \int_{x_4}^{x_6} f(x)\,dx + \ldots + \int_{x_{n-2}}^{x_n} f(x)\,dx$$

$$= \frac{h}{3}[f_0 + 4f_1 + f_2] + \frac{h}{3}[f_2 + 4f_3 + f_4] + \frac{h}{3}[f_4 + 4f_5 + f_6] + \ldots$$

$$+ \frac{h}{3}[f_{n-2} + 4f_{n-1} + f_n]$$

$$= \frac{h}{3}[(f_0 + f_n) + 4(f_1 + f_3 + f_5 + \ldots + f_{n-1}) + 2(f_2 + f_4 + f_6 + \ldots + f_{n-2})]$$

$$= \frac{h}{3}\left[(f_0 + f_n) + 4\sum_{\substack{i=1 \\ i=odd}}^{n-1} f_i + 2\sum_{\substack{i=2 \\ i=even}}^{n-2} f_i\right] \qquad \ldots (5.8)$$

It is obvious from (5.8) that with the exception of the first and the last functional values, all odd functional values are multiplied by 4 and all even functional values are multiplied by 2. The formula is used only when n is even. Simpson's rule gives a more accurate result than the trapezoidal rule and is easier to program and manipulate as well.

5.3.3 Combination of Trapezoidal and Simpson's Rules

Since Simpson's $\frac{1}{3}$rd rule is used when n is even, but if, in some cases, the number of intervals n is odd, we can still find the solution.

For example, we have the following data:

x	f(x)
0	f_0
1	f_1
2	f_2
3	f_3
4	f_4
5	f_5
6	f_6
7	f_7

If we use the values against the points $x = 0$ to $x = 6$ (i.e. $n = 6$) in Simpson's $\frac{1}{3}$rd rule, we get the solution.

$$I_S = \frac{h}{3}[(f_0 + f_6) + 4(f_1 + f_3 + f_5) + 2(f_2 + f_4)]$$

We add to this the result obtained using trapezoidal rule for $x = 6$ to $x = 7$.

$$I_T = \frac{h}{2}[f_6 + f_7]$$

Result = $I_S + I_T$, which is the integral over the entire range.

Consequently, we can also select the first interval to integrate by the Trapezoidal rule and the remainder by Simpson's $\frac{1}{3}$rd rule. However, this criterion seems to work for choosing the end for applying the Trapezoidal rule. There may be a little difference in the two results we obtain but the former is slightly better.

5.3.4 Simpson's $\frac{3}{8}$th Rule

If we truncate the expression in (5.3) after the third-order differences and impose the limits (0, 3), we have,

$$I_{SR} = \int_{x_0}^{x_3} f(x)\,dx$$

$$= \int_0^3 f_p\,dp$$

$$= \left[pf_0 + \frac{1}{2}p^2\Delta f_0 + \frac{1}{2}\left(\frac{p^3}{3} - \frac{p^2}{2}\right)\Delta^2 f_0 + \frac{1}{6}\left(\frac{p^4}{4} - p^3 + p^2\right)\Delta^3 f_0\right]_0^3$$

Simplifying and rearranging terms, we get,

$$I_{SR} = \frac{3h}{8}[f_0 + 3(f_1 + f_2) + f_3] \qquad \ldots (5.9)$$

This is called **Simpson's $\frac{3}{8}$th rule.**

Extending the formula (5.9) upto n intervals, we get,

$$I_{SR} = \frac{3h}{8}[f_0 + 3(f_1 + f_2) + 2f_3 + 3(f_4 + f_5) + 2f_6 + 3(f_7 + f_8) + \ldots$$
$$+ 3(f_{n-2} + f_{n-1}) + f_n] \qquad \ldots (5.10)$$

The above formula does not yield more accurate result than the simple Simpson's rule. One useful application is the calculation of a tabulated function with an odd number of panels by doing the first (or the last) three with the $\frac{3}{8}$th rule and the rest with the $\frac{1}{3}$rd rule. There may be a little difference, although the former is slightly better.

5.3.5 Boole's Rule

If we truncate the expression in (5.3) after fourth-order differences and impose the limits (0, 4), we have,

$$I_B = \int_{x_0}^{x_4} f(x)\,dx$$

$$= \int_0^4 f_p\,dp$$

$$= \frac{2h}{45}\{7f_0 + 32f_1 + 12f_2 + 32f_3 + 7f_4\} \qquad \ldots (5.11)$$

This is called the **Boole's rule**.

5.3.6 Weddle's Rule

If we truncate the expression in (5.3) after sixth differences and impose the limits (0, 6), we have,

$$I_W = \int_{x_0}^{x_6} f(x)\,dx$$

$$= \int_0^6 f_p\,dp$$

$$= \frac{3h}{10}\{f_0 + 5f_1 + f_2 + 6f_3 + f_4 + 5f_5 + f_6\} \qquad \ldots (5.12)$$

This is called the **Weddle's rule**.

In order to illustrate the above methods, we consider the following simple example.

Example 1 The following table represents the values of sine function:

x	0.0	0.1	0.2	0.3	0.4	0.5	0.6
f(x)	0.0000	0.0998	0.1987	0.2955	0.3894	0.4794	0.5646

Compute $\int_0^{0.6} f(x)\,dx$ based on,

(a) Trapezoidal rule, (b) Simpson's $\frac{1}{3}$rd rule, (c) Simpson's $\frac{3}{8}$th rule,

(d) Boole's rule, and (e) Weddle's rule.

Solution As the number of functional values is seven, the number of intervals, n = 6, and h = 0.1.

(a) Trapezoidal Rule

$$I_T = \frac{h}{2}[f_0 + 2(f_1 + f_2 + f_3 + f_4 + f_5) + f_6]$$

$$= \frac{0.1}{2}\{0.0000 + 2(0.0998 + 0.1987 + 0.2955 + 0.3894 + 0.4794) + 0.5646\}$$

$$= \frac{0.1}{2} \times 3.4902 = 0.1745$$

(b) Simpson's $\frac{1}{3}$rd Rule

$$I_S = \frac{h}{3}[f_0 + 4(f_1 + f_3 + f_5) + 2(f_2 + f_4) + f_6]$$

$$= \frac{0.1}{3}\{0.0000 + 4(0.0998 + 0.2955 + 0.4794) + 2(0.1987 + 0.3894) + 0.5646\}$$

$$= \frac{0.1}{3} \times 5.2396 = 0.1747$$

(c) Simpson's $\frac{3}{8}$th Rule

$$I_{SR} = \frac{3h}{8}\{f_0 + 3(f_1 + f_2) + 2f_3 + 3(f_4 + f_5) + f_6\}$$

$$= 3 \times \frac{0.1}{8}\{0.0000 + 3(0.0998 + 0.1987) + 2 \times 0.2955 + 3(0.3894 + 0.4794) + 0.5646\}$$

$$= 3 \times \frac{0.1}{8} \times 4.6575 = 0.1747$$

(d) Boole's Rule

$$I_B = \frac{2h}{45}\{7(f_0 + f_6) + 32(f_1 + f_3 + f_5) + 12(f_2 + f_4)\}$$

$$= 2 \times \frac{0.1}{45}\{7(0.0000 + 0.5646) + 32(0.0998 + 0.2955 + 0.4794) + 12(0.1987 + 0.3894)\}$$

$$= \frac{0.2}{45}[3.9522 + 27.9904 + 7.0572]$$

$$= \frac{0.2}{45} \times 38.9998 = 0.1733$$

(e) Weddle's Rule

$$I_W = 3\times\frac{0.1}{10}[(0.0000+0.5646)+5(0.0998+0.4794)+(0.1987+0.3894)$$
$$+6\times0.2955]$$

$$=\frac{0.3}{10}[0.5646+2.8960+0.5881+1.7730]$$

$$=\frac{0.3}{10}\times5.8217 = 0.1747$$

Example 2 Given the following integral:

$$\int_0^2 \frac{e^{2x}}{1+x^2}$$

Use Simpson's $\frac{1}{3}$rd rule to evaluate the integral with n = 8.

Solution

$$\boxed{\begin{aligned} &n = 8,\quad a = 0, b = 2 \\ &h = \frac{b-a}{n} = \frac{2-0}{8} = 0.25 \end{aligned}}$$

Table of Values:

x	$f = \frac{e^{2x}}{1+x^2}$
$x_0 = 0$	$f_0 = 1.0000$
$x_0 = 0.25$	$f_1 = 1.5500$
$x_1 = 0.50$	$f_2 = 2.1746$
$x_2 = 0.75$	$f_3 = 2.8683$
$x_3 = 1.00$	$f_4 = 3.6945$
$x_4 = 1.25$	$f_5 = 4.7542$
$x_5 = 1.50$	$f_6 = 6.1802$
$x_6 = 1.75$	$f_7 = 12.8132$
$x_7 = 2.00$	$f_8 = 10.9197$

Using Simpson's $\frac{1}{3}$ rd Rule:

$$I_S = \frac{h}{3}[(f_0 + f_8) + 4(f_1 + f_3 + f_5 + f_7) + 2(f_2 + f_4 + f_6)]$$

$$= \frac{0.25}{3}[(0.0000 + 10.9197) + 4(1.5500 + 2.8683 + 4.7542 + 11.8132)$$
$$+ 2(2.1746 + 3.6945 + 6.02)]$$

$$= \frac{0.25}{3}[11.9197 + 83.9428 + 24.0986] = 9.9968$$

5.4 ESTIMATION OF ERRORS IN SOME NEWTON-COTES FORMULAS

In this section, we explain ways of analysing errors in the Trapezoidal and Simpson rules:

Let $F(x) = \int f(x)\,dx$ .... (5.13)

Then $I = \int_{x_0}^{x_0+h} f(x)\,dx$

$$= F(x_0 + h) - F(x_0) \quad \text{.... (5.14)}$$

5.4.1 Error in Trapezoidal Rule

From (5.5), we have,

$$I_T = \frac{h}{2}[f(x_0) + f(x_0 + h)]$$

The error E_T in the Trapezoidal rule can be defined by the following relation:

$$E_T = I - I_T$$

$$= [F(x_0 + h) - F(x_0)] - \frac{h}{2}[f(x_0) + f(x_0 + h)] \quad \text{.... (5.15)}$$

Expanding terms $F(x_0 + h)$ and $f(x_0 + h)$ in (5.15) by Taylor series and setting,

$F'(x_0) = f(x_0)$

$F''(x_0) = f'(x_0)$, etc., we get,

$$E_T = \left[F(x_0)+hF'(x_0)+\frac{h^2}{2!}F''(x_0)+\frac{h^2}{3!}F'''(x_0)+\cdots-F(x_0)\right]$$

$$-\frac{h}{2}\left[f(x_0)+f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\cdots\right]$$

$$= h\left[f(x_0)+\frac{h}{2}f'(x_0)+\frac{h^2}{6}f''(x_0)+\cdots\right]$$

$$-\frac{h}{2}\left[2f(x_0)+hf'(x_0)+\frac{h^2}{2}f''(x_0)+\cdots\right]$$

$$= \frac{-h^3}{12}f''(x_0) \qquad \text{... (5.16)}$$

The above error is the error in a single step and is called the **local error**. When using Trapezoidal rule over n intervals, the error is as follows:

$$E_T = \frac{-nh^3}{12}f''(Z) \qquad \text{... (5.17(a))}$$

$$= \frac{-(b-a)h^2}{12}f''(Z) \qquad \text{... (5.17(b))}$$

where $a \le Z \le b$, and $h = \dfrac{(b-a)}{n}$.

The above error is called the **global error**, which is the total error.

In order to obtain the upper bound, choose Z in (a, b) such that $f''(Z)$ is the largest in magnitude; similarly lower bound can be obtained choosing Z in (a, b) such that $f''(Z)$ is the smallest in magnitude. It follows from (5.17) that the error in the Trapezoidal rule is of the order h^2 and is conventionally written as "error $0(h^2)$". Its significance lies in the fact that as $h \to 0$, the error falls quadratically with h.

5.4.2 Error in Simpson's $\frac{1}{3}$ rd Rule

The error in Simpson's rule is derived in the following manner:

Let $I = \int_a^{a+2h} f(x)\,dx$

$$= F(a+2h) - F(a) \qquad \text{... (5.18)}$$

From (5.7), we have

$$I_s = \frac{h}{3}[f(a) + 4f(a+h) + f(a+2h)]$$

The Error in Simpson's rule can be defined by,

$$E_s = I - I_s$$

$$E_s = [F(a+2h) - F(a)] - \frac{h}{3}[f(a) + 4f(a+h) + f(a+2h)] \qquad \ldots (5.19)$$

Expanding terms f(a + h), f(a + 2h) and F(a + 2h) in (5.19), we get,

$$E_s = \Big[F(a) + 2hF'(a) + \frac{(2h)^2}{2!}F''(a) + \frac{(2h)^3}{3!}F'''(a) + \frac{(2h)^4}{4!}F^{(iv)}(a)$$

$$+ \frac{(2h)^5}{5!}F^{(v)}(a) + \cdots - F(a)\ \Big] - \frac{h^3}{3!}\Big[f(a) + 4\{f(a) + hf'(a)$$

$$+ \frac{h^2}{2!}f''(a) + \frac{h^3}{3!}F'''(a) + \frac{h^4}{4!}F^{(iv)}(a) + \cdots\} + \{f(a) + 2hf'(a)$$

$$+ \frac{(2h)^2}{2!}f''(a) + \frac{(2h)^3}{3!}f'''(a) + \frac{(2h)^4}{4!}f^{(iv)}(a) + \cdots\}\ \Big]$$

$$= \Big[2hF'(a) + (2h)^2F''(a) + \frac{4h^3}{3}F'''(a) + \frac{2h^4}{3}F^{(iv)}(a) + \frac{4}{15}h^5F^{(v)}(a) + \cdots\Big]$$

$$- \frac{h}{3}f(a) - \frac{4h}{3}f(a) - \frac{4h^2}{3}f'(a) - \frac{2h^3}{3}f''(a) - \frac{2h^4}{9}f'''(a) - \frac{h^5}{18}f^{(iv)}(a)\Big]$$

$$- \cdots - \frac{h}{3}f(a) - \frac{2h^2}{3}f'(a) - \frac{2h^3}{3}f''(a) - \frac{4h^4}{9}f'''(a) - \frac{2h^5}{9}f^{(iv)}(a) - \cdots$$

$$\ldots (5.20)$$

Let $F'(a) = f(a)$

$F''(a) = f'(a)$

$F'''(a) = f''(a)$, and so on.

Simplifying (5.20), we get,

$$E_s = 2hf(a) + 2h^2f'(a) + \frac{4h^3}{3}f''(a) + \frac{2h^4}{3}f'''(a) + \frac{4h^5}{15}f^{(iv)}(a) - 2hf(a)$$

$$- 2h^2f'(a) - \frac{4h^3}{3}f''(a) - \frac{2h^4}{3}f'''(a) - \frac{4h^5}{18}f^{(iv)}(a)$$

$$= -\frac{h^5}{90}f^{(iv)}(a)\cdots \qquad \ldots (5.21)$$

The error in (5.21) is called the **local error**. If we integrate over n intervals, we get the **global error** and is as follows,

$$E_s = \frac{-nh^5}{90} f^{(iv)}(Z) \qquad \text{... (5.22(a))}$$

$$E_s = \frac{-(b-a)h^4}{90} f^{(iv)}(Z) \qquad \text{... (5.22(b))}$$

where $a \le Z \le b$.

The error in Simpson's $\frac{1}{3}$rd rule of the order of h^4, i.e., "$0(h^4)$". This is equivalent to saying that for h (small enough), the error is proportional to h^4.

Example 3 Evaluate $\int_1^2 \sqrt{x}\, dx$, using

(a) Trapezoidal and Simpson's $\frac{1}{3}$rd rules, taking h = 0.25 in each case. Write computer programs in each case also.

(b) Calculate exact value to 4 dp. Compare the results obtained in (a) above with the exact value.

(c) Compute the error bounds in each case.

Solution Tabular Values:

x	$f(x) = \sqrt{x}$
1.00	1.0000
1.25	1.1180
1.50	1.2247
1.75	1.3229
2.00	1.4142

(a) Trapezoidal Rule

$$I_T = \int_1^2 \sqrt{x}\, dx$$

$$= \frac{h}{2}[(f_0 + f_4) + 2(f_1 + f_2 + f_3)]$$

$$= \frac{0.25}{2}[(1.0000 + 1.4142) + 2(1.1180 + 1.2247 + 1.3229)]$$

$$= \frac{0.25}{2} \times 9.7454 = 1.2182$$

(b) Simpson's $\frac{1}{3}$rd Rule

$$I_S = \frac{h}{3}\left[(f_0 + f_4) + 4(f_1 + f_3) + 2f_2\right]$$

$$= \frac{0.25}{3}\left[(1.0000 + 1.4142) + 4(1.1180 + 1.3229) + 2 \times 1.2247\right]$$

$$= \frac{0.25}{3}\left[2.4142 + 9.7636 + 2.4494\right]$$

$$= \frac{0.25}{3} \times 14.6272 = 1.2189$$

Computer Program No 7: Trapezoidal Rule

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

float returnval;

float f(float x);
  {
    returnval = 0;
    returnval = sqrt (x);
    cout<<"\n\tX: "<<x<<\t\tf(x) : "<< returnval";
    return returnval;
  }

void main ( )
  {
    float low, up, interval, sum=0, steplen;

    clrscr ( );
    cout<<"\n\tENTER THE LOWER LIMIT :  "; cin>>low;
    cout<<"\n\tENTER THE UPPER LIMIT :  "; cin>>up;
    cout<<"\n\tENTER THE INTERVAL :  "; cin>>interval;
    steplen = (up – low / interval;
    sum = f(low) + f(up);
    cout<<"\n\n\tTHE STEPLENGTH IS :  "; >>steplen;
    cout<<"\n\tTHE SUM IS :  "; "<<sum<<"\n";
    for(int i=1; i< interval; i++)
```

```
    {
      sum += 2 * f(low + i*steplen);
      cout<<"\tSUM : "<<sum;
    }
  sum =(sum*steplen) / 2.0;
  cout<<"\n\n\tFINAL RESULT BY TRAPEZOIDAL RULE IS :  "sum;
}
```

Computer Output

```
ENTER THE LOWER LIMIT  : 1

ENTER THE UPPER LIMIT  : 2

ENTER THE INTERVAL :  4

      X: 1            f(x) : 1
      X: 2            f(x) : 1.414214

THE STEPLENGTH IS :  0.25
THE SUM IS :  2.414214

      X: 1.25         f(x) : 1.118034        SUM : 4.650282
      X: 1.5          f(x) : 1.224745        SUM : 7.099771
      X: 1.75         f(x) : 1.322876        SUM : 9.745522

FINAL RESULT BY TRAPEZOIDAL RULE IS :  1.21819
```

Computer Program No 8: Trapezoidal Rule

Note: This program takes given functional values as input to solve the problem.

```
# include<iostream.h>
# include<conio.h>

void trapezoidal ( );
  {
    clrscr( );
    cout<<"\t\t\t\tTrapezoidal Rule Input\n\n";
    double *x, *fx, interval,start,end;
```

```
int n;
cout<<"Enter total number of elements : ";
cin>>n;
cout<<"Enter interval between the elements : ";
cin>>interval;
cout<<"Enter the first value of x : ";
cin>>start;

x = new double[n];
fx = new double[n];
for(int i=0; i<n; i++)
{
  x[i] = start;
  start = start + interval;
  cout<<"Enter fx "<<i<<": ";
  cin>>fx[i];
}

cout<<"\n\n\n\t\tTable of Values \n\n";
cout<<"\nX                    fx";
for(i=0; i<n; i++)
{
  cout<<"\n"<<x[i]<<"\t\t";
  cout<<fx[i];
}

float h = interval;
double result=0,sum=0;

  if(n>1)
  {
      cout<<"\n\n\nIt = h/2 [(f0+f "<<n-1<<") + 2 (";
      for(i=1;i<n-1;i++)
        {
          cout<<"f"<<i;
          if(i<n-2)
          cout<<"+";
        }
        cout<<")]\n\n"
        result = fx[0] + fx[n-1];

        for(i=1;i<n-1;i++)
          sum = sum + fx[i];
```

```
            sum = sum*2;
            result = result + sum;
            result = h/2 * result;
            cout<<"\nIt = "<<h<<"/2 [("<<fx[0]<<"+"<<fx[n-1]<<") + 2 (";
              for(i=1;i<n-1;i++)
              {
                cout<<fx[i];
                if(i<n-2)
                cout<<"+";
              }
                cout<<")]";
      cout<< "\n\n\nResult is                                    : <<result"
      }
    else
        cout<<"\n\n\t\t\tYou must have more than one elements";
  }
  void main( )
  {
      clscr( );
      trapezoidal ( );
      getch( )
  }
```

Computer Output

Enter total number of elements : 5
Enter interval between the elements : 0.25
Enter first value of x : 1
Enter fx0 : 1
Enter fx1 : 1.1180
Enter fx2 : 1.2247
Enter fx3 : 1.3229
Enter fx4 : 1.4142

Table of Values

X	fx
1	1
1.25	1.1180
1.50	1.2247
1.75	1.3229
2	1.4142

$I_t = h/2\ [(f0+f4) + (f1 + f2 + f3)]$

$I_t = 0.25/2\ [(1 + 1.4142) + 2(1.1180 + 1.2247 + 1.3229)]$

Result is : 1.2182

Program No. 9: Simpson's $\frac{1}{3}$rd Rule

Note: The input functional values are generated using the given function.

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

float returnval;

float f(float x);
  {
    returnval = 0;
    returnval = sqrt (x);
    cout<<"\n\tX: "<<x<<\t\tf(x) : "<< returnval";
    return returnval;
  }

void main ( );
  {
    float low, up, interval, sum=0, steplen, multi=4;

    clrscr ( );
    cout<<"\n\tENTER THE LOWER LIMIT : "; cin>>low;
    cout<<"\n\tENTER THE UPPER LIMIT : "; cin>>up;
    cout<<"\n\tENTER THE INTERVAL : "; cin>>interval;
    steplen = (up – low / interval;
    sum = f(low) + f(up);
    cout<<"\n\n\tTHE STEPLENGTH IS : "; >>steplen;
    cout<<"\n\tTHE SUM IS : "; "<<sum<<"\n";
    for(int i=1; i<interval; i++)
      {
        sum += multy * f(low + i*steplen);
        multy =6 – multi;
        cout<<"\tsum : "<<sum;
      }
    sum =(sum*steplen) / 3.0;
```

```
  cout<<"\n\n\tFINAL RESULT BY SIMPSON'S RULE IS : "sum;
 }
```

Computer Output

```
ENTER THE LOWER LIMIT : 1

ENTER THE UPPER LIMIT : 2

ENTER THE INTERVAL : 4

    X: 1          f(x) : 1
    X: 2          f(x) : 1.414214

THE STEPLENGTH IS : 0.25
THE SUM IS : 2.414214

    X: 1.25       f(x) : 1.118034      SUM : 6.88635
    X: 1.5        f(x) : 1.224745      SUM : 9.335839
    X: 1.75       f(x) : 1.322876      SUM : 14.627342
FINAL RESULT BY SIMPSON'S RULE IS : 1.21819
```

Program No. 9: Simpson's $\frac{1}{3}$rd Rule

Note: The program takes the given functional values as input.

```
# include<iostream.h>
# include<conio.h>

void Simpson ( );
 {
   clrscr( );
   double *x, *fx, interval,start,end;
   int n;
   cout<<"\t\t\t\tInput of Simpson Rule";
   cout<<"\n\n\nEnter total number of elements (should be even) : ";
   cin>>n;
   cout<<"Enter interval between the elements : ";
   cin>>interval;
```

```
cout<<"Enter the first value of x : ";
cin>>start;

x = new double[n];
fx = new double[n];
for(int i=0; i<n; i++)
{
  x[i] = start;
  start = start + interval;
  cout<<"Enter fx "<<i<<": ";
  cin>>fx[i];
}

cout<<"\n\n\t\t\tTable of Values\n\n          ";
cout<<"\nX                          fx";
for(i=0; i<n; i++)
{
  cout<<"\n"<<x[i]<<"\t\t";
  cout<<fx[i];
}

double h = interval;
double result=0,sum=0;
   if(n>1)
   {
       cout<<"\n\nIs = h/3 [(f0+f "<<n-1<<") + 4 ( ";
       for(i=1;i<n-1;i=i+2)
         {
           cout<<"f"<<i;
           if(i<n-3)
           )
           cout<<"+";
         }
         cout<<") + 2( ";
         for(i=2;i<=n-2;i=i+2)
           {
             cout<<"f"<<i;
             if(i<n-3)
                 cout<< "+";
           }
         cout<<")]"
         float sum 1=0;
         result = fx[0] + fx[n-1];
```

```
        for(i=1;i<n-1;i=i+2)
            sum = sum + fx[i];
        sum = sum*4;

        for(i=2;i<=n-2;i=i+2)
            sum1 = sum1 + fx[i];
        sum1 = sum1*2;
        result = result + sum + sum1;
        result = h/3 * result;

        cout<<"\nIs = "<<h<<"/3 [("<<fx[0]<<"+"<<fx[n-1]<<") + 4 (";
          for(i=1;i<n-1;i=i+2)
          {
            cout<<fx[i];
            if(i<n-3)
            cout<<"+";
          }
          cout<<") + 2 ( ";

          for(i=2;;i<=n-2;i=i+2)
          {
            cout<<fx[i];
            if(i<n-3)
            cout<<"+";
          }
          cout<< ")]";

          cout<< "\n\n\nResult is          :          "<<result;
    }
  else

      cout<<"\n\n\n\t\t\tYou must have more than two elements";
}
void main( )
{
    clscr( );
    Simpson( );
    getch( )
}
```

Computer Output

Enter total number of elements : 5

Enter interval between the elements : 0.25

Enter first value of x : 1

Enter fx0 : 1

Enter fx1 : 1.1180

Enter fx2 : 1.2247

Enter fx3 : 1.3229

Enter fx4 : 1.4142

Table of Values

X	fx
1	1
1.25	1.1180
1.50	1.2247
1.75	1.3229
2	1.4142

$I_s = h/3\ [(f0+f4) + 4\ (f1 + f3) + 2\ f2]$

$I_s = 0.25/3\ [(1 + 1.4142) + 4(1.1180 + 1.3229\ ) + 2 * 1.2247]$

Result is : 1.2189

(b) Exact value of the integral:

$$\int_1^2 \sqrt{x}\ dx = 1.2190$$

This value is closer to the one obtained from Simpson's rule.

(c) (i) Error Using Trapezoidal Rule

$$E_T = \frac{-(b-a)h^2}{12} f''(Z)$$

$$f(x) = x^{\frac{1}{2}}$$

$$f'(x) = \frac{-1}{2}x^{\frac{1}{2}}$$

$$f''(x) = \frac{-1}{4}x^{-\frac{3}{2}}$$

$$\max f''(Z) = \frac{-1}{4}(Z)^{-\frac{3}{2}}$$

$$= \frac{-1}{4}(1)^{-\frac{3}{2}} = -0.25$$

$$\min f''(Z) = \frac{-1}{4}(2)^{-\frac{3}{2}} = -0.0884$$

Minimum error is, $E_{min} = \frac{-(2-1)}{12} \times (.25)^2 \times -0.0884$

$$= -\frac{1}{12} \times (.25)^2 \times -0.0884 = 0.0005$$

Maximum error is, $E_{max} = \frac{-(2-1)}{12} \times (.25)^2 \times -0.25$

$$= -\frac{1}{12} \times (.25)^2 \times -0.25 = 0.0013$$

Hence, the error bound is : $0.0005 \le E_T \le 0.0013$

(ii) Error Using Simpson's $\frac{1}{3}$rd Rule

$$f'''(x) = \frac{3}{8}x^{-\frac{3}{2}}$$

$$f^{(iv)}(x) = \frac{-15}{16}x^{-\frac{7}{2}}$$

$$\min f^{(iv)}(Z) = \frac{-15}{16}(1)^{-\frac{7}{2}} = -0.9375$$

$$\max f^{(iv)}(Z) = \frac{-15}{16}(2)^{-\frac{7}{2}} = -0.0829$$

Minimum error is, $E_{min} = \frac{-(b-a)h^4}{12} f^{(iv)}(Z)$

$$= \frac{-(2-1) \times (.25)^4}{90} \times -0.0829 = 0.000004$$

Maximum error is, $E_{max} = \frac{-(2-1)}{90} \times (.25)^4 \times -0.9375 = 0.000041$

Hence, the error bound is : $0.000004 \le E_S \le 0.000041$

5.5 AUTOMATIC SUBDIVISION OF INTERVALS

The most difficult problem in numerical integration lies in choosing the right number of intervals. It may not be sensible to start solving the problem with a large number of intervals and then hope for the best. It will not only result in inconvenience but also in the wastage of computer time. In many cases, accuracy can be improved by the sub-division of intervals rather than by using high-order Newton-Cotes formulas.

In this section, we shall show how to tackle this problem in a systematic and efficient manner. Two methods concerning subdivision of intervals will be discussed which are as follows:

- Repeated use of Trapezoidal rule
- Romberg's integration method

5.5.1 Repeated Use of Trapezoidal Rule

Suppose, we wish to evaluate the integral,

$$I = \int_a^b f(x)\,dx$$

Let I_n be an approximation to I, obtained by using Trapezoidal rule with n intervals. One possible method to decide how large n should be so that I_n approximates I to the desired accuracy would be to evaluate I_1, I_2, I_4, ... until the two successive estimates agree close enough to the desired accuracy. This can be done by halving h and comparing the two results I_n and I_{2n}. We can continue halving h, and calculating I_n, I_{2n}, I_{4n}, ... This will, in theory, converge to the true value I and the process is estimated when two successive estimates agree to the required accuracy, but this would be very labourious.

The procedure is as follows (with $h = b - a$):

i) Let $T_1 = \frac{1}{2}[f(a) + f(b)]$; then $I_1 = hT_1$,

ii) Let $T_2 = T_1 + \left(a + \frac{h}{2}\right)$; then $I_2 = \frac{h}{2} T_2$,

iii) Let $T_4 = T_2 + \left[f\left(a + \frac{h}{2}\right) + f\left(a + \frac{3h}{4}\right)\right]$; then $I_4 = \frac{h}{4} T_4$,

iv) Let $T_8 = T_4 + \left[f\left(a+\frac{h}{8}\right)+f\left(a+\frac{3h}{8}\right)+f\left(a+\frac{5h}{8}\right)+f\left(a+\frac{7h}{8}\right)\right]$; then

$I_8 = \frac{h}{8} T_8$, and so on.

Because of the relatively large error of Trapezoidal rule, it can hardly be considered as an efficient approach. If we look for an accuracy of 0.00000001, the method goes as far as n = 2048, and it takes a lot of computer time to obtain the result.

Example 4 Write a computer program to implement the Trapezoidal rule with automatic interval halving. Use the following test data:

$$\int_1^2 e^{-\frac{x}{2}}\,dx\ ;\ \text{accuracy, } E = 0.000001$$

Computer Program No. 10: Repeated Use of Trapezoidal Rule

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

double f(double x);
    {
      return (exp (-x/2.0));
    }
void main ( )
  {
    double i=0, i1, a, b, e, h, t;
    int k,n=1;
    cout<<"\n\tINTEGRATION USING REPEATED RULE";
    cout<<"\n\n\tENTER THE LOWER LIMIT A : ";
    cin>>a;
    cout<<"\n\tENTER THE UPPER LIMIT B : ";
    cin>>b;
    cout<<"\n\tENTER THE ACCURACY E : ";
    cin>>e;

    h=b - a;
    t = (f(a) + f(b))/2.0;
    i1=t*h/n;
    cout<<"\n\n\tNO. OF INTERVALS\tESTIMATE OF I\n\n";
```

```
  while (fabs*il-i) > e)
   {
     cout<<"\t"<<n<<"\t\t\t"<<i1<<end1;
     n=n*2;
     i=i1;
     for(k=1;k<n;K+=2
       {
         t=t+f(a+k\8h/n);
       }
     i1=t*h/n;
   }
  cout<<"\t"<<n<<"\t\t\t"<<i1<<end1;
  cout<<"\n\tAFTER"<<n<<" INTERVALS VALUE OF INTEGRAL IS :  "<<i1;
  getch ( );
}
```

Computer Output

INTEGRATION USING REPEATED RULE

ENTER THE LOWER LIMIT A : 1

ENTER THE UPPER LIMIT B : 2

ENTER THE ACCURACY E : 0.000001

NO. OF INTERVALS	ESTIMATION OF I
1	0.487205
2	0.479786
4	0.477924
8	0.477485
16	0.477341
32	0.477312
64	0.477305
128	0.477303
256	0.77303

AFTER 256 INTERVALS VALUE OF INTEGRAL IS : 0.77303

5.5.2 Romberg Integration

Although the Trapezoidal rule is the easiest Newton-Cotes formula to apply, it lacks the degree of accuracy generally required. **Romberg Integration** is a method that has wide application because it improves the approximation fairly rapidly. Romberg integration is mostly designed for cases where the function to be integrated is known. This is because knowledge of the function permits the evaluation required for the initial implementations of the Trapezoidal rule.

Let f(x) be known either explicitly or as a tabulation of equispaced data:

x	x_0	x_1	x_2	...	x_n
f(x)	f_0	f_1	f_2	...	f_n

The first step in Romberg's method is to define a series of sums: I_{11}, I_{12}, I_{13}, ..., where

$$I_{11} = \frac{1}{2}(f_0 + f_n);\ h' = \frac{(b-a)}{n},\ \text{where } n = 1.$$

$$I_{12} = \left[I_{11} + f\left(a + \frac{h'}{2}\right)\right]$$

$$I_{13} = \left[I_{12} + f\left(a + \frac{h'}{4}\right) + f\left(a + \frac{3h'}{4}\right)\right]$$

$$I_{14} = \left[I_{13} + f\left(a + \frac{h'}{8}\right) + f\left(a + \frac{3h'}{8}\right) + f\left(a + \frac{5h'}{8}\right) + f\left(a + \frac{7h'}{8}\right)\right]$$

From these sums, various other values T_{11}, T_{12}, T_{13}, ..., are computed using the following relations:

$$T_{11} = h' I_{11}$$

$$T_{12} = \frac{h'}{2} I_{12}$$

$$T_{13} = \frac{h'}{4} I_{13}$$

$$T_{14} = \frac{h'}{8} I_{14}, \text{ and so on.}$$

Note: h is the difference between consecutive values of x, but h' is the difference between the upper and lower limits of the integral.

Romerg's table is as follows:

Integration Sums	Calculation of Approximations
I_{11}	T_{11}
	T_{22}
I_{12}	T_{12} T_{33}
	T_{23} T_{44}
I_{13}	T_{13} T_{34}
	T_{24}
I_{14}	T_{14}

With the values of T_{11}, T_{12}, ..., we compute the first-order Romberg integration as follows:

$$T_{22} = T_{12} + \frac{1}{3}(T_{12} - T_{11})$$

$$T_{23} = T_{13} + \frac{1}{3}(T_{13} - T_{12})$$

$$T_{24} = T_{14} + \frac{1}{3}(T_{14} - T_{13})$$

We now compute the second-order Romberg integration:

$$T_{33} = T_{23} + \frac{1}{15}(T_{23} - T_{22})$$

$$T_{34} = T_{24} + \frac{1}{15}(T_{24} - T_{23})$$

Calculation of third-order Romberg integration:

$$T_{44} = T_{34} + \frac{1}{63}(T_{34} - T_{33}), \text{ etc.}$$

General formula to calculate various values in the table is,

$$T_{j+1,K+1} = T_{j,K+1} + \frac{1}{4^j - 1}\left[T_{j,K+1} + T_{jK}\right] \qquad \ldots (5.37)$$

The procedure continues until the difference between two successive values on the diagonal agree to the desired accuracy. In each column, the bottom number is hopefully the most accurate number. Trapezoidal and Simpson's rules are sometimes inadequate for problem contexts where high efficiency and low errors are needed.

Romberg method is one technique that is designed to obviate these shortcomings. It has been reported in literature that the error in column k of the Romberg table diminishes by about a factor of $\frac{1}{4^{k+1}}$ as one progresses down its rows. The algorithm is clear, although the justification is quite hard.

Finally, one might feel that accuracy of these integration formulas can be increased using higher-order formulas until sufficient accuracy is obtained. However, there are two reasons why this strategy might fail; namely, that the function may not be adequately approximated by a polynomial, in which case the truncation error becomes large, or, that the formulas may be subject to excessive rounding error. One interesting experience about the usage of formulas with an even number of strips is that they not only give zero error for polynomials upto degree n but also for polynomials of degree n + 1. In view of this extra accuracy an even-order formulas would normally be used. However, the exception to this is the Trapezoidal rule, which is valuable because of its simplicity. The other point of significance is that the error depends on a derivative of the function to be integrated.

To summarize there is clearly a major gain in efficiency in using methods which are higher order than the Trapezoidal rule, such as Simpson's rule and especially Romberg integration. All in all, Romberg integration is a powerful but quite simple method, which we recommend for general use. For a given number of intervals, it is much more accurate than the Trapezoidal rule, and quite a bit more accurate than Simpson's rule, but does not need any more function evaluations.

Example 5 (a) Using Romberg integration method, evaluate the integral: $\int_1^{2.6} \frac{dx}{x}$. Let n = 8.

(b) Write a computer program to implement the above procedure.

Solution (a) Tabulated values are as follows:

$$h = \frac{2.6-1}{8} = 0.2$$

Since, $a = 1$, $b = 2.6$ and $n = 8$, the functional values are calculated as below:

x	1.0	1.2	1.4	1.6	1.8	2.0	2.2	2.4	2.6
f(x)	1.000	0.833	0.714	0.625	0.556	0.500	0.455	0.417	0.385

Calculations of I_{11}, I_{12}, I_{13} and I_{14}.

$$I_{11} = \frac{1}{2}(f_0 + f_8)$$

$$= \frac{1}{2}(1.00 + 0.385) = 0.6925$$

$$h' = \frac{(b-a)}{n} = 2.6 - 1 = 1.6 \text{ [since } n = 1]$$

$$T_{11} = h' \, I_{11} = 1.6 \times 0.6925 = 1.1080$$

$$I_{12} = \left[I_{11} + f\left(a + \frac{h'}{2}\right) \right]$$

$$= [I_{11} + f_4] = 0.6925 + 0.556 = 1.2485$$

$$T_{12} = \frac{h'}{2} I_{12} = \frac{1.6}{2} \times 1.2485 = 0.9988$$

$$I_{13} = \left[I_{12} + f\left(a + \frac{h'}{4}\right) + f\left(a + \frac{3h'}{4}\right) \right]$$

$$= [I_{11} + f_2 + f_6]$$

$$= 1.2485 + 0.714 + 9.455 = 2.4175$$

$$T_{13} = \frac{h'}{4} I_{13} = \frac{1.6}{4} \times 2.4175 = 0.9670$$

$$I_{14} = \left[I_{13} + f\left(a + \frac{h'}{8}\right) + f\left(a + \frac{3h'}{8}\right) + f\left(a + \frac{5h'}{8}\right) + f\left(a + \frac{7h'}{8}\right) \right]$$

$$= [I_{13} + f_1 + f_3 + f_5 + f_7]$$

$$= 2.4175 + 0.833 + 0.625 + 0.500 + 0.417 = 4.7925$$

$$T_{14} = \frac{h'}{8} I_{14} = \frac{1.6}{8} \times 4.7925 = 0.9585$$

Calculating other values in the table:

$$T_{22} = T_{12} + \frac{1}{3}(T_{12} - T_{11})$$

$$= 0.9988 + \frac{1}{3}(0.9988 - 1.1080)$$

$$= 0.9988 - 0.0364 = 0.9624$$

$$T_{23} = T_{13} + \frac{1}{3}(T_{13} - T_{12})$$

$$= 0.9670 + \frac{1}{3}(0.9670 - 0.9988)$$

$$= 0.9670 - 0.0106 = 0.9564$$

$$T_{24} = T_{14} + \frac{1}{3}(T_{14} - T_{13})$$

$$= 0.9585 + \frac{1}{3}(0.9585 - 0.9670)$$

$$= 0.9557$$

$$T_{33} = T_{23} + \frac{1}{15}(T_{23} - T_{22})$$

$$= 0.9564 + \frac{1}{15}(0.9564 - 0.9624)$$

$$= 0.9564 - 0.0004 = 0.9560$$

$$T_{34} = T_{24} + \frac{1}{15}(T_{24} - T_{23})$$

$$= 0.9557 + \frac{1}{15}(0.9557 - 0.9564) = 0.9557$$

$$T_{44} = T_{34} + \frac{1}{63}(T_{34} - T_{33})$$

$$= 0.9557 + \frac{1}{63}(0.9557 - 0.9560)$$

$$= 0.9557 - 0.0000 = 0.9557$$

Displaying these values in tabular form, we have,

Interval	Trapezoidal Sums	Romberg Values		
		First-Order	Second-Order	Third-Order
1	1.1080			
		0.9624		
2	0.9988		0.9560	
		0.9564		0.9557
4	0.9670		0.9557	
		0.9557		
8	0.9585			

We note that the final result, 0.9557, is accurate upto 4 dp.

It is often useful to have predetermined a specific value for n and instead modify the algorithm slightly to allow the procedure to continue until a value of n is found that satisfied $|T_{n,n} - T_{n-1,n-1}| < \in$, for a given tolerance $\in$.

Computer Program No. 11: Romberg Integration

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

float f(float x);
    {
      return (1/x);
    }

void main ( )
    {
      float a, b, h, t[12][12]={0}. i=0;
      int j, k, p, no;
      cout<<"\n\tINTEGRATION USING ROMBERG METHOD",
      cout<<"\n\n\tENETER THE LOWER LIMIT A : ";
      cin<< a;
      cout<<"\n\tENETER THE UPPER LIMIT B : ";
      cin<< b;
      cout<<"\n\tENETER N : ";
      cin<< no;
      h = (b–a);
      t[0][0] = (f(a) + f(b))/2;                          //i11
      t[0][1] = h * t[0][0];                              //T11
      k=1; i=0;
      while (k<no)
        {
          i++;
          k=k*2;
          t[i][0] = t[i–1][0];
          for (p=1;p<k;p+=2)
            {
              t[i][0] = t[i][0] + f(a + p * h/k);
            }
          t[i][1] = h/k * t[i][0];
        }
      for(j=2;j<=i+2;j++)
        {
          for(k=0;k<i+2-j;k++)
            {
              t[k][j] = t[k+1][j–1]+(t[k+1][j–1] –t[k][j–1]) / (pow(4j –1) –1);
            }
        }
```

```
    cout<<"\n\n\tTHE BEST ESTIMATE IS :  <<[0][j-2]";
  }
```

Computer Output

```
INTEGRATION USING ROMBERG'S METHOD
ENTER THE LOWER LIMIT A : 1
ENTER THE UPPER LIMIT B : 2.6
ENTER N : 8
THE BEST ESTIMATE IS :  0.955517
```

PROBLEMS

1. (a) The values of a certain function f(x) are given in the following table:

x	0	.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
f(x)	1	1.649	2.718	4.482	7.389	12.18	20.09	33.12	45.60

By using Trapezoidal and Simpson's $\frac{1}{3}$rd rules, compute the integral: $\int_0^4 f(x)\,dx$.

(b) Given the following table:

x	−1	0	1	2	3	4	5	6	7
f(x)	0.9848	1	0.9848	0.9397	0.8660	0.7660	0.6428	0.5000	0.3420

Evaluate $\int_{-1}^{7} f(x)\,dx$ using Trapezoidal and Simpson's $\frac{1}{3}$rd rules.

(c) Use Trapezoidal and Simpson's rules to estimate the numbers of square feet of land in given lots when x and y are measured in feet:

(i)

x	0	10	20	30	40	50	60	70	80	90	100	110	120
y	75	81	84	76	67	68	69	72	68	56	42	44	0

(ii)

x	0	100	200	300	400	500	600	700	800	900	1000
y	125	125	120	112	90	90	95	88	75	35	0

2. (a) Evaluate the integral $e^{\sqrt{x}}$ correct to 3 dp, using (i) Trapezoidal rule and (ii) Simpson's rule from the values given below:

x	0	1	2	3	4
$e^{\sqrt{x}}$	1	2.7185	4.1132	5.6522	7.3891

Using a suitable substitution to evaluate the integral, determine which of these numerical answers is nearer to the exact value.

(b) Compute $\int_0^1 \frac{dx}{2+x^2}$ by Simpson's $\frac{1}{3}$rd rule with n = 8. Evaluate the function analytically and comment on the outcomes in each case.

(c) Evaluate the integral $\int_{-1}^{1} x^2 e^{-x} dx$ using Simpson's $\frac{1}{3}$rd rule with n = 8.

3. Evaluate $\int_0^1 \frac{dx}{1+x}$ using Trapezoidal and Simpson rules. Take n = 8. Compare your results with the exact answer. Compute the error bounds in both cases.

4. (i) The function f(x) is well-defined by the following table and is well-behaved in the given domain:

x	2.03	2.04	2.05	2.06	2.07	2.08	2.09
f(x)	10.13916	10.26167	10.34737	10.45643	10.56905	10.68531	10.80547

a) The value given for f(2.07) is in error by 3×10^{-5}. Find the correct value and show why this is likely to be correct.

b) Compute the integral $\int_{2.03}^{2.09} f(x)\,dx$ from the values given above in 4(i) using Trapezoidal and Simpson rules.

(ii) Evaluate $\int_0^{\frac{\pi}{3}} \sqrt{\sin x}\, dx$ by Simpson's $\frac{1}{3}$rd rule, using 6 intervals.

(iii) Evaluate $\int_0^{\frac{\pi}{2}} \sin dx$ using the Trapezoidal and Simpson's $\frac{1}{3}$rd rules. Find the exact solution and the error involved. Take n = 6.

(iv) A pin moves along a straight guide so that its velocity v(cm/s) when it is a distance x(cm) from the beginning of the guide at time t(s) is given in the table below:

t(s)	0	0.5	1.0	1.5	2.0	2.5	3.0	3.5	4.0
v(cm/s)	0	4.00	7.94	11.68	14.97	17.39	18.25	16.08	0.00

Apply to Simpson's $\frac{1}{3}$rd rule to find the approximate total distance travelled by the pin between $t = 0$ to $t = 4$.

(v) (a) Evaluate the integral:

$$\int_0^3 \frac{x^2}{1+x^3}\,dx$$

Using Trapezoidal and Simpson's $\frac{1}{3}$rd rules, with $n = 10$. Round your answers to 4 dp.

(b) Evaluate this integral mathematically and compare it with the results already obtained. Which is a better solution?

5. (a) Use Simpson's and Trapezoidal rules to evaluate the following integral to an accuracy of 0.0001 with $h = \frac{1}{8}$,

$$I = \int_0^1 \frac{dx}{1+x^2}$$

Evaluate the integral analytically and find the errors in both cases.

(b) Let 10^{-8} be the largest error which can be tolerated when

$$\int_0^1 \log(1+x)\,dx$$

is evaluated by the methods in (a) above. Calculate the number of sub-divisions required to obtain this accuracy in using the Trapezoidal and Simpson's rules.

(c) Let 5×10^{-8} be the largest error which can be tolerated when

$$\int_0^2 x\,e^{-x}\,dx$$

is evaluated by Trapezoidal and Simpson's rules. Calculate the number of sub-divisions required to obtain this accuracy in both methods.

[Hint: $f^{(IV)}(x) = (x-4)\,e^{-x}$]

(d) (i) Use Trapezoidal and Simpson's rules to estimate the integral:

$$\int_0^1 \ln(x^2+1)\,dx \quad \text{with } n = 8.$$

Round your answer to 4 dp.

(ii) Compute the error bound in both cases.

(iii) Let 0.001 be the largest error which can be tolerated when the integral in (i) above is evaluated. Calculate the number of subdivisions required to meet this accuracy in using Trapezoidal and Simpson's rules.

(e) If the composite Trapezoidal rule is to be used to evaluate $\int_0^1 e^{-x^2}\,dx$ with an error of at most $\frac{1}{2}\times 10^{-4}$, how many points should be required?

6. (a) The table below represents a function $y = f(x)$:

x	0	1	2	3	4
y	1.000	1.027	1.110	1.255	1.485

The distance of the centroid $\mathring{x}$ from the axis Oy is given by the equation:

$$\mathring{x} = \frac{\int_0^4 xy\,dx}{\int_0^4 y\,dx}$$

Find $\mathring{x}$ to 3 dp using Simpson's $\frac{1}{3}$rd rule to perform the necessary integration.

(b) The root mean square (RMS) value of a function $y = f(x)$ in the range $x = a$ to $x = b$ is given by the expression:

$$\text{RMS} = \sqrt{\int_a^b \frac{y^2}{b-a}\,dx}$$

Using Simpson's $\frac{1}{3}$rd rule with 9 ordinates to evaluate the RMS value of the function:

$f(x) = (1+x)^{\frac{3}{2}}$ in the range $x = 1$ to $x = 3$:

x	1.00	1.25	1.50	1.75	2.00	2.25	2.50	2.75	3.00
y	2.8284	3.3750	3.9928	4.5804	5.1962	5.8590	6.5479	7.2818	8.0000

(c) A solid of revolution is formed by rotating about the x-axis; the area between the x-axis, the lines $x = 0$ and $x = 1$, and a curve through the points with the following ordinates:

x	0.00	0.25	0.50	0.75	1.00
y	2.8284	0.9826	0.9589	0.9089	0.8415

Estimate the volume, $V = \pi \int_0^1 y^2\,dx$ using Simpson's $\frac{1}{3}$rd rule and giving the answer to 3 dp.

(d) The arc length of the curve $y = f(x)$ over the interval $a \le x \le b$ is given by the following formula:

$$\text{Length} = \int_a^b \sqrt{1+[f'(x)]^2}\, dx$$

Find the arc length for $f(x) = 0.1\, x(30 - x)$, for $0 \le x \le 30$, using Trapezoidal and Simpson rules with $n = 12$.

(e) The length of the curve given by $y = f(x)$, $a < x < b$ is

$$I = \int_a^b \sqrt{1+[f'(x)]^2}\, dx$$

Calculate the length of the parabolic arc: $y^2 = 4x;\ 2 < x < 4$

using Simpson's $\frac{1}{3}$rd rule. with $n = 8$.

(f) The solid of revolution obtained by rotating the region under the curve $y = f(x)$, $a \le x \le b$, about the x-axis has surface area given by,

$$\text{Area} = a\pi \int_a^b f(x)\sqrt{1+[f'(x)]^2}\, dx$$

Find the area of the function $f(x) = x^3$, $0 \le x \le 1$, using Trapezoidal and Simpson rules. Take $n = 10$.

7. (a) Given the following function tabulated at evenly-spaced intervals:

x	0	1	2	3	4	5	6	7	8	9
f(x)	0	0.5687	0.7909	0.5743	0.1350	–0.1852	–0.1802	0.0811	0.2917	0.3031

Evaluate $\int_0^9 f(x)\, dx$ using some suitable methods.

(b) Given the following data at equally-spaced intervals:

x	0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0	1.1
y	93	87	68	55	42	37	35	39	48	53	51	39

The data are felt to be relatively error-free. Evaluate the integral as accurately as possible.

8. (a) Evaluate $\int_0^1 e^{x^2} dx$ to 6 dp, using Romberg's method. Take $n = 8$.

(b) Evaluate $\int_0^1 (1+x^2)^{-1}\,dx$ to 6 dp, using Romberg's method. Take n = 8.

(c) Calculate $\int_0^8 f(x)\,dx$, for the following table using Romberg's method:

x	0.0	0.1	0.2	0.3	0.4	0.5
f(x)	1.000000	0.990050	0.960789	0.913831	0.852144	0.778801

0.6	0.7	0.8
0.697676	0.612626	0.527292

9. (a) Calculate $\int_1^2 \frac{dx}{\sqrt{x}}$ using Romberg's method. Taking n = 8.

(b) Calculate $\int_1^{1.5} e^{-x^2}\,dx$, for the following table using Romberg's method:

x	1	1.125	1.250	1.375	1.5
f(x)	0.3678794	0.2820629	0.2096113	0.1509774	0.1053992

10. Solve the following integrals using Romberg's integration method correct to 4 dp:

a) $\int_1^2 \frac{\sqrt{1-e^{-x}}}{x}\,dx$; Take n = 8

b) $\int_1^2 \frac{dx}{\sqrt{e^x+x-1}}$; Take n = 8

c) $\int_0^3 xe^{2x}\,dx$; Take n = 8

d) $\int_0^{0.8} e^{-x^2}\,dx$; Take n = 8

e) $\int_1^2 \ln x \cos x\,dx$; Take n = 8

11. Evaluate the following integrals, correct to 6 dp, using Romberg's integration method. Compare these results with their exact answers. What can you say about this comparison?

a) $\int_0^{\frac{\pi}{4}} \sec\,dx$; Take n = 8

b) $\int_0^{\frac{\pi}{2}} \sin x \, dx$; Take n = 8

c) $\int_{-1}^{2} \left(x^7 + 2x^3 - 1\right) dx$; n = 8

12. Evaluate the following integrals, correct to 8 dp, using Romberg's integration method.

a) $\int_0^3 \frac{\sin 2x}{1+x^2} dx$; with n = 4.

b) $\int_0^2 \sqrt{4-x^2} \, dx$; with n = 8.

c) $\int_0^2 \sqrt{4-x^2} \, dx$; with n = 4.

d) $\int_0^2 \frac{1}{x^2+0.1} dx$; with n = 4.

e) $\int_1^{\pi} x \cos 3x \, dx$; Take n = 6.

13.(a) Find approximation of

$$\int_0^1 e^{-x^2} dx$$

using the Trapezoidal rule with (i) one (ii) two and (iii) four panels of equal width.

(b) Find approximation of

$$\int_0^1 x^4 dx$$

using (i) the Trapezoidal rule,

(ii) Simpson's rule

in each case with two, four and eight panels of equal width. By using the exact result, calculate and compare each of the errors and check with theoretical order of convergence as $h \to 0$.

(c) (i) It is required to use the Trapezium rule to compute

$$\int_0^{\frac{\pi}{2}} \sin^2 x \, dx$$

to 4 dp accuracy. Use the error bound formula to recommend a panel size 4.

(ii) Find the Trapezium rule to integrate with $h = \pi/4$ and compare with the exact value. Does this result contradict your part (i) result.

Chapter 6

Ordinary Differential Equations

6.1 INTRODUCTION

A **differential equation** is an equation involving functions and their derivatives. Thus, an equation of the form,

$$y' = \frac{dy}{dx} = f(x, y) \qquad \ldots (6.1)$$

subject to an initial condition that $y = y_0$ at $x = x_0$ is called a **differential equation**. Here, y is called the **dependent variable** and x is the **independent variable**. A **solution** of a differential equation is a relationship between the dependent and independent variables that satisfies the original differential equation. For example, $y = 3x^2 + x$ is the solution of $y' = 6x + 1$.

Since most physical laws of biology, business, chemistry, ecology, economics, etc., are expressed in terms of differential equations, the need to solve such equations occurs quite often. The predator-prey problem has become a classic example of differential equations. In this chapter, we shall describe several methods to solve differential equations.

6.1.1 Classification of Differential Equations

If there is only one independent variable x, the equation is called an **ordinary differential equation** (abbreviated as ODE). The equation,

$$y' = 3x^2 + \sin x; \text{ subject to } y(0) = 2 \text{ is an ODE.}$$

If more than one independent variable exists, the derivatives must be partial and the equation is called the **partial differential equation** (abbreviated as PDE). The equation,

$$\frac{\delta^2 y}{\delta^2 t} = c\frac{\delta^2 y}{\delta^2 x}$$

is a PDE with two independent variables x and t and with dependent variable y. This book deals only with solutions of ODEs.

6.1.2 Categories of ODEs

(a) Due to Order

The ODEs can be categorised according to their **order**. The order of an ODE is defined as an integer equal to the maximum number of times the dependent variable is differentiated. Thus, $y' = x + y^2$ is of order 1, since the highest derivative that appears is of first order, whereas $x(y'')^3 + y' + x = 0$, is of order 2, despite the power 3 on the second derivative. Restricting ourselves for the time being to the case in which there is only one dependent variable y, the most general nth-order ODE can be written:

$$F(x, y, y', y'', \ldots, y^{(n)}) = 0 \qquad \ldots (6.2)$$

The nth-order differential equation (6.2) can be replaced by an equivalent system of n first-order equations as below:

$$y_1 = y; \quad y_2 = y', \quad y_3 = y''; \ldots; y_n = y^{(n-1)}$$

We will first focus our discussion on the solution of first-order differential equations.

(b) Linear and Non-linear ODEs

The equation (6.2) is said to be **linear**, if F is a linear function of the variables y, y', ..., $y^{(n)}$, in a linear ODE–terms such as x^2, x^4, y, etc., may be present. If it contains terms such as yy', $y'y''$, etc., the ODE is called **non-linear**. The following ODEs are non-linear:

$$yy'' = x + 1;$$

$$y^3 y'' = 2(xy - 1);$$

$$y''' = 2y^3 + x, \text{ etc.}$$

(c) Boundary Conditions

We may also classify the problems in differential equations, according to the nature of boundary conditions. If all the required values are given simply at one point, the mathematical problem is called an **initial value problem** (also called **starting problem** or **marching problem** as the solution is advanced in steps).

An initial value problem consists of two parts, the differential equation $y'(x) = f(x, y)$, which gives the relationship between y(x) and $y'(x)$ and the initial condition $y(x_0) = y_0$.

If the conditions are given at two or more distinct points, the problem is known as a **boundary value** (or sometimes **jury**) **problem**. The solutions of initial-value problems are easily obtained by using direct methods, whereas the solutions of boundary-value problems are, in general, determined iteratively.

A typical boundary-value problem might be of the form,

$$\frac{d^2y}{dx^2}+\frac{dy}{dx}+cy=h(x);$$

$$y(0)=y_0,\ y(L)=y_1$$

This problem could describe the steady state temperature distribution in a one-dimensional heat transfer problem with temperatures y_0 at $x = 0$ and y_1 at $x = L$. These are called the boundary conditions whether or not the points $x = 0$ and $x = L$ represent actual typical boundaries. The problem is a boundary value problem if conditions are specified at two or more different values of the independent variable, thus,

$$\frac{d^4y}{dx^4}+ay=f(x);$$

$$y(0)=y_0;\ \left(\frac{dy}{dx}\right)_0=w_0;\ \left(\frac{d^2y}{dx^2}\right)_0=v_0;\ y(L)=y_1$$

is a boundary-value problem. In this book, we will discuss only the initial-value problems. This does not imply that initial-value problems are more important or more frequently encountered in practice than boundary value problems.

Different methods need to be used to solve boundary-value problems, for example, the shooting method, multiple shooting or global methods like finite differences or collocation.

6.2 METHODS TO SOLVE ODEs

There are a variety of methods to solve ODEs, for instance,

a) Analytical methods

b) Graphical methods

c) Numerical methods

If a problem can be solved **analytically**, it is usually considered to be the most accurate solution. Most of the ODEs encountered in practice either cannot be solved by analytical methods or they have too laborious analytical solutions, because of large number of integrals involved. Therefore, we look for some other methods to obtain approximations of the solutions.

The **graphical methods** may give very useful insight into the nature of the solutions of ODEs, but they suffer from several serious disadvantages, a few of them are as follows:

i) Accuracy is limited by the draughtsman's technique;

ii) The judgment is subjective;

iii) The error is difficult to compute.

In the absence of any elementary methods, we attempt to solve ODEs numerically.

6.3 NUMERICAL METHOD TO SOLVE ODEs

By a numerical method for solving the initial value problem (6.1) is meant a procedure for finding approximate values, $y_0, y_1, y_2, \ldots, y_n$, of the exact y(x) at the points:

$$x_0, x_1, x_2, \ldots, x_n$$

The first step is to estimate y_1, from the initial conditions, and $y_0' = f(x_0, y_0)$ from (6.1). After finding y_1, we determine y_2 and so on. In general, methods that require only a knowledge of y_n to determine y_{n+1} are called **starting** (or **single-step**) **methods**. On the other hand, we make use of [illegible] at more than one previous points, say, $y_n, y_{n-1}, y_{n-2}, \ldots$ to determine y_{n+1}; such m[illegible]hods are called **continuing** (or **multi-step**) **methods**.

The success in using an appropriate numerical-method depends primarily on the skill and insight of the practitioner. One should be very careful in using numerical methods, because they can have inherent difficulties of their own. In the first place, there is the question of convergence, i.e., as the difference h between the successive points, $x_0, x_1, x_2, \ldots, x_n$, approaches zero, do the values of the numerical solutions, $y_1, y_2, \ldots, y_n$, approach the exact solution of the differential equation?

This is generally not a practical concern, since all standard numerical techniques are convergent, when applied to virtually any differential equation. This does not mean that in practice, the numerical solution will always approach the exact solution to the differential equation as $h \to 0$, since round-off error will inevitably be present in any real computation.

There is also a serious question of estimating error, which arises generally from the following causes:

i) The formulas used in numerical methods are only approximate formulas, which introduce truncation error (also called **discretization error**).

ii) It is possible to carry only a limited number of digits in any computation, which gives rise to **rounding error**.

iii) Any error that an approximate scheme introduces at an early step will be carried along in the computation process till later steps. It is due to the **propagation error**.

Type of Numerical Methods

Since the numerical solution of ODEs is of considerable importance to many fields of science and engineering, this topic has always received much attention. Many methods have been developed for the solution of such equations.

Numerical methods, generally, fall into the following three classes:

a) A class of methods which produces expressions for y, in terms of functions of x, from which values of y, can be obtained by direct substitution. Under this, the following methods will be described:

- Picard's method
- Taylor series method

b) Another class of methods finds the numerical values of the change in y, due to a given increment in x. Under this, the following methods will be described:

- Euler's method and its variations
- Runge-Kutta methods

c) **Predictor-Corrector Methods**

They make use of two formulas; one is called a **predictor**, which first predicts a value for y_{n+1}; and then the second is called a **corrector** to improve upon y_{n+1}. Under this, the following methods will be discussed:

- Milne-Simpson method
- Adams-Bashforth method
- Adams-Moulton method

Let us begin our discussion of the above methods one by one.

6.4 PICARD'S METHOD

The ODE (6.1) can be rewritten as:

$$y = y_0 + \int_{x_0}^{x} f(x, y)\, dx \qquad \ldots (6.3)$$

A solution is obtained in the form of a power series in x to represent y, over a given range of values of x. Thus, the numerical values of y can be generated by direct substitution of the desired values of x.

For the first approximation, we substitute the initial values of y in f(x, y) and denote it by $y^{(0)} = y_0$. Then, we have,

$$y^{(1)} = y_0 + \int_{x_0}^{x_0+h} f(x, y^{(0)})\, dx$$

If we replace the value of $y^{(0)}$ by $y^{(1)}$ in the above relation, we get .the second approximation,

$$y^{(2)} = y_0 + \int_{x_0}^{x_1+h} f(x, y^{(1)})\, dx$$

For the third approximation, we have,

$$y^{(3)} = y_0 + \int_{x_0}^{x_2+h} f(x, y^{(2)})\,dx$$

$$\vdots$$

In general, $y^{(n)} = y_0 + \int_{x_0}^{x_{n-1}+h} f(x, y^{(n-1)})\,dx$ ... (6.4)

The process is continued till $|y^{(n-1)} - y^{(n)}|$ is less than or equal to the pre-assigned accuracy or till the required number of approximations is reached. To get a reasonable accuracy, it is advisable to use more than four approximations. This method is sometimes also called the **successive approximation method**.

Picard's method is of considerable theoretical interest for solving ODEs. It is also helpful sometimes to generate starting values, which are required in the use of predicator-corrector methods. It is successfully employed only when f(x, y) can be integrated, but breaks down when further integration becomes difficult to perform. We thus look for some other methods. Moreover, Picard's method does not have much practical value for computer solution.

The following example illustrates the practical details of Picard's method.

Example 1 (a) Use Picard's method to solve $y' = x + y^2$, subject to the initial condition

$$y = y_0 \text{ at } x_0 = 0.$$

(b) Tabulate the values of y corresponding to x = 0(0.1)0.5 correct to 5 dp.

(c) Determine roughly over what range this solution will hold to 5 dp if terminated at x^8.

Solution (a) Picard's Method

Initial Approximation

$$y^{(0)} = y_0 = 0.$$

First Approximation

$$y^{(1)} = y_0 + \int_{x_0}^{x} f(x, y^{(0)})\,dx$$

$$= 0 + \int_{x_0}^{x} \left[x + (y^{(0)})^2\right]dx$$

$$= 0 + \int_{x_0}^{x} [x + 0]\,dx = \frac{x^2}{2}$$

Second Approximation

$$y^{(2)} = y_0 + \int_{x_0}^{x} f(x, y^{(1)})\,dx$$

$$= 0 + \int_{x_0}^{x} \left[x + \left(\frac{x^2}{2} \right)^2 \right] dx$$

$$= 0 + \int_{x_0}^{x} \left[x + \frac{x^4}{2} \right] dx = \frac{x^2}{2} + \frac{x^5}{20}$$

Second Approximation

$$y^{(3)} = y_0 + \int_{x_0}^{x} f(x, y^{(2)})\,dx$$

$$= 0 + \int_{x_0}^{x} \left[x + \left(\frac{x^2}{2} + \frac{x^5}{20} \right)^2 \right] dx$$

$$= 0 + \int_{x_0}^{x} \left[x + \frac{x^4}{4} + \frac{x^7}{20} + \frac{x^{10}}{400} \right] dx$$

$$= \frac{x^2}{2} + \frac{x^5}{20} + \frac{x^6}{160} + \frac{x^{11}}{4400} \qquad \ldots (6.5)$$

(b) Determination of Values for y

We rewrite the expression (6.5) in the nested polynomial form, which is faster and more efficient computationally:

$$y = \frac{x^2}{2}\left(1 + \frac{x^3}{10}\left(1 + \frac{x^3}{4}\left(\frac{1}{2} + \frac{x^3}{55}\right)\right)\right) \qquad \ldots (6.6)$$

Substituting the values of x = 0(0.1)0.5 in (6.6), we get,

x	0	0.1	0.2	0.3	0.4	0.5
y	0.0000	0.0050	0.02002	0.04512	0.08052	0.12659

(c) Truncating (6.5) at x^8 gives rise to the following error:

$$\frac{x^{11}}{4400} \le \frac{1}{2} \times 10^{-5}$$

$$x^{11} \le \frac{1}{2} \times 10^{-5} \times 4400$$

$$x \le \left(\frac{1}{2} \times 10^{-5} \times 4400\right)^{\frac{1}{11}} = 0.71$$

So, the range of values for x : $0 \le x \le 0.71$.

6.5 TAYLOR SERIES METHOD

The Taylor series method is of great general applicability and it is the standard to which we compare the accuracy of various other numerical methods for solving an initial value problem. It can be devised to have any specified degree of accuracy.

We attempt to develop the relation between y and x, by finding the coefficients of the Taylor series in which we expand y about the point $x = x_0$:

$$y = y_0 + (x - x_0) y_0' + \frac{(x - x_0)^2}{2!} y_0'' + \frac{(x - x_0)^3}{3!} y_0''' + \cdots + \frac{(x - x_0)^n}{n!} y_0^{(n)} \quad \ldots (6.7)$$

where $y_0', y_0'', \ldots, y_0^{(n)}$ are derivatives, and can be calculated from (6.1)

$$y' = f(x, y) \quad \ldots (6.8)$$

$$y_0' = f(x_0, y_0)$$

Differentiating (6.8) partially gives

$$y'' = \frac{\delta}{\delta x} f(x, y) + \frac{\delta}{\delta y} f(x, y) \cdot y'$$

$$= \frac{\delta}{\delta x} f(x, y) + \frac{\delta}{\delta y} f(x, y) \cdot f(x, y)$$

$$= f_1(x, y)$$

$$y_0'' = f_1(x_0, y_0)$$

Similarly,

$$y''' = \frac{\delta}{\delta x} f_1(x, y) + \frac{\delta}{\delta y} f_1(x, y) \cdot y'$$

$$= \frac{\delta}{\delta x} f_1(x, y) + \frac{\delta}{\delta y} f_1(x, y) \cdot f(x, y)$$

$$= f_2(x, y)$$

$$y_0''' = f_2(x_0, y_0)$$

Substituting the expressions for y_0', y_0'', y_0''', ... and $x - x_0 = h$ in (6.7), we rewrite the series as:

$$y = y_0 + hf(x_0, y_0) + \frac{h^2}{2!}f_1(x_0, y_0) + \frac{h^3}{3!}f_2(x_0, y_0) + \cdots + \frac{h^n}{n!}f_n(x_0, y_0) \quad \ldots (6.9)$$

An upper bound for the approximate error in the Taylor series is given by:

$$E = \frac{h^{n+1}}{(n+1)!}y^{n+1}(Z); \text{ where } x_0 \le Z \le x_n.$$

The Taylor series method (like Picard's method) can be easily applied to a higher order equation. For example, if we are given: $y'' = x^2 + y^3$; $y(0) = 1$, $y'(0) = -2$.

We can find the derivative-terms in the Taylor series as follows:

- $y(0)$ and $y'(0)$ are given by initial conditions.
- $y''(0)$ comes from substitution into the differential equation from $y(0)$ and $y'(0)$.
- $y'''(0)$ and higher derivatives are found by differentiating the equation, for the previous order of derivatives and substituting previously computed values.

Taylor series method can be very effective, but its main disadvantages lie in the calculation of higher-order derivatives, which are sometimes complex and difficult to compute. This method breaks down if it is not possible to compute derivatives any further. The Taylor series method is often used to provide starting values of y required in the predictor-corrector methods. This method is not very amenable to computer solution. However, it is normal to use a series only to find a few values of x, near x_0 and then to continue the solution by one of the step-by-step methods given later.

Example 2 (a) Find an expression for y including first six derivatives in the series, given that $y' = 0.1(x^3 + y^2)$, subject to the initial condition $y(0) = 1$.

(b) Determine roughly over what range this solution will hold to 4 dp if terminated at x^5.

(c) Terminate the series at x^4 and then evaluate the series for $x = 0(0.2)1.0$.

Solution (a) Given $y' = 0.1(x^3 + y^2)1$; $y_0 = 1$, $x_0 = 0$

$$y_0' = f(x_0 + y_0) = 0.1(0 + 1) = 0.1$$

Differentiating the given ODE with respect to x, we have,

$$y'' = 0.1(3x^2 + 2yy')$$

$$y_0'' = f_1(x_0, y_0)$$
$$= 0.1(3x_0^2 + 2y_0 y_0')$$
$$= 0.1(3 \times 0 + 2 \times 1 \times 0.1) = 0.02$$
$$y''' = 0.1(6x + 2yy'' + 2(y')^2)$$
$$y_0''' = 0.1(6x_0 + 2y_0 y_0'' + 2(y_0')^2)$$
$$= 0.1(6 \times 0 + 2 \times 1 \times 0.02 + 2(0.1)^2)$$
$$= 0.1(0 + 0.04 + 0.2) = 0.006$$

Similarly,

$$y_0^{(iv)} = 0.6024$$
$$y_0^{(v)} = 0.12120$$
$$y_0^{(vi)} = 0.08472$$

Also, $h = x - x_0 = x$.

Substituting respective values in the Taylor series (6.9), we get,

$$y = y_0 + hy_0' + \frac{h^2}{2}y_0'' + \frac{h^3}{6}y_0''' + \frac{h^4}{24}y_0^{(iv)} + \frac{h^5}{120}y_0^{(v)} + \frac{h^6}{720}y_0^{(vi)}$$
$$= 1.0 + 0.1x + \frac{0.02}{2}x^2 + \frac{.006}{6}x^3 + \frac{.6024}{24}x^4 + \frac{.1212}{120}x^5 + \frac{.8472}{720}x^6$$
$$= 1.0 + 0.1x + 0.01x^2 + 0.001x^3 + 0.0251x^4 + 0.00101x^5 + 0.00012x^6 \quad \ldots (6.10)$$

(b) If the series (6.10) is terminated at x^5, x must be small enough, so that

$$0.00012x^6 \le \frac{1}{2} \times 10^{-4}$$
$$x^6 \le \frac{1}{2} \times 10^{-4} \times \frac{1}{.00012} = 0.41667$$
$$x \le (0.41667)^{\frac{1}{6}} = 0.86$$

Thus, the fifth degree polynomial represented by the first six terms of (6.10) gives y correct to 4 dp over the range: $0 \le x \le 0.86$.

(c) To tabulate y when x = 0(0.2)1.0, we use the truncated series in the nested polynomial form:

$$y = 1 + 0.1x + 0.01x^2 + 0.001x^3 + 0.0251x^4$$
$$= 1 + x\,(0.1 + x\,(0.01 + x\,(0.001 + 0.0251\,x)))$$

Thus, the values of y are calculated below:

x	0.0	0.2	0.4	0.6	0.8	1.0
y	1.0000	1.0204	1.0423	1.0671	1.0972	1.1361

6.6 EULER'S METHOD AND ITS VARIATIONS

There are many ways to derive Euler's method, but the simplest is by using Taylor series. If we truncate the expression (6.9) after the first derivative term, we get

$$y_{n+1} = y_n + h y_n'$$

$$= y_n + hf(x_n, y_n); \text{ where } h = x - x_0. \qquad \ldots (6.11)$$

This is called **Euler's method**. It works iteratively and does not require the computation of higher-order derivatives.

The maximum truncation error per step in Euler's method is given by the following relation:

$$E = \frac{h^2}{2} y''(Z); \quad x_0 \le Z \le x_0 + h \text{ for some } Z. \qquad \ldots (6.12)$$

This shows that the **local truncation error** in Euler's method is $0(h^2)$, i.e., proportional to the square of the step-size. This implies that halving the step-size will reduce the truncation at each step by a factor of 4. **Global error** at any point in the computation is the difference between the computed value of the solution and the exact solution. Thus, the global error accounts for the total accumulation of error from the start of the computational process.

Euler's method is so inaccurate that it is virtually rarely used in practice. However, because of its simplicity, it is convenient to use it as an introduction to numerical techniques for solving ODEs.

To reduce the inherent error in the simple Euler's method, two variations are used:

The first variation,

$$y_{n+1} = y_n + hf\left[x_n + \frac{h}{2}, y_n + \frac{h}{2} f(x_n, y_n)\right] \qquad \ldots (6.13)$$

is called the **modified Euler's method.**

The second variation,

$$y_{n+1} = y_n + \frac{h}{2}[f(x_n, y_n) + f(x_n + h, y_n + hf(x_n, y_n))] \qquad \ldots (6.14)$$

is called the **improved Euler's method** (also called **Heun's method**). Both methods give a definite improvement in accuracy, and are special cases of the second-order Runge-Kutta method to be studied in section 6.7. The local truncation error for the improved Euler's method is proportional to the cube of the step-size. A major drawback of Euler's methods as mentioned earlier is that the orders of accuracy are low. This

disadvantage is two-fold: to maintain a high-accuracy, very small h is essential. It increases computational-time and causes round-off errors.

Example 3 Solve the differential equation, $y' = x + y$, subject to the initial condition $y(0) = 1$, on the interval [0, 0.5], using Euler's method. Write a computer program to implement the above method. Also estimate the global error if the exact solution of the differential equation is $y = 2e^x - (x + 1)$. Take $h = 0.1$.

Solution Assume $k_1 = f(x_n, y_n) = x_n + y_n$

Given: $x_0 = 0$, $y_0 = 1$, and $x = 0(0.1)0.5$

n	x_n	y_n	$k_1 = x_n + y_n$	$y_{n+1} = y_n + hk_1$
0	0	1.0	$0 + 1.0 = 1.0$	$1 + 0.1 \times 1.0 = 1.1$
1	.1	1.1	$.1 + 1.1 = 1.2$	$1.1 + 0.1 \times 1.2 = 1.22$
2	.2	1.22	1.42	1.362
3	.3	1.362	1.662	1.5282
4	.4	1.5282	1.9282	1.72102
5	.5	1.72102	...	

$Y(0.5) = 2e^{.5} - (0.5 + 1) = 1.79744$ and $y(0.5) := 1.77102$

Global error $= |Y(0.5) - y(0.5)| \le |1.79744 - 1.72102| = 0.0764$

Program No. 12: Euler's Method

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

float f(float x, float y)
  {
    return (x +y);
  }

void main ( )
  {
    float x, y, xup, h, n, ynew;

    cout<<"\n\tSIMPLE EULER'S METHOD";
```

```
    cout<<"\n\n\tENTER VALUE OF X        : " ; cin >>x ;
    cout<<"\n\tENTER VALUE OF Y  : "; cin >> y ;
    cout<<"\n\tENTER UPPER LIMIT OF X : "; cin >> xup;
    cout<<"\n\tENTER THE INTERVAL      : "; cin >> h ;
    n = (xup-x) / h;
    cout<<"\n\tX\tYn\t\tY(n+1)";
    cout<<"\n\t-----------------------------\n";
    for(int i=0;i<=n;i++)
    {
        ynew = y + h * f(x,y);
        cout<<"\n\t"<<x<<"\t"<<y<<"\t\t"<<ynew;
        y = ynew;
        x = x+h;
    }
}
```

Computer Output

```
SIMPLE EULER'S METHOD
ENTER THE VALUE OF X    : 0.0
ENTER THE VALUE OF Y    : 1.0
ENTER UPPER LIMIT OF X  : 0.5
ENTER THE INTERVAL      : 0.1
```

X	Yn	Y(n+1)
0.0	1.0	1.1
0.1	1.1	1.22
0.2	1.22	1.362
0.3	1.362	1.5282
0.4	1.5282	1.72102
0.5	1.72102	1.943122

7 RUNGE-KUTTA METHODS

The Runge-Kutta methods are a family of methods derived from the Taylor series ethod. The rigorous development of these methods is beyond the scope of this

elementary book, because their derivations involve complicated algebraic manipulations and may be found in only a few advanced texts on the subject. See the bibliography given at the end of this book.

In this section, we shall discuss the second-order Runge-Kutta method to demonstrate the essential ideas.

Let us write the Taylor series with first three terms:

$$y_{r+1} = y_r + hy'_r + \frac{h^2}{2!}y''_r + 0(h^3) \qquad \ldots (6.15)$$

$$= y_r + h f_r + \frac{h^2}{2!}\left(\frac{\delta f_r}{\delta x} + f_r \frac{\delta f_r}{\delta y}\right) \qquad \ldots (6.16)$$

Since, $y'_r = \left[\frac{\delta y}{\delta x}\right]_r = (f(x, y))_r = f_r$

$$y''_r = \left[\frac{d}{dx} f(x, y)\right]_r = \frac{\delta f_r}{\delta x} + f_r \frac{\delta f_r}{\delta y}$$

Let us now define the two parameters k_1 and k_2 as follows:

$$k_1 = hf(x_r, y_r) = hf_r \qquad \ldots (6.17)$$

$$k_2 = hf(x_r + \alpha h, y_r + \beta k_1)$$

and form

$$y_{r+1} = y_r + w_1 k_1 + w_2 k_2 \qquad \ldots (6.18)$$

Two values of α are worth mentioning because setting $\alpha = \frac{1}{2}$ gives the modified Euler's method and $\alpha = 1$ gives the improved Euler's method. Expanding (6.17) and substituting in (6.18) gives the following relation:

$$y_{r+1} = y_r + (w_1 + w_2) hf_r + w_2 h^2 \left[\alpha \frac{\delta f_r}{\delta x} + \beta f_r \frac{\delta f_r}{\delta y}\right] \qquad \ldots (6.19)$$

Comparing the coefficients of powers of h in (6.16) and (6.19), we get,

$$w_1 + w_2 = 1$$

$$w_1 \alpha = \frac{1}{2} \qquad \ldots (6.20)$$

$$w_2 \beta = \frac{1}{2} \qquad \ldots (6.20)$$

There are three equations with four unknowns. With these choices, we observe that $\beta = \alpha$, $w_2 = \frac{1}{2}\alpha$. and $w_1 = 1 - \frac{1}{2}\alpha$.. Substituting the values of w_1 and w_2 in (6.18), we get,

$$y_{r+1} = y_r + \left(1 - \frac{1}{2}\alpha\right) k_1 + \frac{1}{2}\alpha k_2 \qquad \ldots (6.21)$$

If we set $\alpha = 1$, we obtain $w_1 = w_2 = \frac{1}{2}$.

Substituting these values in (6.18) or (6.21), we get one of the several second-order Runge-Kutta formulas of the form:

$$y_{r+1} = y_r + \frac{1}{2}(k_1 + k_2) \qquad \ldots (6.22)$$

where $k_1 = hf(x_r, y_r)$, and $k_2 = hf\left[x_r + \frac{h}{2}, y_r + \frac{k_1}{2}\right]$.

Other higher-order Runge-Kutta methods can be developed in much the same way as the second-order Runge-Kutta method. However, we shall mention here some of the higher-order Runge-Kutta methods.

A third-order Runge-Kutta method is given below:

$$y_{r+1} = y_n + \frac{1}{6}(k_1 + 4k_2 + k_3) \qquad \ldots (6.23)$$

where $k_1 = hf(x_n, y_n)$

$$k_2 = hf\left[x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right]$$

$$k_3 = hf\left(x_n + h, y_n - k_1 + 2k_2\right)$$

Since the term containing h^4 in the Tayler series is ignored, the error is said to be of order h^4. Another variation of the third-order Runge-Kutta method is as follows:

$$y_{r+1} = y_n + \frac{1}{9}(2k_1 + 3k_2 + 4k_3) \qquad \ldots (6.24)$$

where $k_1 = hf(x_n, y_n)$

$$k_2 = hf\left[x_n + \frac{h}{2}, y_n + \frac{k_1}{2}\right]$$

$$k_3 = hf\left[x_n + \frac{3h}{4}, y_n + \frac{3k_2}{4}.\right]$$

None of the above methods are widely used. A well-known fourth-order Runge-Kutta method often referred to as the classic formula is as follows:

$$y_{r+1} = y_n + \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4) \qquad \ldots (6.25)$$

where $k_1 = hf(x_n, y_n)$

$$k_2 = hf\left[x_n + \frac{h}{2},\ y_n + \frac{k_1}{2}\right]$$

$$k_3 = hf\left[x_n + \frac{h}{2},\ y_n + \frac{k_2}{4}\right]$$

$$k_4 = hf(x_n + h,\ y_n + k_3)$$

In this method, the per-step-error is of the order h^5. The classic Runge-Kutta method is the most popular. It is a good choice for common purpose because it is quite accurate, stable, self-starting and easy to computerise. The main disadvantages are the requirement that the function f(x, y) has to be evaluated for different values of x and y in every step. This repeated evaluation of functional-values takes much computer time as compared to other methods of comparable accuracy. Moreover, local error estimates are somewhat difficult to obtain.

Most authorities proclaim that it is not necessary to use a higher-order method, because the increased accuracy is offset by extra computational effort. If more accuracy is desired, then a smaller step-size is recommended. It has been suggested in literature that in using the fourth-order method, the step-size used may be based upon the relationship:

$$\left|\frac{k_2 - k_3}{k_1 - k_2}\right|$$

When this quantity exceeds a few hundreds for a given h, then h should be decreased so as to obtain a better result with less truncation error. In general, the step-size can be large when the solution is slowly varying but should be small when rapidly varying. However, it has been reported in literature that if the step-size in this method is reduced by a factor of $\frac{1}{2}$, we can expect that the overall final global error will be reduced by a factor of $\frac{1}{16}$.

All Runge-Kutta methods can be shown to be convergent, i.e.,

$$\lim_{h \to 0}(y_i - y(x_i)) = 0.$$

Another criterion for selecting an algorithm for the solution of a differential equation with given initial conditions is **instability**. Stability is a somewhat ambiguous term and appears in the literature with a variety of qualifying adjectives (inherent, partial, relative, weak, strong, absolute, etc.). In general, a solution is said to be **unstable** if errors introduced at some stage in the calculations (for example, from erroneous initial

conditions or local truncation or round-off errors) are propagated without bound throughout subsequent calculations.

Example 4 Given the ODE, $y' = \dfrac{y-x}{y+x}$, subject to initial condition y(0) = 1. Use the classic Runge-Kutta method to solve the problem in the range $0 \le x \le 0.5$, taking h = 0.1. Write also a computer program to implement this method.

Solution Given $x_0 = 0,\ y_0 = 1$ and $h = 0.1$

First approximation

$$k_1 = hf(x_0, y_0) = h\left[\frac{y_0 - x_0}{y_0 + x_0}\right]$$

$$= 0.1\frac{(1-0)}{(1+0)} = 0.1$$

$$k_2 = hf\left(x_0 + \frac{h}{2},\ y_0 + \frac{k_1}{2}\right) = hf\,(0.5, 1.05)$$

$$= 0.1\frac{(1.05-0.05)}{(1.05+0.05} = \frac{0.1\times 1}{1.1} = 0.0909$$

$$k_3 = hf\left(x_0 + \frac{h}{2},\ y_0 + \frac{k_2}{2}\right) = hf\,(0.5, 1.0455)$$

$$= 0.1\frac{(1.0455-0.05)}{(1.0455+0.05} = \frac{0.1\times 9955}{1.0955} = 0.0909$$

$$k_4 = hf\,(x_0 + h,\ y_0 + k_3) = hf\,(0.1, 1.0909)$$

$$= 0.1\frac{(1.0909-0.1)}{(1.0909+0.1} = 0.0832$$

$$k = \frac{1}{6}\{k_1 + 2(k_2 + k_3) + k_4\}$$

$$= \frac{1}{6}\{0.1 + 2(0.0909 + 0.0909) + 0.0832\} = 0.0911$$

$$y_1 = y_0 + k$$

$$= 1.0 + 0.0911 = 1.0911$$

Other values $h = 0.1,\ x_{n+1} = x_n + h$

n	x_n	y_n	k_1	k_2	k_3	k_4	$y_{n+1} = y_n + h$
0	0	1.0	1.0	0.0909	0.0909	0.0832	$1.0911 = y_1 = y_0 + k$
1	.1	1.0911	0.0832	0.0766	0.0766	0.0708	1.1678
2	.2	1.1678	0.0708	0.0656	0.0655	0.0609	1.2335
3	.3	1.2335	0.0609	0.0566	0.0566	0.0527	1.2902
4	.4	1.2902	0.0527	0.0491	0.0490	0.0456	1.3393
5	.5	1.3393	...	...	...	...	...

Computer Program No 13: Runge-Kutta Method

```
# include<iostream.h>
# include<conio.h>
# include<math.h>

float function(float x0, float y0)
  {
    float result;
    result=(y0-x0)/(y0+x0);
    return results;
  }

void main(void)
  {
    float k1,k2,k3,k4,k,h,x0,y0,yn;
    int n, i, row, col;
    clrscr( );

    cout<<"\n\tCLASSIC RUNGE-KUTTA METHOD";
    cout<<"\n\tENTER THE VALUE OF X0: ";
    cin>>x0;
    cout<<"\n\tENTER THE VALUE OF Y0: ";
    cin>>y0;
    cout<<"\n\tENTER THE VALUE OF h: ";
    cin>>h;
    cout<<"\n\tENTER THE VALUE OF n: ";
    cin>>n;
    cout<<"\nn  xn yn  k1  k2  k3  k4  y(n+1)=y(n)+k";
    cout<<"\n-----------------------------------------------";
```

```
row=12;
col=0;
for(i=0;i<n+1; i++)
  {
    k1=h*function(x0,y0);
    k2=h*function(x0+h/2,y0+k1/2);
    k3=h*function(x0+h/2,y0+k2/2);
    k4=h*function(x0+h,y0+k3);
    k=(k1+2*k2+2*k3+k4)/6;
    yn=y0+k;
    gotoxy(col, row);
    cout<<i;
    gotoxy(col+4, row);
    cout<<x0;
    gotoxy(col+8, row);
    cout<<y0;
    gotoxy(col+17, row);
    cout<<k1;
    gotoxy(col+26,row);
    cout<<k2;
    gotoxy(col+35,row);
    cout<<k3;
    gotoxy(col+44, row);
    cout<<k4;
    gotoxy(col+56, row);
    cout<<yn;

    y0+=k;
    x0+=h;
    row+=2;
  }
}
```

Computer Output

CLASSIC RUNGE-KUTTA METHOD

ENTER THE VALUE OF X0 : 0

ENTER THE VALUE OF Y0 : 1

ENTER THE VALUE OF h : 0.1

ENTER THE VALUE OF n : 5

n	Xn	Yn	K1	K2	K3	K4	Y(n+1)=Y(n)+k
0	0	1	0.1	0.90909	0.090871	0.083206	1.091128
1	0.1	1.091128	0.083209	0.076612	0.076552	0.070753	1.167843
2	0.2	1.167843	0.070757	0.065594	0.065532	0.060871	1.233490
3	0.3	1.23349	0.060874	0.056628	0.05657	0.052664	1.290145
4	0.4	1.290145	0.052667	0.049051	0.048999	0.045627	1.339211
5	0.5	1.339211	0.045629	0.042469	0.042422	0.039444	1.381681

6.8 PREDICTOR-CORRECTOR METHODS

The methods discussed in the previous sections are called **single-step methods** because they use only the information from one previous point to compute successive point that is only the initial point (x_0, y_0) is used to compute (x_1, y_1) and in general y_n is required to compute y_{n+1}.

The predictor-corrector methods, which are also called **multi-step methods**, are not self-starting. They require four initial points, (x_0, y_0), (x_1, y_1), (x_2, y_2) and (x_3, y_3), in order to generate the point (x_4, y_4).

The basic principle behind the multi-step method is to utilize past-values of y to construct a polynomial that approximates the derivatives of the function f(x, y) and to extrapolate this into the next interval. The degree of the polynomial depends on the number of past points concerned. If we use two past points, the approximating polynomial will be of fist-order. If we use three past points, the approximating polynomial will be a quadratic and if we use four past points, the approximating polynomial will be cubic. The more points we use, the higher is the order of the approximating polynomial and the better is the accuracy.

Suppose that integration has already progressed some way and that a table is formed giving the values of y as: y_0, y_1, y_2, y_3, ..., and thus the corresponding derivatives are:

x_0	y_0	f_0
x_1	y_1	f_1
x_2	y_2	f_2
x_3	y_3	f_3
x_4	y_4	f_4

In order to compute y_4, the following two types of formulas are used:

i) A **predictor formula** which is used to predict (determine an estimate) the value y_4 in terms of the values of y's and f's already computed. This formula is used once in an iteration.

ii) A **corrector formula** which is used to find y_4 in terms of the values of y's already known, together with the newly predicted value. This formula is repeated as many times as necessary to obtain the required level of accuracy, i.e., until the two successive corrected values in an iteration are same or in agreement to the required number of decimal places.

In these methods, the accuracy is controlled by the corrector formula, whereas the predictor formula simply helps to provide an initial approximation. Both formulas usually depend on values of the function already obtained for prior points. There are several predictor-corrector methods but we shall discuss those methods, which are easy to develop and are commonly used. They are as follows:

- Milne-Simpson method
- Adams-Bashforth method and its special cases
- Adams-Moulton (or Modified Adams) method

Advantages of the predictor-corrector methods

These methods are widely used for solving ODEs because of the following reasons:

a) They are faster computationally.

b) The difference between the predictor and corrector values provides a measure of the error being made at each step and hence can be used to control the step-size employed in the integration.

c) Only one or perhaps two evaluations of the derivatives need to be computed at each step (as compared with the four for the classic Runge-Kutta method) and on higher-order systems this can save considerable computing effort.

Disadvantages of the predictor-corrector methods

Some disadvantages are that these methods are complex to program and are not self-starting. The three main sources of trouble in these methods for integrating ODEs are as follows:

a) Truncation errors that arise from the finite approximations for the derivatives.

b) Propagation errors (instability) that arise from solutions of the approximate difference equations that do not correspond to solutions of the differential equations.

c) Amplification of round-off errors due to certain combinations of coefficients in finite difference formulas.

Let us now derive the above mentioned predictor-corrector formulas.

6.8.1 Milne-Simpson Predictor-Corrector Method

a) Derivation of Predictor Formula due to Milne

We derive this formula using Newton's forward difference interpolation formula (3.2) neglecting differences beyond third-order and integration f_p between the limits (0, 4):

$$f_p = f_0 + p\Delta f_0 + \frac{1}{2}(p^2 - p)\Delta^2 f_0 + \frac{1}{6}(p^3 - 3p^2 + 2p)\Delta^3 f_0 + \ldots, \quad \ldots (6.26)$$

Hence,

$$\int_{x_0}^{x_4} f(x, y)\, dx = h\int_0^4 f_p\, dp$$

$$= h\int_0^4 \left[f_0 + p\Delta f_0 + \frac{1}{2}(p^2 - p)\Delta^2 f_0 + \frac{1}{6}(p^3 - 3p^2 + 2p)\Delta^3 f_0\right] dp$$

$$= h\left[pf_0 + \frac{p^2}{2}\Delta f_0 + \frac{1}{2}\left(\frac{p^3}{3} - \frac{p^2}{2}\right)\Delta^2 f_0 + \frac{1}{6}\left(\frac{p^4}{4} - p^3 + p^2\right)\Delta^3 f_0\right]_0^4$$

$$= h\left[4f_0 + 8\Delta f_0 + \frac{20}{3}\Delta^2 f_0 + \frac{8}{3}\Delta^3 f_0\right]$$

$$= \frac{4h}{3}\left[3f_0 + 6\Delta f_0 + 5\Delta^2 f_0 + 2\Delta^3 f_0\right] \quad \ldots (6.27)$$

We now express the above relation into simple functions.

Since, $\Delta f_0 = f_1 - f_0$

$$\Delta^2 f_0 = f_2 - 2f_1 + f_0$$

$$\Delta^3 f_0 = f_3 - 3f_2 + 3f_1 - f_0$$

Substituting these functional values in (6.27), we get,

$$\int_{x_0}^{x_4} f(x, y)\, dx = \frac{4h}{3}\left[3f_0 + 6(f_1 - f_0) + 5(f_2 - 2f_1 + f_0) + 2(f_3 - 3f_2 + 3f_1 - f_0)\right]$$

$$= \frac{4h}{3}\left[2f_1 - f_2 + 2f_3\right]$$

Thus, Milne's predictor formula is as follows:

$$y_4 = y_0 + \int_{x_0}^{x_4} f(x, y)\, dx$$

$$= y_0 + \frac{4h}{3}\left[2f_1 - f_2 + 2f_3\right] \quad \ldots (6.28)$$

More generally,

$$y_{n+1} = y'_{n-3} + \frac{4h}{3}[2f_{n-2} - f_{n-1} + 2f_n] \qquad \ldots (6.29)$$

for $n = 0, 1, 2, \ldots$

b) Derivation of Corrector Formula due to Simpson

Simpson's corrector formula can be derived by integrating (6.26) between the limits (0, 2):

$$\int_{x_0}^{x_2} f(x, y)\,dx = h\int_0^2 f_p\,dp$$

$$= h\left[pf_0 + \frac{p^2}{2}\Delta f_0 + \frac{1}{2}\left(\frac{p^3}{3} - \frac{p^2}{2}\right)\Delta^2 f_0 + \frac{1}{6}\left(\frac{p^4}{4} - p^3 + p^2\right)\Delta^3 f_0\right]_0^2$$

$$= h\left[2f_0 + 2\Delta f_0 + \frac{1}{3}\Delta^2 f_0\right]$$

$$= h\left[2f_0 + 2(f_1 - f_0) + \frac{1}{3}(f_2 - 2f_1 + f_0)\right]$$

$$= \frac{h}{3}[f_0 + 4f_1 + f_2]$$

Hence, Simpson's rule which is used as a corrector formula is as follows:

$$y_2 = y_0 + \int_{x_0}^{x_2} f(x, y)\,dx$$

$$= y_0 + \frac{h}{3}[f_0 + 4f_1 + f_2]$$

or, $$y_4 = y_2 + \frac{h}{3}[f_2 + 4f_3 + f_4] \qquad \ldots (6.30)$$

More generally,

$$y_{n+1} = y_{n-1} + \frac{h}{3}[2f_{n-1} + 4f_n + f_{n+1}] \qquad \ldots (6.31)$$

for $n = 0, 1, 2, \ldots$.

The need for a corrector formula arises because the predictor alone is numerically unstable; it gives spurious solutions growing exponentially. Milne's predictor uses four previous values of y, hence extra starting formulae are needed to find y_1, y_2 and y_3 when y_0 is given. The starting problem is a weakness of predictor-corrector methods in general; nevertheless they are serious competitors to Runge-Kutta methods.

Starting Values

Since the predictor-corrector formula is not self-starting, we require three additional values of y to start the process. While solving numerical examples manually or on the computer, it is advisable to repeat (or iterate) the corrector formula more than once. In this way, we can get a more accurate value of y. We may iterate to improve the value obtained from the corrector or step-size may be reduced to obtain a more accurate value with one or more applications of the corrector. However, the predictor formula is used only once.

If starting values are not given, they can be computed by either of the ways:

(i) Using Picard or Taylor series method to generate a series expansion and then using this series to compute y_1, y_2, and y_3; or

(ii) Using a self-starting method like Euler's or Runge-Kutta methods. As already mentioned, the most widely used starting methods are those due to Runge-Kutta.

Stopping Criteria

In a predictor-corrector method, the stopping of iterations may be controlled either by comparing the difference between two successive values of y's to some pre-assigned accuracy or by pre-determining the number of iterations or combining both of them.

Truncation error due to predictor formula,

$$E = \frac{28}{90} h^5 y^{(v)}(Z); \text{ where } x_{n-3} < Z < x_{n+1}. \qquad \ldots (6.29(a))$$

Truncation error due to predictor formula,

$$E = \frac{1}{90} h^5 y^{(v)}(Z); \text{ where } x_{n-3} < Z < x_{n+1}. \qquad \ldots (6.31(a))$$

Example 5 Use of classic Runge-Kutta method to solve the differential equation, $y' = x - y$ with initial values (0, 1), gives the following tabular values:

x	0.0	0.1	0.2	0.3
y	1.0000	0.9097	0.8375	0.7816

Using Milne-Simpson predictor-corrector formula, find y(.4) correct to 4 dp. If $y = 2e^{-x} + x - 1$ is the analytical solution of the equation, what can you say about your result?

Solution The values of x, y and f are given below:

x	y	$f = x - y$
0	$1.0000 = y_0$	$-1.0000 = f_0$
.1	$0.9097 = y_1$	$-0.8097 = f_1$
.2	$0.8375 = y_2$	$-0.6375 = f_2$
.3	$0.7816 = y_3$	$-0.4816 = f_3$
.4	$0.7407 = y_4$	$-0.3407 = f_4$

Using predictor formula (6.28), we get,

$$y(.4) = y_0 + \frac{4h}{3}[2f_1 - f_2 + 2f_3]$$

$$= 1.0000 + \frac{4\times.1}{3}(2x - 0.8097 + 0.6375 + 2x - 0.4816)$$

$$= 1.0000 + \frac{0.4}{3}(-1.6194 + 0.6375 - 0.9632)$$

$$= 1.0000 - 0.2593 = 0.7407$$

Using predictor formula (6.30), we get,

$$y(.4) = y_2 + \frac{h}{3}[f_2 + 4f_3 + f_4]$$

$$= 0.8375 + \frac{0.1}{3}(-0.6375 + 4x - 0.4816 - 0.3407)$$

$$= 0.8375 + \frac{0.1}{3} \times -0.9046 = 0.7407$$

Both the predicted and corrected values agree to 4 dp.

Exact answer, $y(.4) = 2e^{-4} + .4 - 1 = 0.7406$.

Obviously, the exact answer and the numerical result both agree to 3 dp.

6.8.2 Adams-Bashforth Predictor-Corrector Method

a) Derivation of Predictor Formula due to Adams

This formula is derived using Newton's backward difference formula. Integrating (3.3) and using limits (0, 1), we get,

$$\int_{x_0}^{x_1} f(x, y)\,dx = h\int_0^1 f_p\,dp$$

$$= h\int_0^1\left[f_0 + p\nabla f_0 + \frac{1}{2}(p^2 + p)\nabla^2 f_0 + \frac{1}{6}(p^3 + 3p^2 + 2p)\nabla^3 f_0 + \frac{1}{24}(p^4 + 24p^3 + 11p^2 + 6p)\nabla^4 f_0 + \cdots\right]dp$$

$$= h\left[pf_0 + \frac{p^2}{2}\nabla f_0 + \frac{1}{2}\left(\frac{p^3}{3} + \frac{p^2}{2}\right)\nabla^2 f_0 + \frac{1}{6}\left(\frac{p^4}{4} + p^3 + p^2\right)\nabla^3 f_0 + \frac{1}{24}\left(\frac{p^5}{5} + 6p^4 + \frac{11}{3}p^3 + 3p^2\right)\nabla^4 f_0 + \cdots\right]_0^1$$

$$= h\left[f_0 + \frac{1}{2}\nabla f_0 + \frac{5}{12}\nabla^2 f_0 + \frac{3}{8}\nabla^3 f_0 + \frac{251}{720}\nabla^4 f_0 + \cdots\right]$$

Predictor formula is as follows:

$$y_1 = y_0 + \int_{x_0}^{x_1} f(x, y)\,dx$$

$$y_1 = y_0 + h\left[f_0 + \frac{1}{2}\nabla f_0 + \frac{5}{12}\nabla^2 f_0 + \frac{3}{8}\nabla^3 f_0 + \frac{251}{720}\nabla^4 f_0 + \cdots\right] \quad \ldots (6.32)$$

More generally,

$$y_{n+1} = y_n + h\left[f_n + \frac{1}{2}\nabla f_n + \frac{5}{12}\nabla^2 f_n + \frac{3}{8}\nabla^3 f_n + \frac{251}{720}\nabla^4 f_n + \cdots\right] \quad \ldots (6.33)$$

for $n = 0, 1, 2, \ldots.$

b) Derivation of Corrector Formula due to Bashforth

The corrector formula is derived as follows:

$$f_p = E^{p-1} f_i$$

$$= (1-\nabla)^{-(p-1)} - f_i$$

Expanding by Binomial Theorem, we get,

$$= \left[1 + (p-1)\nabla + \frac{1}{2}p(p-1)\nabla^2 + \frac{1}{6}p(p-1)(p+1)\nabla^3 + \frac{1}{24}p(p-1)(p+1)(p+2)\nabla^4\right] f_1$$

$$= f_1 + (p-1)\nabla f_1 + \frac{1}{2}(p^2 - p)\nabla^2 f_1 + \frac{1}{6}(p^3 - p)\nabla^3 f_1 + \frac{1}{24}(p^4 - 2p^3 - p^2 - p)\nabla^4 f_1 + \cdots$$

As before,

$$\int_{x_0}^{x_1} f(x, y)\,dx = h\int_0^1 f_p\,dp$$

$$= h\int_0^1 \left[f_1 + (p-1)\nabla f_1 + \frac{1}{2}(p^2 - p)\nabla^2 f_1 + \frac{1}{6}(p^3 - p)\nabla^3 f_1 + \frac{1}{24}(p^4 - 2p^3 - p^2 - 2p)\nabla^4 f_1 + \cdots\right] dp$$

$$= h\left[pf_1 + \left(\frac{p^2}{2} - p\right)\nabla f_1 + \frac{1}{2}\left(\frac{p^3}{3} - \frac{p^2}{2}\right)\nabla^2 f_1 + \frac{1}{6}\left(\frac{p^4}{4} - \frac{p^2}{2}\right)\nabla^3 f_1 + \frac{1}{24}\left(\frac{p^5}{5} + \frac{p^4}{4} - \frac{p^3}{3} - p^2\right)\nabla^4 f_0 + \cdots\right]_0^1$$

$$= h\left[f_1 - \frac{1}{2}\nabla f_1 - \frac{1}{12}\nabla^2 f_1 - \frac{1}{24}\nabla^3 f_1 - \frac{19}{720}\nabla^4 f_1 - \cdots\right] \quad \ldots (6.34)$$

Corrector formula is as follows:

$$y_1 = y_0 + \int_{x_0}^{x_1} f(x, y)\, dx$$

$$= y_0 + h\left[f_1 - \frac{1}{2}\nabla f_1 - \frac{1}{12}\nabla^2 f_1 - \frac{1}{24}\nabla^3 f_1 - \frac{19}{720}\nabla^4 f_1 + \cdots\right] \quad \ldots (6.35)$$

More generally,

$$y_{n+1} = y_n + h\left[f_{n+1} - \frac{1}{2}\nabla f_{n+1} - \frac{1}{12}\nabla^2 f_{n+1} - \frac{1}{24}\nabla^3 f_{n+1} - \frac{19}{720}\nabla^4 f_{n+1} - \cdots\right] \quad \ldots (6.36)$$

for $n = 0, 1, 2, \ldots$.

Truncation error for the predictor formula,

$$E = \frac{251}{720} h^5 y^{(v)}(Z); \quad x_n \le Z \le x_{n+1}$$

Truncation error for the predictor formula,

$$E = -\frac{19}{720} h^5 y^{(v)}(Z); \qquad x_n \le Z \le x_{n+1}$$

c) Special Case

Truncating predictor formula (6.32) after ∇f_0, we get,

$$\int_{x_0}^{x_1} f(x, y)\, dx = h\left[f_0 + \frac{1}{2}\nabla f_0\right]$$

$$= h\left[f_0 + \frac{1}{2}(f_0 - f_{-1})\right]$$

$$= \frac{h}{2}[3f_0 - f_{-1}]$$

$$y_1 = y_0 + \frac{h}{2}[3f_0 - f_{-1}]$$

More generally,

$$y_{n+1} = y_n + \frac{h}{2}[3f_n - f_{n-1}] \qquad \ldots(6.37)$$

for $n = 0, 1, 2, \ldots.$

Truncating corrector formula (6.35) after ∇f_1, we get,

$$\int_{x_0}^{x_1} f(x, y)\,dx = h\left[f_1 - \frac{1}{2}\nabla f_1\right]$$

$$= h\left[f_1 - \frac{1}{2}(f_1 - f_0)\right]$$

$$= \frac{h}{2}[f_0 + f_1]$$

$$y_1 = y_0 + \frac{h}{2}[f_0 + f_1]$$

More generally,

$$y_{n+1} = y_n + \frac{h}{2}[f_n + f_{n+1}] \qquad \ldots(6.38)$$

for $n = 0, 1, 2, \ldots.$

Example 6 Given the ODE, $y' = 1 + 2xy$, with (0, 0).

(a) Show that the series expansion of y in power of x as far as x^5,

$$y = x + \frac{2}{3}x^3 + \frac{4}{15}x^5 + \ldots$$

(b) Tabulate y and x to 4 dp for $x = 0(0.1)0.3$ and apply Adams-Bashforth method to compute y(0.4).

Solution Given $y' = 1 + 2xy$, $x = 0, y = 0$.

(a) Using three approximations, Picard's method generates the required series:

$$y = x + \frac{2}{3}x^3 + \frac{4}{15}x^5 + \ldots$$

$$= x\left[1 + x^2\left(\frac{2}{3} + \frac{4}{15}x^2\right)\right]$$

(b) Substituting values of x in the above series, we get,

x	y	$f = 1 + 2xy$	∇	∇^2	∇^3	∇^4
0.0	$.0000 = y_{-3}$	$1.0000 = f_{-3}$				
			201			
0.1	$.1007 = y_{-2}$	$1.0201 = f_{-2}$		420		
			621		49	
0.2	$.2054 = y_{-1}$	$1.0822 = f_{-1}$		469		42
			1090		91	
0.3	$.3186 = y_0$	$1.1912 = f_0$		560		44
			1650		93	
0.4	$.4453 = y_p$	$1.3562 = f_1$		562		
			1652			
0.4	$.4455 = y_p$	$1.3564 = f_1$				

Using predictor formuia (6.32), we get,

$$y_1 = y_0 + h\left\{f_0 + \frac{1}{2}\nabla f_0 + \frac{5}{12}\nabla^2 f_0 + \frac{3}{8}\nabla^3 f_0\right\}$$

$$= .3186 + 0.1\left\{1.1912 + \frac{1}{2}\times .1090 + \frac{5}{12}\times .0409 + \frac{3}{8}\times .0049\right\}$$

$$= .3186 + 0.1\{1.1912 + .0545 + .01954 + .00184\}$$

$$= .3186 + 0.1 \times 1.2671 = 0.4453$$

Using corrector formula (6.35), we get,

$$y_1 = y_0 + h\left[f_1 - \frac{1}{2}\nabla f_1 - \frac{1}{12}\nabla^2 f_1 - \frac{1}{24}\nabla^3 f_1 - \frac{19}{720}\nabla^4 f_1 + \cdots\right]$$

$$= .3186 + 0.1\left\{1.3564 - \frac{1}{2}\times .1650 - \frac{1}{12}\times .0560 - \frac{1}{24}\times .0049 - \frac{19}{720}\times .0042\right\}$$

$$= .3186 + 0.1 \times 1.2686 = 0.4455$$

Using corrector formula once again, we get,

$$y_1 = .3186 + 0.1\left\{1.3564 - \frac{1}{2}\times .1652 - \frac{1}{12}\times .0562 - \frac{1}{24}\times .0093 - \frac{19}{720}\times .0044\right\}$$

$$= .3186 + 0.1 \times 1.2686$$

$$= 0.4455$$

Using corrector formula twice, the answer correct to 4 dp is, y(.4) = 0.4455

If we are interested to extend the solution to x = 0.5 and 0.6, we get y(0.5) = 0.5923 and y(0.6) = 0.7671.

6.8.3 Adams-Moulton Method

Adams-Moulton formula (also called the **modified Adams method**) is as follows:

Predictor

Truncating (6.33) after the third differences and expressing in terms of functional values, we get,

$$y_{n+1} = y_n + \frac{h}{24}[55f_n - 59f_{n-1} + 37f_{n-2} - 9f_{n-3}] \qquad \ldots(6.39)$$

for $n = 0, 1, 2, 3, \ldots.$

Corrector

Truncating (6.36) after the third differences and expressing in terms of functional values, we get,

$$y_{n+1} = y_n + \frac{h}{24}[9f_{n+1} + 19f_n - 5f_{n-1} + f_{n-2}] \qquad \ldots(6.40)$$

Local truncation error in predictor,

$$E = \frac{251}{720}h^5 y^{(v)}(Z); \quad \text{where } x_n \le Z \le x_{n+1} \qquad \ldots(6.41)$$

Overall, the Adams-Moulton method seems to exhibit the best features and is recommended. Milne-Simpson method is fairly accurate but it has some instabilities. Small errors introduced earlier become relatively large errors over a large number of steps. So, some multi-step formulas are not computationally satisfactory because of rapid error growth due to the phenomenon of numerical instability; further consideration of which is beyond the scope of this book.

The efficiency of predictor-corrector methods is one of the primary reasons for their current popularity. Thus, it is good to add something more here regarding the use of these methods. Some sources recommend that the corrector formulas should be iterated only once, regardless of whether or not any convergence criterion is satisfied. However, this can occasionally be dangerous, particularly if the step-size is fairly coarse. In others opinion, the best approach is usually to iterate the corrector as many times as necessary to meet a reasonable convergence criterion, although it is also usually-desirable to set some upper limit on the number of iterations. This number might be approximately three for efficiency and more for ensured accuracy. If this limit is exceeded, the result can either be flagged or the program terminated if desired.

Example 7 Reconsider Example 4 of this chapter. Solve it using Adams-Moulton formula for y(0.4). Assume that the starting values given below are computed using the classic Runge Kutta method:

x	0	.1	.2	.3
y	1.0000	1.0911	1.1678	1.2335

Solution Given $y' = \dfrac{y-x}{y+x}$; $x_0 = 0$, $y_0 = 1$.

x	y	$f = \dfrac{y-x}{y+x}$
0.0	$1.0000 = y_{-3}$	$1.0000 = f_{-3}$
0.1	$1.0911 = y_{-2}$	$0.8321 = f_{-2}$
0.2	$1.1678 = y_{-1}$	$0.7076 = f_{-1}$
0.3	$1.2335 = y_0$	$0.6087 = f_0$
0.4	$1.2898 = y_p$	$0.5266 = f_1$, Predicted value
	$1.2902 = y_c$	$0.5267 = f_1$, Corrected value

Using predictor formula (6.39), we get,

$$y_1 = y_0 + \frac{h}{24}\{55f_0 - 59f_{-1} + 37f_{-2} - 9f_{-3}\}$$

$$= 1.2335 + \frac{0.1}{24}\{55 \times 0.6087 - 59 \times 0.7076 + 37 \times 0.8321 - 9 \times 1.0000\}$$

$$= 1.2335 + \frac{0.1}{24}\{33.4785 - 41.7484 + 30.7877 - 9.0000\}$$

$$= 1.2335 + \frac{0.1}{24} \times 13.5178 = 1.2898$$

Using corrector formula (6.40), we get,

$$y_1 = y_0 + \frac{h}{24}\{9f_1 + 19f_0 - 5f_{-1} + f_{-2}\}$$

$$= 1.2335 + \frac{0.1}{24}\{9 \times 0.5266 + 19 \times 0.6087 - 5 \times 0.7076 + 0.8321\}$$

$$= 1.2335 + \frac{0.1}{24} \times 13.5988 = 1.2902$$

Using corrector formula once more, we get,

$$y_1 = 1.2335 + \frac{0.1}{24}\{9 \times 0.5267 + 19 \times 0.6087 - 5 \times 0.7067 + 0.8321\}$$

$$= 1.2335 + \frac{0.1}{24}\{4.7403 + 11.5653 - 3.538 + 0.8321\}$$

$$= 1.2335 + \frac{0.1}{24} \times 13.5997 = 1.2902$$

Using corrector formula twice, answer correct to 4 dp, y(0.4) = 1.2902.

Example 8 If $y' = 1 + 2xy$ and $y = 0$, when $x = 0$. Write a computer program to calculate the values of y correct to 6 dp for x = 0(0.1)0.3, using the classical Runge-Kutta method and, then solve the differential equation for x = 0.4 by (i) Milne-Simpson method, (ii) Adams-Bashforth method and (iii) Adams-Moulton method.

Solution

Program No. 14: Predictor-Corrector Methods

Given the data: $x = 0, y = 0, h = 0.1, y' = 1 + 2xy$

```
# include<iostream.h>
# include<conio.h>
# include<process.h>

float ray[50][50];
int dg=4, lt;

float func(float a, float b)
  {
    float temp;
    temp=1+2*a*b;
    return temp;
  }

void initial (float x[ ], float y[ ], float f [ ])
  {
    int p, q;
    for (p=0; p<50; p++)
      {
        x[p]=0;
        y[p ]=0;
        f [p ]=0;
        for (q=0; q<50; q++)
            ray [p][q]=0;
```

```
      }
  }

entry (float x[ ], float y[ ], float &h, float &xp)
  {
    clrscr ( );
    cout<<"\n\tIMPLEMENTATION OF PREDICTOR-CORRECTOR METHODS";
    cout<<"\n\tINITIAL VALUE OF X : ";
    cin>>x[0];
    cout<<"\n\tINITIAL VALUE OF Y : ";
    cin>>y[0];
    cout<< "\n\tSTEP LENGTH H .    :          ";
    cin>>h;
    cout<<"\n\tVALUE OF X FOR WHICH\n\tY IS TO BE PREDICTED : ";
    cin>>xp;
  }

void calc(int n, float h, float x[ ], float y[ ], float f[ ])
{
    int i;
    float yn,k1,k2,k3,k4,k;
    for(i=0; i<=n; i++)
      {
        k1=h*func(x[i], y[i];
        k2=h*func(x[i]+h/2,y[i]+k 1/2);
        k3=h*func(x[i]+h/2,y[i]+k2/2);
        k4=h*func(x[i]+h,y[i]+k3);
        k=(k1+2*(k2+k3)+k4)/6;
        yn=y[i]+k;
        f[i]=func(x[i],y[i]);
        x[i+1]=x[i]+h;
        y[i+1]=yn;
      }
}

void table(int m, float xp)
  {
      int a,b,c,i,j;
      c=m;
      b=5;
      dg=4;
      clrscr( );
      cout<<"\t\tDIFFERENCE TABLE";
```

```
    cout<< "\n\t\tY\t          F(X)\t\tDIFFERENCES";
    for(i=1;i<m;i++)
      {
        a=1;
        gotoxy(a,b);
        cout<<ray[i][1];
        gotoxy(a+= 11,b);
        cout<<ray[i][2];
        gotoxy(a+= 12,b);
        cout<<ray[i][3];
        b+=2;
      }
    a+=10;
    for(i=4;i<=dg+3;i++)
      {
        b=i+1;
        for(j=1; j<c-1; j++)
          {
            gotoxy(a,b);
            cout<<ray[j] (i];
            b+=2;
          }
        a+=11;
        c--;
      }
    gotoxy(5,30);
    cout<< "RESULTANT - VALUE = "<<ray[m][2];
    getch( );
  }
void msimp(int n, float h, float x[ ], float y[ ], float xp,float f [ ])
  {
    float yc, yp;
    int i;
    clscr( );
    cout<< "\n\tMILNE-SIMPSON METHOD";
    cout<<"n\tX\tY\t\t\tF(X)";
    for (i=0; i<=n;i++)
      {
        cout<<"\n\t"<<x[i]<<"\t"<<y[i]<<"\t\t"<<f [i];
      }
    y[n+1]=y[n-3]+4*h/3*(2*f [n-2]-f [n-1]+2*f [n]);
    yp=y[n+1];
    y[n+1]=y[n-1]+h/3*(f [n-1)+4*f [n]+f [n+1]);
```

```
    yc=y[n+1];
    while(yc!=yp)
      {
        yp=yc;
        f [n+1]=func(x[n+1],y[n+1]);
        y[n+1]=y[n-1]+h/3*(f [n-l]+4*f [n]+f[n+1])
        yc=y[n+1];
      }
    cout<<"\n\n\tTHE PREDICTED & CORRECTED VALUE OF Y("<<xp+h<<") is "<<yc;
  }

void setting(int m, int n)
  {
    int i, j, count=0, v=n;
    dg=4;
    for(j=4;j<=dg+3;j++)
      {
        for(i= 1;i<=m-j+ 3;i++)
          {
            count++;
            if(count<v)
              ray[i][j]=ray[i+1][i-1]-ray[i] [j-1];
            else
              ray[i] [j]=ray[i+1][j-1]-ray[v][j-1];
          }
        count=0;
        v--;
      }
  }

void bash(int n,float h,float x[ ],float y[ ],float xp,float f [ ])
  {
    int t=0;
    float yc,yp;
    setting(n,n);
    for( t=1;t<=n;t++)
      {
        ray[t][1]=x[t-1];
        ray[t][2]=y[t-1];
        ray[t][3]=f [t-1];
      }
    t=0;
    ray[n+l][1]=xp;
```

```
    ray[n+1][2]=ray[n][2]+h*(ray[n][3]+0.5*ray[n-l][4]+0.416667*ray[n-2][5]
    +0.375*ray[n-3][6];
    yp=ray[n+1][2];
    ray[n+1][3]=func(xp,ray[n+1][2]);
    setting(n+1,n);
    ray[n+2][1]=xp;
    yc=ray[n+2][2]=ray[n][2]+h*(ray[n+1][3]-0.5*ray[n][4]-0.083333*ray[n-1][5]
                    -0.041667*ray[n-2}[6]-0.026389*ray[n-3][7]);
    ray[n+2][3]=func(xp,yc);

    setting(n+2,n);
    t=n+3;
    while(yc!=yp)
      {
        yp=yc;
        yc=ray[t][2]=ray[n][2]+h*(ray[t-1][3]-0.5*ray[t-2][4]-0.083333*ray[t-3][5]
                    -0.041667*ray[t-4][6]-0.026389*ray[t-5} [7]);
        ray[t][3]=func(xp,yc);
        ray[t][1]=xp;
        t++;
      }
    table(t-1,xp );
  }

void amoulton(int n, float h, float x[ ],float y[ ],float xp,float f [ ])
  {
    float yc, yp;
    int i;
    clrscr( );
    cout<<"\n\tADAMS-MOULTON METHOD";
    cout<<"\n\tX\tY\t\t\tF(X)";
    for(i=0;i<=n;i++)
    {
      cout<<"\n\t "<<x[iJ<<"\t "<<y[i]<<"\t\t"<<f [i];
    }
  y(n+1]=y(n]+h/24*(55*f(n]-59*f [n-l ]+37*f [n-2]-9*f[n-3]);
  yp=y[n+1];
  y[n+1]=y[n]+h/24*(9*f [n+1]+19*f [n]-5*f [n-1]+f [n-2]);
  yc=y[n+1];
  while(yc!=yp)
    {
      yp=yc;
      f [n+1]=func(x[n+1],y(n+1]);
```

```
    y[n+1]=y(n-l]+h/3*(f [n-1]+4*f [n]+f [n+1]);
    yc=y[n+1];
  }
cout<<"\n\n\tTHE PREDICTED & CORRECTED VALUE OF Y("<<xp+h<<") is "<<yc;
{
  void main(void)
  {
    int n,choice;
    char opt;
    float x[50],y[50];f [50],h,xp;
    cout<<"\n\t\tPREDICTOR-CORRECTOR METHODS";
    cout<<"\n\tTHIS PROGRAM SOLVES Y' = 1 + 2XY";
    while(l)
      {
        initial(x,y,f);
        clrscr( );
        cout<<"\n\n\t\tMENU";
        cout<<"\n\n\tMILNE-SIMPSON METHOD--------------1";
        cout<<"\n\n\tADAMS-BASHFORTH METHOD---------2";
        cout<<"\n\n\tADAMS-MOULTON METHOD-----------3";
        cout<<"\n\n\tEXIT-------------------------------------------4";
        cout<<"\n\n\tYOUR CHOICE";
        cin>>choice;

        if(choice!=4)
          {
            entry(x,y,h,xp);
            xp=xp-h;
            n=(xp-x[0])/h+0.5;
            calc(n,h,x,y,f);
          }
        switch(choice)
          {
            case 1:msimp(n,h,x,y,xp,f);getch( );break;
            case 2:bash(n,h,x,y,xp+h,f);getch( );break;
            case 3:amoulton(n,h,x,y,xp,f);getch( ); break;
            case 4:exit(0);
            default:cout<<"\n\tENTER CORRECT CHOICE."; getch( );
          }
```

Computer Output

```
MENU
MILNE-SIMPSON. METHOD----------------1
ADAMS-BASHFORTH METHOD-----------2
ADAMS-MOULTON METHOD--------------3
EXIT------------------------------------------ 4
YOUR CHOICE :
IMPLEMENTATION OF PREDICTOR-CORRECTOR METHODS
INITIAL VALUE OF X : 0
INITIAL VALUE OF Y : 0
STEP LENGTH H :      0.1

VALUE OF X FOR WHICH
Y IS TO BE PREDICTED :  04

MILNE-SIMPSON METHOD
X        Y        F(X)

0        0        1
0.1      0.100669 1.020134
0.2      0.205419 1.082168
0.3      0.318665 1.191199

THE PREDICTED & CORRECTED VALUE OF Y(0.4) IS 0.445532

ADAMS-MOULTON METHOD
X        Y        F(X)

0        0        1
0.1      0.100669 1.020134
0.2      0.205419 1.082168
0.3      0.318665 1.191199

THE PREDICTED & CORRECTED VALUE OF Y(0.4) IS 0.445532
```

6.9 SOLUTION OF SIMULTANEOUS AND HIGHER-ORDER ORDINARY DIFFERENTIAL EQUATIONS

In the previous sections, various methods to solve the first-order ODEs are discussed. In practice, we often have to solve a set of simultaneous first-order differential equations. Such equations occur frequently in obtaining solutions of higher-order differential equations as part of the solution process. We can solve the present problems, using one of the methods discussed so far. However, we will describe the use of Runge-Kutta methods, which are well-suited for the solution of such equations; manually as well as on computers.

6.9.1 Solution of First-Order Simultaneous Differential Equations

Let us consider the solution of two simultaneous first-order differential equations of the form:

$$y' = f(x, y) \qquad \ldots (6.42)$$

$$z' = g(x, y, z)$$

with the initial conditions $y = y_0$ and $z = z_0$ when $x = x_0$ are given.

Using the classic Runge-Kutta method, we get,

$$y_{n+1} = y_n + \frac{1}{6}[k_1 + 2(k_2 + k_3) + k_4]$$

$$z_{n+1} = z_n + \frac{1}{6}[\ell_1 + 2(\ell_2 + \ell_3) + \ell_4]$$

where

$$k_1 = hf(x_n, y_n, z_n) \qquad \ell_1 = hg(x_n, y_n, z_n)$$

$$k_2 = hf\left[x_n + \frac{h}{2}, y_n + \frac{k_1}{2}, z_n + \frac{\ell_1}{2}\right] \qquad \ell_2 = hg\left[x_n + \frac{h}{2}, y_n + \frac{k_1}{2}, z_n + \frac{\ell_1}{2}\right]$$

$$k_3 = hf\left[x_n + \frac{h}{2}, y_n + \frac{k_2}{2}, z_n + \frac{\ell_2}{2}\right] \qquad \ell_3 = hg\left[x_n + \frac{h}{2}, y_n + \frac{k_2}{2}, z_n + \frac{\ell_2}{2}\right]$$

$$k_4 = hf[x_n + h, y_n + k_3, z_n + \ell_3] \qquad \ell_4 = hg[x_n + h, y_n + k_3, z_n + \ell_3]$$

If the number of equations is more than two, the method is modified accordingly.

6.9.2 Solution of Nth-Order Differential Equations

An nth-order differential equation can be solved by transforming the given equation for a set of n simultaneous first-order differential equations and applying Runge-Kutta formula as discussed above.

Consider the: second-order differential equation,

$$y'' = f(x, y, y') \qquad \ldots (6.43)$$

subject to the initial conditions: $y = y_0$, $y' = y_0$ at $x = x_0$.

Let $z = y'$, then (6.43) can be transformed into two first-order differential equations. Differentiating, we get,

$$\left.\begin{aligned} z' &= y'' = g(x, y, z) \\ y' &= z \end{aligned}\right\} \qquad \ldots (6.44)$$

The equations in (6.44) can be viewed as,

$$y' = z$$

$$z' = f(x, y, z)$$

and can be solved as a pair of first-order equations.

Example 9 (a) Use the classic Runge-Kutta method to solve the following system of equations for x = 0(0.1)0.5.

$$y' = x + yz$$

$$z' = y^2 + z^2$$

subject to the initial conditions, y = z = 1.0 when x = 0.

(b) Write also the computer program for the purpose.

Solution (a) Given $h = 0.1$, $y_0 = 1.0$, $z_0 = 1.0$, $x_0 = 0.0$.

Values of y and z are required for x = 0(0.1)0.5.

The sequence of computations is given below:

$$k_1 = hf(x, y, z) = 0.1[x_0 + y_0 * z_0]$$

$$= 0.1(0 + 1 \times 1) = 0.1$$

$$\ell_1 = hg(x, y, z) = 0.1\left[x_0^2 + z_0^2\right]$$

$$= 0.1(1 \times 1 + 1 \times 1) = 0.2$$

$$k_2 = hf\left[x + \frac{h}{2},\ y + \frac{k_1}{2},\ z + \frac{\ell_1}{2}\right]$$

$$= 0.1[(0 + 0.05) + (1 + 0.05)(1 + 0.1)] = 0.1205$$

$$\ell_2 = hg\left[x+\frac{h}{2},\ y+\frac{k_1}{2},\ z+\frac{\ell_1}{2}\right]$$

$$= 0.1[(0+0.05)^2+(1+0.1)^2] = 0.2313$$

$$k_3 = hf\left[x+\frac{h}{2},\ y+\frac{k_2}{2},\ z+\frac{\ell_2}{2}\right]$$

$$= 0.1\left[(0+0.5)+\left(1+\frac{0.1205}{2}\right)\times\left(1+\frac{0.2313}{2}\right)\right] = 0.1233$$

$$\ell_3 = hg\left[x+\frac{h}{2},\ y+\frac{k_2}{2},\ z+\frac{\ell_2}{2}\right]$$

$$= 0.1\left[\left(1+\frac{0.1205}{2}\right)^2+\left(1+\frac{0.2313}{2}\right)^2\right] = 0.2369$$

$$k_4 = hf(x+h,\ y+k_3,\ z+\ell_3)$$

$$= 0.1[0.1+(1.1233)\times(1.2369)] = 0.1489$$

$$\ell_4 = hg(x+h,\ y+k_3,\ z+\ell_3)$$

$$= 0.1[(1.1233)^2+(1.2369)^2] = 0.2792$$

$$k = \frac{1}{6}(k_1+2k_2+2k_3+k_4)$$

$$= \frac{1}{6}[0.2+2(0.1205+0.1233)+0.1489] = 0.1228$$

$$\ell = \frac{1}{6}(\ell_1+2\ell_2+2\ell_3+\ell_4)$$

$$= \frac{1}{6}[0.2+2(0.2313+0.2369)+0.2792] = 0.2359$$

$$y_1 = y_0+k$$

$$= 1.0+0.1228 = 1.1228$$

$$z_1 = z_0+\ell$$

$$= 1.0+0.2359 = 1.2359$$

The other values can be calculated in a similar manner and are given in the following table:

x	Y	z
0.0	1.0000	1.0000
0.1	1.1228	1.2359
0.2	1.3069	1.5778
0.3	1.5961	2.1217
0.4	2.1021	3.1208
0.5	3.2249	5.5258

Program No. 15: Runge-Kutta Method for N Equations

```
#include<iostream.h>
#include<conio.h>
#include<math.h>

float function_f(float x0, float y0, float z0)
  {
    float result;
    result=x0+y0*z0;
    return result;
  }
float function-g(float x0, float y0, float z0)
  {
    float result;
    result=(y0*y0)+(z0*z0);
    return result;
  }
void main(void)
  {
    float k1,k2,k3,k4,k;
    float 11,12,13,14,1;
    float h,x0,y0,z0;
    int n,i;
    cout<<"\n\tRUNGE-KUTTA METHOD FOR HIGHER ORDER DIFFERENTIAL
      EQUATIONS";
    cout<<"\n\tEnter the value of X0 : ";
    cin>>x0;
    cout<<"\n\tEnter the value of Y0 : ";
```

```
  cin>>y0;
  cout<<"\n\tEnter the value of Z0 : ";
  cin>>z0;
  cout<<"\n\tEnter the value of h : ";
  cin>>h;
  cout<<"\n\tEnter the value of n : ";
  cin>>n;
  cout<<"\tn\t xn\t yn\t\t zn";
  for(i=0;i<n+1 ;i++)
{
  cout<<"\n\t"<<i;
  cout<<"\t"<<x0;
  cout<<"\t"<<y0;
  cout<<"\t"<<z0;

  k1=h*function_f(x0,y0,z0);
  l1=h*function_g(x0,y0,z0);
  k2=h*function_f(x0+h/2,y0+k1/2,z0+l1/2);
  l2=h*function_g(x0+h/2,y0+k1/2,z0+l1/2);
  k3=h*function_f(x0+h/2,y0+k2/2,z0+l2/2);
  l3=h*function_g(x0+h/2,y0+k2/2,z0+l2/2);
  k4=h*function_f(x0+h,y0+k3,z0+l3);
  l4=h*function_g(x0+h,y0+k3,z0+l3);
  k=(k1+2*(k2+k3)+k4)/6;
  l=(l1+2*(l2+l3)+l4)/6;
  x0+=h;
  y0+=k;
  z0+=l;
}
}
```

Computer Output

RUNGE-KUTTA METHOD FOR HIGHER ORDER DIFFERENTIAL EQUATIONS

Enter The Value Of X0 : 0.0

Enter The Value Of Y0 : 1.0

Enter The Value Of Z0 : 1.0

Enter The Value Of h : 0.1

Enter The Value Of n : 5

n	x	Y	z
0	0.0	1	1
1	0.1	1.122751	1.235902
2	0.2	1.306859	1.577818
3	0.3	1.596055	2.121701
4	0.4	2.102132	3.120769
5	0.5	3.224887	5.5258

PROBLEMS

1. (a) Show by successive approximation method or otherwise that the differential equation,

$$y' = 1 + x\,y + x^2\,y^2$$

subject to $y(0) = 0$, is

$$y = x + \frac{1}{3}x^3 + \frac{4}{15}x^5 + \frac{13}{105}x^7 + \ldots$$

(b) Prove that the differential equation,

$y' = y^2 - 2$; subject to $y(0) = I$, has a series

solution of the form about the initial point:

$$y = 1 - x - x^2 - \frac{1}{3}x^3 + \frac{2}{3}x^4 + \ldots$$

2. Given the following differential equation,

$$y' = x(y - 1),$$

subject to the initial condition $y(0) = 2.0$.

(a) Derive the Taylor series expansion as far as the third derivative. Evaluate the series for y(0.2), correct to 4 dp.

(b) Find the true answer if the solution of the differential equation is,

$$y = e^{\frac{x^2}{2}} + 1$$

Compute the error between the true and numerical solutions.

(c) Find the maximum truncation error to 4 dp.

(d) How small must the step-size be to ensure 4 dp accuracy per step?

3. (a) Explain how an approximate solution of a differential equation may be obtained by, (i) the Taylor series, and (ii) Picard's methods.

Stating briefly the advantages and disadvantages of each method.

(b) Explain why one of the methods is inapplicable to the equation,

$$y' = y + \frac{3}{\sqrt{\sin x}}$$

for which y(0) = 1 and employ the other method to evaluate y(0.1) and y(1.2) correct to 4 dp.

c) Explain why one of the methods is not applicable to the equation:

$$y' = y x^{-\frac{1}{2}};\ y(0) = 1$$

Use the other method to evaluate y(1.5).

4. Consider the initial value problem,

$$y' = x - y;\ y(0) = 1.$$

(a) Derive the Taylor series formula as far as third derivatives for the above problem.

(b) With h = 0.1, use the result to compute an approximate solution to ensure 4 dp on the initial conditions (0, 1).

(c) Estimate the maximum truncation error per step. How small must the step-size be in order to ensure 6 dp accuracy per step?

(d) If the analytical solution to the given problem is $y = 2e^{-x} + x - 1$, calculate y(0.1). What can you say when compared with the result obtained in (b) above.

5. (a) Consider the initial value problem, $y' = x^2 + x + y$; y(0) = 1.

(i) Use Picard's method to find the solution in the form of a series, including upto four approximations.

(ii) Use the series obtained above to tabulate for x = 0(0.1)1.

(iii) If the exact solution of the given differential equation is,

$$Y = 4e^x - (x^2 + 3x + 3), \text{ compute } Y(1)$$

Obtain the global error $|Y(1) - y(1)|$.

(b) (i) Derive the Taylor series expansion formula of order 5 using the differential equation given in (a) above.

(ii) Determine roughly over what range this solution will hold to 4 dp if terminated at x^4. Using the terminated series, tabulate y for $x = 0(0.1)1.0$.

(iii) Since the exact solution is known, obtain the global error.

(c) Consider the initial value problem,

$$y'' = xy; \; y(0) = y'(0) = 1.$$

Show by Taylor series method that

$$y = 1 + x + \frac{1}{6}x^3 + \frac{1}{12}x^4 + \frac{1}{80}x^6$$

Obtain the values of y and y'' for $x = -0.2(02)0.6$, correct to 5 dp.

(d) (i) Find the numerical solution of the problem,

$$y' = y + 2x - 1; \; y(0) = 1,$$

over the interval [0, 1] using Taylor series methods of orders 1, 2, 3, and 4, with $h = 0.1$.

(ii) Find also the exact solution of the given differential equation.

6. (a) Using Euler's method, solve the differential equation,

$$y' = x^2 - y, \text{ over the interval } [0, 0.2] \text{ with } y(0) = 1 \text{ and } h = 0.05.$$

(b) If the exact solution is,

$$y = -e^{-x} + x^2 - 2x + 2$$

Compare the solution obtained in (a) above with the exact solution.

(c) Find the local and global errors.

7. (a) Use the simple Euler's method to solve for y(0.1) from:

$$y' = x + y + xy, \text{ with } y(0) = 1; \text{ with } h = 0.01.$$

Estimate how small h would need to be to obtain 4 dp accuracy.

(b) Determine y at $x = 0(0.2)0.6$ by the classic Runge-Kutta method, given that,

$$y' = \frac{1}{x+y}, \; y(0) = 2.$$

(c) Perform two iterations of second-order Runge-Kutta method for the solution of the equation:

$$y' = xy + y^2; \quad y(1) = 2, \text{ with } h = 0.$$

(d) Find the values of y(2.1) and y(2.2) as a solution of the differential equation:

$$y' = x^2 + y^2; \quad y(2) = 3.$$

Take h = 0.

(e) Repeat the problem under (d) above for y(0) = 1 for x = 0.4. Take h = 0.2

8. If $y' = x^2 + xy$, subject to y(0) = 1, find a series expansion using Picard's (or Taylor series) method for y in as far as x^6. Calculate the values of y, correct to 4 dp for x =0(0.1)0.3.

Using Milne-Simpson predictor-corrector method formula, find y(0.4) correct to 4 dp.

9. If $y' = x + y^2$ and y = 1 when x = 0, use the classical Runge-Kutta method to calculate the values of y correct to 4 dp for x = 0(0.2)0.6.

Solve the differential equation by Milne-Simpson predictor-corrector method for y when x = 0.8

10. If $y' = 1 + y^2$ and y(0) = 0, use the classical Runge-Kutta method to calculate values of y correct to 4 dp for x = 0(0.2)0.6.

Solve the differential equation by Milne-Simpson predictor-corrector method for y when x = 0.8 and 1.0.

11. The differeritial equation $y' = x - 0.1\,y^2$ is to be solved with the initial value y = I when x = 0. Assuming that the following starting values have been obtained:

x	−.2	−.1	.1	.2
y	1.04068	1.01513	0.99507	1.00013

Find the value of y correct to 5 dp at x = 0.3 using Adams-Bashforth formula.

12. (a) Solve the following simultaneous equations using Runge-Kutta method of order 4,

$$x' = y - t$$

$$y' = x + 1$$

with the initial conditions x = 0, and y = 1, when t = 0.

Complete the following table considering the answers correct to 4 dp:

t	0	.1	.2	.3	.4
x	0.0000				
y	1.0000				

(b) Consider the second-order differential equation,

$$y'' = x y' + x^2$$

with $y(0) = 1$, $y'(0) = 2$ and $h = 0.2$. Write down the equivalent system of two first-order differential equations. Determine $y(0.2)$ and $z = 0.2$.

13. (a) The differential equation,

$$y' = 2x(y - 1),\ y(0) = 0,$$

has initial values as follows:

x	y	f
0	0	0
.1	– 0.01005	– 0.20201
.2	– 0.04081	– 0.41632
.3	– 0.09417	– 0.65650
.4	– 0.17351	– 0.93881

Use Adams-Bashforth method to compute $y(0.5)$. Compare your answer with the exact answer: – 0.28403.

(b) The differential equation $y' + y^2 - x^2 = 0$ with the boundary condition $y = 0$ when $x = 0$ is satisfied by the value of x and y in the following table:

x	– 0.4	– 0.2	0.2	0.4
y	0.02131	0.00267	– 0.00267	– 0.02131

Use Adams-Bashforth formula to obtain correct to 4 dp, the value of y when $x = 0.6$.

14. (a) Given the differential equation,

$$y' + y + 2x = 0, \text{ with } y(0) = -1.$$

The starting values, correct to 7 dp, have been obtained using some method:

x	0.1	0.2	0.3
y	– 0.9145122	– 0.8561923	– 0.824547

Use the Adams-Moulton method to compute the solutions for $x = 0.4$ and 0.5.

(b) For the differential equation,

$$y' = x^3 + y^2,\ y(0) = 0,$$

using $x = 0(0.2)0.6$, compute three new values correct to 4 dp by the Runge-Kutta method. Then extend the solution for $x = 0.8$ and 1.0 using the Adams-Moulton method.

(c) Given the differential equation $y' = x - 2y$, with $y(0) = 0.75$. Assume that the other starting values are given as:

x	.2	.4	.6
y	.52032	.39933	.35119

Use the Adams-Moulton method to find the value of y at $x = 0.8$.

(d) Consider the initial value problem:

$$y' = x - y + 2;\ \ y(0) = 0$$

(i) Use Taylor series method or Picard's method to find the values of y, correct to 6 dp, for $x = 0(0.1)0.3$.

(ii) Compute the values of y for $x = 0.4$ and $x = 0.5$ using Adams-Bashforth method.

(iii) Check the answers obtained in (b) above using Adams-Moulton method.

(e) Given the differential equation:

$$y' = 2y/x;\ \ \text{with } y(1) = 0$$

The following starting values are computed using Runge-Kutta of fourth order:

x	1	1.25	1.50	1.75
y	2.00	3.13	4.50	6.13

Estimate y(2) using the Milne-Simpson's predictor-corrector method.

15. Given the following two differential equations:

$$x' = 2x + 3y$$

$$y' = 2x + y$$

with initial conditions $x(0) = -2.7$ and $y(0) = 2.8$.

(a) Solve the system of equations using the Runge-Kutta method for $t = 0(0.05)0.20$.

(b) If the analytical solution is given as,

$$x(t) = -\frac{69}{25}e^{-t} + \frac{3}{50}e^{4t} \text{ and}$$

$$y(t) = \frac{69}{25}e^{-t} + \frac{1}{25}e^{4t},$$

find the exact solutions for x(t) and y(t) for the given interval.

(c) Find also the local and global errors. Comment on your results.

16. Consider the system of two first-order differential equations:

$$x' = x + 2y$$

$$y' = 3x + 2y$$

with x = 6 and y = 4, when t = 0.

(a) Use the Runge-Kutta method to solve the above problem over the range of values 0(.02).2.

(b) Compare the numerical solution with the true solutions:

$$x(t) = 4e^{4t} + 2e^{-t} \text{ and}$$

$$x(t) = 6e^{4t} - 2e$$

17. Given $y' = \frac{y}{x} - 1$ with the initial condition y(1) = 2.

a) Find the series expansion using Taylor series or Picard's method. Tabulate the values of y corresponding to x = 1(0.02)1.08 working to 5 dp.

b) (i) Use Adams-Bashforth predictor-corrector formula to find the value of y(1.10).

(ii) If the exact solution of the differential equation is Y = x(2 − ln x), find Y(1.10). Comment on these two results.

c) Check your answer as obtained in b(i) above with Adams-Moulton method.

d) Use the computer program to extend the values of y for x = 1.10(0.02)1.20.

18. Consider the second-order initial value problem:

$$x''(t) + 4x'(t) + 5x(t) = 0$$

with the initial conditions: x(0) = 3 and $x'(0) = -5$.

a) Write down the equivalent system of two first-order equations.

b) Use the Runge-Kutta method to solve the reformulated problem in (a) above over the range of values 0(0.1)0.5.

c) Compare the numerical solution with the true solution:

$$x(t) = 3e^{-2t}\cos(t) + e^{-2t}\sin(t)$$

Display your output in the following format:

t_n	x_n	$x_n(t)$	Error = $\lvert x_n - x_n(t)\rvert$
0.0	3.000000	3. 000000	0.000000
0.1	...	...	...
0.2			
⋮			
5.0			

19. (a) Given the following system of equations:

$$\frac{dy}{dx} = 6x - 3z - 5$$

$$\frac{dz}{dx} = (x - y + 5)/3$$

with $x_0 = 0$, $y_0 = 2$ and $z_0 = -1$, $h = 0.1$.

Solve the equations for $x = 0.5$ and 1 respectively using Runge-Kutta method.

(b) Convert the following second-order equation to two first-order equations:

$$\frac{d^2 y}{d x^2} + 2x\frac{dy}{dx} - 3y = x^2 + 2, \quad h = 0.1$$

(c) Convert the following second-order equation to two first-order equations:

$$\frac{d^2 y}{d x^2} + y\frac{dy}{dx} + y = 2x$$

with $y(1) = 1$ and $y'(1) = 1$.

Chapter 7

Non-Linear Equations

7.1 INTRODUCTION

This chapter is concerned with the most commonly used methods for solving equations of the form,

$$f(x) = 0 \qquad \ldots (7.1)$$

where f(x) is a given function. The roots of (7.1), which are the required answers, are those values of x for which f(x) is true.

For example, if $f(x) = x^2 + 4x + 4$, the equation $x^2 + 4x + 4 = 0$ has two roots, – 2 and –2. The roots of an equation are also called the **zeros** of the equation.

The function f(x) can be **linear** or **non-linear**.

A linear function is of the form:

$$f(x) = ax + b,$$

where a and b are constants. In this case, the solution of f(x) = 0, is simple and is given by $ax + b = 0$, or, $x = \frac{-b}{a}$, provided $a \neq 0$.

A non-linear function may be one of the following types:

(a) f(x) may be an **algebraic function** (or a **polynomial of degree n**) expressible in the form:

$$f(x) = a_n x^n + a_{n-1} x^{n-1} + a_{n-2} x^{n-2} + \ldots + a_0$$

If $n \geq 5$, the solution cannot be obtained easily by some direct methods, and we have to use some other methods.

(b) f(x) can be a **transcendental function**. A transcendental function is one which involves trigonometric, exponential, logarithmic functions, etc. Some examples of transcendental functions are as follows:

$$\tan x - x + 1 = 0,$$

$$\log x + e^x - 1 = 0, \text{ etc.}$$

An algebraic equation of degree n has n (real and/or complex) roots, while a transcendental equation may have no root, a finite, or an infinite number of (real and/or complex) roots. It is obvious that many non-linear equations cannot be solved easily by analytical or direct methods and hence there is a need to use numerical methods for finding their roots.

7.2 METHODS TO SOLVE NON-LINEAR EQUATIONS

The numerical techniques designed to find roots are powerful, although each has its own limitations and pitfalls. Therefore, students should learn pros and cons of each method, particularly its difficulties and become familiar with the methods through practice using computer.

In this book, we shall consider the following methods for finding the real roots of algebraic and transcendental equations:

i) Simple iterative method

ii) Newton Raphson method

iii) Bisection method

iv) Secant method

v) Rule of false position

If f(x) is a polynomial in x with real coefficients, it may have both real and complex roots. Of the various methods available for finding all roots, the following two are worth mentioning:

i) Synthetic division method

ii) Bairstow's method

The interested reader is referred to some specialized books (given in the bibliography at the end of this book) for the description of Bairstow's method.

Let us discuss the above methods one by one.

7.3 SIMPLE ITERATIVE METHOD

A fundamental concepts in computer science is **iteration**. It means that a process is repeated until an answer is achieved. An iterative procedure may be defined as the trial and error method in which the subsequent trials are selected by a systematic technique for finding the root to a desired accuracy, based on some initial approximation (or approximations) to the real root of (7.1). As already mentioned, finding the roots of an equation is equivalent to finding the value of x for which f(x) = 0.

Let us proceed to find a real root, say α, of (7.1).

To start with, we need an initial approximation, say x_0, to α. In the simple iterative procedure, one of the ways is to rewrite (7.1) in the following form:

$$x = \Phi(x) \qquad \ldots (7.2)$$

Substituting the value of x_0 for α in the right hand side of (7.2), we proceed as follows:

$$x_1 = \Phi(x_0)$$

$$x_2 = \Phi(x_1)$$

$$x_3 = \Phi(x_2)$$

$$\vdots$$

$$x_{n+1} = \Phi(x_n);\ n \geq 0. \qquad \ldots (7.3)$$

What can we learn from this sequence of numbers? If the numbers tend to a limit, then we say that something has been achieved:

$$\underset{n \to \alpha}{\text{Lt}} = \Phi(\alpha)$$

Hence, $x = \alpha$ satisfies the equation (7.2). This does not mean that we necessarily have found the root, but under the given conditions we cannot improve approximations using this process. This procedure of finding successive approximations to an initial approximation is called an **iterative procedure**, each use of this procedure is called an **iteration** and each approximation achieved is called an **iterate**.

7.3.1 Termination of an Iterative Procedure

An iterative procedure may converge or diverge. If the divergence occurs, the procedure should be terminated because there may not be any solution to the problem. We may restart the procedure by changing the initial approximation if necessary.

In case of convergence, one of the following criteria for stopping computations may be used:

a) Continue the computations for a fixed number of iterations, say n, and then terminate the process. This is to safeguard against slow convergence. The final value of x_n may then be accepted as the value of the root.

b) Continue the computations till absolute difference between two successive values x_n and x_{n-1} is less than a pre-assigned accuracy, say $\in$, i.e.,

$$|x_n - x_{n-1}| < \in, \text{ where } \in > 0.$$

c) A better criterion for stopping the process is to use the following:

$$\left| \frac{x_n - x_{n-1}}{x_n} \right| < \in, \text{ provided } x_n \neq 0, \text{ and } \in > 0.$$

In considering, whether an iteration converges or not, it may be necessary to ignore the first few iterations since the procedure may appear to diverge initially, even though it ultimately may converge.

An iterative procedure to find the root of an equation consists of three parts:

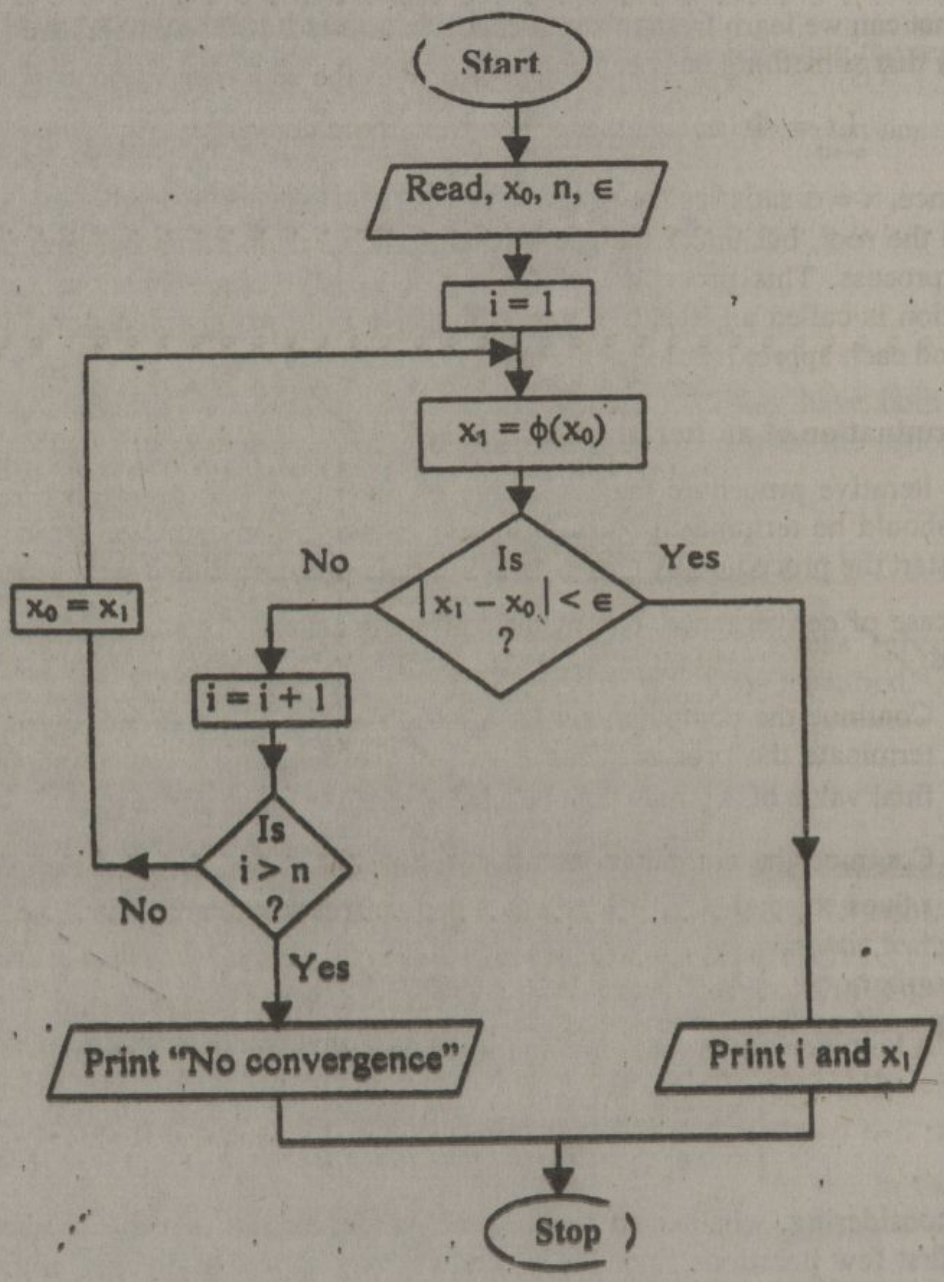
a) an initial guess for the solution,

b) an algorithm for improving the approximate solution, and

c) a criterion for stopping the computations.

7.3.2 Flowchart for a Simple Iterative Procedure

If x_0, $\in$ and n are the initial guess, pre-assigned accuracy and the number of iterations respectively, then a flowchart to find the root x_1 using a simple iterative procedure is as follows:

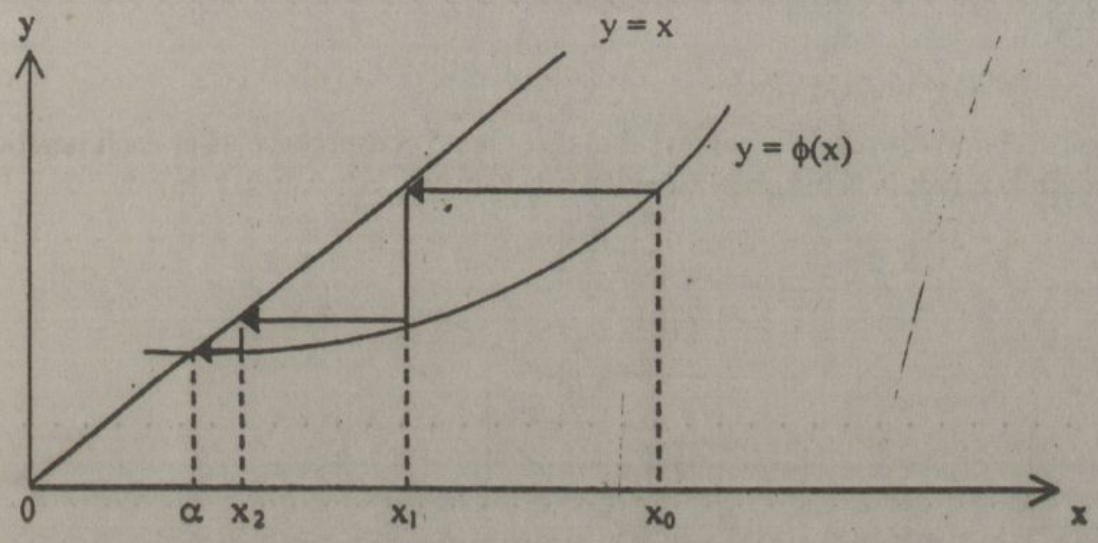


7.3.3 Graphical Representation of Convergence

After inventing an iterative procedure, it is necessary to test it for convergence. This will be shown by the following graphs drawn for $y = x$ and $y = \Phi(x)$ where $\Phi(x)$ represents a function of the forms: cosx, logx, $\frac{1}{\sqrt{1+x}}$, etc.

a) Graph for Fast Convergence

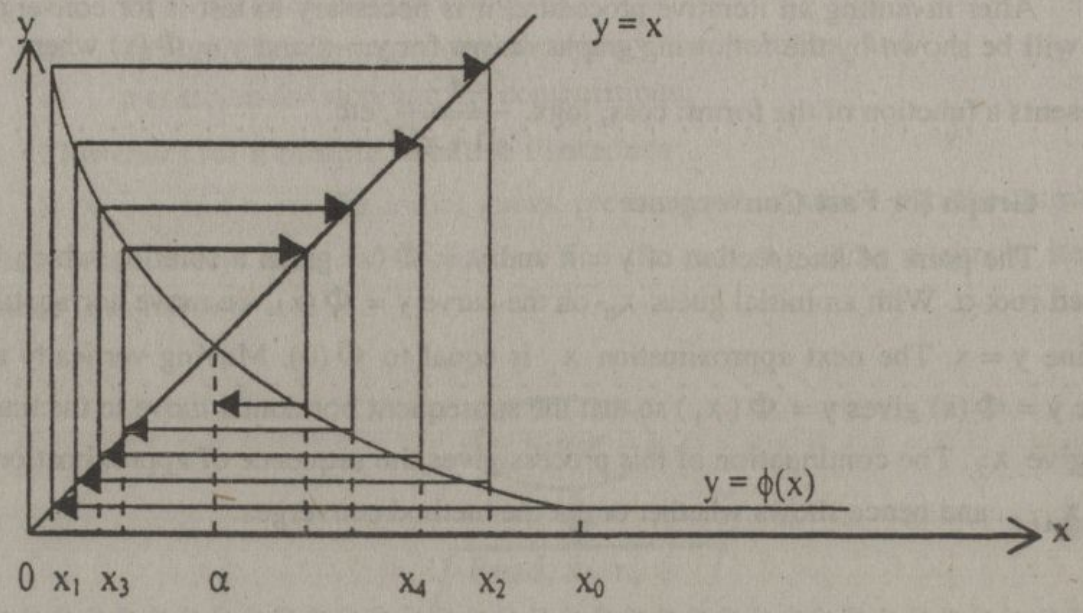
The point of intersection of $y = x$ and $y = \Phi(x)$ gives a solution which is the desired root α. With an initial guess x_0 on the curve $y = \Phi(x)$, we move horizontally to the line $y = x$. The next approximation x_1 is equal to $\Phi(0)$. Moving vertically to the curve $y = \Phi(x)$ gives $y = \Phi(x_1)$ so that the subsequent horizontal move to the line y=x will give x_2. The continuation of this process gives the sequence of approximation x_2, x_3, x_4, ... and hence shows whether or not the method converges.



The above graph shows a very rapid convergence because a better approximation is coming closer to the true root α. This is called **staircase convergence.**

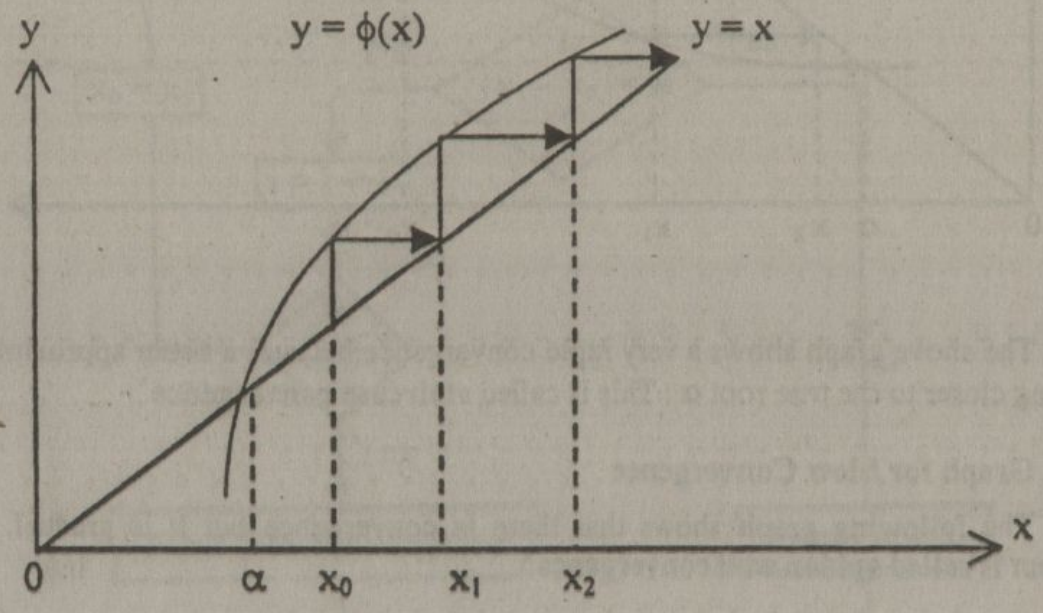
b) Graph for Slow Convergence

The following graph shows that there is convergence but it is gradual. This behaviour is called **spider web convergence.**



c) Divergent Behaviour

The following graph shows that there is no convergence as at each iteration, the approximate root is going away from the true root α.



7.3.4 Localization (Approximation) of Roots

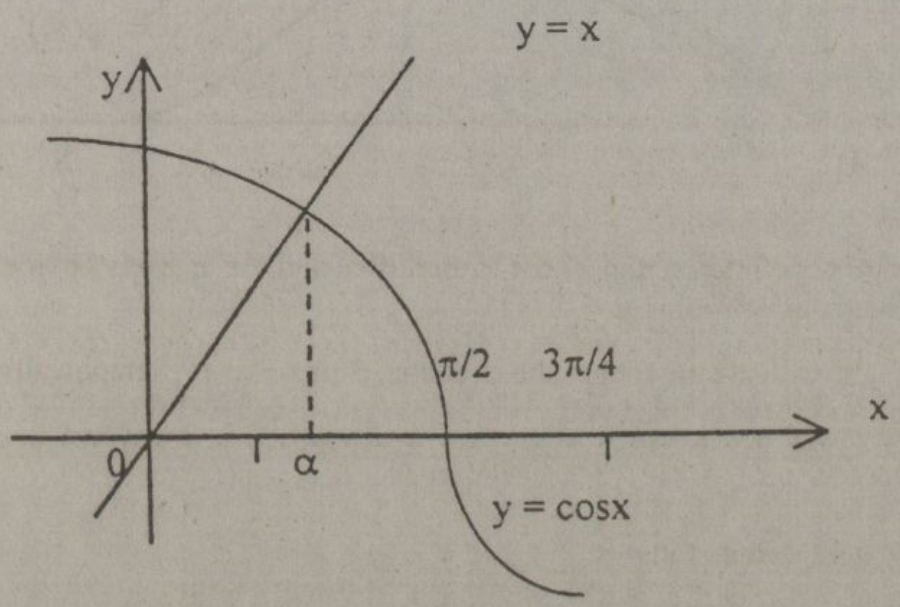
The choice in the selection of an initial guess may lead to a convergent or divergent situation. There does not exist a universally accepted hard and fast rule to select a suitable initial guess (or guesses). If it is not given in the problem, some idea about the

starting value is necessary. An initial guess may be found by using the context in which the problem first arose. The use of a graphical method can be appropriate for the purpose. We draw roughly the graph of f(x) and then see how many roots the equation has. These roots can then be read off and used as initial approximations to solve the problem at hand.

Example 1 Investigate graphically the roots of the equation, $\cos x - x = 0$.

Solution Rearranging the equation: $x = \cos x$

Plot: $y = x$; $y = \cos x$



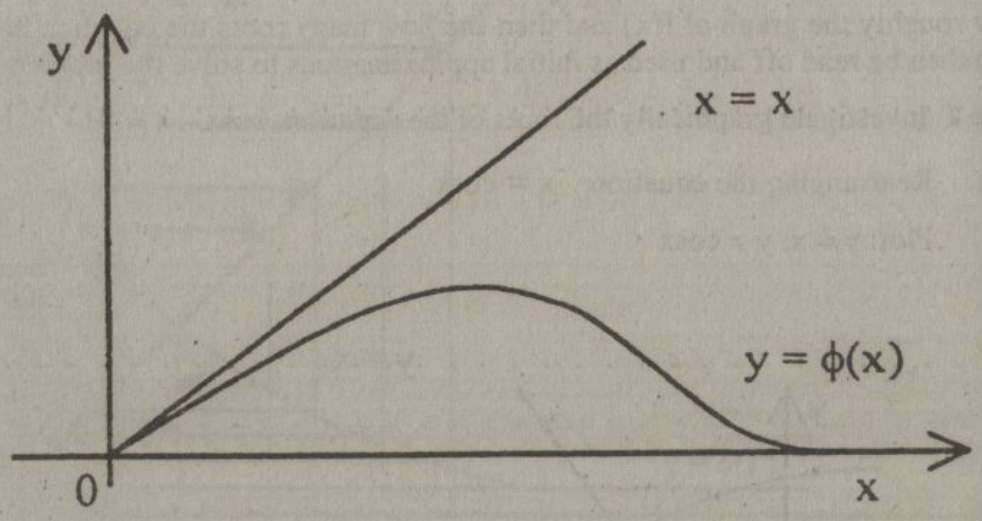
The point of intersection of the curve and straight line gives the root of the equation. It is clear from the graph that the equation has only one root, which is about $\alpha = \frac{\pi}{4} = 0.786$. It may be possible that the solution lies between 0.7 and 0.8.

Example 2 Investigate the root of the equation, $x - \sin^2 x = 0$ graphically.

Solution Rearranging the given equation:

Plot: $y = x$; $y = \sin^2 x$

Graph is as follows:

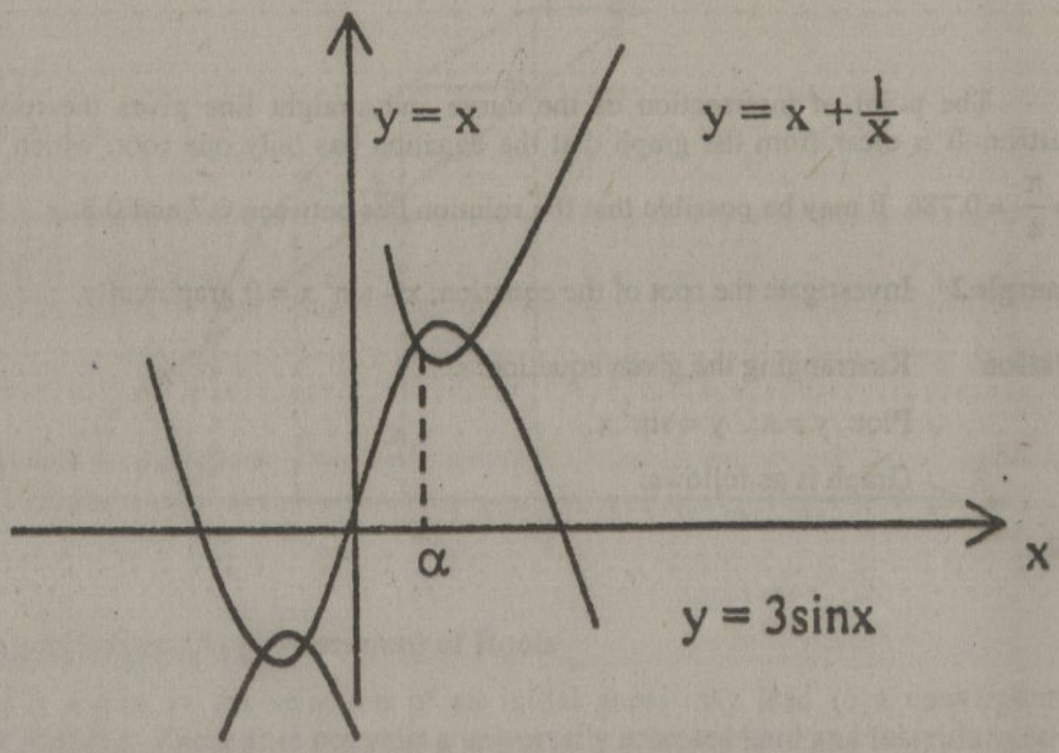


The point of intersection of the straight line and the curve is obviously at $x = 0$. Therefore, the required root is $x. = 0$.

Example 3 Investigate the root of the equation, $3 \sin x = x + \frac{1}{x}$ graphically

Solution Plot: $y = x + \frac{1}{x}$; $y = 3 \sin x$

Graph is as follows:



It is clear from the graph that the root α lies at the point of intersection of the two curves and it is to the left of $x = 1$. Hence, it is safe to take the approximate root $x_0 = 0.8$.

7.3.5 Convergence

The following three questions concerning convergence arise:

a) Does the sequence of iterates $x_0, x_1, x_2, \ldots$, always converge to some number α ?

b) If it does, will α be a root of the equation $x = \Phi(x)$?

c) How shall we choose x_0 so that the sequence of iterates $x_1, x_2, \ldots, x_n$ converges to the root α ?

Let us briefly answer the above questions.

a) The answer for the first question is no. For example, let us consider the equation:

$$x = 10^x + 1.$$

If $x_0 = 0$, then $x_1 = 10^0 + 1 = 2$ and subsequently,

$$x_2 = 10^2 + 1 = 101$$

$$x_3 = 10^{101} + 1, \text{ etc.}$$

It shows that as n increases, x_n also increases without limit. Hence, the sequence $x_1, x_2, x_3, \ldots$ does not always converge.

b) The second question is easy to answer.

Let us reconsider $x_{n+1} = \Phi(x_n)$, which gives the relation between nth and (n+1)th iterates. As n increases the left hand side tends to the root α, and if Φ is continuous, the right hand side tends to $\Phi(\alpha)$. Hence, in the limit, we have $\alpha = \Phi(\alpha)$ which shows that α is a root of (7.3). It means that the sequence converges to the true root.

c) The answer to the third question is contained in the following theorem (stated without proof):

Theorem: Let $x = \alpha$ be a root of $f(x) = 0$ and let I be an interval containing the point $x = \alpha$. Let $\Phi(x)$ and $\Phi'(x)$ be continuous in I, where $\Phi(x)$ is defined by the equation $x = \Phi(x)$ which is equivalent to $f(x) = 0$.

Then, if $|\Phi'(x)| < 1$ for all x in I, the sequence of iterates $x_0, x_1, \ldots, x_n$ converges to the root α., provided that the initial approximation x_0 is chosen in I. Thus the error decreases.

As $|\Phi'(x)|$ increases toward 1, the rate of convergence decreases. Gradually, we do get divergence in the interval where $|\Phi'(x)| > 1$. Thus, the error grows. If $|\Phi'(x)| = 1$, the error remains constant.

Example 4 Find a root correct to 4 dp of the equation, $x^3 + x^2 - 1 = 0$. Suppose that the actual root lies in the interval (0, 1).

Solution (i) Let us first find whether or not the iterative method is applicable.

Let us rewrite $f(x) = 0$ in the form $x = \Phi(x)$.

From $x^3 + x^2 = 1$, we get,

$$x(x^2 + 1) = 1; \quad x^2 = \frac{1}{1+x}$$

or

$$x = \frac{1}{\sqrt{1+x}}$$

so that $\Phi(x) = \dfrac{1}{\sqrt{1+x}}$. There may be many other choices for $\Phi(x)$.

$$\Phi'(x) = -\frac{1}{2}(1+x)^{-\frac{3}{2}}$$

Since, $|\Phi'(x)| < 1$ for all $x \le 1$, the iterative method is applicable.

(ii) Assume that $x_0 = 0.75$. We, therefore, set up the scheme:

$$x_{n+1} = \Phi(x_n) = \frac{1}{\sqrt{1+x_n}}$$

Putting $n = 0$, we get,

$$x_1 = \frac{1}{\sqrt{1+x_0}} = \frac{1}{\sqrt{1+0.75}} = 0.7559$$

$$x_2 = \frac{1}{\sqrt{1+x_1}} = \frac{1}{\sqrt{1+0.7559}} = 0.7547$$

$$x_3 = \frac{1}{\sqrt{1+x_3}} = \frac{1}{\sqrt{1+0.7547}} = 0.7549$$

$$x_4 = \frac{1}{\sqrt{1+x_3}} = \frac{1}{\sqrt{1+0.7549}} = 0.7549$$

The root, correct to 4 dp, is $x = 0.7549$.

Example 5 Find the root of the equation $2x - \cos x - 3 = 0$ correct to 3 dp. Check also if the simple iterative method is applicable. Take $x_0 = \frac{\pi}{2}$. Write a computer program to implement the method.

Solution (a) Rewriting $f(x) = 0$ in the form $x = \Phi(x)$, we get,

$$x = \frac{1}{2}(\cos x + 3)$$

so that $\Phi(x) = \frac{1}{2}(\cos x + 3)$

$$\Phi'(x) = -\frac{1}{2}\sin x$$

$$|\Phi'(x)| < 1.$$

Hence, the iterative method can be applied. We, therefore, set up the formula:

$$x_{n+1} = \Phi(x_n) = \frac{1}{2}(\cos x_n + 3).$$

Putting $n = 0$, we get,

$$x_1 = \Phi(x_n)$$

$$= \frac{1}{2}(\cos x_0 + 3) = \frac{1}{2}(\cos\frac{\pi}{2} + 3) = 1.5$$

$$x_2 = \frac{1}{2}(\cos 1.5 + 3) = 1.535$$

$$x_3 = \frac{1}{2}(\cos 1.535 + 3) = 1.518$$

$$x_4 = \frac{1}{2}(\cos 1.518 + 3) = 1.527$$

$$x_5 = \frac{1}{2}(\cos 1.527 + 3) = 1.522$$

$$x_6 = \frac{1}{2}(\cos 1.522 + 3) = 1.524$$

$$x_7 = \frac{1}{2}(\cos 1.524 + 3) = 1.523$$

$$x_8 = \frac{1}{2}(\cos 1.523 + 3) = 1.524$$

Hence, the required root $x = 1.524$, which is correct to 3 dp.

Program No. 16: Simple Iterative Method

```
#include<iostream.h>
#include<math.h>
#include<conio.h>
#include<process.h>
# define f(x)  0.5*(cos(x)+3)

int n;
float e,x0,x1;

void main(void)
{
  int i,n,flag=1;
  float x2;
  char ch:

    cout<<"\n\n\tSIMPLE ITERATIVE METHOD";
    cout<<"\n\n\tEnter The Value Of X0\t";
    cin>>x0;
    cout<<"\n\n\tEnter The Value Of N\t";
    cin>>n;
    cout<<"\n\n\tEnter The Value Of E\t";
    cin>>e;
    x1 =f(x0);
    for(i=0;i<n && flag;i++)
    {
      x2=x1-x0;
      if (x2<0)
        x2=x2*(-1);
          if(x2<e)
            flag=0;
            else
              {
                x0=x1;
                x1=f(x0);
              }
    }
    if(flag==1)
      {
        cout<<"\n\n\tNo convergence";
        getch( );
      }
```

```
    else
      {
        cout<<"\n\nAfter\t"<<i<<"\tIterations,Root Is\t"<<x1;
        getch( );
      }
}
```

Computer Output

SIMPLE ITERATIVE METHOD

Enter The Value Of XO 1.5

Enter The Value Of N 30

Enter The Value Of E 0.0005

After 11 Iterations, Root is 1.523604

7.3.6 Theoretical Study of Convergence

Let us now demonstrate convergence theoretically. Let x_n be an approximation to the real root α. Therefore, $e_n = x_n - \alpha$ is the error in the approximation. So,

$$x_n = e_n + \alpha.$$

From (7.3), $x_{n+1} = \Phi(x_n)$

or, $e_{n+1} + \alpha = \Phi(e_n + \alpha)$

Expanding $\Phi(e_n + \alpha)$ in the Taylor series, we get,

$$e_{n+1} + \alpha = \Phi(\alpha) + e_n \Phi'(\alpha) + \frac{1}{2}e_n^2 \Phi''(\alpha) + \frac{1}{6}e_n^3 \Phi'''(\alpha) + \ldots$$

Since α is a root of the equation $x = \Phi(x)$, therefore, $\alpha = \Phi(\alpha)$.

$$e_{n+1} = e_n \Phi'(\alpha) + \frac{1}{2}e_n^2 \Phi''(\alpha) + \frac{1}{6}e_n^3 \Phi'''(\alpha) + \ldots \qquad \ldots (7.4)$$

We shall now consider the result (7.4) in some detail.

Case 1 When $\Phi'(\alpha) \neq 0$ (Simple or First-order Iteration)

If $\Phi'(\alpha) \neq 0$, neglecting squares and higher powers of e_n in (7.4), we have,

$$e_{n+1} = e_n \Phi'(\alpha)$$

If follows that if $|\Phi'(\alpha)| < 1$, then the sequence $x_0\ x_1,\ x_2, \ldots$ will tend to be α. Since, we do not, in general, know that what the value of α is, we usually replace α in the above condition by x_0 and hence, the practical criterion for simple iteration to lead to a root is $|\Phi'(x_0)| < 1$. Note also that since $e_{n+1} = e_n\ \Phi'(x_0)$, the closer $\Phi'(x_0)$ is to zero the more quickly the sequence $x_0\ x_1,\ x_2, \ldots$ will converge. This is a sufficient but not a necessary condition for convergence.

Case 2 When $\Phi'(\alpha) = 0$ but $\Phi''(\alpha) \neq 0$, (Second-order Iteration)

If $\Phi''(\alpha) \neq 0$, neglecting cubes and higher powers of e_n in (7.4), we get;

$$e_{n+1} = \frac{1}{2}\Phi''(\alpha)\, e_n^2$$

It means that each error in this case is proportional to the square of the previous one. It shows that convergence in the second-order iteration is normally very rapid.

Case 3 When $\Phi'(\alpha) = \Phi''(\alpha) = 0$ but $\Phi'''(\alpha) \neq 0$, (Third-order Iteration)

If $\Phi'(\alpha) = \Phi''(\alpha) = 0$, then $e_{n+1} = \frac{1}{6}\Phi'''(\alpha)\, e_n^3$ and there is a very rapid convergence. However, this advantage tends to be offset by the fact that higher the order of the iterative process the more complicated $\Phi(x)$ tends to be, so that time saved by the speed of convergence is lost again in evaluating $\Phi(x)$ at each stage.

7.4 ACCELERATION OF CONVERGENCE

The slow rate of convergence of a first-order iterative-process can be accelerated, by using Aitken's Δ^2-process, which is described below:

Let x_{i-1}, x_i and x_{i+1} be three successive approximations to the desired root $x = \alpha$ of the equation $x = \Phi(x)$.

For any first-order process, we can write: $e_{n+1} = \Phi'(\alpha)\, e_n = k e_n$ ... (7.5)

where $e_0 = \alpha - x_{i-1}$

$e_1 = \alpha - x_i$

$e_2 = \alpha - x_{i+1}$

Setting n = 0 and 1 respectively in the relation (7.5) and simplifying, we get:

$$\alpha - x_i = k(\alpha - x_{i-1}) \qquad \ldots (7.5.1)$$

$$\alpha - x_{i-1} = k(\alpha - x_i) \qquad \ldots (7.5.2)$$

Subtracting (7.5.2) from (7.5.1), we obtain,

$$x_{i-1} - x_i = k(x_i - x_{i-1})$$

or
$$k = \frac{x_{i+1} - x_i}{x_i - x_{i-1}}$$

Substituting k in (7.5.1), we obtain,

$$\alpha - x_i = \frac{x_{i+1} - x_i}{x_i - x_{i-1}}(\alpha - x_{i-1})$$

Simplifying, we obtain,

$$\alpha = \frac{x_{i+1}x_{i-1} - x_i^2}{x_{i+1} - 2x_i + x_{i+1}}$$

$$= \frac{x_{i+1}x_{i-1} - x_i^2 + x_{i+1}^2 - x_{i+1}^2 + 2x_ix_{i+1} + 2x_ix_{i+1}}{x_{i+1} - 2x_i + x_{i-1}}$$

(Note the above step)

$$= \frac{x_{i+1}x_{i-1} - 2x_ix_{i+1} + x_{i+1}^2 - (x_{i+1}^2 - 2x_ix_{i+1} + x_{i+1}^2)}{x_{i+1} - 2x_i + x_{i-1}}$$

$$= x_{i+1} - \frac{(x_{i+1} - x_i)^2}{x_{i+1} - 2x_i + x_{i-1}} \qquad \ldots (7.6)$$

Let us define Δx_i and $\Delta^2 x_{i-1}$ by the relations:

$$\Delta x_i = x_{i+1} - x_i$$

$$\Delta^2 x_{i-1} = \Delta(\Delta x_{i-1})$$

$$= \Delta(x_i - x_{i-1})$$

$$= \Delta x_i - \Delta x_{i-1}$$

$$= (x_{i+1} - x_i) - (x_i - x_{i-1})$$

$$= x_{i+1} - 2x_i + x_{i-1}$$

Substituting in (7.6), we get,

$$\alpha = x_{i+1} - \frac{(\Delta x_i)^2}{\Delta^2 x_{i-1}}; \text{ for all } i \geq 0. \qquad \ldots (7.7)$$

which explains the term Δ^2-process. It should be noted that Aitken's method cannot be applied to second- or higher-order iterative processes. It can only be used to accelerate the convergence of any sequence that is linearly convergence.

Example 7 Three successive approximations to a root of equation $x^3 - x^2 - x + 1 = 0$, obtained by a linearly convergent iterative process, are: $x_0 = 0.6$, $x_1 = 0.5435$ and $x_2 = 0.5582$. Use Aitken's delta procedure to obtain a better approximation.

Solution **Difference Table**

			Δ	Δ^2
x_0	=	0.6000		
			−565	
x_1	=	0.5435		721
			147	
x_2	=	0.5582		

Putting i = 1 in Aitken's delta process (7.7), we get the accelerated root as,

$$\alpha = x_2 - \frac{(\Delta x_i)^2}{\Delta^2 x_0}$$

$$= 0.5582 - \frac{0.0147^2}{0.0712} = 0.55$$

It means that we have jumped ahead about two iterations, using Aitkeri's process.

Example 8 Find the root of the equation 2x = eosx + 3 correct to 3 dp, using Aitken's delta process. Take x = 1.5.

Solution Re-arranging the equation: $x = \frac{1}{2}(\cos x + 3)$ and then calculating three of its roots, we get:

			Δ	Δ^2
x_0	=	1.5		
			35	
x_1	=	1.535		−52
			−17	
x_2	=	1.518		

Substituting the required values in (7.7), we get,

$$\alpha = 1.518 - \frac{(-0.017)^2}{(-0.052)}$$

$$= 1.518 + 0.006$$

$$= 1.524$$

The process can be iterated up to the desired degree of accuracy.

7.5 NEWTON-RAPHSON METHOD

The Newton-Raphson (or simply Newton's) method is one of the most powerful and well-known method, used for finding a root of f(x) = 0. There are many ways to derive Newton-Raphson method. The simplest way to derive this formula is by using the first two terms in the Taylor series expansion of the form,

$$f(x_{n+1}) = f(x_n) + (x_{n+1} - x_n) f'(x_n)$$

Setting $f(x_{n+1}) = 0$ gives,

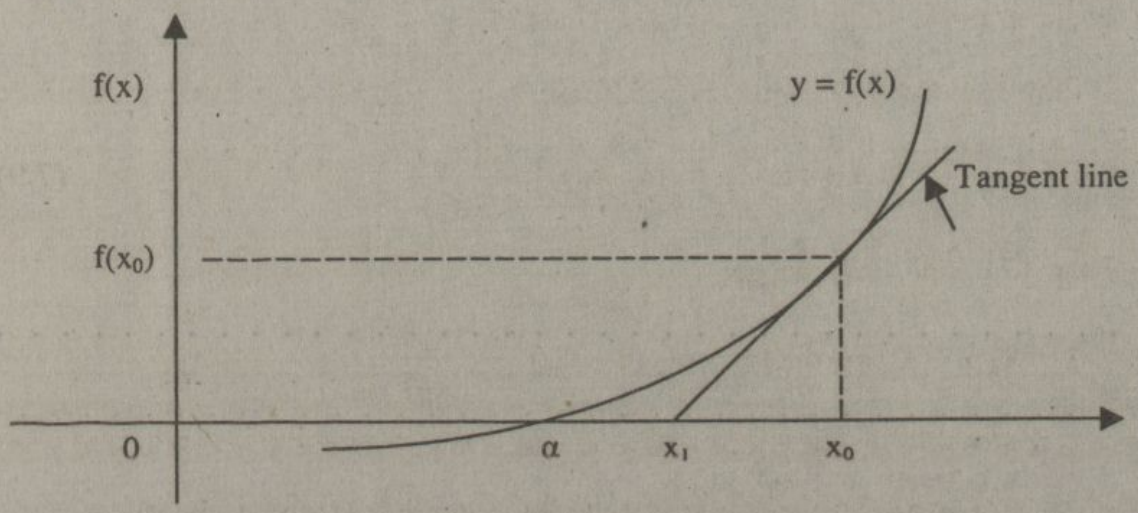
$$f(x_n) + (x_{n+1} - x_n) f'(x_n) = 0$$

Thus, on simplification, we get,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}; \quad \text{for } n = 0, 1, \ldots \qquad \ldots (7.8)$$

7.5.1 Geometrical Interpretation

The geometrical interpretation of Newton-Raphson method is quite simple and is given in the following figure:



The root α is given by the point of intersection of the curve y = f(x) and the x-axis. If an iterative procedure is being designed to approximate real root α, one simple approach can be to replace the curve by a straight line the intersection of which with the x-axis can easily be found. Starting with an arbitrary initial approximation x_0, we then calculate a sequence of iterates, $x_1, x_2, x_3, \ldots$.

Now the question is how to select the direction of the straight line. In Newton-Raphson method, the direction of the straight line is that of the tangent to the curve at the given point. That is why this method is also known as **Newton's method of tangents**.

Let an initial guess be x_0. Move vertically a tangent line from that point to the curve. The point x_1 where the tangent line crosses the x-axis will be the new iterate, i.e.,

the improvement over x_0. The point x_1 is then used as the next starting point. We repeat this process several times until our solution is sufficiently accurate.

7.5.2 Order of Newton-Raphson Method

Now we shall establish the relationship between e_n and e_{n+1}. We know that $e_n = \alpha - x_n$.

The Taylor series expansion about x_n gives:

$$f(\alpha - x_n) = f(x_n) + (\alpha - x_n) f'(x_n) + \frac{1}{2}(\alpha - x_n)^2 f''(x_n)$$

Setting the above relation equal to 0, we get,

$$f(x_n) + (\alpha - x_n) f'(x_n) + \frac{1}{2}(\alpha - x_n)^2 f''(x_n) = 0$$

Dividing both sides by $f'(x_n)$ we get,

$$\frac{f(x_n)}{f'(x_n)} + (\alpha - x_n) + \frac{1}{2}(\alpha - x_n)^2 \frac{f''(x_n)}{f'(x_n)} = 0$$

It follows that

$$-\frac{f(x_n)}{f'(x_n)} = (\alpha - x_n) + \frac{1}{2}(\alpha - x_n)^2 \frac{f''(x_n)}{f'(x_n)} \qquad \ldots (7.9)$$

From (7.8) and (7.9), we get,

$$x_{n+1} - x_n = (\alpha - x_n) + \frac{1}{2}(\alpha - x_n)^2 \frac{f''(x_n)}{f'(x_n)}$$

$$x_{n+1} - \alpha = \frac{1}{2}(\alpha - x_n)^2 \frac{f''(x_n)}{f'(x_n)}$$

Setting $x_{n+1} - \alpha = e_{n+1}$ and $x_n - \alpha = e_n$, we get,

$$e_{n+1} = \frac{1}{2} \frac{f''(x_n)}{f'(x_n)} e_n^2$$

Since $\dfrac{f''(x_n)}{2f'(x_n)}$ is a constant quantity (say k), we can write,

$$e_{n+1} = k e_n^2 \qquad \ldots (7.10)$$

It is obvious from (7.10) that the error at (n + 1)st step is proportional to the square of the error of the nth step. Hence, we say that Newton-Raphson method has

quadratic convergence order. In practical terms, this means that the number of accurate significant figures is approximately doubled with each iteration. For example, if we start with one correct digit for an approximation, then after one iteration we should have two correct digits; after three iterations, eight correct digits. So, if proper care has been observed, we need to use Newton-Raphson process only three or four times.

Several problems may arise using Newton-Raphson method:

i) x_0 must often be chosen very close to a root for convergence to the result.

ii) $f'(x)$ must not be very easy to compute. This is a major difficulty in this method.

iii) If $|f'(x)|$ is very small compared to $|f(x)|$, there could be slow convergence or even divergence.

Despite some of the difficulties mentioned above, Newton-Raphson method is still the most popular method for finding a root of the equation. The attraction of this method is that it converges very rapidly.

To illustrate Newton-Raphson procedure, consider the following examples. These examples do not involve any specific stopping criteria.

Example 9 (a) Find the real root of $f(x) = x^2 - 2x - 2$, correct to 3 dp, using Newton-Raphson method.

(b) Write a computer program to implement the method.

Solution (a) Let $x_0 = 2$

$$f(x) = x^2 - 2x - 2$$

$$f(x_0) = f(2) = -2$$

$$f'(x) = 2x - 2$$

$$f'(x_0) = f'(2) = 2$$

Using Newton-Raphson method:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Putting $n = 0$, we get,

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

$$= 2 - \frac{(-2)}{2} = 3$$

$$f(x_1) = x_1^2 - 2x_1 - 2$$

$$= 9 - 6 - 2 = 1$$

$$f'(x_1) = 2x_1 - 2$$

$$= 2 \times 3 - 2 = 4$$

$$x_2 = x_1 - \frac{f(x_1)}{f'(x_1)}$$

$$= 3 - \frac{1}{4} = 2.75$$

The subsequent iterates are as follows:

n	x_n
0	2.0
1	3.0
2	2.75
3	2.7321
4	2.7320
5	2.7321

Hence, x = 2.7321 is the root correct to 3 dp.

Program No. 17: Newton-Raphson Method

```
#include<conio.h>
#include<iostream.h>
#include<math.h>
# define f(x) (pow(x,2) -2*x -2)
# define fl(y) (2*y-2)

void main ( )
  {
    int i,flag=1;
    cout<<"\n\n\tNEWTON RAPHSON METHOD: ";
    cout<<"\n\n\tENTER THE VALUE OF X0 : ";
    cin>>x0;
    cout<<"\n\tENTER THE VALUE OF N : ";
    cin>>n;
    cout<<"\n\tENTER THE VALUE OF E : ";
    cin>>e;
```

```
for (i=0;i<n && flag;i++)
  {
    f = f(x0);
    fd = fl (x0);
    x1= x0-f/fd;
    x2 = x1-x0;
    if (x2<0)
      x2=x2*(-1);
    if (x2<e)
      flag=0;
    else
      x0=x1;
  }
if(flag==l)
  {
    cout<<"\n\n\n\n\tNO CONVERGENCE\n";
    getch ( );
  }
else
  {
    cout<<"\n\n\n\tAFTER "<<i<<"ITERATIONS, THE ROOT IS :  "x1
    getch ( );
  }
}
```

Computer Output

NEWTON RAPHSON METHOD:

ENTER THE VALUE OF X0 : 2.0

ENTER THE VALUE OF N : 20

ENTER THE VALUE OF E : 0.0005

AFTER 7 ITERATIONS, THE ROOT IS : 2.732051

7.5.3 Special Cases of Newton-Raphson Method

a) Determination of Square and Cube Roots

Let us compute the square root of a number, say a (a > 0), by Newton-Raphson method. In other words, we shall obtain a recurrence to evaluate $\sqrt{a}$.

Consider, $x = \sqrt{a}$.

$$f(x) = x^2 - a$$

$$f(x_n) = x_n^2 - a$$

$$f'(x) = 2x$$

$$f'(x_n) = 2x_n$$

Using Newton-Raphson method:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$= x_n - \frac{\left(x_n^2 - a\right)}{2x_n}$$

$$= \frac{\left(x_n^2 + a\right)}{2x_n}$$

$$= \frac{1}{2}\left(x_n + \frac{a}{x_n}\right); \text{ for } n = 0, 1, 2, \ldots \qquad \ldots (7.11)$$

The above formula is widely used, as the basis for evaluating square roots on all digital computers as well as on calculators, which include square root capability.

Similarly, we can determine the **cube root** of a number, using the following formula:

$$x_{n+1} = \frac{1}{3}\left(2x_n + \frac{a}{x_n^2}\right); \text{ for } n = 0, 1, 2, \ldots \qquad \ldots (7.12)$$

b) Determination of pth root of a Number

Let $x = a^{\frac{1}{p}}$, where $a > 0$ and p is any positive integer.

$$x_{n+1} = x_n - \frac{\left(x_n^p - a\right)}{p\,x_n^{p-1}}$$

$$= \frac{(p-1)x_n^p + a}{p\,x_n^{p-1}}$$

$$= \left(1 - \frac{1}{p}\right)x_n^{p-p+1} + \frac{a}{p}x_n^{1-p}$$

$$= \left(1 - \frac{1}{p}\right)x_n + \frac{a}{p}x^{1-p} \qquad \ldots (7.13)$$

c) **Determination of Reciprocal of a Number**

Given a number a (a > 0), we would like to find its reciprocal.

Let $x = \frac{1}{a}$, then $f(x) = \frac{1}{x}$, and $f(x) = \frac{-1}{x^2}$

$$f(x_n) = \frac{1}{x_n} - a$$

$$f'(x_n) = \frac{-1}{x_n^2}$$

Applying Newton-Raphson method (7.8), we get,

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

$$= x_n - \frac{\left(\frac{1}{x_n} - a\right)}{\left(\frac{-1}{x_n^2}\right)} \qquad \ldots (7.14)$$

$$= x_n(2 - a x_n); \quad \text{for } n = 0, 1, 2. \ldots$$

In the same-way, we can find the formula for $\frac{1}{\sqrt{a}}$ which is as follows:

$$x_{n+1} = \frac{1}{2} x_n (3 - a x_n^2) \qquad \ldots (7.15)$$

Example 10 Use the iterative formula for $\sqrt{a}$ to evaluate $\sqrt{3}$ correct to 6 dp, taking $x_0 = 1$.

Solution Given: $x_0 = 1$, $a = 3$.

Putting n = 0 in the formula (7.11), we get,

$$x_1 = \frac{1}{2}\left(x_0 + \frac{a}{x_0}\right)$$

$$= \frac{1}{2}\left(1 + \frac{3}{1}\right) = 2.0$$

$$x_2 = \frac{1}{2}\left(x_1 + \frac{a}{x_1}\right)$$

$$= \frac{1}{2}\left(2 + \frac{3}{2}\right) = 1.75$$

$$x_3 = \frac{1}{2}\left(x_2 + \frac{a}{x_2}\right)$$

$$= \frac{1}{2}\left(1.75 + \frac{3}{1.75}\right) = 1.732143$$

$$x_4 = \frac{1}{2}\left(x_4 + \frac{a}{x_3}\right)$$

$$= \frac{1}{2}\left(1.732143 + \frac{3}{1.732143}\right) = 1.732051$$

True value of $\sqrt{3} = 1.7320508$. The numerical solution and the true value agree to 6 dp.

Example 11 (a) By applying Newton-Raphson method to the function $f(x) = 1 - \frac{5}{x^2}$, devise an iterative procedure for evaluating $\sqrt{5}$.

(b) If the initial approximation is 2, calculate $\sqrt{5}$ correct to 2 dp.

(c) Show that if x_n is in error by a small quantity, e_n, then the next approximation x_{n+1} is in error by about $0.67\,e_n^2$.

Solution (a) $f(x) = 1 - \frac{5}{x^2}$

$$f(x_n) = 1 - \frac{5}{x_n^2}$$

$$f'(x) = \frac{10}{x^3}$$

$$f'(x_n) = \frac{10}{x_n^3}$$

Applying Newton-Raphson method, we get,

$$x_{n+1} = x_n - \frac{f(x_0)}{f'(x_0)}$$

$$x_{n+1} = x_n - \frac{\left(1 - \frac{5}{x_n^2}\right)}{\left(\frac{10}{x_n^3}\right)}$$

$$x_{n+1} = x_n - \frac{\left(x_n^3 - 5x_n\right)}{10}$$

$$= \frac{10x_n - x_n^3 + 5x_n}{10}$$

$$x_{n+1} = \frac{\left(15x_n - x_n^3\right)}{10}$$

(b) Let $x_0 = 2$. Putting $n = 0$ in the above formula; we get,

$$x_1 = \frac{\left(15x_0 - 5x_n^3\right)}{10}$$

$$= \frac{(2 \times 15 - 8)}{10} = 2.2$$

$$x_2 = \frac{\left(15x_1 - x_0^3\right)}{10}$$

$$= \frac{\left(15 \times 2.2 - 2.2^3\right)}{10} = 2.2352$$

Writing subsequent iterates in the tabular form:

n	x_n
0	2
1	2.2
2	2.2352
3	2.2360
4	2.2362
5	2.2370
6	2.2373

As $|x_5 - x_6| < \frac{1}{2} \cdot 10^{-2}$ we may accept $x_6 = 2.2373$ as the required root, which is very close to $\sqrt{5} = 2.24$.

(c) If e_n is the error in x_n, we have,

$$x_n = \alpha + e_n$$
$$= \sqrt{5} + e_n$$

Using the iterative formula derived in part (a) above, we get,

$$x_{n+1} = \frac{(15x_n - x_n^3)}{10}$$
$$= \frac{15(\sqrt{5}+e_n) - (\sqrt{5}+e_n)^3}{10}$$
$$= \frac{10\sqrt{5} - 3\sqrt{5}e_n^2 - e_n^3}{10}$$

Ignoring higher powers of e_n^2 being small, we get,

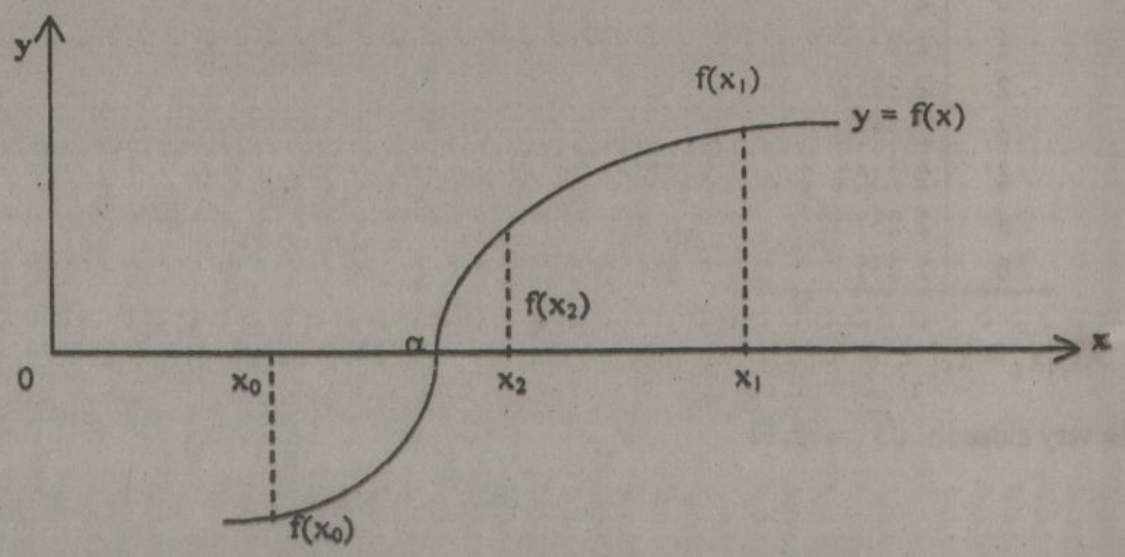
$$= \frac{10\sqrt{5} - 3\sqrt{5}e_n^2}{10}$$

Thus, $e_{n+1} = 0.67e_n^2$.

7.6 THE BISECTION METHOD

The **bisection method** (also called the **binary-search method**) is probably the most primitive procedure for finding a real root of (7.1) and is described as follows:

It requires two starting values x_0 and x_1 for the solution such that $f(x_0)f(x_1)<0$. Then, the equation (7.1) has at least one real root in the interval (x_0, x_1). We shall illustrate bisection method graphically by the following figure in which x_2, x_3, ... denote successive midpoints:



The basic procedure for the bisection method relies on repeated application of the following:

1. Find x_2, the new iterate, using $x_2 = \dfrac{x_0 + x_1}{2}$ and evaluate $f(x_2)$.
2. If $f(x_2) = 0$, then x_2 is a root of f(x).
3. If $f(x_2) \neq 0$, two things are possible:

   a) If $f(x_0)\,f(x_2) < 0$, we compute the new iterate x_3 as

   $$x_3 = \frac{x_0 + x_2}{2} \text{ and evaluate } f(x_3).$$

   b) If $f(x_1)\,f(x_2) < 0$, we compute the new iterate x_3 as

   $$x_3 = \frac{x_1 + x_2}{2} \text{ and evaluate } f(x_3).$$

The process is then repeated with new points until it is felt that the root is determined with sufficient accuracy. We note, however, that this method uses little information about the function, but only its sign. In other words, the heart of bisection method is the assumption that an interval $x_1 \le x \le x_2$ has been found, such that $f(x_1)f(x_2) < 0$ and the method undertakes to decrease the size of the interval.

This method is very simple, very slow to converge, always works for real roots when there is an odd number of roots in an interval [a, b], but the convergence is guaranteed. For this reason, this method is often used for solving non-linear equations. This method is particularly useful when we have to find the roots using a computer program. Most commercial root finding routines use a combination of the Newton-Raphson and the bisection methods. If one fails to converge, the routines switch to the other method to obtain a new estimate of the root. In some cases to avoid the pitfalls of Newton-Raphson method is to use this combination. We can use bisection to obtain an estimate of the root and then use Newton-Raphson method to find a more exact solution. Bisection method is also known as the half-interval method and the Bolzano method.

It is necessary to know that this method does not work for double roots.

Example 12 (a) Use the bisection method to find, correct to 4 dp, the root between 0.4 and 0.6 of the equation $\sin x - 5x + 2 = 0$.

(b) Write a computer program to implement the method.

Solution (a) Let $f(x) = \sin x - 5x + 2 = 0$

$$x_0 = 0.4, \quad x_1 = 0.6$$

$$f(x_0) = f(0.4) = \sin(0.4) - 5 \times 0.4 + 2 = 0.3894$$

$$f(x_1) = f(0.6) = \sin(0.6) - 5 \times 0.6 + 2 = -0.4354$$

Applying the bisection algorithm:

$$f(x_0)\,f(x_1) = 0.3894 \times -0.4354 = -0.1695 < 0$$

So, $x_2 = \dfrac{x_0 + x_1}{2} = \dfrac{0.4 + 0.6}{2} = 0.5$

$$f(x_2) = \sin(0.5) - 5 \times 0.5 + 2 = -0.0206$$

Since, $f(x_0)\,f(x_1) < 0$, $x_3 = \dfrac{x_0 + x_2}{2} = 0.45$

$$f(x_3) = \sin(0.5) - 5 \times 0.45 + 2 = 0.1850$$

$$x_4 = 0.475,\ f(x_4) = 0.0823$$

$$x_5 = 0.4875,\ f(x_5) = 0.0309$$

We continue in this manner until the required accuracy is achieved.

b) **Program No. 18: BISECTION METHOD**

```
#include<iostream.h>
#include<math.h>
#include<conio.h>
#define f(x) (sin(x) -5*x +2)

float e,x1,x2;

void main(void)
  {
    int i,x1,x2;
    float x3,f,f1 ,f2,f3,temp;
    cout<< "\n\n\tBISECTION METHOD \n";
    cout<<"\n\tEnter The Value Of X1 : ";
    cin>>x1;
    cout<<"\n\tEnter The Value Of X2 :  ";
    cin>>x2;
    cout<<"\n\tEnter The Value Of E : ";
    cin>>e;

    // x1=f(x1);
    for(i=0;i<50 && flag;i++)
      {
        temp=x2-x1;
        if(temp<0)
          temp*= (-1);
```

```
        if(temp<e)
          {
            flag=0;
            break;
          }
        f1=f(x1);
        f2=f(x2);
        f3=f1*f2;
        x3=(x1+x2)/2;
        f3=f(x3);
        if(f3==0)
          {
            cout<<"\n\n\tAFTER "<<i<<" ITERATIONS, ROOT IS  "<<x3;
            break;
          }
        f=f1*f3;
        if(f<0)
          x2=x3;
        else
          {
            f=f2*f3;
            if(f<0)
              x1=x3;
          }
      }
    if(flag==1)
      cout<<"\n\n\tSOLUTION DOES NOT EXIST";
    else
      cout<<"\n\n\tAFTER "<<i<<" ITERATIONS, ROOT IS   "<<x1;
    getch( );
  }
```

Computer Output

```
BISECTION METHOD
Enter The Value Of X1 : 0.4
Enter The Value Of X2 : 0.6
Enter The Value Of E  : 0.0005

AFTER 9 ITERATIONS, ROOT IS 0.494922
```

7.7 THE SECANT METHOD

The secant method is a modified form of Newton-Raphson method. If in Newton-Raphson method, we replace the derivative $f'(x_n)$ by the following difference ratio, i.e.,

$$f'(x_n) = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}}$$

where x_n and x_{n-1} are two approximations of the root, we get.

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

$$= \frac{x_n f(x_n) - x_n f(x_{n-1}) - f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

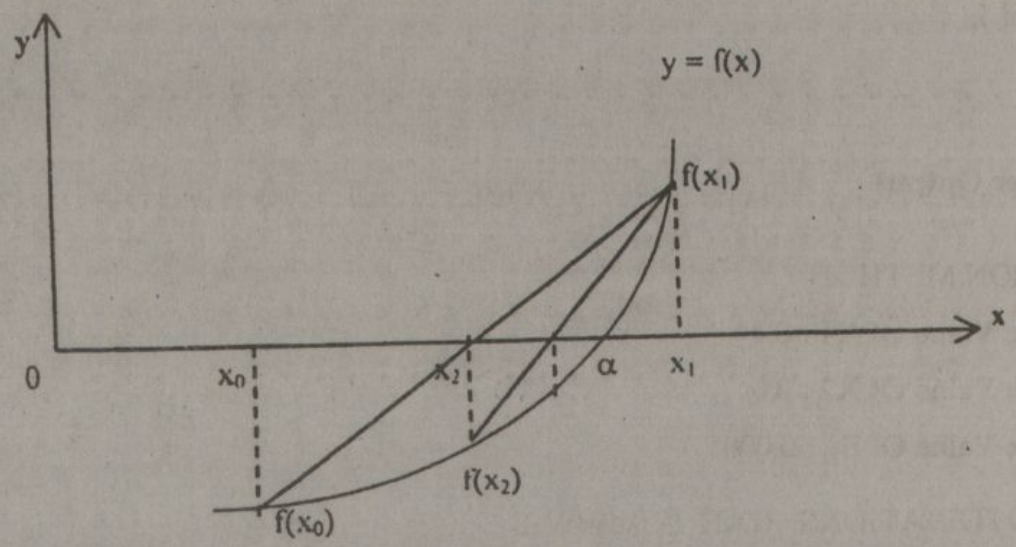
$$= \frac{x_{n-1} f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})} \qquad \ldots (7.16)$$

provided $f(x_n) \neq f(x_{n-1})$.

The secant method requires two starting values x_0 and x_1; values of $f(x_0)$ and $f(x_1)$ are calculated which give two points on the curve. The new point x_2 is obtained using (7.16). We can continue this process to get better estimates of the root. So, in this formula, we do not need $f'(x_n)$.

Geometric Interpretation

Geometrically, the secant method corresponds to drawing secants rather than tangents to obtain various approximations to the root α; hence the name.



To obtain x_2, we find the intersection between the secant through the points $(x_0, f(x_0))$ and $(x_1\ f(x_1))$ and the x-axis.

It is experienced that the secant method converges rather quickly. Then: is a possibility of divergence if the two roots lie on the same side or the curve. The order of convergence of secant method is equal to $\frac{\left(1+\sqrt{5}\right)}{2} = 1.61803$, which shows that this method has the order of convergence slightly inferior to that of Newton-Raphson method. In this method, f(x) is not required to change signs between the estimates.

Example 14 (a) Find the root of 2 cos hx sin x = 1, using the secant method, with an accuracy of 4 dp. Take 0.4 and 0.5 as the two starting values.

(b) Reconsider Example 12 and write computer program using secant method.

Solution (a) Let $x_0 = 0.4$ and $x_1 = 0.5$

$$f(x) = 2\cos hx \sin x - 1$$

$$f(x_0) = 2\cos h\, x_0 \sin x_0 - 1$$

$$= 2 \times 1.081 \times 0.3894 - 1 = -0.1580$$

$$f(x_1) = 2\cos h\, x_1 \sin x_1 - 1$$

$$= 2 \times 1.1276 \times 0.4794 - 1 = -0.0811$$

Putting n = 1 in the secant formula, we get,

$$x_2 = \frac{x_0 f(x_1) - x_1 f(x_0)}{f(x_1) - f(x_0)}$$

$$= \frac{0.4 \times 0.0811 - 0.5 \times -0.1580}{0.0811 - 0.1580}$$

$$= \frac{0.03244 - 0.979}{0.2391} = 0.4661$$

$$f(x_2) = 2\cos h\, x_2 \sin x_2 - 1$$

$$= 2 \times 1.1106 \times 0.4494 - 1 = -0.0018$$

$$x_3 = \frac{x_1 f(x_2) - x_2 f(x_1)}{f(x_2) - f(x_1)}$$

$$= \frac{0.5 \times -0.0018 - 0.4661 \times 0.0811}{-0.0018 - 0.0811}$$

$$= \frac{0.009 - 0.0378}{-0.0828} = 0.4668$$

$$f(x_3) = 2\cos h\, x_3 \ \sin x_3 - 1 = -0.00009$$

$$x_4 = \frac{x_2 f(x_3) - x_3 f(x_2)}{f(x_3) - f(x_2)}$$

$$= \frac{0.4661 \times -0.00009 - 0.4668 \times -0.0018}{-0.00009 + 0.0018}$$

$$= \frac{-0.000042 + 0.00048}{-0.00171} = 0.4667$$

The root, correct to 3 dp, is 0.467.

(b) **Program No. 19: Secant Method**

```
# include<iostream.h>
# include<math.h>
# include<conio.h>
# define f(x) (sin(x) -5*x+2)

int n;
float e,x0,x1;

void main(void)
  {
    int i,flag;
    float x2,f1 ,f0,x3,temp,denom;
    char ch;

    cout<<"\n\n\tSECANT METHOD";
    cout<<"\n\n\tENTER THE VALUE OF X0 : ";
    cin>>x0;
    cout<<"\n\n\tENTER THE VALUE OF X1 : ";
    cin>>x1;
    cout<<"\n\n\tENTER THE VALUE OF N : ";
    cin>>n;
    cout<<"\n\n\tENTER THE VALUE OF E :  ";
    cin>>e;
    for(i=0;i<n && flag;i++)
      {
        f0=f(x0);        f1=f(x1);
        denom=f1-f0;  temp=denom;

        if(temp=<0)
```

```
        temp=temp*(-1 );
      if(temp<e)
        {
          cout<<"\n\n\tDENOMINATOR TOO SMALL";
          flag=0;   break;
        }
      x2=((x0*f1)-(x1*f0))/denom;
      x3=x2-x1;
      if(x3<0)
        x3=x3*(-1);
      if(x3<e)
        {
          cout<<"\n\n\tAFTER "<<i<<" ITERATIONS, ROOT IS :  "<<x2;
          flag=0;
        }
      x0=x1;
      x1=x2;
    }
  if(flag=1)
    cout<<"\n\n\tNO CONVERGENCE";
  getch( );
}
```

Computer Output

SECANT METHOD

ENTER THE VALUE OF X0 : 0.4

ENTER THE VALUE OF X1 : 0.6

ENTER THE VALUE OF N : 20

ENTER THE VALUE OF E : 0.0005

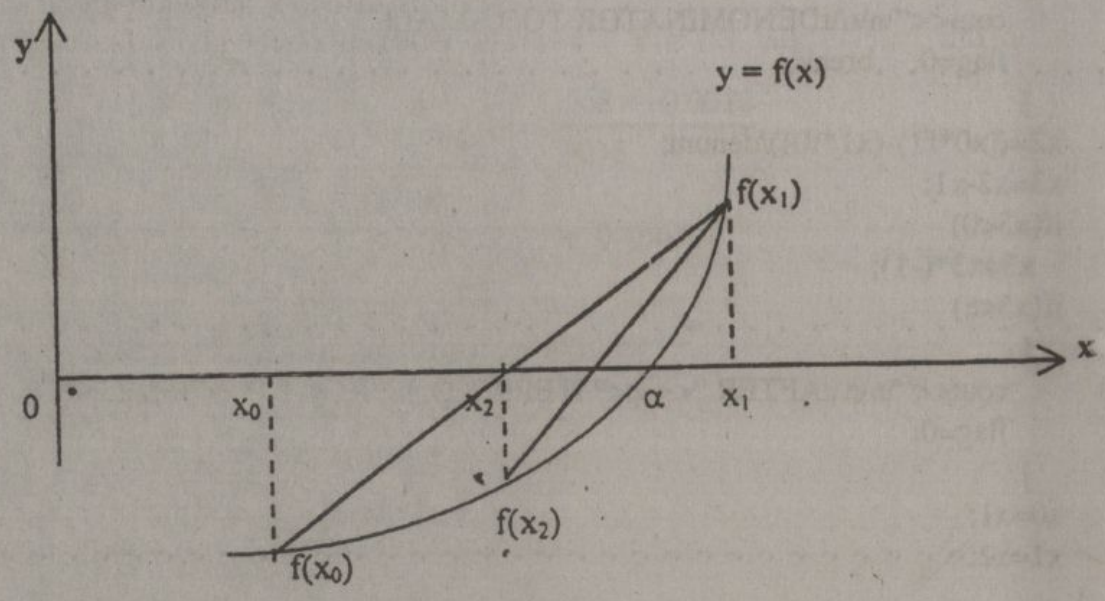
AFTER 2 ITERATIONS, ROOT IS : 0.495008

7.8 METHOD OF FALSE POSITION AND ITS MODIFIED FORM

A simple modification of the secant method produces a method which usually converges. The new method is called **regula falsi (false position)** and also **linear interpolation**.

It needs two initial approximations x_0 and x_1 so that $f(x_0)\, f(x_1) < 0$, i.e., the two functions must have opposite signs. The value of x_2 is found as the intersection

between the chord joining $f(x_0)$ and $f(x_1)$ and the x-axis. It is illustrated graphically below:



The formula for the regula falsi is as follows:

$$x_{n+1} = \frac{x_{n-1} f(x_n) - x_n f(x_{n-1})}{f(x_n) - f(x_{n-1})} \qquad \ldots (7.17)$$

provided $f(x_n)\, f(x_{n-1}) < 0$.

The regula falsi method is the same as the secant method, except that the condition $f(x_n)\, f(x_{n-1}) < 0$ should meet at each step.

Convergence may be more rapid than by the bisection method, but there is no assurance that this will always be the case. The number of iterations required for satisfactory convergence will depend on the shape of the graph of the function in the interval that has been found to contain a root.

The fact that the replacement of the curve by a straight-line gives a **false position** of the root, is the origin of the name method of false position or in Latin, regula falsi. The main weakness in (7.17) is that it is slow.

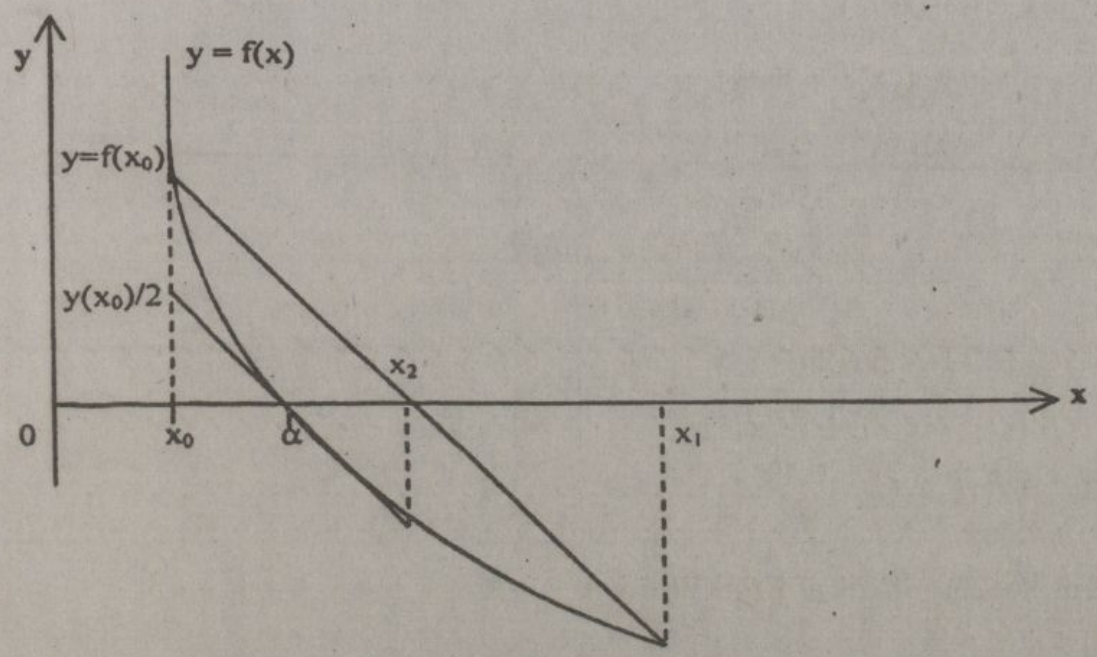
One pitfall of this method is **stagnation** of an end point.

It means that one end of successive intervals does not move from the original end point, so that the approximations for the root denoted by $x_2, x_3, x_4, \ldots$ converge to the exact root a from one side only. Stagnation is not desirable because it slows down convergence, particularly when the initial interval is very large or when the function deviates significantly from a straight line in the interval. The difficulty is avoided by the **modified false position method**, which is explained below:

The Modified False Position Method

In this method, the f(x) value of a stagnant end point is halved, if that point has repeated twice or more. The end point that repeats is called a **stagnant point**. The exception to this rule is that for i = 2, the f(x) value at one end is divided by 2 immediately, if it does not move.

The algorithm is illustrated in the figure below:



The effect of halving the y value is that the solution of the linear interpolation becomes closer to the true root α.

Example 16 (a) Use the method of false position for finding the root correct to 4 dp between 0.4 and 0.6 of the equation sinx = 5x – 2.

(b) Write also the computer program to implement the method.

Solution (a) $f(x) = \sin x - 5x + 2;\ x_0 = 0.4,\ x_1 = 0.6$

$$f(x_0) = \sin(0.4) - 5 \times 0.4 + 2 = 0.389$$

$$f(x_1) = \sin(0.6) - 5 \times 0.6 + 2 = -0.435$$

Since $f(x_0)\,f(x_1) < 0$, therefore,

$$x_2 = \frac{x_0 f(x_1) - x_1 f(x_0)}{f(x_1) - f(x_0)}$$

$$= \frac{0.4 \times -0.435 - 0.6 \times 0.389}{-0.435 - 0.389} = 0.494$$

$$f(x_2) = \sin 0.494 - 5 \times 0.494 + 2 = 0.0042$$

Since $f(x_1)\,f(x_2) < 0$, therefore,

$$x_3 = \frac{x_1 f(x_2) - x_2 f(x_1)}{f(x_2) - f(x_1)}$$

$$= \frac{0.6 \times 0.0042 - 0.494 \times -0.435}{0.0042 - (-0.435)} = 0.4950$$

$$f(x_3) = 0.00003$$

Since $f(x_1)\,f(x_3) < 0$, therefore,

$$x_4 = \frac{x_1 f(x_3) - x_3 f(x_1)}{f(x_3) - f(x_1)}$$

$$= \frac{0.6 \times 0.00003 - 0.4950 \times -0.435}{0.00003 - (-0.435)}$$

$$= \frac{0.00008 + 0.2153}{0.43503} = 0.4949$$

Therefore, the root is 0.4949.

(b) **Program No. 20: Rule of False Position**

```
# include<iostream.h>
# include<math.h>
# include<conio.h>
# include<processing.h>
# define f(x) (sin(x) -5*x+2)

int n;
float e,x0,x1;

void main(void)
  {
    int i,flag=1;
    float x2,f1 ,f0,x3,temp,denom,x4,f,f2;
    char ch;

    cout<<"\n\n\tRegula Falsi Method";
    cout<<"\n\n\ tEnter The Value Of X0\t";
    cin>>x0;
    cout<<"\n\n\ tEnter The Value Of X1 : ";
    cin>>x1;
    cout<<"\n\n\ tEnter The Value Of N : ";
```

```
cin>>n;
cout<<"\n\n\ tEnter The Value Of E :  ";
cin>>e;

for(i=0;i<n && flag;i++)
  {
    f0=f(x0);
    f1=f(x1);
    denom=f1-f0;
    temp=denom;
    if(temp=<0)
      temp=temp*(-l );
    if(temp<e)
      {
        cout<<"\n\n\ tDenominator Too Small";
        flag=0;
        break;
      }
    f=f0*f1;
    if(f>0)
      {
        cout<<"\n\n\tThere is no root";
        flag=0;
        break;
      }
    x2=((x0*f1)-(x1*f0))/denom;
    f2=f(x2);
    f=f2*f1;
    x3=x2-x1;
    if(x3<0)
      x3=x3*(-1);
    if(x3<e)
      {
        cout<<"\n\n\tAfter\t" "<<i<<"\tIterations, Root Is\t"<<x2;
        getch( );
        flag=0;
      }
    if(f<0)
        x0=x2;
    else
        x1=x2;
    x4=x1-x0;
    if(x4<0)
        x4=x4 *( -1);
```

```
        if(x4<e)
          {
            cout<<"\n\nNo Convergence";
            getch( );
            flag=0;
          }
      }
    if(flag=1)
      {
        cout<<"\n\n\tValue of root :  "<<x2;
        getch( );
      }
  }
```

Computer Output

Regula Falsi Method

Enter The Value Of X0	0.4
Enter The Value Of X1	0.6
Enter The Value Of N	10
Enter The Value Of E	0.0005

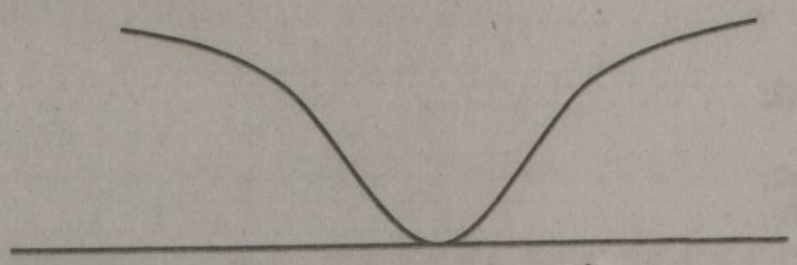
Value Of Root : 0.495008

7.9 DETERMINATION OF MULTIPLE ROOTS

Root-finding methods are widely known to have problems with multiple and nearly multiple, i.e., close roots. A multiple root corresponds to a point where a function is tangential to the x-axis. For example, a double root results from,

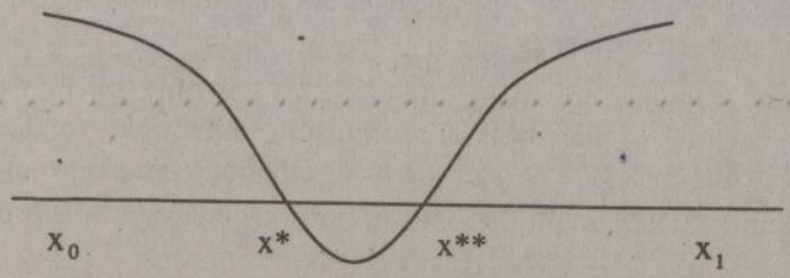
$$f(x) = x^3 - 5x^2 + 7x - 3$$
$$= (x-3)(x-1)(x-1)$$

Graphically, this corresponds to the curve touching the x-axis tangentially at the double root (See figures below):



Double root at x: x_0 x^* x_1

Two close roots:



Mathematically,

i) If $f(x_0) = 0$, then $(x - x_0)$ is a factor of f(x) and conversely, and

ii) If $f(x - x_0)^k$ is a factor, then x is a zero of multiplicity k.

It has been pointed out in the literature that for single roots the rate of convergence of Newton-Raphson method is quadratic, but for multiple roots it is linear; unless the method is modified. Thus, a slight modification in Newton-Raphson method can be made to handle this situation when the multiplicity is known. Instead of the usual procedure, several methods have been suggested and we first mention the following formula:

$$x_{n+1} = x_n - \frac{mf(x_n)}{f'(x_n)} \qquad \ldots (7.18)$$

where m is the multiplicity of the root. Thus, this modification can restore quadratic convergence. Obviously, this may be an unsatisfactory formulation because in actual practice we do not usually know the multiplicity of the root in advance. We might guess at the value for m and see whether we get quadratic convergence. In theory, one might estimate m from the successive iterates, but the labour seems of a questionable value, so the difficulty remains. However, if m is known before the above formula can accelerate Newton's method.

Another formula due to Newton-Raphson has been suggested to deal with multiple roots:

$$x_{n+1} = x_n - \frac{f(x_n)f'(x_n)}{[f'(x_n)]^2 f(x_n)f''(x_n)} \qquad \ldots (7.19(a))$$

Halley's method is another way to deal with multiple roots:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}\left[1 - \frac{f(x_n)f''(x_n)}{2[f'(x_n)]^2}\right]$$

$$= x_n - \frac{2f(x_n)\cdot f'(x_n)}{2[f'(x_n)]^2 - f(x_n)\cdot f''(x_n)} \qquad \ldots (7.19(b))$$

The term in brackets is the modification of the Newton-Raphson method. Halley's method yields cubic convergence at simple zeros of f(x). In these formulas, multiplicity has been removed.

These formulas are more attractive for speeding up rate of convergence and improving accuracy of the roots. Theoretically, drawbacks to these methods are the additional calculation of $f''(x)$ and the more laborious procedure of calculating iterates. In fact, the presence of a multiple root can cause severe round-off problems

7.10 ZEROS OF POLYNOMIALS

Polynomial equations are frequently used in practice and a vast literature is available to find their roots. The methods discussed so far in this chapter are used to find a zero of a polynomial, but the present section is devoted to find all zeros of a polynomial. We shall confine our treatment only to polynomials with real coefficients; thus, we propose to find real linear and real quadratic factors. If the complex zeros are required they are easily found from the real quadratic factors by using quadratic formula for the zeros.

A polynomial of degree n is a function of the form,

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0 \qquad \ldots (7.20)$$

where the coefficients a_i are real constants and $a_n \neq 0$.

This method is merely the Newton-Raphson method applied to the polynomial equation with the synthetic division method.

7.10.1 Evaluation of a Polynomial (Birga-Vieta Method)

We use the nested polynomial (synthetic division) scheme, which is more efficient and faster, because it takes only n additions and n multiplications to evaluate an nth-degree polynomial. It is also known as Horner's scheme.

The polynomial (7.20) is rewritten in the form:

$$p(x) = (((a_n x + a_{n-1}) x + a_{n-2}) x + \ldots + a_1) x + a_0 \qquad \ldots (7.21)$$

and a particular value of x is evaluated from the innermost bracket outwards.

When we perform hand calculations using Horner's method, we first construct the synthetic division table:

Input x	a_n	a_{n-1}	a_{n-2}	$\cdots$	a_2	a_1	a_0
		$x\,b_n$	$x\,b_{-1}$	$\cdots$	$x\,b_3$	$x\,b_2$	$x\,b_1$
	b_n	b_{n-1}	b_{n-2}	$\cdots$	b_2	b_1	$b_0 = p(x)$
							Output (remainder term)

$$\left.\begin{aligned}
b_n &= a_n \\
b_{n-1} &= a_{n-1} + x\,b_n \\
b_{n-2} &= a_{n-2} + x\,b_{-1} \\
&\vdots \\
b_k &= a_k + b_{k+1} \;;\quad k = n-1, n-2, \ldots, 1, 0 \\
&\vdots \\
b_1 &= a_1 + x\,b_2 \\
b_0 &= a_0 + x\,b_1
\end{aligned}\right\} \qquad \ldots (7.22)$$

If input is α and it is not a root of the polynomial, the remainder term due to the synthetic division is the value of p(x) for $x = \alpha$. When p(x) is of degree n and is divided by $(x - \alpha)$, we obtain, after one synthetic division,

$$p_n(x) = (x - \alpha)\,p_{n-1}(x) + r \qquad \ldots (7.23)$$

where p_{n-1} is the quotient and r is the remainder term which is a constant. The above process of division is called **deflation**. If α is an actual root of $p_n(x)$, then the remainder is zero.

Thus, $p_n(x) = (x - \alpha)\,p_{n-1}(x)$ ... (7.24)

7.10.2 Evaluation of Derivatives of Polynomials

Continuing the synthetic division once more, we get,

Input x	a_n	a_{n-1}	a_{n-2}	$\cdots$	a_2	a_1	a_0
		$x\,b_n$	$x\,b_{-1}$	$\cdots$	$x\,b_3$	$x\,b_2$	$x\,b_1$
	b_n	b_{n-1}	b_{n-2}	$\cdots$	b_2	b_1	$b_0 = p(x)$
		$x\,c_n$	$x\,c_{-1}$	$\cdots$	$x\,c_3$	$x\,c_2$	
	c_n	c_{n-1}	c_{n-2}	$\cdots$	c_2		$c_1 =$ Remainder $= p'(x)$

The value of the derivative, $p'(x)$, of a polynomial, for $x = \alpha$ is equal to the remainder obtained in the first synthetic division. Continuing the process, it is possible to compute higher-order derivatives.

Differentiating (7.23) w.r.t.x, we have,

$$p'_n(x) = (x-\alpha)\,p'_{n-1}(x) + 1\cdot p_{n-1}(x) + 0$$

$$= (x-\alpha)\,p'_{n-1}(x) + p_{n-1}(x) \qquad \ldots (7.25)$$

When $x = \alpha$, $p'_n(x) = p_{n-1}(x)$, which is a polynomial of degree $(n-1)$. Since we have computed $p(x)$ and $p'(x)$ at α, using Horner's method, we can now use Newton-Raphson method to evaluate a root of this polynomial.

A problem with applying Newton-Raphson method to polynomials concerns the possibility of the polynomial having complex roots even when all the coefficients are real numbers. If the initial approximation using Newton-Raphson method is a real number, all subsequent approximations will also be real numbers. One way to overcome this difficulty is to begin with a non-real initial approximation and do all the computations using complex arithmetic.

Example 9 Consider the polynomial, $x^3 + x^2 - 10x + 8$.

(a) Evaluate $p(0.5)$ and $p'(0.5)$.

(b) Starting with initial approximation $x_0 = 0.5$ for the real root of the above polynomial, use Horner's scheme and Newton-Raphson method to compute all its roots, correct to 3 dp.

Solution (a) Evaluation of the polynomial and its derivative at $x = 0.5$.

First Iteration:

Input	1	1	−10	8	← coefficients of the polynomial
$x_0 = 0.5$		0.5	0.75	−4.625	
	1	1.5	−9.25	$3.375 = p(x_0)$	
		0.5	1.0		
	1	2.0	−8.25	$= p'(x_0)$	

$p(x_0) = 3.375$; $p'(x_0) = -8.25$

accuracy, $e = 0.0005$

(b) Computation of roots using Newton-Raphson method:

$$x_1 = x_0 - \frac{p(x_0)}{p'(x_0)}$$

$$= 0.5 - \frac{3.375}{-8.25} = 0.909$$

$$|x_1 - x_0| = |0.909 - 0.5| \not< e.$$

Continue the process

Second Iteration:

$x_1 = 0.909$	1	1	−10	8
		0.909	1.735	−7.513
	1	1.909	−8.265	$0.487 = p(x_1)$
		0.909	2.563	
	1	2.818	−5.703	$= p'(x_1)$

$p(x_1) = 0.487$

$p'(x_1) = -5.703$

$$x_2 = x_1 - \frac{p(x_1)}{p'(x_1)}$$

$$= 0.909 - \frac{0.487}{-5.703} = 0.994$$

$$|x_2 - x_1| = |0.994 - 0.909| \not< e.$$

Continue the process

Third Iteration:

$x_2 = 0.994$	1	1	−10	8
		0.994	1.982	−7.970
	1	1.994	−8.018	$0.030 = p(x_2)$
		0.994	2.970	
	1	2.988	−5.048	$= p'(x_2)$

$p(x_2) = 0.0307$

$p'(x_2) = -5.048$

$$x_3 = x_2 - \frac{p(x_2)}{p'(x_2)}$$

$$= 0.994 - \frac{0.030}{-5.048} = 0.9999$$

$|x_3 - x_2| = |0.9999 - 0.994| \not< e.$

Continue the process

Fourth Iteration:

$x_3 = 0.994$	1	1	– 10	8
		0.9999	1.9997	– 7.9994
	1	1.9999	– 8.0003	$0.0005 = p(x_3)$
		0.9999	2.9995	
	1	2.9998	– 5.0008	$= p'(x_3)$

$p(x_3) = 0.0005$

$p'(x_3) = -5.0008$

$$x_4 = x_3 - \frac{p(x_3)}{p'(x_3)}$$

$$= 0.9999 - \frac{0.0005}{-5.0008} = 0.9999$$

$|x_4 - x_3| = |0.9999 - 0.9999| < 0.0005.$

$(x - 0.9999)(x^2 + 1.9999x - 8.0003) = 0$

Approximately, the above equation may be rewritten as:

$$(x - 1)(x^2 + 2x - 8) = 0$$

Factorising, we get:

$$(x - 1)(x + 4)(x - 2) = 0$$

Thus, the possible roots are 1, 2, and – 4.

PROBLEMS

1. Use graphical method to find the approximate root of the following equations:

   (a) $x^3 - x + 1 = 0$, (b) $x^4 + 3x - 1 = 0$, (c) $e^x - 3x = 0$

2. (a) The cubic equation $x^3 - 2x - 5 = 0$ has one real root that is near to $x = 2$. The equation can be rewritten in the following manner:

   (i) $x = \frac{1}{2}(x^3 - 5)$ (ii) $x = \frac{5}{x^2 - 2}$ (iii) $x = (x^2 + 5)^{\frac{1}{3}}$

   Choose the form which satisfies the condition $|\Phi'(x)| < 1$ and find the root correct to 4 dp.

   (b) The cubic equation $x^3 - 3x - 20 = 0$, has one real root that is near to $x_0 = 0.3$. The equation can be rewritten in the following manner:

   (i) $x = \frac{1}{3}(x^3 - 20)$ (ii) $x = \frac{20}{x^2 - 3}$ (iii) $x = \sqrt{3 + \frac{20}{x}}$ (iv) $x = (3x + 20)^{\frac{1}{3}}$

   Choose the form which satisfies the condition $|\Phi'(x)| < I$ and find the root correct to 4 dp. Which of them gives rise to very rapid convergence?

   (c) Given the following variations of the equation, $x^4 + x^2 - 80 = 0$,

   (i) $x = (80 - x^2)^{\frac{1}{4}}$ (ii) $x = \sqrt{80 - x^4}$ (iii) $x = \sqrt{\frac{80}{1 + x^2}}$

   Which of them gives rise to a convergent sequence? Find the real root of the equation correct to 4 dp. Take $x_0 = 3$.

3. (a) To locate the root of $e^{-x} - \cos x = 0$ that is near to 1.29, using iteration, we could rewrite the equation as,

   (i) $x = \cos^{-1}(e^{-x})$ (ii) $x = -\log \cos x = \log \sec x$

   (iii) $x = x - 0.01(e^{-x} - \cos x)$

   Which of these three forms (if any) would yield a convergence iteration scheme? Which would converge the fastest?

   (b) Starting with $x_0 = 6$, perform ten iterations using each of the recurrence relations

   (i) $x_{n+1} = \sqrt{5x_n - 4}$ (ii) $x_{n+1} = \frac{x_n - 4}{2x_n - 5}$

   which of (i) and (ii) has the higher rate of convergence?

(c) Determine which of the following iterative functions, $\Phi(x)$, can be used to locate the zeros of the equation $x^3 + 2x - 1 = 0$ on the interval $\left[\frac{1}{4}, \frac{1}{2}\right]$:

(i) $\frac{1}{2}(1 - x^3)$ (ii) $\frac{(1-2x)}{x^2}$ (iii) $\frac{x^3}{(1-x)}$ (iv) $(1-2x)^{\frac{1}{3}}$

(v) $x - 0.2(x^3 + 2x - 1)$ (vi) $\frac{x^3 + 2x - 1}{3x^3 + 2}$

(d) (i) Starting at $x_0 = 0$, use the simple iterative method to find the first five approximations for the solution of $x^4 - x + 0.12 = 0$.

(ii) Starting at $x_0 = 1$, use the simple iterative method to find the first five approximations for the solution of $x^3 - x\sqrt{x} - 4 = 0$.

(iii) Compute a solution, correct to 6 dp, of $e^{-3x} - \cot x = 0$ by Newton's method starting at $x_0 = 1$.

(iv) Compute a solution, correct to 6 dp, of $x^3 - x \sin x = 0$ by Newton's method starting at $x_0 = 1$.

(v) Find a rearrange of the equation $e^x - 3x - 1 = 0$, which will converge to the unique positive root when the simple iterative method is applied. Take $x_0 = 2$.

(e) The cubic equation $2x^3 + 3x^2 - 3x - 5 = 0$ has a root near $x = 1.25$. Show that the equation can be rearranged into any of the following three forms suitable for the simple (fixed-point) iterative method:

(i) $x = \left\{\frac{(5 - 3x - 3x^2)}{2}\right\}^{\frac{1}{3}}$

(ii) $x = ((5+32)/(22+3))^{\frac{1}{2}}$

(iii) $x = \frac{(2x^3 + 3x^2 - 5)}{3}$

Use simple iterative method on the rearranged equation (i) with an initial guess of $x_0 = 1.2$ in order to find the root to 4 dp.

Repeat part (b) for the rearrangement (ii) using $x_0 = 1.2$. Which method converges faster? Why?

Try a few iterations using rearrangement(iii). What goes wrong?

4. Use Newton-Raphson method to obtain a root of each of the following equations correct to 3 dp:

(a) $x^3 - 2x + 2 = 0$; with $x_0 = 0.2$

(b) $x^3 - 3x - 3 = 0$; with $x = 2$

(c) $x^6 - x - 1 = 0$; with $x_0 = 0.5$

(d) $\sin x - 5x + 2 = 0$; with $x_0 = 0.4$

(e) $\cos x - x = 0$; with $x_0 = 0.74$

(f) $e^{-x} - x = 0$; with $x_0 = 0$

(g) $e^x - 3x^2 = 0$; with $x_0 = 1$

(h) $\sin x - x + 1 = 0$; with $x_0 = 1.5$

(i) $\tan x - 0.5x = 0$; with $x_0 = 4.0$

(j) $x^2 = e^x$; with $x_0 = -1$

(k) $x^4 + x^2 = 80$; with $x_0 = 3$

(l) $x \sin x = 1$; with $x_0 = 1.11$

(m) $x \ln x = 3$; with $x_0 = 2$

(n) $x^3 - 2x^2 + x - 3$; with $x_0 = 4$

5. (a) By applying Newton-Raphson method to the function defined by $f(x) = 1 - \frac{10}{x^2}$, develop an iterative formula for calculating $\sqrt{10}$. Hence, using 2 as an initial approximation to $\sqrt{10}$, calculate $\sqrt{10}$ correct to 2 dp. Show that if x_n, the nth approximation to $\sqrt{10}$, has a small error e_n, then the correct approximation e_{n+1} has an error of magnitude about $0.5\,e_n^2$.

(b) Use the following iterative formula for $\frac{1}{\sqrt{a}}$ to find $\frac{1}{\sqrt{5}}$ to 4 dp:

$$x_{n+1} = \frac{1}{2} x_n (3 - a x_n^2).$$

(c) Show that the curve $f(x) = x^3 - 2x - 1$ crosses the x-axis between $x = 1$ and $x = 2$. Use a recurrence relation of the form,

$$x_{n+1} = x_n - \frac{f(x_n)}{m}$$

where (i) $m = 5$ and (ii) $m = f'(x) = 3x^2 - 2$, to find the value of the root to 3 dp. Take $x_0 = 2$ in both cases.

(d) Given $f(x) = x^3 - a$, use Newton's method to establish the recurrence relation,

$$x_{n+1} = \frac{1}{3}\left[2x_i + \frac{a}{x_i^2}\right]$$

for the cubic root of a number a. With $a = 9$ and $x_0 = 3$, perform the iterations 6 times to find the cubic root of 9.

6. (a) Starting with $x_0 = 8$, perform 10 iterations using the following iterative formula:

$$x_{n+1} = 9 - \frac{20}{x_n}$$

Use Aitken's iterative method with x_7, x_8 and x_9, to improve the rate of convergence.

(b) Use Aitken's iterative method to find the root of $e^x = 5x$ near to $x_0 = 0.3$. Compare the result with iteration $x_{n+1} = \frac{1}{5}e^x$ starting with $x_0 = 0.3$.

7. Use bisection method to find correct to 4 dp, the solutions of the following equations:

(a) $\sin x - \frac{1}{2}x = 0$; in the interval $\left(\frac{\pi}{2}, \pi\right)$.

(b) $x^3 - 9.0x + 1.0 = 0$; $x_1 = 2$, $x_2 = 4$.

(c) $9x^3 + 4x^2 + 5x - 8 = 0$; $x_1 = -5$, $x_2 = 5$.

(d) $8x^3 + 8x - 5 = 0$; $x_0 = 0.3$, $x_1 = 0.6$

(e) $x \sin x - 1 = 0$; $x_0 = 0$, $x_1 = 2.0$

8. Use secant method to find, correct to 4 dp, the solutions of the following equations:

(a) $x^3 - 9x + 1 = 0$; $x_0 = 3$, and $x_1 = 4$

(b) $\sin x - 5x + 2 = 0$; $x_0 = 0.4$, and $x_1 = 0.5$

(c) $x^3 - 5 = 0$; $x_0 = 0$, and $x_1 = 3.0$

(d) $x^3 = x - 2$ $x_0 = 2.6$ and $x_1 = -2.4$

(e) $x^3 - 3.23x^2 - 5.54x + 9.84 = 0$; $x_0 = 0.9$ and $x_1 = 1.0$

9. Use Regula Falsi method to find, correct to 4 dp, the solutions of the following equations:

(a) $x^6 = x + 1$; $x_0 = 1$, $x_1 = 1.2$

(b) $x^3 - 9x + 1 = 0;\ x_0 = 2.0,\ x_1 = 4.0$

(c) $2x^3 + 7x - 1 = 0;\qquad x_0 = 0,\ x_1 = 1$

(d) $e^x - 2 = 0;\qquad x_0 = 0,\ x_1 = 1$

(e) Find $\sqrt{7}$; $x_0 = 0,\ x_1 = 1$

(f) $x \sin x - 1 = 0;\ x_0 = 0,\ x_1 = 2$

10.(a) Let $x_0 = 0.5$ and $x_1 = 0.8$ be the initial approximations to a root of the equation $x - \frac{1}{4}\sin x - \frac{1}{2} = 0$. Compute the next three approximations correct to 5 dp to the root using the following iterative methods:

i) Linear iteration

ii) Newton-Raphson method

iii) Secant method

iv) Bisection method and

v) Regula falsi method

(b) Consider the equation $230x^4 + 18x^3 + 9x^2 - 22x - 9 = 0$ has two real roots, one in $[-1, 0]$ and the other in the interval $[0, 1]$. Attempt to approximate these roots to within 10^{-6} using (i) Method of false position, (ii) Secant method and (iii) Newton's method. Use the end points of each interval as the initial approximations in (i) and (ii); the midpoints as the initial approximation in (iii).

11.(a) Find the value of the polynomial $5x^4 - 2x^3 + x - 1$ and its derivative at the point $x_0 = 2$, using the remainder theorem and the synthetic division process.

(b) Find the value of the polynomial, $4x^4 - 2x + 5x - 3$ at the point $x = -2$ by the rested multiplication method.

(c) Find the value of the polynomial $3x^4 - x^3 + 2x^2 + x - 7$ and its first two derivatives at $x = -2$.

12.(a) Consider the polynomial,

$$p(x) = x^5 - 6x^4 + 8x^3 + 8x^2 + 4x - 40$$

Starting with initial approximation $x_0 = 3$, evaluate $p(3)$ and $p'(3)$. Using Horner's scheme with Newton-Raphson method, compute the next two approximations correct to 2 dp.

(b) Find the real roots of the polynomial equation $x^4 - 5x^3 + 5x^2 + 5x - 7 = 0$, correct to 4 dp, given that the equation has roots near 3 and -1.

13.(a) Show that the polynomial, $p(x) = x^4 - 3x^3 - 7x^2 + 15x + 18$, has a root of multiplicity 2 at $x = 3$. Find the iterates and the values of $p(x)$ and $p'(x)$ at each iterate.

(b) Given the function, $f(x) = x^3 - 3x + 2$. If $m = 2$ and $x_0 = 1.5$, find the roots x_1, x_2 and x_3 using Newton-Raphson formula (7.18).

(c) Use formula (7 .19(a)) and find the roots x_1, x_2 and x_3 of $f(x) = x^3 + 4x^2 - 10$. Let $x_0 = 1.5$.

(d) (i) Perform three iterations of Newton's method to obtain the double root of $x^3 - 2x^2 - 0.75x + 2.25 = 0$ which is close to 1, such that iterations converge quadratically,

(ii) Compare your results with Newton's method without modification which converges linearly.

14.(a) Start with $f(x) = x^2 - a$ and find Halley's iterative formula for computing $\sqrt{a}$. Taking $a = 5$, and $x_0 = 2$, compute x_1, x_2 and x_3.

(b) Start with $f(x) = x^3 - 3x + 2$ and use Halley's formula to compute x_1, x_2 and x_3. Let $x_0 = -2.4$.

Chapter 8

Linear Systems of Equations

8.1 BASIC CONCEPTS

Consider a set of m simultaneous linear algebraic equations in n unknown, x_1, x_2, ..., x_n:

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + \cdots + a_{2n}x_n &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + \cdots + a_{3n}x_n &= b_3 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + a_{m3}x_3 + \cdots + a_{mn}x_n &= b_m \end{aligned}\right\} \quad \ldots (8.1)$$

In a more compact notation, the above equations can be rewritten as:

$$\sum_{j=1}^{n} a_{ij}x_j = b_i \text{; for } i = 1, 2, \ldots, m.$$

Three type of quantities occur here:

(a) The unknowns, $x_1, x_2, \ldots, x_n$.

(b) The coefficients a_{ij}, where $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, n$

(a) The right hand sides, $b_1, b_2, \ldots, b_m$.

Equations (8.1) can also be written in the matrix notation, as,

$$Ax = b \quad \ldots (8.2)$$

$$\text{where } A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ a_{31} & a_{32} & \cdots & a_{3n} \\ \cdots & \cdots & \cdots & \cdots \\ & & \vdots & \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}; \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ \vdots \\ x_n \end{bmatrix}; \quad b = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ \vdots \\ b_m \end{bmatrix}$$

A is a rectangular matrix having m rows and n columns, x and b are column vectors.

Problems of this type occur in almost all disciplines. Our aim is to develop methods, which can solve such problems and are easily implemented on a digital computer.

8.2 METHODS TO SOLVE A SYSTEM OF LINEAR EQUATIONS

Various methods have been devised to solve systems of linear equations. This shows that no single method is best suited to all situations. These methods should be judged on the basis of their speed and accuracy. Speed is of importance in solving large systems because of the large volume of computations involved and accuracy is necessary because of the round off errors involved in performing these computations.

The methods for solving systems of linear equations can be classified as:

i) Direct methods

ii) Indirect (iterative) methods

By a **direct method**, we mean a method which calculates the required solution without any initial or intermediate approximations in a finite number of steps. Amongst the direct methods, we will describe the following:

a) Cramer's rule and its modified form

b) Gaussian elimination method and its variations

c) Triangular decomposition method

d) Solution of tridiagonal system of equations

An **indirect method** starts with an initial sequence of approximations and proceeds by calculating a sequence of further approximations, which eventually gives the solution as accurately as desired. The most commonly used methods in this category are:

a) Jacobi's method

b) Gauss-Seidel method

Even when a direct method does exist, an iterative method may be preferable because it is more efficient or more stable.

8.3 CRAMER'S RULE AND ITS MODIFIED FORM

According to Cramer's rule, the system (8.2) can be solved using,

$$x_r = \frac{\det(A_r)}{\det(A)} \qquad \ldots (8.3)$$

where the determinant $\det(A_r)$ is exactly the same as det(A), except the rth column of det(A), has been replaced by the column of constants $b_1, b_2, \ldots, b_m$.

Let us illustrate this method using the following example.

Example 1 Solve the following system of equations:

$$7x_1 + 6x_2 + 3x_3 = 19$$
$$3x_1 + 2x_2 - x_3 = 7$$
$$x_1 + 4x_2 + 2x_3 = -2$$

Solution From the given system of equations, we obtain,

$$A = \begin{bmatrix} 7 & 6 & 3 \\ 3 & 2 & -1 \\ 1 & 4 & 2 \end{bmatrix}; \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}; \quad b = \begin{bmatrix} 19 \\ 7 \\ -2 \end{bmatrix}$$

$$\det(A) = \det\left[\begin{vmatrix} 7 & 6 & 3 \\ 3 & 2 & -1 \\ 1 & 4 & 2 \end{vmatrix}\right]$$

$$= 7\left[\begin{vmatrix} 2 & -1 \\ 4 & 2 \end{vmatrix}\right] - 6\left[\begin{vmatrix} 3 & -1 \\ 1 & 2 \end{vmatrix}\right] + 3\left[\begin{vmatrix} 3 & 2 \\ 1 & 4 \end{vmatrix}\right]$$

$$= 7(4+4) - 6(6+1) + 3(12-2)$$
$$= 7 \times 8 - 6 \times 7 + 3 \times 10$$
$$= 56 - 42 + 30 = 44$$

$$\det(A_1) = \det\left[\begin{vmatrix} 19 & 6 & 3 \\ 7 & 2 & -1 \\ -1 & 4 & 2 \end{vmatrix}\right] = 176$$

$$\det(A_2) = \det\left[\begin{vmatrix} 7 & 19 & 3 \\ 3 & 7 & -1 \\ 1 & -2 & 2 \end{vmatrix}\right] = -88$$

$$\det(A_3) = \det\left[\begin{vmatrix} 7 & 6 & 19 \\ 3 & 2 & 7 \\ 1 & 4 & -2 \end{vmatrix}\right] = 44$$

$$x_1 = \frac{\det(A_1)}{\det(A)} = \frac{176}{44} = 4$$

$$x_2 = \frac{\det(A_2)}{\det(A)} = \frac{-88}{44} = -2$$

$$x_3 = \frac{\det(A_3)}{\det(A)} = \frac{44}{44} = 1$$

The solution of the equations:

$$x_1 = 4,\ x_2 = -2,\ x_3 = 1$$

Alternative Method

Pre-multiplying both sides of (8.2) by the inverse of matrix A (i.e., A^{-1}), we get,

$$A^{-1}Ax = A^{-1}b$$

$$1 . x = A^{-1}b$$

$$x = A^{-1}b \qquad \ldots (8.4)$$

where $A^{-1} = \dfrac{\text{adj}(A)}{\det(A)}$ and adj(A) is the adjoint of the matrix A. Cramer's rule, of course, is identical to the formula (8.4).

The adjoint (or adjugate) of a square matrix A is the transpose of the matrix obtained by replacing each element of A by its cofactor. It is written as,

$$\text{adj}(A) = [A_{ij}]^T = [A_{ji}]$$

Example 2 Given the following system of equations:

$$x_1 + x_2 - x_3 = 10$$

$$x_1 - 2x_2 + 3x_3 = -4$$

$$x_1 + x_2 + 2x_3 = 10$$

a) Find the determinant, adjoint and inverse of A.

b) Solve also the system of equations

Solution

$$A = \begin{bmatrix} 2 & 1 & -1 \\ 1 & -2 & 3 \\ 1 & 1 & 2 \end{bmatrix}; \quad x = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}; \quad b = \begin{bmatrix} 10 \\ -4 \\ 10 \end{bmatrix}$$

a) $$\det(A) = \det\left[\begin{vmatrix} 2 & 1 & -1 \\ 1 & -2 & 3 \\ 1 & 1 & 2 \end{vmatrix}\right] = -16$$

$$\text{Minor } M_{11} \text{ of } a_{11} = \det\left[\begin{vmatrix} -2 & 3 \\ 1 & 2 \end{vmatrix}\right] = -7$$

$$\text{Cofactor } A_{11} \text{ of } a_{11} = (-1)^{1+1} M_{11} = -7$$

$$\text{Minor } M_{12} \text{ of } a_{12} = \det\left[\begin{vmatrix} 1 & 3 \\ 1 & 2 \end{vmatrix}\right] = -1$$

$$\text{Cofactor } A_{12} \text{ of } a_{12} = (-1)^{1+2} M_{12} = 1$$

Similarly, other cofactors are computed and written as below:

$$A_{13} = 3$$

$$A_{21} = -3; \quad A_{22} = 5; \quad A_{23} = -1$$

$$A_{31} = 1; \quad A_{32} = -7; \quad A_{33} = -5$$

$$\text{adj}(A) = \begin{bmatrix} -7 & 1 & 3 \\ -3 & 5 & -1 \\ 1 & -7 & -5 \end{bmatrix}^T = \begin{bmatrix} -7 & -3 & 1 \\ 1 & 5 & -7 \\ 3 & -1 & -5 \end{bmatrix}$$

$$A^{-1} = \frac{\text{adj}(A)}{\det(A)} = \frac{-1}{16}\begin{bmatrix} -7 & -3 & 1 \\ 1 & 5 & -7 \\ 3 & -1 & -5 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{7}{16} & \frac{3}{16} & \frac{-1}{16} \\ \frac{-1}{16} & \frac{-5}{16} & \frac{7}{16} \\ \frac{-3}{16} & \frac{1}{16} & \frac{5}{16} \end{bmatrix}$$

b) Solution of Equations

Using the formula, $x = A^{-1} b$, we get,

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} \frac{7}{16} & \frac{3}{16} & \frac{-1}{16} \\ \frac{-1}{16} & \frac{-5}{16} & \frac{7}{16} \\ \frac{-3}{16} & \frac{1}{16} & \frac{5}{16} \end{bmatrix} \times \begin{bmatrix} 10 \\ -4 \\ 10 \end{bmatrix} = \begin{bmatrix} 3 \\ 5 \\ 1 \end{bmatrix}$$

On simplification, we get,

$$x_1 = 3,\ x_2 = 5,\ x_3 = 1$$

Some remarks on the above methods:

If the number of equations in a problem is small (i.e., three or four), we may use Cramer's rule safely, but if the problem involves more equations and unknowns, we have to be careful. Suppose a problem has n equations and the same number of unknowns and, if we have to use determinants, then $[(n^2 - 1)n! + n]$ multiplications are required to solve the system of equations by Cramer's rule. For large n, $n^2 \times n!$ is a good estimate of the number of multiplications.

We regard determinants as a useful tool in developing theory, but in practical, solving numerical analysis, we should disregard them. Some other methods, which are better than Cramer's rule, should be used. They do not require computation of determinants and cofactors. These methods are discussed in the subsequent sections and can be used for any number of equations.

8.4 GAUSSIAN ELIMINATION METHODS

The Gaussian elimination method reduces a system of linear equations to a simpler form. The method works in two stages:

- **Forward stage**

  This stage is concerned with the manipulation of equations in order to eliminate some unknowns from the equations and produce an upper triangular system.

- **Backward (or Back substitution) stage**

  This stage is concerned with the actual solution of the equations and uses the back substitution process on the reduced upper triangular system.

We shall describe this method by considering the following system of four equations, for the sake of convenience and simplicity:

$$\left.\begin{aligned} a_{11}x_1 + a_{12}x_2 + a_{13}x_3 + a_{14}x_4 &= b_1 \\ a_{21}x_1 + a_{22}x_2 + a_{23}x_3 + a_{24}x_4 &= b_2 \\ a_{31}x_1 + a_{32}x_2 + a_{33}x_3 + a_{34}x_4 &= b_3 \\ a_{41}x_1 + a_{42}x_2 + a_{43}x_3 + a_{44}x_4 &= b_4 \end{aligned}\right\} \quad \ldots (8.5)$$

Let us describe these stages.

Forward Stage

Step 1 Operations on the equation

The first equation in (8.5) is called the **pivotal equation** and the coefficient of x_1 is called the **pivot**.

a) Divide the pivotal equation by its pivot a_{11}. This gives a new equation with 1 as the coefficient of x_1:

$$x_1 + a_{12}^{(1)}x_2 + a_{13}^{(1)}x_3 + a_{14}^{(1)}x_4 = b_1^{(1)}$$

For convenience, the coefficients calculated in the first step are denoted by a superscript (1), those calculated in the second step are denoted by a superscript (2), and so on.

b) Eliminate x_1 from the remaining equations:

i) x_1 from the second equation can be eliminated by adding to it $-a_{21}$ times the new first equation. This gives a new second equation in which x_1 is eliminated:

$$0 + a_{22}^{(1)}x_2 + a_{23}^{(1)}x_3 + a_{24}^{(1)}x_4 = b_2^{(1)}$$

ii) x_1 from the third equation can be eliminated by adding to it $-a_{31}$ times the new first equation. This gives a new third equation in which x_1 is eliminated:

$$0 + a_{32}^{(1)}x_2 + a_{33}^{(1)}x_3 + a_{34}^{(1)}x_4 = b_3^{(1)}$$

iii) Similarly, the new fourth equation in which x_1 is eliminated is obtained,

$$0 + a_{42}^{(1)}x_2 + a_{43}^{(1)}x_3 + a_{44}^{(1)}x_4 = b_4^{(1)}$$

iv) In general, the new ith equation is obtained by adding to it $-a_{i1}$ times the new first equation. So, x_1 is eliminated:

$$0 + a_{i2}^{(1)}x_2 + a_{i3}^{(1)}x_3 + a_{i4}^{(1)}x_4 = b_i^{(1)}.$$

After the completion of first step, the new set of equations with x_1 eliminated (except the pivot equation) is written below:

$$\left.\begin{aligned} a_{22}^{(1)} x_2 + a_{23}^{(1)} x_3 + a_{24}^{(1)} x_4 &= b_2^{(1)} \\ a_{32}^{(1)} x_2 + a_{33}^{(1)} x_3 + a_{34}^{(1)} x_4 &= b_3^{(1)} \\ a_{42}^{(1)} x_2 + a_{43}^{(1)} x_3 + a_{44}^{(1)} x_4 &= b_4^{(1)} \end{aligned}\right\} \quad \ldots (8.6)$$

Step 2

The first equation in the new set of equations in (8.6) is the new pivot equation with a_{22} as its pivot.

a) Divide the new pivot equation by its pivot. This gives a new first equation in the set with 1 as the coefficient of x_2:

$$x_2 + a_{23}^{(2)} x_3 + a_{24}^{(2)} x_4 = b_2^{(2)}$$

b) Eliminate x_2 from the remaining equations:

i) x_2 from the second equation of (8.6) can be eliminated by adding to it $-a_{32}$ times the current first equation: This gives a new second equation in which x_2 is eliminated:

$$0 + a_{33}^{(2)} x_3 + a_{34}^{(2)} x_4 = b_3^{(2)}$$

ii) Similarly, x_2 from the third equation is eliminated which is as below:

$$0 + a_{43}^{(2)} x_3 + a_{44}^{(2)} x_4 = b_4^{(2)}$$

After the completion of second step, the set of new equations with x_2 eliminated (except the pivotal equation) is written below:

$$\left.\begin{aligned} a_{33}^{(2)} x_3 + a_{34}^{(2)} x_4 &= b_3^{(2)} \\ a_{43}^{(2)} x_3 + a_{44}^{(2)} x_4 &= b_4^{(2)} \end{aligned}\right\} \quad \ldots (8.7)$$

Step 3

The first equation in (8.7) is the pivotal equation and $a_{33}^{(2)}$ is its pivot:

a) Divide the pivot equation by its pivot. This gives the new first equation in (8.8) with 1 as the coefficient of x_3:

$$x_3 + a_{34}^{(3)} x_4 = b_3^{(3)}$$

b) Eliminate x_3 from the remaining equation:

$$a_{44}^{(3)} x_4 = b_4^{(3)} \qquad \ldots (8.8)$$

Divide the final equation (8.8) by its pivot, we get,

$$x_4 = b_4^{(4)}$$

Rewriting all pivotal equations together, we have the following set:

$$\left.\begin{aligned} x_1 + a_{12}^{(1)} x_2 + a_{13}^{(1)} x_3 + a_{14}^{(1)} x_4 &= b_1^{(1)} \\ x_2 + a_{23}^{(2)} x_3 + a_{24}^{(2)} x_4 &= b_2^{(2)} \\ x_3 + a_{34}^{(3)} x_4 &= b_3^{(3)} \\ x_4 &= b_4^{(4)} \end{aligned}\right\} \qquad \ldots (8.9)$$

So, the equations under (8.9) have been reduced to an **upper triangular matrix**.

Backward (Back Substitution) Stage

The values of x_4, x_3, x_2 and x_1 can be obtained from (8.9) by back substitution.

Let us illustrate the method by means of an example.

Example 3 Solve the following system of equations using Gaussian elimination process:

$$4x_1 - 4x_2 - 3x_3 + 7x_4 = 1.3 \qquad \ldots \text{(i)}$$

$$8x_1 - 3x_2 - 8x_3 + 17x_4 = 6.6 \qquad \ldots \text{(ii)}$$

$$12x_1 - 12x_2 - 16x_3 + 29x_4 = -2.1 \qquad \ldots \text{(iii)}$$

$$-8x_1 + 33x_2 - 25x_3 + 36x_4 = 10.4 \qquad \ldots \text{(iv)}$$

Write also the computer program to implement the method.

Solution

Forward Stage

Step 1 Dividing equation (i) by 4, we get,

$$x_1 - x_2 - 0.75x_3 + 1.75x_4 = 0.325 \qquad \ldots \text{(i)}'$$

Multiplying equation (i)′ by 8 and subtracting from (ii), we get,

$$5x_2 - 2x_3 + 3x_4 = 4 \qquad \ldots \text{(ii)}'$$

Multiplying equation (i)′ by 12 and subtracting from (iii), we get

$$0-7x_3+8x_4 = -6 \qquad \ldots \text{(iii)}'$$

Multiplying equation (i)′ by 8 and subtracting from (iv), we get

$$25x_2-31x_3+50x_4 = 13 \qquad \ldots \text{(iv)}'$$

After the completion of Step-1, the new set of equations (except equation (i)′), with x_1 eliminated is as

$$5x_2 - 2x_3 + 3x_4 = 4 \qquad \ldots \text{(ii)}'$$

$$0 - 7x_3 + 8x_4 = -6 \qquad \ldots \text{(iii)}'$$

$$25x_2-31x_3+50x_4 = 13 \qquad \ldots \text{(iv)}'$$

Step 2 Dividing equation (ii)′ by 5, we get,

$$x_2 - 0.4x_3+0.6x_4 = 0.8 \qquad \ldots \text{(ii)}''$$

Multiplying equation (ii)″ by 25 and subtracting from (iv)′, we get,

$$-21x_3 + 35x_4 = -7 \qquad \ldots \text{(iv)}''$$

After tile completion of Step-2, the new set of equations (except equation (ii)″), with x_1 and x_2 eliminated as:

$$-7x_3+8x_4 = -6 \qquad \ldots \text{(iii)}'$$

$$-21x_3 + 35x_4 = -7 \qquad \ldots \text{(iv)}''$$

Step 3 Dividing equation (iii)′ by 7, we get,

$$-x_3+1.143x_4 = -0.857 \qquad \ldots \text{(iii)}'$$

Multiplying equation (iii)″ by 21 and subtracting from equation (iv)″, we get,

$$10.997x_4 = 10.997 \qquad \ldots \text{(iv)}'''$$

or, $$x_4 = \frac{10.997}{10.997} = 1 \qquad \ldots \text{(iv)}^{(iv)}$$

Rewriting the pivot equations, we get,

$$x_1 - x_2 - 0.75x_3 + 1.75x_4 = 0.325 \qquad \ldots \text{(i)}'$$

$$x_2 - 0.4x_3 + 0.6x_4 = 0.8 \qquad \ldots \text{(ii)}''$$

$$- \quad x_3 + 1.143x_4 = -0.857 \qquad \ldots \text{(iii)}'$$

$$x_4 = 1 \qquad \ldots \text{(iv)}^{(iv)}$$

Backward Substitution

Substituting the value of x_4 in (iii)″, we get,

$$x_3 = 0.857 + 1.143 = 2$$

Substituting the values of x_4 and x_3 in (ii)″, we, get,

$$x_2 - 0.4 \times 2 + 0.6 \times 1 = 0.8$$

$$x_2 = 0.8 + 0.8 - 0.6 = 1$$

Substituting the values of x_2, x_3 and x_4 in (i)′, we get,

$$x_1 - 1 - 0.75 \times 2 + 1.75 \times 1 = 0.325$$

$$x_1 = 0.325 + 1 + 1.50 - 1.75 = 1.075$$

The solution is as follows:

$$x_1 = 1.075,\ x_2 = 1,\ x_3 = 2 \text{ and } x_4 = 1$$

Solution in Tabular Form:

(a) Forward Step

Action	x_1	x_2	x_3	x_4	b	Number of Equations
	4	– 4	– 3	7	1.3	(i)
	8	– 3	– 8	17	6.6	(ii)
	12	–12	–16	29	–2.1	(iii)
	–8	33	–25	36	10.4	(iv)
(i) / 4	1	–1	–0.75	1.75	0.325	(i)′
(ii) – 8(i)′	0	5	– 2	3	4	(ii)′
(iii) – 12(i)′	0	0	– 7	8	– 6	(iii)′
(iv) – 8(i)′	0	25	– 31	50	13	(iv)′
		5	– 2	3	4	(ii)′
		0	– 7	-8	–6	(iii)′
		25	– 31	50	13	(iv)′
(ii)′ / 5		1	–0.4	0.6	0.8	(ii)″
		0	– 7	8	– 6	(iii)′
(iv)′ - 25(ii)″		0	–21	35	–7	(iv)″
			–7	8	– 6	(iii)′
			–21	35	– 7	(iv)″
(iii)′ / -7			1	–1.143	0.857	(iii)‴
(iv)‴ +21 (iii)‴			0	10.997	10.997	(iv)″
				10.997	10.997	(iv)‴
(iv)‴ / 10.997				1	1	$(iv)^{(iv)}$

b) Backward Substitution

$$x_1 - x_2 - 0.75x_3 + 1.75x_4 = 0.325 \quad \ldots (i)'$$

$$x_2 - 0.4x_3 + 0.6x_4 = 0.8 \quad \ldots (ii)''$$

$$x_3 - 1.143x_4 = 0.857 \quad \ldots (iii)'$$

$$x_4 = 1 \quad \ldots (iv)^{(iv)}$$

Solving by backward substitution, we get,

$x_1 = 1.075,$

$x_2 = 1.0000$

$x_3 = 2.000$

$x_4 = 1.000$

Program No. 21: Gaussian Elimination Method

```
# include<iostream.h>
# include<conio.h>

float temp,arr[10][6],answer[ 6];
int noofeq;

void readata( )
  {
    short i,j,k=8,I=20;
    gotoxy(24, 1);
    cout<<"GAUSSIAN ELIMINATION METHOD";
    gotoxy(24,2);
    cout<<"*********************************"
    cout<<"\n\tHOW MANY EQUATIONS: ";
    cin>>noofeq;
    cout<<"\n\tENTER DATA FOR EQUATIONS";
    coutt<<"\n\t==============================";
    for(i=0;i<=noofeq-1;i++)
    for(j=0;j<=noofeqj++)
      {
        gotoxy(l,k);
        cin>>arr[i][j];
        I+=8;
        if(j=noofeq)
```

```
        {
          k++;
          I=20;
        }
      }
  }
void print_result( )
  {
    int i,J=1;
    cout<<"\n\tFINAL RESULT                    : ";
    coutt<<"\n\t==============================\n";
    for(i=noofeq-1;i>=0;i--)
      {
        cout<<"\tX"<<j++<<":\t"<<answer[i]<<end 1;
      }
  }
void main( )
  {
    int i,j,k,I=1,p=1;
    clrscr( );
    readata( );
    for(i=1;i<=noofeq;i++)
      {
        for(j=i-1j<=noofeq\j++)
          arr[noofeq+i-1] [j]=arr[i-1][j] / arr[i-1] [i-1];
        for(j=i\i<noofeq\i++)
          {
            temp=arr[j][i-1];
            for(k=i-1;k<=noofeq;k++)
              arr[j][k]- =arr[noofeq+i-1][k]*temp;
          }
      }
    temp=0; k=1;
    answer[0]=arr[2*noofeq-1][noofeq];
    for(j=2*noofeq-1;j>noofeq\j--)
      {
        for(k=1;k<=p;k++)
          temp+=answer[k-1]*arr[j-1][noofeq-k];
        answer[p++]=arr[j-1](noofeq]-temp;
        temp=0;
      }
    print_result( );
  }
```

Computer Output

```
                    GAUSSIAN ELIMINATION METHOD
                    ***********************************

HOW MANY EQUATIONS: 4

ENTER DATA FOR EQUATIONS
==========================
    4        -4        -3        7        1.3
    8        -3        -8        17       6.6
    12       -12       -16       29       -2.1
    -8       33        -25       36       10.4

FINAL RESULT
=============
X1:   1.075
X2:   1
X3:   2
X4:   1
```

8.4.1 Pivot Strategy

There are two difficulties in using the simple Gaussian elimination method. They are as follows:

i) One of the pivot elements may be zero.

ii) If any pivot element is very small (very close to zero) division by this element tends to magnify the round-off error.

These difficulties can easily be avoided. It is not necessary to use the first available equation as the pivotal equation. It is quite safe to interchange the row having zero pivot with any other row which does not have a zero element in that position. This raises the question of whether there is any preference as to which row is exchanged with the one having zero pivot. Greater accuracy can be achieved if the pivot has the greatest magnitude. In other words, the row with a zero pivot should be exchanged with any row, which has the largest (in absolute value) element in the same column.

The above procedure will not only eliminate zero pivots, but will also increase overall computing accuracy.

In practice, we select the pivots by either of the following two ways:

- Partial pivoting
- Complete pivoting

8.4.2 Partial Pivoting Scheme

In this scheme, we eliminate the unknowns in order starting with x_1 but at each stage, we select the pivotal equation as the one with the largest pivot (in absolute value) of the unknowns being eliminated. The reason for this is that, when calculations are performed using finite-digit arithmetic, as would be the case for calculator or, even computer-generated solutions, a pivot that is small compared to the entries below it in the same column can lead to substantial round-off error. Generally, partial pivoting scheme improves the accuracy of the solution.

Let us solve the previous example using partial pivoting scheme.

Example 4 Solve the following system of equations using partial pivoting technique:

$$\begin{aligned} 4x_1 - 4x_2 - 3x_3 + 7x_4 &= 1.3 \\ 8x_1 - 3x_2 - 8x_3 + 17x_4 &= 6.6 \\ 12x_1 - 12x_2 - 16x_3 + 29x_4 &= -2.1 \\ -8x_1 + 33x_2 - 25x_3 + 36x_4 &= 10.4 \end{aligned}$$

Solution **Forward Stage**

Let us rewrite equations in the following form:

Action	x_1	x_2	x_3	x_4	b	No.
	4	−4	−3	7	1.3	(i)
	8	−3	−8	17	6.6	(ii)
	12	−12	−16	29	−2.1	(iii)
	−8	33	−25	36	10.4	(iv)
i) − 4(iii)′		0	2.332	−2.668	2.0	(i)′
ii) − 8(iii)′		5	2.664	−2.336	8.0	(ii)′
iii) /12	1	−1	−1.333	2.417	−0.175	(iii)′
iv) + 8(iii)′		25	−35.664	55.336	9.0	(iv)′
		0	2.332	−2.668	2.0	(i)′
		5	2.664	−2.336	8.0	(ii)′
		25	−35.664	55.336	9.0	(iv)′
			2.332	−2.668	2.0	(i)′
ii)′ − 5(iv)″			9.799	−13.401	6.2	(ii)″
iv)′ / 125		1	−1.427	2.213	0.36	(iv)″
			2.332	−2.668	2.0	(i)′
			9.799	−13.401	6.2	(ii)″
(i)′ −2.332(ii)‴				0.523	0.524	(i)″
(ii)′ /9.799			1	−1.368	0.633	(ii)‴
(i)″ / 0.523				1	1	(i)‴

Rewriting pivot equation (iii)′, (iv)″, (ii)‴ and (i)‴, we get,

$$x_1 - x_2 - 1.33x_3 + 2.417x_4 = -0.175 \quad \ldots \text{(iii)}'$$

$$x_2 - 1.427x_3 + 2.213x_4 = 0.36 \quad \ldots \text{(iv)}''$$

$$x_3 - 1.368x_4 = 0.633 \quad \ldots \text{(ii)}'''$$

$$x_4 = 1 \quad \ldots \text{(i)}'''$$

Back Substitution

Substituting the value of x_4 from (i)‴ in (ii)‴, we get,

$$x_3 - 1.368 \times 1 = 0.633$$

$$x_3 = 0.633 + 1.368 = 2.0$$

Substituting the values of x_3 and x_4 in (iv)″, we get,

$$x_2 - 1.427 \times 2 + 2.213 \times 1 = 0.36$$

$$x_2 = 0.36 + 2.854 - 2.213 = 1.0$$

Substituting the values of x_2, x_3 and x_4 in (iii)′, we get,

$$x_1 - 1.0 - 1.333 \times 2.0 + 2.417 \times 1 = -0.175$$

$$x_1 = -0.175 + 1.0 + 2.666 - 2.417 = 1.074$$

The solution is,

$$x_1 = 1.074,\ x_2 = 1.0,\ x_3 = 2.0 \text{ and } x_4 = 1$$

8.4.3 Complete Pivoting Scheme

In complete pivoting scheme, we select the equation with the largest coefficient (in absolute value) as the pivot equation; it can be anywhere in the body of the table.

Let us solve the previous example using the complete pivoting strategy.

Example 5 Solve the following system of equations using complete pivoting strategy:

$$\begin{aligned}
4x_1 - 4x_2 - 3x_3 + 7x_4 &= 1.3 \\
8x_1 - 3x_2 - 8x_3 + 17x_4 &= 6.6 \\
12x_1 - 12x_2 - 16x_3 + 29x_4 &= -2.1 \\
-8x_1 + 33x_2 - 25x_3 + 36x_4 &= 10.4
\end{aligned}$$

Solution **Forward Stage**

Action	x_1	x_2	x_3	x_4	b	No.
	4	– 4	– 3	7	1.3	(i)
	8	– 3	– 8	17	6.6	(ii)
	12	–12	–16	29	–2.1	(iii)
	– 8	33	–25	36	10.4	(iv)
i) – 7(iv)′	5.554	–10.419	1.858		–0.723	(i)′
ii) – 17(iv)′	11.774	–18.589	3.798		1.687	(ii)′
iii) – 29(iv)′	18.438	–38.593	4.126		–10.481	(iii)′
iv) / 36	–0.222	0.917	–0.694	I	0.289	(iv)′
	5.554	–10.419	1.858		– 0.723	(i)′
	11.774	–18.589	3.798		1.687	(ii)′
	18.438	–38.593	4.126		–10.481	(iii)′
i)′ + 10.419(iii)″	0.574		0.743		2.111	(i)″
ii)′ + 18.589(iii)″	2.888		1.809		6.734	(ii)″
iii)′ / –38.593	–0.478	1	–0.107		0.272	(iii)″
	0.574		0.743		2.111	(i)″
	2.888		1.809		6.734	(ii)″
i)″ –0.574(ii)‴			0.384		0.771	(i)‴
ii)″ / 12.888	1		0.626		2.335	(ii)‴
i)‴ / 0.384			1		2.0	$(i)^{(iv)}$

Back Substitution

Rewriting pivot equations (iv)′, (iii)″, (ii)‴ and $(i)^{(iv)}$, we get,

$$-0.222x_1 + 0.917x_2 - 0.694x_3 + x_4 = 0.289 \quad \text{(iv)}'$$
$$-0.487x_1 + x_2 - 0.107x_3 = 0.272 \quad \text{(iii)}''$$
$$x_1 + 0.626x_3 = 2.335 \quad \text{(ii)}'''$$
$$x_3 = 2.0 \quad (i)^{(iv)}$$

Substituting the value of x_3 in (ii)‴, we get,

$$x_1 + 0.626 \times 2 = 2.335$$
$$x_1 = 2.335 - 1.252 = 1.083$$

Substituting the values of x_1 and x_3 in (iii)″, we get,

$$-0.487 \times 1.083 + x_2 - 0.107 \times 2 = 0.272$$
$$x_2 = 0.272 + 0.214 + 0.518 = 1$$

Substituting the values of x_1, x_2 and x_3 in (iv)′, we get,

$$-0.222 \times 1.083 + 0.917 \times 1 - 0.694 \times 2 + x_4 = 0.289$$

$$x_4 = 0.289 - 0.917 + 0.240 + 1.388 = 1$$

The solution is,

$$x_1 = 1.083,\ x_2 = 1.0,\ x_3 = 2.0 \text{ and } x_4 = 1$$

8.5 TRIANGULAR DECOMPOSITION (FACTORIZATION) METHOD

This method is based on the fact, that a non-singular square matrix A can be replaced by the product of lower and upper triangular matrices. That is why this process is also known as **LU-decomposition method**.

Consider the following linear system of equations,

$$a_{11} x_1 + a_{12} x_2 + a_{13} x_3 = b_1$$

$$a_{21} x_1 + a_{22} x_2 + a_{23} x_3 = b_2$$

$$a_{31} x_1 + a_{32} x_2 + a_{33} x_3 = b_3$$

which can be written in the form,

$$Ax = b \qquad \ldots (8.10)$$

Then, A takes the form,

$$A = LU \qquad \ldots (8.11)$$

$$\text{where } L = \begin{vmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{vmatrix}$$

$$\text{and } U = \begin{vmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{vmatrix}$$

L is a lower-triangular matrix (that has 1's along the diagonal) and U is an upper-triangular matrix (with non-zero diagonal elements). Hence, (8.10) becomes:

$$LUx = b \qquad \ldots (8.12)$$

This method can be used both for solving a system of equations and computing the inverse of the given matrix.

8.5.1 Solution of Systems of Equations

If we set $Ux = y$, $\qquad \ldots (8.13)$

then (8.12) may be rewritten as:

$$Ly = b \qquad \ldots (8.14)$$

which is equivalent to the system,

$$\begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

On simplification, we get,

$$\begin{aligned} y_1 &= b_1 \\ l_{21} y_2 + y_2 &= b_2 \\ l_{31} y_1 + l_{32} y_2 + y_3 &= b_3 \end{aligned}$$

Solve the above for y_1, y_2 and y_3 using forward substitution.

Now, we compute the values of x's as:

$$Ux = y$$

$$\begin{bmatrix} u_{11} & u_{12} & u_{13} \\ & u_{22} & u_{23} \\ & & u_{33} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix}$$

On multiplying, we get,

$$\begin{aligned} u_{11} x_1 + u_{12} x_2 + u_{13} x_3 &= y_1 \\ u_{22} x_2 + u_{23} x_3 &= y_2 \\ u_{33} x_3 &= y_2 \end{aligned}$$

which can be solved for x_1, x_2 and x_3 by backward substitution.

8.5.2 Inverse of a Matrix A using L and U

In order to compute the inverse of A using L and U, .the following steps are used:

i) $A = LU$

ii) $A^{-1} = (LU)^{-1} = U^{-1}.L^{-1}$

iii) Multiplying both sides of (ii) by U

Scheme for Computing L and U

We shall now describe the scheme for computing the matrices L and U and illustrate the procedure with a matrix of order 3. From (8.11), we get,

$$\begin{vmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{vmatrix} \begin{vmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{vmatrix} = \begin{vmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{vmatrix}$$

Multiplying the matrices on the left-hand side and equating the corresponding elements of both sides, we get,

$$u_{11} = a_{11} \qquad u_{12} = a_{12} \qquad u_{13} = a_{13}$$

$$l_{11} \cdot u_{11} = a_{21}$$

or,

$$l_{21} = \frac{a_{21}}{u_{11}} = \frac{a_{21}}{a_{11}}$$

$$l_{21} u_{11} + u_{22} = a_{22}$$

or,

$$a_{22} = a_{22} - \frac{a_{21} \cdot a_{12}}{a_{11}} .$$

$$l_{31} \cdot u_{11} = a_{31}$$

or,

$$l_{31} = \frac{a_{31}}{u_{11}} = \frac{a_{31}}{a_{11}}$$

$$l_{31} u_{11} = a_{31}$$

or,

$$l_{31} = \frac{a_{31}}{u_{11}} = \frac{a_{31}}{a_{11}}$$

$$l_{21} u_{12} + u_{22} = a_{22}$$

or,

$$u_{22} = a_{22} - \frac{a_{21} \cdot a_{12}}{a_{11}}$$

$$l_{21} u_{13} + u_{23} = a_{23}$$

or,

$$l_{23} = a_{23} - \frac{a_{21} \cdot a_{13}}{a_{11}}$$

$$l_{31} \cdot u_{12} + l_{32} \cdot u_{22} = a_{32}$$

$$l_{32} = \frac{1}{u_{22}} \left(a_{32} - \frac{a_{31} \cdot a_{12}}{a_{11}} \right)$$

$$l_{31} \cdot u_{13} + l_{32} \cdot u_{33} = a_{33}$$

$$u_{33} = a_{33} - l_{31} \cdot u_{31} - l_{32} \cdot u_{23}$$

The above is a systematic procedure to compute the elements of L and U. We can easily generalize the scheme.

Let us illustrate the procedure by means of the following example.

Example 6 (a) Solve the following system of equations,

$$\begin{aligned} 4x_1 + 5x_2 - 2x_3 &= 5.6 \\ 2x_1 + 3x_2 + x_3 &= 1.3 \\ 2x_1 + 4x_2 + 4x_3 &= 9.5 \end{aligned}$$

using the decomposition method.

(b) Invert $A = \begin{bmatrix} 4 & 5 & -2 \\ 2 & 3 & 1 \\ 2 & 4 & 4 \end{bmatrix}$

using the decomposition method.

c) Using A^{-1}, solve the system of equations given in (a) above. Compare the two results.

Solution (a) Solution of Equations using L and U.

We know that $LU = A$.

$$\text{So,} \quad \begin{vmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{vmatrix} \begin{vmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{vmatrix} = \begin{bmatrix} 4 & 5 & -2 \\ 2 & 3 & 1 \\ 2 & 4 & 4 \end{bmatrix}$$

Multiplying, we get:

$$\begin{vmatrix} u_{11} & u_{12} & u_{13} \\ l_{21}u_{11} & l_{21}u_{12}+u_{22} & l_{21}u_{13}+u_{23} \\ l_{31}u_{11} & l_{31}u_{12}+l_{32}u_{22} & l_{31}u_{13}+l_{32}u_{23}+u_{33} \end{vmatrix} = \begin{bmatrix} 4 & 5 & -2 \\ 2 & 3 & 1 \\ 2 & 4 & 4 \end{bmatrix}$$

Computing coefficients, we get,

$u_{11} = 4$	$u_{12} = 5$	$u_{13} = -2$
$l_{21}u_{11} = 2$	$l_{21}u_{12}+u_{22} = 3$	$l_{21}u_{13}+u_{23} = 1$
or, $l_{21} = \frac{2}{u_{11}} = \frac{2}{4} = \frac{1}{2}$	$u_{22} = \frac{1}{2}$	$u_{23} = 2$
$l_{31}u_{11} = 2$	$l_{31}u_{12}+l_{32}u_{22} = 4$	$l_{31}u_{13}+l_{32}u_{23}+u_{33} = 4$
or, $l_{31} = \frac{2}{u_{11}} = \frac{2}{4} = \frac{1}{2}$	$l_{32} = 3$	$u_{33} = -1$

It follows that $A = LU$

$$= \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 3 & 1 \end{bmatrix} \times \begin{bmatrix} 4 & 5 & -2 \\ 0 & \frac{1}{2} & 2 \\ 0 & 0 & -1 \end{bmatrix}$$

Since Ly = b, we have

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 3 & 1 \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \begin{bmatrix} 5.6 \\ 1.3 \\ 9.5 \end{bmatrix}$$

Multiplying the matrices on the left-hand side, we get,

$$y_1 = 5.6$$

$$\frac{1}{2} y_1 + y_2 = 1.3$$

$$\frac{1}{2} y_1 + 3y_2 + y_3 = 9.5$$

Solving the above system, we get,

$$y_1 = 5.6,\ y_2 = -1.6 \text{ and } y_3 = 11.2$$

Using Ux = y, we get,

$$\begin{bmatrix} 4 & 0 & -2 \\ 0 & \frac{1}{2} & 2 \\ 0 & 0 & -2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5.6 \\ -1.5 \\ 11.2 \end{bmatrix}$$

Multiplying the matrices on the left-hand side, we get,

$$\begin{aligned} 4x_1 + 5x_2 - 2x_3 &= 5.6 \\ \frac{1}{2}x_2 + 2x_3 &= -1.5 \\ -x_3 &= 11.2 \end{aligned}$$

Solving the above system of equations by backward substitution, we get,

$$x_3 = -11.2,\ x_2 = 41.8 \text{ and } x_1 = -56.5$$

(b) The matrices L and U have already been computed in (a) above.

Now, $LL^{-1} = I$

$$\text{Let } L^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ l'_{21} & 1 & 0 \\ l'_{31} & l'_{32} & 1 \end{bmatrix}$$

Therefore, $\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2} & 1 & 0 \\ \frac{1}{2} & 3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ l'_{21} & 1 & 0 \\ l'_{31} & l'_{32} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Multiplying both matrices and comparing like terms, we get:

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{2}+l'_{21} & 1 & 0 \\ \frac{1}{2}+3l'_{21}+l'_{31} & 3+l'_{32} & 1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\tfrac{1}{2}+l'_{21}=0; \quad l'_{21}=-\tfrac{1}{2}$$

$$\tfrac{1}{2}+3l'_{21}+l'_{31}=0; \quad l'_{21}=1$$

$$3+l'_{32}=0; \quad l'_{32}=-3$$

Substituting values in L^{-1}, we get,

$$L^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ 1 & -3 & 1 \end{bmatrix}$$

Also, $UA^{-1} = L^{-1}$

$$\begin{bmatrix} 4 & 5 & 0 \\ 0 & \frac{1}{2} & 0 \\ 0 & 0 & -1 \end{bmatrix} \begin{bmatrix} a'_{11} & a'_{12} & a'_{13} \\ a'_{21} & a'_{22} & a'_{23} \\ a'_{31} & a'_{32} & a'_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ -\frac{1}{2} & 1 & 0 \\ 1 & -3 & 1 \end{bmatrix}$$

Multiplying and comparing coefficients, we get

$$a'_{31}=-1; \quad a'_{32}=3; \quad a'_{33}=-1$$

$$a'_{21}=3; \quad a'_{22}=-10; \quad a'_{23}=4$$

$$a'_{11}=-4; \quad a'_{12}=14; \quad a'_{13}=-\frac{11}{2}$$

Substituting values in A^{-1}, we get,

$$A^{-1} = \begin{bmatrix} -4 & 14 & -\frac{11}{2} \\ 3 & -10 & 4 \\ -1 & -3 & -1 \end{bmatrix}$$

The system of equations can be solved using: $x = A^{-1} b$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -4 & 14 & -\frac{11}{2} \\ 3 & -10 & 4 \\ -1 & -3 & -1 \end{bmatrix} \begin{bmatrix} 5.6 \\ 1.3 \\ 11.2 \end{bmatrix} = \begin{bmatrix} -56.5 \\ 41.8 \\ -11.2 \end{bmatrix}$$

The solution is as follows:

$$x_1 = -56.5, \quad x_2 = 41.8 \text{ and } x_3 = -11.2$$

Both results are the same.

8.6 TRIANGULAR DECOMPOSITION FOR SYMMETRIC MATRICES

When the given matrix A is real symmetric and positive definite, then we may decompose A as:

$$\begin{aligned} A &= LL^T \\ A^{-1} &= \left(LL^T\right)^{-1} = \left(L^T\right)^{-1}\left(L^{-1}\right) \\ &= \left(L^{-1}\right)^T\left(L^{-1}\right) \qquad \ldots (8.15) \end{aligned}$$

In this case, we have to perform only one inversion of a triangular matrix and one multiplication. This method is due to **Choleski** and is also known as the **square-root method**. It is possible that this method may give a zero or imaginary diagonal element l_{ii} even though the matrix A is real. This is the major disadvantage of Choleski's method.

Positive Definite Matrix

A real matrix A is said *to* be positive definite if $x^T Ax > 0$, for all non-zero vector x.

The matrix $A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix}$ is positive definite because

$$x^T Ax = [x_1\ x_2\ x_3] \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & -1 \\ 0 & -1 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}$$

$$= x_1^2 + 2x_2^2 + x_3^2 + 2x_1 x_2 - 2x_2 x_3$$

$$= (x_1 + x_2)^2 + (x_2 - x_3)^2$$

The right-hand side of the above relation will be positive for all values of x_1, x_2 and x_3. Hence, the matrix A is positive definite.

If the matrix A is not positive definite, there are two possibilities:

i) at some stage, $l_{ii} = 0$, if this happens the method fails, for example.

$$A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}; \; l_{ii} = 0$$

ii) at some stage, $l_{ii}^2 < 0$, and this results in imaginary numbers.

Let us illustrate Choleski's method by means of an example.

Example 7 a) Given the matrix, $A = \begin{bmatrix} 1 & 2 & 6 \\ 2 & 5 & 15 \\ 6 & 15 & 46 \end{bmatrix}$

Find A^{-1} using Choleski's method

b) Using A^{-1} computed in (a) above, solve the following system of equations:

$$\begin{aligned} x_1 + 2x_2 + 6x_3 &= 13 \\ 2x_1 + 5x_2 + 15x_3 &= 15 \\ 6x_1 + 15x_2 + 46x_3 &= 19 \end{aligned}$$

Solution a) Let $A = \begin{vmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{vmatrix}$

$$A = LL^T = \begin{vmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{vmatrix} \begin{vmatrix} l_{11} & l_{12} & l_{13} \\ 0 & l_{22} & l_{23} \\ 0 & 0 & l_{33} \end{vmatrix}$$

$$\begin{vmatrix} l_{11}^2 & l_{11}\, l_{21} & l_{11}\, l_{31} \\ l_{21}\, l_{11} & l_{21}^2 + l_{22}^2 & l_{21}\, l_{31} + l_{22}\, l_{32} \\ l_{31}\, l_{11} & l_{31}\, l_{21} + l_{22} & l_{31}^2\, l_{32}^2 + l_{33}^2 \end{vmatrix} = \begin{bmatrix} 1 & 2 & 6 \\ 2 & 5 & 15 \\ 6 & 15 & 46 \end{bmatrix}$$

Computing elements, we get,

$l_{11}^2 = 1;$ $\qquad l_{11} = 1$

$l_{11}\, l_{21} = 2;$ $\qquad l_{21} = 2$

$l_{11}\, l_{31} = 6;$ $\qquad l_{31} = 6$

$$l_{11}^2 + l_{22}^2 = 5; \qquad l_{22} = 1$$

$$l_{21}\, l_{31} + l_{22}\, l_{32} = 15 \qquad l_{32} = 3$$

$$l_{31}^2 + l_{32}^2 + l_{33}^2 = 46; \qquad l_{33} = 1$$

$$\text{So, } L = \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 6 & 3 & 1 \end{bmatrix}$$

$$\text{Let } L^{-1} = \begin{bmatrix} l'_{11} & 0 & 0 \\ l'_{21} & l'_{22} & 0 \\ l'_{31} & l'_{32} & l'_{33} \end{bmatrix}$$

$$LL^{-1} = I$$

$$\text{So, } \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 6 & 3 & 1 \end{bmatrix} \begin{bmatrix} l'_{11} & 0 & 0 \\ l'_{21} & l'_{22} & 0 \\ l'_{31} & l'_{32} & l'_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Multiplying the above matrices and evaluating the elements of the matrix L^{-1}, we get,

$$L^{-1} = \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{bmatrix}$$

$$A^{-1} = \left(L^{-1}\right)^T \left(L^{-1}\right)$$

$$= \begin{bmatrix} 1 & -2 & 0 \\ 0 & 1 & -3 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -3 & 0 & 1 \end{bmatrix}$$

$$= \begin{bmatrix} 5 & -2 & 0 \\ -2 & 10 & -3 \\ 0 & -3 & 1 \end{bmatrix}$$

Solution of equations: $x = A^{-1} . b$

$$\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 & -2 & 0 \\ -2 & 10 & -3 \\ 0 & -3 & 1 \end{bmatrix} \begin{bmatrix} 13 \\ 15 \\ 19 \end{bmatrix} = \begin{bmatrix} 35 \\ 67 \\ -26 \end{bmatrix}$$

The solution is as follows:

$$x_1 = 35, \ x_2 = 67 \text{ and } x_3 = -26$$

8.7 SOLUTION OF TRIDIAGONAL SYSTEMS OF EQUATIONS

A system of equations is said to be **tridiagonal**, if and only if all elements of matrix A are zero except a_{ii}, a_{jj+1} and a_{j+1j}, where i = 1,2, ..., n and j = 1,2, ..., n – 1.

The following system,

$$\begin{aligned} 4x_1 + x_2 &= 1 \\ x_1 + 3x_2 + 7x_3 &= 0 \\ x_2 + 3x_3 - x_4 &= -1 \\ x_3 + x_4 - x_5 &= 0 \\ x_4 - 2x_5 &= 1 \end{aligned}$$

is tridiagonal.

Tridiagonal matrices occur frequently in a variety of applications:

i) In the solution of certain partial differential equations.

ii) In approximation theory, i.e., where cubic spline functions are used to fit data.

Diagonal Dominance

An n × n matrix A is said to be diagonally dominant if and only if,

$$|a_{ii}| \geq \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} \text{ ; for all } i = 1.2, \ldots, n.$$

The matrix A is said to be **strictly diagonally dominant if,**

$$|a_{ii}| > \sum_{\substack{j=1 \\ j \neq i}}^{n} a_{ij} \text{ ; for all } i = 1.2, \ldots, n.$$

The system,

$$\begin{aligned} 4x_1 + 2x_2 + 2x_3 &= 1 \\ x_1 - 3x_2 - x_3 &= 0 \\ x_1 + x_2 + 2x_3 &= 0 \end{aligned}$$

is diagonally dominant, whereas the following system,

$$
\begin{aligned}
x_1 - \frac{1}{4}x_2 - \frac{1}{4}x_3 &= 7 \\
-\frac{1}{4}x_1 - x_2 + x_3 - \frac{1}{4}x_4 &= 10 \\
-\frac{1}{4}x_1 + x_3 - \frac{1}{4}x_4 &= 15 \\
-\frac{1}{4}x_2 - \frac{1}{4}x_3 + x_4 &= 8
\end{aligned}
$$

is strictly diagonally dominant.

Computational Procedure

A computational procedure to solve tridiagonal system is explained below:

When the system of equations is tridiagonal arid diagonally dominant, its solution exists and is unique. This procedure has been found to be very efficient for use on a digital computer because it uses less memory.

The following steps are used to implement the procedure:

i) Generate the quantities: $w_1, w_2, \ldots, w_n$

and $d_1, d_2, \ldots, d_n$

ii) From $w_1 = a_{11}$

$$d_j = a_{j,j+1}\frac{1}{w_j}; \qquad j = 1, 2, \ldots, n-1$$

$$w_j = a_{j,j} - a_{j,j-1}\, d_{j-1}; \qquad j = 1, 2, \ldots, n.$$

iii) Next generate the quantities: $z_1, z_2, z_3, \ldots, z_n$

From $z_1 = \dfrac{b_1}{w_1}$

$$z_j = \frac{\left(b_j - a_{j,j-1}\, z_{j-1}\right)}{w_j}; \qquad j = 2, 3, \ldots, n.$$

iv) Finally, generate the solution:

$x_1, x_2, \ldots, x_n$ from

$$x_n = z_n$$

$$x_k = z_k - x_{k+1}\, d_k; \qquad k = n-1, n-2, \ldots, 1.$$

The working of this method is illustrated below:

Example 8 Solve the following tridiagonal system of equations:

$$\begin{aligned}
-2x_1 + x_2 &= 1 \\
x_1 - 2x_2 + x_3 &= 0 \\
x_2 - 2x_3 + x_4 &= 0 \\
x_3 - 2x_4 + x_5 &= 0 \\
x_4 - 2x_5 &= 0
\end{aligned}$$

Solution

$$a_{11} = a_{22} = a_{33} = a_{44} = a_{55} = -2$$

$$a_{12} = a_{21} = \ldots = 1$$

$$b_1 = 1;\ b_2 = b_3 = \ldots = 0$$

$$w_1 = a_{11} = -2$$

$$d_1 = \frac{a_{12}}{w_1} = -\frac{1}{2}$$

$$w_2 = a_{22} - a_{21} \cdot d_1$$

$$= -2 - 1x - \frac{1}{2} = -\frac{3}{2}$$

$$d_2 = \frac{a_{23}}{w_2} = -\frac{3}{2}$$

Similarly, $w_3 = -\frac{4}{3};\ d_3 = -\frac{3}{4}$

$$w_4 = -\frac{5}{4};\ d_4 = -\frac{4}{5}$$

$$w_5 = -\frac{6}{5};$$

$$z_1 = \frac{b_1}{w_1} = -\frac{1}{2}$$

$$z_2 = \frac{(b_2 - a_{21} z_1)}{w_2}$$

$$= \frac{\left(0 - 1x - \frac{1}{2}\right)}{-\frac{3}{2}} = -\frac{1}{3}$$

$$z_3 = -\frac{1}{4};\ z_4 = -\frac{1}{5};\ z_5 = -\frac{1}{6}$$

Finally, $x_5 = z_5 = -\frac{1}{6}$

$$x_4 = z_4 - x_5\ d_4$$

$$= -\frac{1}{5} - \left(-\frac{1}{6}\right)\left(-\frac{4}{5}\right) = -\frac{1}{3}$$

$$x_3 = z_3 - x_4\ d_3$$

$$= -\frac{1}{4} - \left(-\frac{1}{3}\right)\left(-\frac{3}{4}\right) = -\frac{1}{2}$$

$$x_2 = z_2 - x_3\ d_2 = -\frac{2}{3}$$

$$x_1 = -\frac{5}{6}$$

The solution is as follows:

$x_1 = -0.8333$

$x_2 = -0.6667$

$x_3 = -0.5000$

$x_4 = -0.3333$

$x_5 = -0.1667$

Program No. 22: Tridiagonal System of Equation

```
# include<conio.h>
# include<stdio.h>
# include<iostream.h>

float a[10][10], w[10],d[10],zeta[10],x[10],b[10];

void main( )
  {
    int n,i,j,k,l,m, ver=10,hor=9;
    clrscr( );
    cout<<"\n\n\tTRIDIAGONAL SYSTEM OF EQUATIONS";
    cout<<"\n\t=================================";
    cout<<"\n\n\tSIZE OF MATRIX: ";
```

```
cin>>n;
            //coefficients of A
cout<<"\n\tENTER THE VALVES: ";
for(i =1;i<n;i++)
  {
    j=i+1;
    gotoxy(hor,ver);  cin>>a[i][i];  hor+=6;
    gotoxy(hor, ver); cin>>a[i][j]; ver++; hor=9+6*(i-1);
    gotoxy(hor,ver);  cin>>a[j][i]; hor+=6;
  }
gotoxy(hor,ver); cin>a[i][i];
                //values of B
hor+=4; ver=10;
for(j=1;j<=n;j++)
  {
    gotoxy(hor,ver);
    cout<<"= ";
    cin>>b[j];
    ver++;
  }
                //calculation of w, d & zeta
w[1]=a[1][1];
zeta[1]=b[t]/w[ 1];
for(j=1; j<=n-1;j++)
  {
    k=j+1;
    d[j]=a[j][k]/w[j];
    w[k]=a[k][k]-a[k][j]*d[j];
    zeta[k]=(b[k]-a[k][j]*zeta[j])/w[k];
  }
                //calculation of x
x[n]=zeta[n];
for(l=1; l<=n-1; 1++)
  {
    m=n-1;
    x[m]=zeta[m]-x[m+1]*d[m];
  }
                //printing values of x on screen
for(m=1;m<=n;m++)
  {
    cout<<"\n\tX "<<m<<" = "<<x[m];
  }
getch( );
}
```

Computer Output

```
TRIDIAGONAL SYSTEM OF EQUATIONS
===============================
NO. OF EQUATIONS: 5
ENTER THE VALUES:

 -2   1                    = 1
  1  -2    1               = 0
       1   -2    1          = 0
            1   -2    1     = 0
                 1   -2     = 0

X1 = -0.833333
X2 = -0.666667
X3 = -0.5
X4 = -0.333333
X5 = -0.166667
```

8.8 ITERATIVE METHODS

The direct methods we have discussed so far are not suitable when the number of equations in a system is too large, or when the coefficient matrix is sparse (i.e., when most of the elements in a matrix are zero). Iterative methods are particularly suitable for computer purposes and are efficient in terms of computer storage and time requirements. These methods arc mostly used to solve linear systems arising in numerical solutions of partial differential equations. Linear systems with as many as 100,000 variables often arise in the solution of partial differential equations. The coefficient matrix for these systems is sparse, i.e., the non-zero entries form a pattern. An iterative process provides an efficient method for solving these large systems.

Some of the advantages of iterative methods are as follows:

a) Fewer multiplications are required for large systems.

b) They have less round-off errors than elimination methods.

c) They are self-correcting if an error is made.

d) They use less computer memory when programmed.

e) They are quicker and easier to use when the coefficient matrix is sparse.

We consider here two classical iterative methods:

i) Jacobi's method.

ii) Gauss-Seidel method

Before we discuss the iterative methods for a system of equations, let us study the general concept of these methods:

Let $Ax = b$.

Assume that the equations are scaled so that $a_{ii} = 1$.

(where $i = 1, 2, \ldots, n$).

$$\left.\begin{array}{rcrcrcrcrcl} x_1 & + & a_{12}x_2 & + & & & \cdots & + & a_{1n}x_n & = & b_1 \\ a_{21}x_1 & + & x_2 & + & & & \cdots & + & a_{2n}x_n & = & b_2 \\ a_{31}x_1 & + & a_{32}x_2 & + & x_3 & + & \cdots & + & a_{3n}x_n & = & b_n \\ & & \vdots & & & & & & & & \\ a_{n1}x_1 & + & a_{n2}x_2 & + & & & \cdots & + & x_n & = & b_n \end{array}\right\} \ldots (8.16)$$

The equations (8.16) can be rewritten as follows:

$$\left.\begin{array}{rcrcrcrcl} x_1 & = & b_1 & - & a_{12}x_2 & - & \cdots & - & a_{1n}x_n \\ x_2 & = & b_2 & - & a_{21}x_1 & - & \cdots & - & a_{2n}x_n \\ & & & & \vdots & & & & \\ x_1 & = & b_i & - & a_{i1}x_1 & - & \cdots & - & a_{i,i-1}x_{i-1} - a_{i,i+1} - \cdots - a_{in}x_n \\ & & & & \vdots & & & & \\ x_n & = & b_n & - & a_{n1}x_1 & - & \cdots & - & a_{n,n-1}x_{n-1} \end{array}\right\} \ldots (8.17)$$

We start with some initial approximations to the unknown variables. Substituting these approximations into the right-hand side of (8.17), we get new approximations. The new approximations are then substituted into the right-hand side of (8.17). We get a second set of approximations and the procedure is repeated until successive values of each of the variables are sufficiently correct.

These iterative methods will not converge for all sets of equations, nor for all possible rearrangements of the equations. When the equations can be ordered so that each diagonal entry is larger in magnitude than the sum of the magnitudes of the other coefficients in that row, the iteration will converge for any initial guess. The speed with which the iterations converge is obviously related to the degree of dominance of the diagonal terms.

8.8.1 Jacobi's Method

Jacobi's method is valid only if all the a_{ij}'s are non-zero or if the equations can be suitably rearranged to make this so. This can always be done if A is invertible. Faster convergence can be achieved if we rearrange the rows so that the diagonal elements have

magnitudes as large as possible relative to the magnitudes of other coefficients in the same row. If this is not done sometimes Jacobi's method may not converge.

To solve the system of equations by Jacobi's method, the following steps are used:

a) Choose initial guesses:

$x_1^{(0)} = x_2^{(0)} = \ldots = x_n^{(0)} = 0$ if no better initial guesses are available.

b) Set $r = 0$

c) For each $i = 1, 2, \ldots, n$, compute

$$x_i^{(r+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{\substack{j=1 \\ j\neq i}}^{n} a_{ij} x_j^{(r)}\right) \qquad \ldots (8.18)$$

Assuming all $a_{ii} \neq 0$; $i = 1, 2, \ldots, n$; $r \geq 0$.

d) If solution vector $x^{(r)}$ is sufficiently accurate, then go to the last step. If $x^{(r)}$ is not sufficiently accurate, then add 1 to r and go to Step c.

e) **Termination of the process:**

The following two possible stopping criteria are used:

i) Use a fixed number of iterations.

ii) Use pre-assigned accuracy $\in$ as: $\left|x^{(r+1)} - x^{(r)}\right| < \in$.

We can combine both of them as well.

Jacobi's method is also known as the **method of simultaneous displacements** because each of the equations is simultaneously changed, by using the most recent set of x-values available.

Example 9 Solve the following system of equations using Jacobi's method:

$$\begin{aligned} 4x_1 - x_2 - x_3 &= 0.5 \\ -x_1 + 4x_2 - x_4 &= 1.3 \\ -x_1 + 4x_3 - x_4 &= 1.0 \\ -x_2 - x_3 + 4x_4 &= 1.8 \end{aligned}$$

Use $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = x_4^{(0)} = 0$.

Write also the computer program to implement the method.

Solution Dividing each equation by the coefficients of its diagonal term, we get the following:

$$
\begin{aligned}
x_1 - 0.25x_2 - 0.25x_3 &= 0.125 \\
-0.25x_1 + x_2 - 0.25x_4 &= 0.325 \\
-0.25x_1 + x_3 - 0.25x_4 &= 0.25 \\
-0.25x_2 - 0.25x_3 + x_4 &= 0.45
\end{aligned}
$$

Rewriting the above equations as follows:

$$
\begin{aligned}
x_1 &= 0.125 + 0.25(x_2 + x_3) \\
x_2 &= 0.325 + 0.25(x_1 + x_4) \\
x_3 &= 0.25 + 0.25(x_1 + x_4) \\
x_4 &= 0.45 + 0.25(x_2 + x_3)
\end{aligned}
$$

or,

$$x_1^{(r+1)} = 0.125 + 0.25\left(x_2^{(r)} + x_3^{(r)}\right)$$

$$x_2^{(r+1)} = 0.325 + 0.25\left(x_1^{(r)} + x_4^{(r)}\right)$$

$$x_3^{(r+1)} = 0.25 + 0.25\left(x_1^{(r)} + x_4^{(r)}\right)$$

$$x_4^{(r+1)} = 0.45 + 0.25\left(x_2^{(r)} + x_3^{(r)}\right)$$

Setting $r = 0$, we get,

$$x_1^{(1)} = 0.125 + 0.25\left(x_2^{(0)} + x_3^{(0)}\right)$$

$$x_2^{(1)} = 0.325 + 0.25\left(x_1^{(0)} + x_4^{(0)}\right)$$

$$x_3^{(1)} = 0.25 + 0.25\left(x_1^{(0)} + x_4^{(0)}\right)$$

$$x_4^{(1)} = 0.45 + 0.25\left(x_2^{(0)} + x_3^{(0)}\right)$$

Substituting the values of $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = x_4^{(0)} = 0$, we get,

$$x_1^{(1)} = 0.125$$

$$x_2^{(1)} = 0.325$$

$$x_3^{(1)} = 0.25$$

$$x_4^{(1)} = 0.45$$

Second approximation:

$$x_1^{(2)} = 0.125 + 0.25\left(x_2^{(1)} + x_3^{(1)}\right)$$

$$= 0.125 + 0.25\,(0.325 + 0.25) = 0.2688$$

$$x_2^{(2)} = 0.325 + 0.25\left(x_1^{(1)} + x_4^{(1)}\right)$$

$$= 0.325 + 0.25\,(0.125 + 0.25) = 0.4688$$

$$x_3^{(2)} = 0.25 + 0.25\left(x_1^{(1)} + x_4^{(1)}\right)$$

$$= 0.25 + 0.25\,(0.125 + 0.45) = 0.3938$$

$$x_4^{(0)} = 0.45 + 0.25\left(x_2^{(1)} + x_3^{(1)}\right)$$

$$= 0.45 + 0.25\,(0.325 + 0.25) = 0.5938$$

The process is repeated several times and the results are represented in the following table:

Iterations	x_1	x_2	x_3	x_4
0	0	0	0	0
1	0.125	0.325	0.25	0.45
2	0.2688	0.4688	0.3938	0.5938
3	0.3406	0.4506	0.4656	0.6656
⋮	⋮	⋮	⋮	⋮
8	0.4103	0.6103	0.5353	0.7353
9	0.4114	0.6114	0.5364	0.7364
10	0.4120	0.6120	0.5370	0.7370
True answer	0.4125	0.6125	0.5375	0.7375

It is obvious from the above results that the new values are better than the values obtained in the previous iteration.

Program No. 23: Jacobi's Method

```
# include<iostream.h>
# include<conio.h>
# include<stdio.h>
# define size 8
static float temp,data[size][size],x[10],xl [10];
short no_eq,itr;

void main(void)
  {
    short i,j,k=8,l=20,count=1;
    clrscr( );
    gotoxy(24,1);cout<<"JACOBI METHOD";
```

```
    gotoxy(17,2);
    cout<<"*************************\n";
    cout<<"\nHOW MANY EQUATIONS : ";
    cin >>no_eq;
    cout<<"\nENTER DATA FOR EQUATIONS";
    cout<<"\n=========================";
    for(i=0;i<no_eq-l;i++)
    for(j=0;j<=no_eq;j++)
      {
        gotoxy(I,k);
        cin>>data[i][j];
        I+8;
        If(j==no_eq)
          {
            k++;
            I+20;
          }
  }
cout<<"\nHOW MANY ITERATIONS YOU WANT TO DO:  ";
cin  >>itr;
for(i=0;i<=no_eq-l; i++)
  {
    temp=data[i][i];
    for(j=0;j<=no_eq;j++)
      data[i][j]/=temp;
  }
cout<<"\n\nITERATIONS                    RESULT\n\n";
while(count<=itr)
  {
    for(i=0;i<==no_eq-1;i++)
      {
        x1(i)- =data[i][no_eq];
        for(j=no_eq-1;j>=0;j--)
          {
            if(i==j)
              j--;
            x1[i]- =data[i][j]*x[j];
          }
      }
    cout<<"                    "<<count;
    for(k=0;k<=no_eq-1;k++)
      {
        x[k]=x1[k];
```

```
        printf("        %9.4f ',x[k]);
      }
    cout<<"\n";
    count++;
  }
getch( );
}
```

Computer Output

JACOBI METHOD

****

HOW MANY EQUATIONS : 4

ENTER DATA FOR EQUATIONS

==========

4	– 1	– 1	0	0.5
– 1	4	0	– 1	1.3
– 1	0	4	– 1	1
0	– 1	– 1	4	1.8

HOW MANY ITERATIONS YOU WANT TO DO : 12

ITERATIONS	RESULTS			
1	0.1250	0.3250	0.2500	0.4500
2	0.2688	0.4688	0.3938	0.5938
3	0.3406	0.5406	0.4656	0.6656
4	0.3766	0.5766	0.5016	0.7016
5	0.3945	0.5945	0.5195	0.7195
6	0.4035	0.6035	0.5285	0.7285
7	0.4080	0.6080	0.5330	0.7330
8	0.4103	0.6103	0.5353	0.7353
9	0.4114	0.6114	0.5364	0.7364
10	0.4119	0.6119	0.5369	0.7369
11	0.4122	0.6122	0.5372	0.7372
12	0.4124	0.6124	0.5374	0.7374

8.8.2 Gauss-Seidel Method

Jacobi's method is modified so that new approximations to the unknown are used as soon as they are available.

Formula (8.18) for $x_i^{(r+1)}$ is then modified as follows:

$$x_i^{(r+1)} = b_i - a_{i1}\, x_i^{(r+1)} - \ldots - a_{i,i-1}\, x_{i-1}^{(r+1)} - a_{i,i_1}\, x_{i+1}^{(r)} - \ldots - a_{in}\, x_n^{(r)} \quad \ldots (8.19)$$

or,
$$x_i^{(r+1)} = b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(r+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(r)} \quad \ldots (8.20)$$

Gauss-Seidel method will always converge if Jacobi's method converges and will do so more rapidly. If Jacobi's method diverges, so does Gauss-Seidel. Since the most recently computed values of x are used in the subsequent iterations, this makes the method more realistic and converges faster. This is generally the case but is not always true. In fact, there are linear systems for which Jacobi's method converges and the Gauss-Seidel method does not and conversely. On the whole, we can say that Gauss-Seidel method converges faster than Jacobi's method.

- As already mentioned that the solutions are quickly reached if $a_{11}, a_{22}, \ldots, a_{nn}$ are numerically larger compared with other coefficients. If necessary, the equations are rearranged so that the bigger coefficients are on the main diagonal.

The following steps are used to solve the system of equations by Gauss-Seidel method.

a) Choose the initial guess

b) Set $r = 0$

c) For each $i = 1, 2, \ldots, n$, compute

$$x_i^{(r+1)} = \frac{1}{a_{ii}}\left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(r+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(r)}\right) \quad \ldots (8.21)$$

d) If the solution vector

i) $x^{(r+1)}$ is sufficiently accurate, go to step (e),

ii) Otherwise, add 1 to r and go to Step (c).

e) Stop the process. Stopping criteria are the same as mentioned under Jacobi's method.

Example 10 Solve the previous example using Gauss-Seidel method.

Solution Initial approximation: $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = x_4^{(0)} = 0$.

The equations are now rewritten as follows:

$$x_2^{(r+1)} = 0.325 + 0.25\left(x_1^{(r+1)} + x_4^{(r)}\right)$$

$$x_3^{(r+1)} = 0.25 + 0.25\left(x_1^{(r+1)} + x_4^{(r)}\right)$$

$$x_4^{(r+1)} = 0.45 + 0.25\left(x_2^{(r+1)} + x_3^{(r+1)}\right)$$

Setting $r = 0$ and substituting the required values in the right-hand side, we get,

$$x_1^{(1)} = 0.125 + 0.25\left(x_2^{(0)} + x_3^{(0)}\right)$$

$$= 0.125 + 0.25\ (0 + 0) = 0.125$$

$$x_2^{(1)} = 0.325 + 0.25\left(x_1^{(1)} + x_4^{(0)}\right)$$

$$= 0.325 + 0.25\ (0.125 + 0) = 0.3563$$

$$x_3^{(1)} = 0.25 + 0.25\left(x_1^{(1)} + x_4^{(0)}\right)$$

$$= 0.25 + 0.25\ (0.125 + 0) = 0.2813$$

$$x_4^{(1)} = 0.45 + 0.25\left(x_2^{(1)} + x_3^{(1)}\right)$$

$$= 0.45 + 0.25\ (0.3563 + 0.2813) = 0.6094$$

The subsequent iterations are given as below:

Iterations	x_1	x_2	x_3	x_4
0	0	0	0	0
1	0.125	0.3563	0.2813	0.6094
2	0.2844	0.5484	0.4734	0.7055
3	0.3805	0.5965	0.5215	0.7295
⋮	⋮	⋮	⋮	⋮
7	0.4124	0.6124	0.5374	0.7375
8	0.4125	0.6125	0.5374	0.7375
True answer	0.4125	0.6125	0.5375	0.7375

This example shows that Gauss-Seidel method is faster and gives more accurate results than the previous method.

Program No. 24: Gauss-Seidel Method

```
# include<conio.h>
# include<stdio.h>
# include<iostream.h>

int noofeq,iteration;
float temp,arr[8][8],x[10];
```

```
void main(void)
  {
    int i,j, k=8, count=1,noofeq,iteration,xpos=10,ypos=8;
    float temp,arr[8][8],x[10];

    clrscr( );
    cout<< "\n\n\tGAUSS-SEIDEL METHOD";
    cout<<"\n\t======================="
    cout<<"\n\tNo Of Equations:  ";
    cin>>noofiq;
    cout<<"\n\tPlease Enter Data Of Equations :\n";
    for(i=0;i<= noofeq-1;i++)
      {
        xpos=10; ypos+=2;
        for(j=0;j<=noofeq;j++ )
          {
            gotoxy(xpos,ypos);
            cin>>arr[i][j];
            xpos+=8;
          }
      }
    cout<<"\n\tHow Many Iterations:  ";
    cin>>iteration;
    for(i=0;i<=noofeq-l;i++)
      {
        temp=arr[i][i];
        for(j=0;j<=noofeq;j++ )
                arr[i][j]/=temp;
      }
    cout<<"\n\nITERATIONS\t\tRESULTS\n\n";

    while(count<=iterations)
      {
        for(i=0;i<=noofeq-l;i++)
          {
            x[i]=arr[i][noofeq];
            for(j=noofeq;j>=o;j--;
              {
                if(i==j)j--;
                x[i]==[i][j]*x[j];
              }
          }
        cout<<count;
```

```
        i=0;
        for(k=0;k<=noofeq-1;k++)
      }
        printf("          %9.4f ",x[k];
    count<<"\n";
    count++;
  }
  getch( );
}
```

Computer Output

GUASS-SEIDEL METHOD

==========

No Of Equations: 4

Please Enter Data Of Equations:

4	– 1	– 1	0	0.5
– 1	4	0	– 1	1.3
– 1	0	4	– 1	1.0
0	– 1	– 1	4	1.8

How Many Iterations : 9

ITERATIONS		RESULTS		
1	0.1250	0.3562	0.2812	0.6094
2	0.2844	0.5484	0.4734	0.7055
3	0.3805	0.5965	0.5215	0.7295
4	0.4045	0.6085	0.5335	0.7355
5	0.4105	0.6115	0.5365	0.7370
6	0.4120	0.6122	0.5372	0.7374
7	0.4124	0.6124	0.5374	0.7375
8	0.4125	0.6125	0.5379	0.7375
9	0.4125	0.6125	0.5357	0.7375

Example 11 Given the following non-linear system of equations:

$$\begin{aligned} 4x + y^2 + z &= 11 \\ x + 4y + z^2 &= 18 \\ x^2 + y + 4z &= 15 \end{aligned}$$

Solve this system of equations using Gauss-Seidel method.

Solution

Certain linear systems of equation can be solved easily by Gauss-Seidel method. After a simple generalization, this method can also be used for solving some non-linear systems.

A solution to the problem at hand can be obtained by using Gauss-Seidel by arranging the system in diagonally dominant form:

Thus,

$$x = \frac{1}{4}\left(11 - y^2 - z\right)$$

$$y = \frac{1}{4}\left(18 - x - z^2\right)$$

$$z = \frac{1}{4}\left(15 - y - x^2\right)$$

Let the initial guess be $x^{(0)} = y^{(0)} = z^{(0)} = 1$.

Iterative Form:

$$\therefore x^{(n+1)} = \frac{1}{4}\left(11 - y^{(n)^2} - z^{(n)^1}\right)$$

$$y^{(n+1)} = \frac{1}{4}\left(18 - x^{(n+1)} - z^{(n)^2}\right)$$

$$z^{(n+1)} = \frac{1}{4}\left(15 - y^{(n+1)} - x^{(n+1)^2}\right)$$

Putting $n = 0$, we get

$$x^{(1)} = \frac{1}{4}\left(11 - y^{(0)^2} - z^{(0)}\right)$$

$$y^{(1)} = \frac{1}{4}\left(18 - x^{(1)} - z^{(0)^2}\right)$$

$$z^{(1)} = \frac{1}{4}\left(15 - y^{(1)} - x^{(1)^2}\right)$$

Substituting the values in the above form, we get

$$x^{(1)} = \frac{1}{4}\left[11-(1)^2-1\right] = 2.25$$

$$y^{(1)} = \frac{1}{4}\left[18-2.25-(1)^2\right] = 3.6875$$

$$z^{(1)} = \frac{1}{4}\left[15-3.6875-(2.25)^2\right] = 1.5625$$

After 67 iterations, we get the answer as:

$$x = 1.000112 \simeq 1.$$

$$y = 1.999962 \simeq 2.$$

$$z = 2.999953 \simeq 3.$$

It must be emphasized that we had no guarantee that this iteration would converge despite the diagonal dominance of the system, since a general theory for the iterative solution of nonlinear equation is not yet available.

PROBLEMS

1. Solve the following systems of equations using Cramer's rule:

a)
$$\begin{aligned} x_1 + x_2 + x_3 &= -2 \\ 3x_1 - x_2 + 2x_3 &= 4 \\ 4x_1 + 2x_2 + x_3 &= -8 \end{aligned}$$

b)
$$\begin{aligned} 4x_1 - x_2 + x_3 &= 3 \\ 2x_1 - 5x_2 - x_3 &= 2 \\ x_1 + 2x_2 + 6x_3 &= 5 \end{aligned}$$

c)
$$\begin{aligned} 6x_1 - 2x_2 + x_3 &= 7 \\ x_1 + 5x_2 - 2x_3 &= 6 \\ 2x_1 - x_2 + x_3 &= 4 \end{aligned}$$

d)
$$\begin{aligned} x_1 - 7x_2 + 4x_3 &= 9 \\ x_1 + 9x_2 - 6x_3 &= 1 \\ -3x_1 + 8x_2 + 5x_3 &= 6 \end{aligned}$$

e)
$$\begin{aligned} 0.3a + 0.52b + c &= -0.01 \\ 0.5a + b + 1.9c &= 0.67 \\ 0.1a + 0.3b + 0.5c &= -0.44 \end{aligned}$$

f)
$$\begin{aligned} x + y - z &= 1 \\ 2x + y + 2z &= 3 \\ 3x - y + z &= -2 \end{aligned}$$

2. Given the matrix,

$$A = \begin{pmatrix} 1 & 2 & 1 \\ 3 & 1 & -1 \\ 1 & 1 & 1 \end{pmatrix}$$

a) Compute adj(A) and then find A^{-1}.

b) Hence or otherwise, solve the following system of equations;

$$\begin{aligned} x_1 + 2x_2 + x_3 &= 3 \\ 3x_1 + x_2 - x_3 &= -11 \\ x_1 + x_2 + x_3 &= 3 \end{aligned}$$

3. (a) Find the inverse of the matrix,

$$A = \begin{pmatrix} 1 & -1 & 1 \\ 3 & -9 & 5 \\ 1 & -3 & 3 \end{pmatrix}$$

Hence or otherwise solve the system of equations:

$$\begin{aligned} x_1 - y + z &= 3 \\ 3x_1 - 9y + 5z &= 6 \\ x_1 - 3y + 3z &= 13 \end{aligned}$$

(b) Find the inverse of the following matrices:

i) $\begin{pmatrix} 2 & -3 & -5 & 2 \\ 1 & -4 & 7 & 4 \\ 0 & 2 & 0 & -1 \\ 2 & 1 & 4 & 1 \end{pmatrix}$ ii) $\begin{pmatrix} 13 & 14 & 6 & 4 \\ 8 & -1 & 13 & 9 \\ 6 & 7 & 3 & 2 \\ 9 & 5 & 16 & 11 \end{pmatrix}$

iii) $\begin{pmatrix} 3 & 7 & 8 & 15 \\ 2 & 5 & 6 & 11 \\ 2 & 6 & 10 & 19 \\ 4 & 11 & 19 & 38 \end{pmatrix}$ iv) $\begin{pmatrix} 0 & 4 & 1 & 2 \\ 3 & 1 & 2 & 0 \\ 1 & 2 & 3 & -1 \\ 2 & 5 & 1 & 3 \end{pmatrix}$

4. Solve the following systems of equations using simple Gaussian elimination, partial pivoting and complete pivoting methods. At each stage clearly indicate the pivots and multipliers you have used. Give reasons for any interchange you made.

a) (i)
$$\begin{aligned} x_1 + x_2 - 2x_3 &= 3 \\ 4x_1 - 2x_2 + x_3 &= 5 \\ 3x_1 - x_2 + 3x_3 &= 8 \end{aligned}$$

(ii)
$$\begin{aligned} 0.5x_1 + 0.4x_2 + 0.2x_3 &= 0.7 \\ 0.2x_1 + 20.1x_2 + 0.4x_3 &= 0.3 \\ 0.4x_1 + 0.3x_2 + 0.6x_3 &= 0.2 \end{aligned}$$

b)
$$\begin{aligned} x_1 - x_2 + x_3 - x_4 &= 1 \\ -x_1 - x_2 + x_3 + x_4 &= -2 \\ 2x_1 + 4x_2 + 3x_3 + 5x_4 &= -2 \\ 3x_1 + x_2 + x_3 + x_4 &= -1 \end{aligned}$$

c)
$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &= 1 \\ 2x_1 + x_2 + x_3 - x_4 &= 2 \\ 2x_1 + 2x_2 + x_3 + x_4 &= 3 \\ 3x_1 - x_2 - x_3 - 5x_4 &= 4 \end{aligned}$$

d)
$$\begin{aligned} 2x_1 + 8x_2 + 4x_3 + 10x_4 &= 7.6 \\ 3x_1 - 9x_2 + 15x_3 &= -8.1 \\ -x_1 - 2x_2 + 17x_3 + 35x_4 &= 9.4 \\ 2x_1 + 5x_2 - 14x_3 + 21x_4 &= 14.1 \end{aligned}$$

e)
$$\begin{aligned} x_1 + x_2 + 3x_4 &= 4 \\ 2x_1 + x_2 - x_3 - x_4 &= 1 \\ 3x_1 - x_2 - x_3 + 2x_4 &= -3 \\ -x_1 + 2x_2 + 3x_3 - x_4 &= 4 \end{aligned}$$

f)
$$\begin{aligned} x_1 + 2x_2 + x_3 + 4x_4 &= 13 \\ 2x_1 + 4x_3 + 3x_4 &= 28 \\ 4x_1 + 2x_2 + 2x_3 + x_4 &= 20 \\ -3x_1 + x_2 + 3x_3 + 2x_4 &= 6 \end{aligned}$$

g)
$$\begin{aligned} x_1 + x_2 + x_3 + x_4 &= 4 \\ 2x_1 + 3x_2 + 7x_3 - x_4 &= 11 \\ 3x_1 - 2x_2 + 5x_3 - 3x_4 &= 3 \\ 4x_1 - 5x_2 - 2x_3 - 3x_4 &= -6 \end{aligned}$$

h)
$$\begin{aligned} x_1 + 2x_2 - 12x_3 + 8x_4 &= 27 \\ 5x_1 + 4x_2 + 7x_3 - 2x_4 &= 4 \\ -3x_1 + 7x_2 + 9x_3 + 5x_4 &= 11 \\ 6x_1 - 12x_2 - 8x_3 + 3x_4 &= 49 \end{aligned}$$

i)
$$\begin{aligned} 10x_1 - 7x_2 + 3x_3 + 5x_4 &= 6 \\ -6x_1 + x_2 - x_3 - 4x_4 &= 5 \\ 3x_1 + x_2 + 4x_3 + 11x_4 &= 2 \\ 6x_1 - 9x_2 - 2x_3 + 4x_4 &= 7 \end{aligned}$$

j)
$$\begin{aligned} 2x_1 + 10x_2 - 6x_3 + 4x_4 + 8x_5 &= 8 \\ -3x_1 - 12x_2 - 9x_3 + 6x_4 + 3x_5 &= 3 \\ -x_1 + x_2 - 34x_3 + 15x_4 + 18x_5 &= 29 \\ 4x_1 + 18x_2 + 4x_4 + 14x_5 &= -2 \\ 5x_1 + 26x_2 - 19x_3 + 25x_4 + 36x_5 &= 23 \end{aligned}$$

k)
$$\begin{pmatrix} 1 & -1 & 2 & 1 \\ 3 & 2 & 1 & 4 \\ 5 & 8 & 6 & 3 \\ 4 & 2 & 5 & 3 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 1 \\ -1 \end{pmatrix}$$

l)
$$\begin{pmatrix} 3 & -13 & 9 & 3 \\ -6 & 4 & 1 & -18 \\ 6 & -2 & 2 & 4 \\ 12 & -8 & 6 & 10 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{pmatrix} = \begin{pmatrix} -19 \\ -34 \\ 16 \\ 26 \end{pmatrix}$$

5. Find the inverse of the following matrices using triangular decomposition method:

a) $A = \begin{pmatrix} 1 & 2 & 3 \\ 4 & 2 & 1 \\ 1 & 1 & 1 \end{pmatrix}$ b) $B = \begin{pmatrix} 50 & 107 & 36 \\ 25 & 54 & 20 \\ 31 & 66 & 21 \end{pmatrix}$

c) Solve the following system of equations,

$$\begin{aligned} 4x_1 + 3x_2 - x_3 &= -2 \\ -2x_1 - 4x_2 + 5x_3 &= 20 \\ x_1 + 2x_2 + 6x_3 &= 7 \end{aligned}$$

using the triangular decomposition method.

d) Solve the following system of equations using triangular decomposition method:

$$\begin{aligned} x_1 + 2x_2 + 3x_3 &= 14 \\ 2x_1 + 5x_2 + 2x_3 &= 18 \\ 3x_1 + x_2 + 5x_3 &= 20 \end{aligned}$$

e) Solve the following system of equations using triangular decomposition method:

$$\begin{aligned} 2x_1 + x_2 + 4x_3 &= 12 \\ 8x_1 - 3x_2 + 2x_3 &= 20 \\ 4x_1 + 11x_2 - x_3 &= 33 \end{aligned}$$

f) Solve the following system of equations using triangular decomposition method:

$$\begin{aligned} x_1 + 3x_2 + 8x_3 &= 4 \\ x_1 + 4x_2 + 3x_3 &= -2 \\ x_1 + 3x_2 + 4x_3 &= 1 \end{aligned}$$

6. (a) Find the inverse of the given matrix using Choleski's method,

$$A = \begin{pmatrix} 4 & 6 & 8 \\ 6 & 10 & 17 \\ 4 & 17 & 25 \end{pmatrix}$$

Hence, solve the following system of equations:

$$\begin{aligned} 4x_1 + 6x_2 + 8x_3 &= 6 \\ 6x_1 + 10x_2 + 17x_3 &= 5 \\ 8x_1 + 17x_2 + 25x_3 &= 7 \end{aligned}$$

(b) Find the inverse of the following matrix using Choleski's method,

$$A = \begin{pmatrix} 4 & 2 & 1 \\ 2 & 5 & -2 \\ 1 & -2 & 7 \end{pmatrix}$$

Hence or otherwise solve the system of equations:

$$\begin{aligned} 4x_1 + 2x_2 + x_3 &= 1.5 \\ 2x_1 + 5x_2 - 2x_3 &= 4.0 \\ x_1 - 2x_2 + 7x_3 &= 6.5 \end{aligned}$$

7. (a) Using Choleski's method, find the inverse of A:

$$A = \begin{pmatrix} 5 & 7 & 6 & 5 \\ 7 & 10 & 8 & 7 \\ 6 & 8 & 10 & 9 \\ 5 & 7 & 9 & 10 \end{pmatrix}$$

Hence or otherwise solve the following system of equations:

$$\begin{aligned} 5a + 7b + 6c + 5d &= 23 \\ 7a + 10b + 8c + 7d &= 32 \\ 6a + 8b + 10c + 9d &= 33 \\ 5a + 7b + 9c + 10d &= 33 \end{aligned}$$

(b) Find the inverse of the following matrix using Choleski's method:

$$A = \begin{pmatrix} 1 & 2 & \frac{1}{2} & 1 \\ 2 & 5 & 0 & -2 \\ \frac{1}{2} & 0 & 2\frac{1}{2} & 7\frac{1}{2} \\ 1 & -2 & 7\frac{1}{2} & 27 \end{pmatrix}$$

(c) Using decomposition method for symmetric matrices, find the inverse of the following matrix:

$$A = \begin{pmatrix} 1 & \frac{1}{2} & \frac{1}{3} \\ \frac{1}{2} & \frac{1}{3} & \frac{1}{4} \\ \frac{1}{3} & \frac{1}{4} & \frac{1}{5} \end{pmatrix}$$

(d) Find the Choleski's decomposition of matrix:

$$A = \begin{pmatrix} 2 & -1 & 0 & 0 \\ -1 & 2 & -1 & 0 \\ 0 & -1 & 2 & -1 \\ 0 & 0 & -1 & 2 \end{pmatrix}$$

Find also the inverse of the matrix.

8. Find the following systems of equations using traditional method:

a)
$$\begin{aligned} 2x_1 - x_2 &= 1 \\ -x_1 + 2x_2 - x_3 &= 0 \\ -x_2 + 2x_3 - x_4 &= 0 \\ -x_3 + 2x_4 &= 1 \end{aligned}$$

b)
$$\begin{aligned} 2x_1 - x_2 &= 1 \\ -x_1 + 2x_2 - x_3 &= 2 \\ -x_2 + 2x_3 - x_4 &= 3 \\ -x_3 + 2x_4 &= 4 \end{aligned}$$

c)
$$\begin{aligned} 2x_1 - x_2 &= 1 \\ -x_1 + 2x_2 - x_3 &= 1 \\ -x_2 + 2x_3 - x_4 &= 1 \\ -x_3 + 2x_4 - x_5 &= 1 \\ -x_4 + 2x_5 - x_6 &= 1 \\ -x_5 + 2x_6 &= 1 \end{aligned}$$

d)
$$\begin{pmatrix} -4 & 1 & & & & & & & & \\ 1 & -4 & 1 & & & & & & & \\ & 1 & -4 & 1 & & & & & & \\ & & 1 & -4 & 1 & & & & & \\ & & & 1 & -4 & 1 & & & & \\ & & & & 1 & -4 & 1 & & & \\ & & & & & 1 & -4 & 1 & & \\ & & & & & & 1 & -4 & 1 & \\ & & & & & & & 1 & -4 & 1 \\ & & & & & & & & 1 & -4 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \\ x_7 \\ x_8 \\ x_9 \\ x_{10} \end{pmatrix} = \begin{pmatrix} -27 \\ -15 \\ -15 \\ -15 \\ -15 \\ -15 \\ -15 \\ -15 \\ -15 \\ -15 \end{pmatrix}$$

e) Let A be the 50×50 tridiagonal matrix:

$$\begin{pmatrix} 5 & -1 & & & & \\ -1 & 5 & -1 & & & \\ & -1 & 5 & -1 & & \\ & & \ddots & \ddots & \ddots & \\ & & & -1 & 5 & -1 \\ & & & & -1 & 5 \end{pmatrix}$$

Consider the problem $Ax = b$ for 50 different vectors b of the form:

$$[1, 2, \cdots, 49, 50]^T,\ [2, 3, \cdots, 50, 1]^T,\ [3, 4, \cdots, 50, 1, 2]^T, \ldots$$

Solve this problem using the computer.

9. Solve the following systems of equations using Jacobi's method:

a)
$$\begin{aligned} 10x + y + z &= 24 \\ -x + 20y + z &= 21 \\ -x - 2y + 100z &= 300 \end{aligned}$$

Take starting values as (0, 0, 0).

b)
$$\begin{aligned} 8x_1 - 3x_2 + 2x_3 &= 20 \\ 4x_1 + 11x_2 - x_3 &= 33 \\ 6x_1 + 3x_2 + 12x_3 &= 36 \end{aligned}$$

Take the initial guess as (0, 0, 0).

c)
$$\begin{aligned} 5x_1 - x_2 + x_3 &= 10 \\ -2x_1 + 8x_2 - x_3 &= 11 \\ -x_1 + x_2 + 4x_3 &= 3 \end{aligned}$$

Take $x_1 = x_2 = x_3 = 0$.

d)
$$\begin{aligned} 10x_1 + x_2 + 8x_3 &= 16 \\ x_1 + 10x_2 - 2x_3 &= 22 \\ 2x_1 + 22x_2 + 10x_3 &= 51 \end{aligned}$$

e) Consider the following systems of equations:

i)
$$\begin{aligned} 5x_1 + 3x_2 &= 6 \\ 4x_1 - 2x_2 &= 8 \end{aligned}$$

ii)
$$\begin{aligned} 2x_1 + x_2 - 5x_3 &= 9 \\ x_1 - 5x_2 - x_3 &= 14 \\ 7x_1 - x_2 - 3x_3 &= 26 \end{aligned}$$

iii)
$$\begin{aligned} x + 2y + 10z &= 59 \\ 2x + 8y + z &= -4 \\ 20x - y + 2z &= 74 \end{aligned}$$

Can Jacobi's method be used to solve them? Why?

f)
$$\begin{aligned} 10x_1 - x_2 + 2x_3 &= 6 \\ -x_1 + 11x_2 - x_3 + 3x_4 &= 25 \\ 2x_1 - x_2 + 10x_3 - x_4 &= -11 \\ 3x_2 - 4x_3 + 8x_4 &= 15 \end{aligned}$$

Take $x_1 = x_2 = x_3 = x_4 = 0$.

g)
$$\begin{aligned} 10x_1 + x_2 - 2x_3 &= 6 \\ x_1 + 10x_2 - x_3 + 3x_4 &= 25 \\ -2x_1 - x_2 + 8x_3 - x_4 &= -11 \\ 3x_2 - x_3 + 5x_4 &= -11 \end{aligned}$$

Take $x_1 = x_2 = x_3 = x_4 = 0$.

10. Solve the following systems of equations using Gauss-Seidel's method taking the initial guess as: (0, 0, 0, 0, 0, 0)

a)
$$\begin{aligned} 11.84x_1 + 9.15x_2 + 2.15x_3 &= 6.88 \\ 4.26x_1 + 15.36x_2 - 2.89x_3 &= -8.61 \\ 6.30x_1 - 5.88x_2 + 3.85x_3 &= 12.95 \end{aligned}$$

b)
$$\begin{aligned} x_1 + 2x_2 - x_3 &= 0 \\ 2x_1 + \frac{2}{3}x_2 + \frac{1}{3}x_3 &= 1 \\ 6x_1 - 2x_2 + 2x_3 &= -2 \end{aligned}$$

c)
$$\begin{aligned} 16x_1 + 2x_2 - 4x_3 + x_4 &= 15 \\ x_1 + 10x_2 - 15x_3 + 3x_4 &= 1 \\ 3x_1 + x_2 + 15x_3 + 2x_4 &= -40 \\ x_1 + 2x_2 - 14x_3 + 18x_4 &= 61 \end{aligned}$$

d)
$$\begin{aligned} x_1 + 10x_2 + x_3 &= 10 \\ 2x_1 + 20x_3 + x_4 &= 10 \\ 3x_2 + 10x_5 + 3x_6 &= 0 \\ 10x_1 + x_2 + x_6 &= 5 \\ 2x_4 + 2x_5 + 20x_6 &= 5 \\ x_3 + 10x_4 + x_5 &= 5 \end{aligned}$$

e)
$$\begin{aligned} -16x_1 + 2x_2 + x_4 &= -30 \\ 12x_2 + x_3 - x_4 &= 9 \\ 2x_2 + 11x_3 + 2x_4 &= 16 \\ x_1 + x_2 - 15x_4 &= -11 \end{aligned}$$

11. Using an iterative method solve the following nonlinear systems of equations:

a)
$$\begin{aligned} \frac{2}{x} + \frac{1}{y} + \frac{3}{z} &= 2.5 \\ \frac{1}{x} + \frac{5}{y} + \frac{1}{z} &= -1.8 \\ \frac{4}{x} + \frac{3}{y} + \frac{4}{z} &= 2.8 \end{aligned}$$

b)
$$\frac{5.22}{x} + \frac{1.75}{y} - \frac{2.13}{z} = 30.74$$
$$\frac{3.82}{x} + \frac{7.11}{y} + \frac{4.91}{z} = 34.60$$
$$\frac{4.76}{x} + \frac{0.87}{y} + \frac{6.41}{z} = 2.40$$

c)
$$x - \frac{1}{10} y^2 + \frac{5}{100} z^2 = 0.7$$
$$y + \frac{3}{10} x^2 - \frac{1}{10} x z = 0.5$$
$$z + \frac{4}{10} y^2 + \frac{1}{10} x y = 1.2$$

d)
$$\begin{aligned} 5x_1 + x_2^3 + x_3 + x_4 &= 8.7 \\ x_1^2 - 6x_2 + 2x_3 - x_4 &= 7.3 \\ x_1 - x_2 + 4x_3 + x_4^2 &= 17.29 \\ 2x_1 + x_2 + x_3^2 + 11x_4 &= 34.7 \end{aligned}$$

e)
$$\begin{aligned} 4x + y^2 + z &= 11 \\ x + 4y + z^2 &= 18 \\ x^2 + y + 4z &= 15 \end{aligned}$$

Assume all initial values to be 1.

12. (a) Solve the following system of equations,

a)
$$\begin{aligned} 9x_1 + 4x_2 + x_3 &= -17 \\ x_1 - 2x_2 - 6x_3 &= 14 \\ x_1 + 6x_2 &= 4 \end{aligned}$$

using Jacobi's and Gauss-Seidel methods. Which one is faster?

Use $x_1 = x_2 = x_3 = 0$.

(b) Solve the following systems of equations using Jacobi's and Seidel methods.

Let $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = 0$.

i)
$$\begin{aligned} 9x_1 - x_2 + 2x_3 &= 9 \\ -x_1 + 10x_2 - 2x_3 &= 15 \\ 2x_1 - 2x_2 - 11x_3 &= -22 \end{aligned}$$

ii)
$$\begin{aligned} 25x_1 + 2x_2 + x_3 &= 70 \\ 2x_1 + 10x_2 + x_3 &= 60 \\ x_1 + 2x_2 + 4x_3 &= 40 \end{aligned}$$

iii)
$$\begin{aligned} 10x_1 + x_2 + x_3 &= 15 \\ x_1 + 10x_2 + x_3 &= 24 \\ x_1 + x_2 + 10x_3 &= 33 \end{aligned}$$

iv)
$$\begin{aligned} 8x_1 - 3x_2 + 2x_3 &= 20 \\ 4x_1 + 11x_2 - x_3 &= 33 \\ 6x_1 + 3x_2 + 12x_3 &= 36 \end{aligned}$$

v)
$$\begin{aligned} 14x_1 + 3x_2 - x_3 &= 5 \\ 2x_1 + 5x_2 + 13x_3 &= 9 \\ 5x_1 + 13x_2 + 7x_3 &= 8 \end{aligned}$$

vi)
$$\begin{aligned} -2x_1 + x_2 &= -1 \\ x_1 - 2x_2 + x_3 &= 0 \\ x_2 - 2x_2 + x_4 &= 0 \\ x_3 - x_4 &= 0 \end{aligned}$$

13. Use the computer program to solve the following system of equations by an iterative method:

$$
\begin{aligned}
x_1 + 10x_2 + x_3 &= 10 \\
2x_1 + 20x_3 + x_4 &= 10 \\
3x_2 + 30x_5 + x_6 &= 0 \\
10x_1 + x_2 - x_6 &= 5 \\
2x_4 - 2x_5 + 20x_6 &= 5 \\
x_3 + 10x_4 - x_5 &= 0
\end{aligned}
$$

Take $x_1^{(0)} = x_2^{(0)} = x_3^{(0)} = x_4^{(0)} = x_5^{(0)} = x_6^{(0)} = 0.$

Chapter 9

Eigenvalues and Eigenvectors

9.1 INTRODUCTION

In this chapter, we study some basics of computing **eigenvalues** and **eigenvectors**. They play a prominent role in the study of differential equations and in many applications in engineering and physical sciences.

Let A be a square matrix, $[a_{ij}]_{n\times n}$. We shall investigate the problem of finding, λ, and non-trivial vector $x_{1\times n}$ (A vector is non-trivial if its all components are not equal to zero), such that,

$$Ax = \lambda x \qquad \text{... (i)}$$

or $\quad (Ax - \lambda x) = 0$

or $\quad (A - \lambda I)\, x = 0 \qquad \text{... (ii)}$

It is known that a solution x ($\neq 0$) exists provided

$$\det\,(A - \lambda) = 0 \qquad \text{... (iii)}$$

More explicitly,

$$\begin{pmatrix} a_{11}-\lambda & a_{12} & a_{13} & \cdots & a_{1n} \\ a_{21} & a_{22}-\lambda & a_{23} & \cdots & a_{2n} \\ a_{31} & a_{32} & a_{33}-\lambda & \cdots & a_{3n} \\ \vdots & \vdots & \vdots & & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \cdots & a_{nn}-\lambda \end{pmatrix} = 0 \qquad \text{... (iv)}$$

If we are to expand the above determinant, we obtain an nth degree polynomial in λ:

$$\det\,(A - \lambda I) = (-1)^n \lambda^n + a_{n-1}\lambda^{n-1} + a_{n-2}\lambda^{n-2} + \cdots + a_1 \lambda + a_0$$

The above polynomial called the **characteristic polynomial** of A. Each value of λ which satisfies (iv), yields a system of homogeneous equations of equation (ii). Thus the problem, of finding the values of λ for which (ii) possesses non-trivial solutions is the same as finding the roots of the characteristic polynomial:

$$(-1)^n \lambda^n + a_{n-1}\lambda^{n-1} + a_{n-2}\lambda^{n-2} + \cdots + a_1 \lambda + a_0 = 0 \qquad \ldots (v)$$

Here, (v) is called the **characteristic equation** (also sometimes called the **secular equation**) whose roots are $\lambda_1, \lambda_2, \ldots, \lambda_n$ and are called the **eigenvalues** (or **latent roots**) of A and x is called the **eigenvector** (or **latent vector**) of A corresponding to λ. These roots can be **distinct** (i.e., $\lambda_1 \neq \lambda_2 \neq \ldots \neq \lambda_n$), or **complex** or **repeated**. In the case of multiple roots, say a p-fold root λ_j, the problem is more involved.

9.2 METHODS TO SOLVE EIGENVALUE PROBLEMS

The eigenvalue problem reduces to the problem of finding the roots of the characteristic equation, det $(A - \lambda I) = 0$. This can be done directly by expanding the determinant in power of λ if A is of order 3×3 or less. As the size of the given matrix grows; this method rapidly becomes inefficient and time-consuming. However, for particular cases, for instance, for sparse matrices, it may still be quite useful.

If the given matrix is a real symmetric matrix, all its roots are real and Newton's method may be used to find the roots of the characteristic equation (See Section 255). If the given matrix is real but not symmetric, there may be complex roots of the characteristic equation. If $\lambda = a + i\,b$ is a root, then $\lambda = a - i\,b$ is also a root of the characteristic equation and corresponding to these two complex roots is the real quadratic factor $\lambda^2 - 2a\lambda + a^2 + b^2 = 0$. It is necessary therefore to first seek quadratic factor, if the given matrix is real and non-symmetric.

To solve eigenvalue problem (determination of eigenvalues and the corresponding eigenvectors) has grown into an extensive special area of numerical methods. The methods developed for this purpose are numerous and it is not possible to describe them one by one or even summarize them comprehensively in this book. To keep our study to a reasonable length, we restrict our attention to the following three methods:

- General method
- Leverrier-Faddeev method
- Power method

Let us describe the above methods one buy one.

9.2.1 General Method

This is a simple method and we illustrate it by the following two examples:

Example 1 Find the eigenvalues and the corresponding eigenvectors of the following matrix:

$$A = \begin{vmatrix} 2 & 1 & -1 \\ 0 & -2 & -2 \\ 1 & 1 & 0 \end{vmatrix}$$

Solution **Characteristic Polynomial**

$$\det(A - \lambda I) = \det\left(\begin{vmatrix} 2-\lambda & 1 & -1 \\ 0 & -2-\lambda & -2 \\ 1 & 1 & -\lambda \end{vmatrix}\right)$$

Expanding the determinant:

$$\det(A - \lambda I) = (2-\lambda)\begin{vmatrix} -2-\lambda & -2 \\ 1 & -\lambda \end{vmatrix} - 0\begin{vmatrix} 1 & -1 \\ 1 & -\lambda \end{vmatrix} + 1\begin{vmatrix} 1 & -1 \\ -2-\lambda & -2 \end{vmatrix}$$

$$= (2-\lambda)[2\lambda + \lambda^2 + 2] + 0 + [-2-2-\lambda]$$

$$= 4\lambda + 2\lambda^2 + 4 - 2\lambda^2 - \lambda^3 - 2\lambda - 4 - \lambda$$

$$= -\lambda^3 + \lambda$$

Roots of the Characteristic Equation

$$-\lambda^3 + \lambda = 0$$

or $\lambda^3 - \lambda = 0$

or $\lambda(\lambda^2 - 1) = 0$

$\lambda_1 = 0$, $\lambda_2 = 1$ and $\lambda_3 = -1$

So the eigenvalues are $0, 1, -1$.

The set of all eigenvalues of matrix A, usually denoted by the symbol $\sigma(A)$, is called the **spectrum** of A.

The eigenvectors corresponding to the above eigenvalues may now be calculated.

(i) When $\lambda_1 = 0$, eigenvector, $x^{(1)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\therefore (A - \lambda I) = \begin{bmatrix} 2 & 1 & -1 \\ 0 & -2 & -2 \\ 1 & 1 & 0 \end{bmatrix}$$

$$(A - \lambda I)x = \begin{bmatrix} 2 & 1 & -1 \\ 0 & -2 & -2 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving:

$$\begin{aligned} 2x + y - z &= 0 \\ -2y - 2z &= 0 \\ x + y &= 0 \end{aligned}$$

Thus, from the last two equations, we get

$$x = -y = z$$

$$x^{(1)} = \begin{bmatrix} x \\ -x \\ x \end{bmatrix}$$

Since eigenvalues are of arbitrary length, we are free to choose one component. So, we may choose any **non-zero** value of x.

Let us use x = 1 and we get

When $\lambda_1 = 0$, $x^{(1)} = \begin{bmatrix} 1 \\ -1 \\ 1 \end{bmatrix} = [1 \quad -1 \quad 1]^T$

(ii) When $\lambda_2 = 1$, eigenvector, $x^{(2)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\therefore (A - \lambda I) = \begin{bmatrix} 1 & 1 & -1 \\ 0 & -3 & -2 \\ 1 & 1 & -1 \end{bmatrix}$$

$$(A - \lambda I)x = \begin{bmatrix} 1 & 1 & -1 \\ 0 & -3 & -2 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving:

$$\begin{aligned} x + y - z &= 0 \\ -3y - 2z &= 0 \\ x + y - z &= 0 \end{aligned}$$

These equations yield:

$$y = \frac{-2}{3} z$$

$$x = -y + z$$

$$= +\frac{2}{3} z + z = \frac{5}{3} z$$

$$\therefore \quad x^{(2)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} \frac{5}{3} z \\ -\frac{2}{3} z \\ z \end{bmatrix}$$

Letting $z = 1$, we get

When $\lambda_2 = 0$, $x^{(2)} = \begin{bmatrix} \frac{5}{3} \\ -\frac{2}{3} \\ 1 \end{bmatrix} = \begin{bmatrix} \frac{5}{3} & \frac{-2}{3} & 1 \end{bmatrix}^T$

(iii) When $\lambda_3 = -1$, eigenvector, $x^{(3)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\therefore (A-\lambda I) = \begin{bmatrix} 3 & 1 & -1 \\ 0 & -1 & -2 \\ 1 & 1 & 1 \end{bmatrix}$$

$$(A-\lambda I)x = \begin{bmatrix} 3 & 1 & -1 \\ 0 & -1 & -2 \\ 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$\begin{aligned} 3x + y - z &= 0 \\ -y - 2z &= 0 \\ x + y - z &= 0 \end{aligned}$$

Solving:

$$y = -2z$$

$$x = -y - z$$

$$= +2z - z = z$$

$$x^{(3)} = \begin{bmatrix} z \\ -2z \\ z \end{bmatrix}$$

Letting $z = 1$, we get

When $\lambda_3 = -1$, $x^{(3)} = \begin{bmatrix} 1 \\ -2 \\ 1 \end{bmatrix} = [1 \quad -2 \quad 1]^T$

Answers:

When $\lambda_1 = 0$, $x^{(1)} = [1 \quad -1 \quad 1]^T$

When $\lambda_2 = 0$, $x^{(2)} = \left[\frac{5}{3} \quad \frac{-2}{3} \quad 1\right]^T$

When $\lambda_3 = -1$, $x^{(3)} = [1 \quad -2 \quad 1]^T$

Example 2 Find the eigenvalues and their corresponding eigenvectors from the following matrix:

$$A = \begin{bmatrix} 2 & 0 & 0 \\ 2 & 2 & 1 \\ 1 & 1 & 2 \end{bmatrix}$$

Solution **Characteristic Polynomial**

$$\det(A-\lambda I) = \det\left(\begin{bmatrix} 2-\lambda & 0 & 0 \\ 2 & 2-\lambda & 1 \\ 1 & 1 & 2-\lambda \end{bmatrix}\right)$$

$$= (2-\lambda)\begin{bmatrix} 2-\lambda & 1 \\ 1 & 2-\lambda \end{bmatrix} - 0\begin{bmatrix} 2 & 1 \\ 1 & 2-\lambda \end{bmatrix} + 0\begin{bmatrix} 2 & 2-\lambda \\ 1 & 1 \end{bmatrix}$$

$$= (2-\lambda)\,[(2-\lambda)^2 - 1] - 0 + 0$$

$$= (2-\lambda)\,[4 + \lambda^2 - 4\lambda - 1]$$

$$= (2-\lambda)\,[\lambda^2 - 4\lambda + 3]$$

$$= -\lambda^3 + 6\lambda^2 - 11\lambda + 6$$

Characteristic Polynomial

$$\lambda^3 - 6\lambda^2 + 11\lambda - 6 = 0$$

Factorizing to get eigenvalues:

$$(\lambda-1)(\lambda-2)(\lambda-3) = 0$$

$$\therefore\ \lambda = 1, 2, 3$$

(i) When $\lambda_1 = 1$, eigenvector, $x^{(1)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\therefore \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}\begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

Solving:

$$\begin{aligned} x &= 0 \\ 2x + y + z &= 0 \\ x + y + z &= 0 \end{aligned}$$

Thus, $x = 0,\ z = -y$

$$x^{(1)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ y \\ -y \end{bmatrix}$$

Letting $y = 1$, we get

$$\therefore \quad x^{(1)} = \begin{bmatrix} 0 \\ 1 \\ -1 \end{bmatrix}$$

When $\lambda_1 = 1$, $x^{(1)} = [0 \quad 1 \quad -1]^T$

(ii) When $\lambda_2 = 2$; $x^{(2)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\begin{bmatrix} 0 & 0 & 0 \\ 2 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$2x + z = 0$$
$$x + y = 0$$

$x = -y$; or $y = -x$

$$z = -2x$$

$$x^{(2)} = \begin{bmatrix} x \\ -x \\ -2x \end{bmatrix}; \text{ let } x = 1$$

$$= \begin{bmatrix} 1 \\ -1 \\ -2 \end{bmatrix}$$

When $\lambda_2 = 2$, $x^{(2)} = [1 \quad -1 \quad -2]^T$

(iii) When $\lambda_3 = 3$; $x^{(3)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\begin{bmatrix} -1 & 0 & 0 \\ 2 & -1 & 1 \\ 1 & 1 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \end{bmatrix}$$

$$-x = 0$$
$$2x - y + z = 0$$
$$x + y + z = 0$$

$$x = 0$$
$$z = y$$

$$x^{(3)} = \begin{bmatrix} 0 \\ y \\ y \end{bmatrix}; \text{ put } y = 1$$

$$= \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$$

When $\lambda_3 = 3$, $x^{(3)} = [0 \;\; 1 \;\; 1]^T$

Answers:

When $\lambda_1 = 1$, $x^{(1)} = [0 \;\; 1 \;\; -1]^T$

When $\lambda_2 = 2$, $x^{(2)} = [0 \;\; -1 \;\; -2]^T$

When $\lambda_3 = 3$, $x^{(3)} = [0 \;\; 1 \;\; 1]^T$

Some Remarks

It is important to remember that following points in using this method:

a) Eigenvalues and eigenvectors can be real as well as complex valued.

b) The dimension of the eigenspace corresponding to an eigenvalue is less than or equal to the multiplicity of that eigenvalue.

c) The method used above is suitable for 2×2 and 3×3 matrices. Eigenvalues and eigenvectors of larger matrices are often computed using some other techniques described in the later sections.

d) However, this method is not suitable for computer.

9.2.2 Leverrier-Faddeev Method

The **Leverrier-Faddeev method** is used to find all eigenvalues and the corresponding eigenvectors. It is a more efficient method as compared to previously discussed and it can be easily computerized.

It uses the **trace** and proceeds as follows:

Let $A_1 = A$ (where A is the given matrix). Also, $P_1 = \text{trace}(A) = \sum_{i=1}^{n} a_{ii}$

Let $A_2 = A(A_1 - P_1 I)$; $P_2 = \frac{1}{2}\text{trace}(A_2)$

Let $A_3 = A(A_2 - P_2 I)$; $P_3 = \frac{1}{3}\text{trace}(A_3)$

$\vdots$

Let $A_n = A(A_{n-1} - P_{n-1} I)$; $P_n = \frac{1}{n}\text{trace}(A_n)$

The numbers $P_1, P_2, \ldots, P_n$ are required coefficients in the characteristic equation. Then, $\lambda^n - P_1\lambda^{n-1} - P_2\lambda^{n-2} - \cdots - P_n = 0$.

Solve it for $\lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_n$.

Check: From the last step, $A_n - P_n I = 0$

Before solving a numerical example, we will introduce the following terminology:

- Trace and determinant of a matrix
- Inverse of a matrix
- Spectral radius

(a) Trace and Determinant of a Matrix

The sum of diagonal elements of a square matrix is called the **trace** of the matrix and equals the sum of its eigenvalues.

Let $A = [a_{ij}]_{n\times n}$ be an nth order non-singular square matrix, then the trace of A is

$$\text{tr}(A) = \sum_{i=1} a_{ij} \text{ ; sum of the diagonal elements.}$$

$$\text{Also tr}(A) = \sum_{i=1}^{n} \lambda_i \text{ ; sum of } \lambda\text{'s.}$$

Determinant of the matrix A:

$$\det(A) = \lambda_1, \lambda_2, \lambda_3, \ldots, \lambda_i$$

$$= \prod_{i=1}^{n} \lambda_i$$

It means that the product of the eigenvalues of a square matrix is equal to the determinant of that matrix.

(b) Inverse

This method can also be used for finding the inverse of A which is given by:

$$A^{-1} = \frac{1}{P_n}[A_{n-1} - P_{n-1} I]$$

(c) Spectral Radius

The spectral radius of a square matrix A is the largest absolute eigenvalue. It is denoted by $\delta(A)$.

$$\delta(A) = \max |\lambda_i|; \ 1 \le i \le n.$$

Example 3 (a) Determine the eigenvalues for the following matrix:

$$A = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix}$$

(b) Find also the inverse, trace, determinant and spectral radius of A.

Solution

(a) Let $A_1 = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix}$; $P_1 = tr(A_1) = \sum a_{ii} = 3 + 0 + 3 = 6$

$$A_2 = A(A_1 - P_1 I)$$

$$= \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix} \left(\begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix} - 6 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix} \begin{bmatrix} -3 & 2 & 4 \\ 2 & -6 & 3 \\ 4 & 2 & -3 \end{bmatrix} = \begin{bmatrix} 11 & 2 & 4 \\ 2 & 8 & 2 \\ 4 & 2 & 11 \end{bmatrix}$$

$$P_2 = \frac{1}{2} tr(A_2)$$

$$= \frac{1}{2}[11 + 8 + 11] = \frac{1}{2} \times 30 = 15$$

$$A_3 = A(A_2 - P_2 I)$$

$$= \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix} \left(\begin{bmatrix} 3 & 2 & 4 \\ 2 & 8 & 2 \\ 4 & 2 & 11 \end{bmatrix} - 15 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)$$

$$= \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix} \begin{bmatrix} -4 & 2 & 4 \\ 2 & -7 & 3 \\ 4 & 2 & -4 \end{bmatrix}$$

$$= \begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 8 \end{bmatrix}$$

$$P_3 = \frac{1}{3} \operatorname{tr}(A_3)$$

$$= \frac{1}{3}[8 + 8 + 8] = 8$$

Check: $A_3 - P_3 I = 0$

$$\begin{bmatrix} 8 & 0 & 0 \\ 0 & 8 & 0 \\ 0 & 0 & 8 \end{bmatrix} - 8 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} = 0$$

Characteristic Polynomial

$$\lambda^3 - 6\lambda^2 - 15\lambda - 8 = 0$$

Factorizing :

$$(\lambda + 1)(\lambda^2 - 7\lambda - 8) = 0$$

$$(\lambda + 1)(\lambda + 1)(\lambda - 8) = 0$$

$$\lambda = -1, -1, 8$$

(b) **Inverse:** A^{-1}

$$A^{-1} = \frac{1}{P_n}[A_{n-1} - P_{n-1} I]$$

$$= \frac{1}{8} \left(\begin{bmatrix} 11 & 2 & 4 \\ 2 & 8 & 2 \\ 4 & 2 & 11 \end{bmatrix} - 15 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right)$$

$$= \frac{1}{8}\begin{bmatrix} -4 & 2 & 4 \\ 2 & -7 & 3 \\ 4 & 2 & -4 \end{bmatrix}$$

$$= \begin{bmatrix} -\frac{1}{2} & \frac{1}{4} & \frac{1}{2} \\ \frac{1}{4} & -\frac{7}{8} & \frac{1}{4} \\ \frac{1}{2} & \frac{1}{4} & -\frac{1}{2} \end{bmatrix}$$

Trace of A :

$$\text{tr}(A) = \sum_{i=1}^{n} \lambda_i$$

$$= -1 - 1 + 8 = 6$$

Determinant of A :

$$\det(A) = \prod_{i=1}^{n} \lambda_i$$

$$= -1 \times -1 \times 8 = 8$$

Spectral radius of A :

$$\delta(\lambda_i) = 8$$

9.2.3 Power Method

It is the simplest iterative procedure for determining the **largest** (or **principal**) eigenvalue and the corresponding eigenvector of a matrix. It is easy to apply and is probably the most widely used method. The eigenvalue having the greatest absolute value is called the **dominant eigenvalue**. This method is used because in many applications only the dominant eigenvalue of a matrix is needed. Power method fails if there is no dominant eigenvalue.

Assume eigenvalues of an $n \times n$ matrix are arranged to be

$$|\lambda_1| > |\lambda_2| \geq |\lambda_3| \geq \ldots \geq |\lambda_n| \geq 0$$

The process proceeds as follows:

Let $x^{(1)}$ be any non-zero vector and define a sequence of vectors,

$$x^{(2)}, x^{(3)}, x^{(4)}, \ldots, x^{(n)}$$

by the recursive relation:

$$x^{(r+1)} = A x^{(r)}$$

Then as $r \to \infty$, $x^{(r)} \to$ (multiple of) $q^{(1)}$. It is important to know that for finding the roots of a polynomial equation of degree ≥ 4 is not a simple task, it has to be carried out using iterative methods.

Proof Express $x^{(1)}$ in terms of eigenvectors:

$$x^{(1)} = \alpha_1 q^{(1)} + \alpha_2 q^{(2)} + \ldots + \alpha_n q^{(n)} \qquad \text{Linear independence.}$$

Then,
$$x^{(2)} = A[\alpha_1 q^{(1)} + \alpha_2 q^{(2)} + \ldots + \alpha_n q^{(n)}] \qquad (\text{since } Aq = \lambda q)$$

$$= \lambda_1 \alpha_1 q^{(1)} + \lambda_2 \alpha_2 q^{(2)} + \ldots + \lambda_n \alpha_n q^{(n)}$$

$$x^{(3)} = A[\lambda_1 \alpha_1 q^{(1)} + \lambda_2 \alpha_2 q^{(2)} + \ldots + \lambda_n \alpha_n q^{(n)}]$$

$$= \alpha_1 \lambda_1^2 q^{(1)} + \alpha_2 \lambda_2^2 q^{(2)} + \ldots + \alpha_n \lambda_n^2 q^{(n)}$$

$$\vdots$$

$$x^{(r)} = \alpha_1 \lambda_1^{(r-1)} q^{(1)} + \alpha_2 \lambda_2^{(r-1)} q^{(2)} + \ldots + \alpha_n \lambda_n^{(r-1)} q^{(n)}$$

So, if r is large enough, $\left|\lambda_1^{r-1}\right| \gg \left|\lambda_2^{r-1}\right|$, and so, $x^{(r)} \simeq \alpha_1 \lambda_1^{r-1} q^{(1)}$.

In practice, we usually scale down at each iteration by dividing $x^{(r)}$ by its largest element:

i.e. $\quad y^{(r+1)} = A x^{(r)}$

Then as $r \to \infty$, $x^{(r)} \to q^{(1)}$ and ratio of $y^{(r+1)}$ to $x^{(r)} \to \lambda_1$.

This iterative method will converge if the largest eigenvalue is real and is not a multiple root. Convergence is most rapid when the ratio of the largest eigenvlue to the next largest eigenvalue is large.

Computing the smallest eigenvalue of a matrix

The eigenvalue of smallest magnitude of a matrix is the same as the inverse (reciprocal) of the dominant eigenvalue of the inverse of the matrix. Since most applications of eigenvalues need the eigenvalue of smallest magnitude, the inverse matrix is often solved for its dominant eigenvalue. This is why the dominant eigenvalue is so important.

In order to find the smallest eigenvalue of a matrix, we apply the principle that the reciprocals of eigenvalues of a matrix are the eigenvalues of the inverse of the matrix. That is, if λ is an eigenvalue of A, then

$$A^{-1} X = \frac{1}{\lambda^2}$$

Therefore, taking the inverse of A and then using the iteration we have just described will give the largest eigenvalue of the inverse of A. The reciprocal of this value will then be the smallest eigenvalue of A.

Let us now illustrate this method by the following two examples.

Example 4 Given the following square matrix:

$$\begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}$$

Find its dominant eigenvalue and its corresponding eigenvector using power method. Try the initial guess as: $x^{(1)} = [1 \quad 1 \quad 1]^T$.

Solution

Given $y^{(r+1)} = A\,x^{(r)}$

Let r = 1, then $y^{(2)} = A\,x^{(1)}$

$x^{(2)} = \frac{1}{6}\,y^{(2)} = \frac{1}{6}\begin{bmatrix} 6 \\ -1 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -.167 \\ 1 \end{bmatrix}$	$y^{(2)} = A\,x^{(1)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ -1 \\ 6 \end{bmatrix}$ Largest value
$x^{(3)} = \frac{1}{6}\,y^{(3)} = \frac{1}{6}\begin{bmatrix} 6 \\ -.167 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -.0278 \\ 1 \end{bmatrix}$	$y^{(3)} = A\,x^{(2)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ -.167 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ -.167 \\ 6 \end{bmatrix}$
$x^{(4)} = \frac{1}{6}\,y^{(4)} = \frac{1}{6}\begin{bmatrix} 6 \\ -.0278 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -.0046 \\ 1 \end{bmatrix}$	$y^{(4)} = A\,x^{(3)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ .0278 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ -.0278 \\ 6 \end{bmatrix}$
$x^{(5)} = \frac{1}{6}\,y^{(5)} = \frac{1}{6}\begin{bmatrix} 6 \\ .0046 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ .0008 \\ 1 \end{bmatrix}$	$y^{(5)} = A\,x^{(4)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ -.0046 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ .0046 \\ 6 \end{bmatrix}$

$x^{(6)} = \frac{1}{6}\, y^{(6)} = \frac{1}{6}\begin{bmatrix} 6 \\ -.0008 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ -.0001 \\ 1 \end{bmatrix}$	$y^{(6)} = A\,x^{(5)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ -.0008 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ -.0008 \\ 6 \end{bmatrix}$
$x^{(7)} = \frac{1}{6}\, y^{(7)} = \frac{1}{6}\begin{bmatrix} 6 \\ .0001 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$	$y^{(7)} = A\,x^{(6)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ -.0001 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ .0001 \\ 6 \end{bmatrix}$
$x^{(8)} = \frac{1}{6}\, y^{(8)} = \frac{1}{6}\begin{bmatrix} 6 \\ 0 \\ 6 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$	$y^{(8)} = A\,x^{(7)} = \begin{bmatrix} 5 & 0 & 1 \\ 0 & -1 & 0 \\ 1 & 0 & 5 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \\ 6 \end{bmatrix}$

At this stage, $x^{(8)} = x^{(7)}$

Thus, $\lambda_1 \approx 6$

$$q^{(1)} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$$

True answer : $\lambda_1 = 6;\ \lambda_2 = 4;\ \lambda_3 = -1$

Corresponding eigenvectors :

For $\lambda_1 = 6$, $\quad q^{(1)} = \begin{bmatrix} 1 \\ 0 \\ 1 \end{bmatrix}$,

For $\lambda_2 = 4$, $\quad q^{(2)} = \begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$,

For $\lambda_3 = -1$, $\quad q^{(3)} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}$.

Example 5: Use power method to determine the dominant eigenvalue and its corresponding eigenvector of the following matrix:

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix}$$

Use $x^{(1)} = [1 \;\; 1 \;\; 1]^T$.

Write computer program to implement power method.

Solution: Let $y^{(2)} = A\,x^{(1)}$ and $x^{(1)} = [1 \;\; 1 \;\; 1]^T$

$x^{(2)} = \frac{1}{7}\, y^{(2)} = \frac{1}{7}\begin{bmatrix} 0 \\ 4 \\ 7 \end{bmatrix} = \begin{bmatrix} 0 \\ .57 \\ 1 \end{bmatrix}$	$y^{(2)} = A\,x^{(1)} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix}\begin{bmatrix} 1 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0 \\ 4 \\ 7 \end{bmatrix}$
$x^{(3)} = \frac{1}{4.14}\, y^{(3)} = \frac{1}{4.14}\begin{bmatrix} -1 \\ 2.14 \\ 4.14 \end{bmatrix} = \begin{bmatrix} -.24 \\ .52 \\ 1 \end{bmatrix}$	$y^{(3)} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix}\begin{bmatrix} 0 \\ .57 \\ 1 \end{bmatrix} = \begin{bmatrix} -1 \\ 2.14 \\ 4.14 \end{bmatrix}$
$x^{(4)} = \frac{1}{3.56}\begin{bmatrix} -1.24 \\ 1.80 \\ 3.56 \end{bmatrix} = \begin{bmatrix} -0.35 \\ .51 \\ 1 \end{bmatrix}$	$y^{(4)} = A\,x^{(3)} = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix}\begin{bmatrix} -.24 \\ .52 \\ 1 \end{bmatrix} = \begin{bmatrix} -1.24 \\ 1.80 \\ 3.56 \end{bmatrix}$
$\vdots$	$\vdots$ $\vdots$
$x^{(10)} = \frac{1}{3.04}\begin{bmatrix} -1.48 \\ 1.52 \\ 3.04 \end{bmatrix} = \begin{bmatrix} -.49 \\ .50 \\ 1 \end{bmatrix}$	$y^{(10)} = \begin{bmatrix} -1.48 \\ 1.52 \\ 3.04 \end{bmatrix}$
$x^{(11)} = \frac{1}{3.02}\begin{bmatrix} -1.49 \\ 1.51 \\ 3.02 \end{bmatrix} = \begin{bmatrix} -0.49 \\ 0.50 \\ 1 \end{bmatrix}$	$y^{(11)} = \begin{bmatrix} -1.49 \\ 1.51 \\ 3.02 \end{bmatrix}$

$$x^{(10)} = x^{(11)}$$

$$\therefore \lambda_1 = 3$$

$$q^{(1)} \approx \begin{bmatrix} -0.49 \\ 0.50 \\ 1 \end{bmatrix}$$

True answer : $\lambda_1 = 3;\ q^{(1)} = \begin{bmatrix} -.50 \\ .50 \\ 1 \end{bmatrix}$

This program has been taken from the following website:

http://www.net.pk/mtshome/appNumericalAnalysis.html

Computer Program **Power Method**

```
#include <iostream.h>
#include <math.h>
#include <fstream.h>
#include <string.h>

void read_n(int&,ifstream&);
void read_array(double**, int, ifstream&);
void write_array(double**, int);
void deallocate_arrary(double**,int);
void mult(double**, double*, double*,int);
double fun(double*,int);
double vectornorm(double*,int);

void main( )
{
        int n,i,k,M;
        double** array;
        double* x;
        double* y;
        double r,vn,tol,rold;

        M=1000;
        tol=0.0;
        ifstream arrayin;
        arrayin.open("array.dat");

        read_n(n,arrayin);
        x=new double[n];
        y=new double[n];

        x[0]=-1.0;
        x[1]=1.0;
```

```
x[2]=1.0;

for(i=0;i<n;i++)
{
        y[i]=0.0;
}

//allocate array
array=new double*[n];
for (i=0;i<n;i++)
{
        array[i] = new double[n];
}

read_ array(array, n, arrayin);
arrayin.close( );

//array now read into file
cout<<"Original Matrix:"<<endl;
write_array(array,n);

k=0;
rold=1.0;
r=10.0;
while(k<M && fabs(rold-r)>tol)
{
    cout<<"k= "<<k<<" x="<<x[0]<<" "<<x[1]<<" "<<x[2]<<"r="
    <<r<<endl;
    mult(array,x,y,n);
    rold=r;
    r=fun(y,n)/fun(x,n);
    vn=vectornorm(y,n);
    for(i=0;i<n;i++)x[i]=y[i]/vn;
    k++;
}
cout<<"k= "<<k<<" x="<<x[0]<<" "<<x[1]<<" "<<x[2]<<"r="
<<r<<endl;
deallocate_array(array,n);
delete x,y;
cout<<"Press ENTER to end"<<endl;
cin.get( );
return 0;
```

```
void read_n(int &n, ifstream & arrayin)
{
        char temp;
        //read in number of rows
        n=int(arrayin.get( ) )-int('0');
        temp=arrayin.get( );   //get next character
        while(temp != '      ' && temp != '\n')
        {
                n=n*10+int(temp)-int('0');
                temp=arrayin.get( );
        }
}

void read_array(double** array, int n, ifstream &arrayin)
{
    double tempd,div;
    bool divflag;
    char temp;
    int i,j;
    //read in array from file
    for(i=0;i<n;i++)
    {
        for(j=0;j<n;j++)
        {
            div=1;
            divflag=false;
            tempd=0.0;
            if(!arrayin.eof( ) ) temp=arrayin.get( );  //get next character
            while(temp != ' ' && temp != '\n' && !arrayin.eof( ) )
            {
                if (temp== '-')
                {
                    div*=-1;
                    temp=arrayin.get( );
                }
                if (tem== '.')
                {
                        temp=arrayin.get( );
                        divflag=true;
                }
                else
```

```
                {
                        if (divflag==true)
                        {
                            div=div*10.0;
                        }
                        tempd=tempd*10+int(temp)-int('0');
                        if(!arrayin.eof( ) )
                        {
                            temp=arrayin.get( );
                        }
                }
            }
            array[i][j]=tempd/div;
        }
        while(!arrayin.eof( ) && temp != '\n')
        {
            temp=arrayin.get( );
        }
    }
    //array[i][j] is now the ith row jth column element of the array retrun;
}
void deallocate_array(double** array, int n)
{
    // deallocate array
    int i;
    for (i=0;i<n;i++)
    {
      deleat[ ] array[i];
    }
    deleat[ ] array;
}

void write_array(double** array,int n)
{
    int i,j;
    for(i=0;i<n;i++)
    {
            for(j=0;j<n;j++)
            {
                cout<<array[i][j]<< " ";
            }
            cout<<endl;
    }
```

```
    return;
}

void mult(double** A,double* x,double* y,int n)
{
        for (int i=0;i<n;i++)
        {
            y[i]=0;
            for (int j=0;j<n;j++)
            {
                y[i]=y[i]+A[i][j]*x[j];
            }
        }
        return;
}

double fun(double* x,int n)
{
    //cout<<x[0]<< " "<<[1]<<" "<<x[2]<<endl;
    return x[1];
}

double vectornorm(double* x,int n)
{
    double answer=x[0];
    for(int i=1;i<n;i++)
    {
        if(abs(x[i])>=abs(answer))
        {
            answer=x[i];
        }
    }
    return answer;
}
```

9.3 MATRIX DEFLATION

There are different methods for finding subsequent eigenvalues of a matrix, we will discuss only one of these, i.e. the **deflation method** which is a straightforward approach.

Suppose we have applied the power method to a matrix A and have obtained its largest eigenvalue λ_1 and corresponding eigenvector $q^{(1)}$. We now require to find the eigenvalue λ_2, to do so, λ_1 must be removed by a process called **deflation**. Deflation

may be defined as the process of finding a matrix A_1 of order $(n-1)$, whose eigenvalues are identical with those of A, except λ_1.

This process can thus be continued until all the eigenvalues of A have been computed.

Theorem Let q be an eigenvector of A corresponding to an eigenvalue λ and let v be an eigenvector of A^T corresponding eigenvalue μ when $\mu \neq \lambda$, prove that $v^T q = 0$.

Proof We know that

$$A\,q = \lambda\,q \text{ – multiplying both sides by } v^T.$$

Hence, $v^T A\,q = \lambda\, v^T q$ ... (i)

Also, $A^T v = \mu\, v$; Transpose

$$\left(A^T v\right)^T = \left(\mu\, v\right)^T$$

$$v^T A = \mu\, v^T \quad \text{... Multiplying both sides by q}$$

$$v^T A\,q = \mu\, v^T q \quad \text{... (ii)}$$

Comparing (i) and (ii), we get

$$\lambda\, v^T q = \mu\, v^T q$$

Since $\lambda \neq \mu$, $(\lambda - \mu)\, v^T q = 0$

$\therefore\ v^T \cdot q = 0$

Although we have used deflation to find subsequent eigenvector-eigenvalue pairs, there is a point where rounding error reduces the accuracy below acceptable limits. To avoid this difficulty, other methods, like Jacobi's method, are preferred when we need to compute many or all eigenvalues of a given matrix.

9.3.1 Hotelling's Deflation

Hotelling's deflation is based on the result that the matrix,

$$A_1 = A - \lambda_1\, q^{(1)}\, v^{(1)^T}$$

(where $v^{(1)}$ is the eigenvector of A^T corresponding to eigenvalue λ_1) has

$$\left.\begin{array}{ll} \text{eigenvalues:} & 0, \lambda_2, \lambda_3, \ldots, \lambda_n \\ \text{eigenvectors:} & q^{(1)}, q^{(2)}, q^{(3)}, \ldots, q^{(n)} \end{array}\right\} \quad \begin{array}{l} \text{provided } q^{(1)}, v^{(1)} \text{ are scaled} \\ \text{so that } v^{(1)^T} \cdot q^{(1)} = 1 \end{array}$$

Proof

Let $q^{(i)}$ be the eigenvector corresponding to an eigenvalue λ_1.

Then, postmultiplying by $q^{(i)}$

$$A_1 q^{(i)} = A q^{(i)} - \lambda_1 q^{(1)} \cdot q^{(i)} v^{(1)^T}$$
$$= A q^{(i)} - \lambda_1 q^{(1)} \cdot v^{(1)^T} q^{(i)}$$

Two cases will be discussed.

(a) When $i = 1$:

$$A_1 q^{(1)} = A q^{(1)} - \lambda_1 q^{(1)} = 0$$

(Since $v^{(1)^T} q^{(1)} = 1$ and $A q^{(1)} = \lambda_1 q^{(1)}$)

Thus, $q^{(1)}$ is an eigenvector of A_1 corresponding to eigenvalue 0.

(b) When $i \neq 1$, so that $\lambda_i \neq \lambda_1$.

Then $A_1 q^{(i)} = \lambda_i q^{(i)} - 0$

(since by theorem $v^{(1)^T} q^{(i)} = 0$)

Thus, $q^{(i)}$ is an eigenvector of A_1 corresponding to eigenvalue λ_1.

Steps to deflate a matrix

In summary, the process of matrix deflation in any of its forms consists of the following steps:

(i) Find λ_1, the dominant eigenvalue of A by the power method and x_1, the corresponding eigenvector.

(ii) Deflate the matrix A to get a new matrix with dominant eigenvalue λ_2.

(iii) Find this dominant eigenvalue and the corresponding eigenvector. Then find the corresponding eigenvector of A.

(iv) Repeat steps (ii) & (iii), using the last deflated matrix at each step, until as many eigenvalues and eigenvectors of A are to be found.

Example 6 Apply Hotelling's deflation method to deflate the matrix:

$$A = \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix}$$

Given the dominant eigenvalue $\lambda_1 = 3$.

Solution To apply Hotelling's Deflation, we require $v^{(1)}$, eigenvector of A^T:

$$A^T = \begin{bmatrix} 1 & 1 & 2 \\ 0 & 2 & 2 \\ -1 & 1 & 3 \end{bmatrix}$$

Hence, $A^T v^1 = \lambda v^{(1)}$; $(A^T - \lambda I) v^{(1)} = 0$,

$$A^T v^{(1)} - \lambda v^{(1)} = 0$$

$$\left(\begin{bmatrix} 1 & 1 & 2 \\ 0 & 2 & 2 \\ -1 & 1 & 3 \end{bmatrix} - 3 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) v^{(1)} = 0$$

Let $v^{(1)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix}$

$$\therefore \quad -2x + y + 2z = 0 \quad \ldots \text{(i)}$$

$$-y + 2z = 0 \quad \ldots \text{(ii)}$$

$$-x + y = 0 \quad \ldots \text{(iii)}$$

From (iii), $y = x$

From (ii), $z = \frac{1}{2} y = \frac{1}{2} x$

$$\therefore \quad v^{(1)} = \begin{bmatrix} x \\ x \\ \frac{1}{2} x \end{bmatrix} = x \begin{bmatrix} 1 \\ 1 \\ \frac{1}{2} \end{bmatrix}$$

$$v^{(1)T} q^{(1)} = \det \left(x \begin{bmatrix} 1 & 1 & \frac{1}{2} \end{bmatrix} \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix} \right) = 1$$

$$= \det \left(\frac{1}{2} x \right) = 1$$

$$\therefore \quad \frac{1}{2} x = 1 \; ; \; x = 2$$

So, we take $v^{(1)} = x \begin{bmatrix} 1 \\ 1 \\ \frac{1}{2} \end{bmatrix}$

$$= 2 \begin{bmatrix} 1 \\ 1 \\ \frac{1}{2} \end{bmatrix} = [2 \quad 2 \quad 1]^T$$

Thus, $A_1 = A - \lambda_1 q^{(1)} v^{(1)^T}$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix} - 3 \begin{bmatrix} -\frac{1}{2} \\ \frac{1}{2} \\ 1 \end{bmatrix} [2 \quad 2 \quad 1]$$

$$= \begin{bmatrix} 1 & 0 & -1 \\ 1 & 2 & 1 \\ 2 & 2 & 3 \end{bmatrix} - \begin{bmatrix} -3 & -3 & -\frac{3}{2} \\ 3 & 3 & \frac{3}{2} \\ 6 & 6 & 3 \end{bmatrix}$$

$$= \begin{bmatrix} 4 & 3 & \frac{1}{2} \\ -2 & -1 & -\frac{1}{2} \\ -4 & -4 & 0 \end{bmatrix}; \text{ Deflated matrix}$$

We apply the power method on A_1 to find the other eigenvalues λ_2, i.e.,

$$A_2 = A_1 - \lambda_2 q^{(2)} v^{(2)^T}.$$

9.3.2 Hotelling's Deflation for Symmetric Matrices

If A is symmetric, $A^T = A$ and $v^{(1)} = q^{(1)}$. Thus provided $q^{(1)}$ is scaled so that $q^{(1)} q^{(1)} = 1$, the matrix given by Hotelling's deflation is

$$A_1 = A - \lambda_1 q^{(1)} q^{(1)}$$

Example 7 Deflate A when the largest eigenvalue is $\lambda_1 = 7$,

$$A = \begin{bmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{bmatrix}$$

Solution Let eigenvector $q^{(1)} = [x \quad y \quad z]^T$

$$(A^T - \lambda I)\, q^{(1)} = 0$$

$$\left(\begin{bmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{bmatrix} - 7 \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \right) \begin{bmatrix} x \\ y \\ z \end{bmatrix} = 0$$

$$-5x - 4y + 2z = 0 \qquad \text{... (i)}$$

$$-4x - 5y - 2z = 0 \qquad \text{... (ii)}$$

$$2x - 2y - 8z = 0 \qquad \text{... (iii)}$$

Add (i) and (ii) :

$$-9x - 9y = 0;\ y = -x$$

Substitute in (iii) :

$$4x - 8z = 0$$

$$x = 2z$$

$$q^{(1)} = \begin{bmatrix} x \\ y \\ z \end{bmatrix} = \begin{bmatrix} 2z \\ -2z \\ z \end{bmatrix} = z \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}$$

$$q^{(1)} \cdot q^{(1)} = z^2(4+4+1) = 1$$

$$9z^2 = 1,\ z = \frac{1}{3}$$

$$q^{(1)} = z \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix} = \frac{1}{3} \begin{bmatrix} 2 \\ -2 \\ 1 \end{bmatrix}$$

$$= \begin{bmatrix} \frac{2}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix}^T$$

So, $A_1 = A - \lambda_1 q^{(1)} q^{(1)^T}$

$$= \begin{bmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{bmatrix} - 7 \begin{bmatrix} \frac{2}{3} \\ -\frac{2}{3} \\ \frac{1}{3} \end{bmatrix} \begin{bmatrix} \frac{2}{3} & -\frac{2}{3} & \frac{1}{3} \end{bmatrix}$$

$$= \begin{bmatrix} 2 & -4 & 2 \\ -4 & 2 & -2 \\ 2 & -2 & -1 \end{bmatrix} - 7 \begin{bmatrix} \frac{4}{9} & -\frac{4}{9} & \frac{2}{9} \\ -\frac{4}{9} & \frac{4}{9} & -\frac{2}{9} \\ \frac{2}{9} & -\frac{2}{9} & \frac{1}{9} \end{bmatrix}$$

$$= \begin{bmatrix} \frac{10}{9} & -\frac{8}{9} & \frac{4}{9} \\ -\frac{8}{9} & -\frac{10}{9} & -\frac{4}{9} \\ \frac{4}{9} & -\frac{4}{9} & -\frac{16}{9} \end{bmatrix}$$

If A is symmetric, Hotelling's deflation gives a symmetric matrix.

9.4 PROPERTIES OF EIGENVALUES AND EIGENVECTORS

It might be a good idea to highlight briefly some important properties of eigenvalues and eigenvectors.

We mention here the following:

- The absolute value of a determinant ($|\det A|$) is the product of the absolute values of the eigenvalues of matrix A.
- $\lambda = 0$ is an eigenvalue of A if A is a singular (noninvertible) matrix.
- If A is a $n \times n$ triangular matrix (upper triangular or lower triangular) or diagonal matrix, the eigenvalues of A are the diagonal entries of A.
- The matrix A and its transpose have same eigenvalues.
- Eigenvalues of a symmetric matrix are orthogonal, but only for distinct eigenvalues.
- The dominant or principal eigenvector of a matrix is an eigenvector corresponding to the eigenvalue of largest magnitude (for real numbers, largest absolute value) of that matrix.
- For a transition matrix, the dominant eigenvalue is always 1.

- The smallest eigenvalue of a matrix A is the same as the inverse (reciprocal) of the largest eigenvalue of A^{-1}, i.e. of inverse of A.
- If we know an eigenvalue, its eigenvector can be computed. The reverse process is also possible; i.e., given an eigenvector, its corresponding eigenvalue can be calculated.

Example 8 Consider the following upper triangular matrix:

$$A = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 2 & 3 & 4 & 5 \\ 0 & 0 & 3 & 4 & 5 \\ 0 & 0 & 0 & 4 & 5 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix}$$

Find its eigenvalues and spectral radius:

Solution

$$\det(A) = \det\left(\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 2 & 3 & 4 & 5 \\ 0 & 0 & 3 & 4 & 5 \\ 0 & 0 & 0 & 4 & 5 \\ 0 & 0 & 0 & 0 & 5 \end{bmatrix}\right)$$

$$= (\lambda - 1)(\lambda - 2)(\lambda - 3)(\lambda - 4)(\lambda - 5) = 0$$

Eigenvalues are: $\lambda_1 = 1$; $\lambda_2 = 2$; $\lambda_3 = 3$; $\lambda_4 = 4$; $\lambda_5 = 5$.

Spectral radius, $\delta(A) = 5$.

9.5 GERSHGORIN'S THEOREM

In the previous sections, we have studied some methods to compute eigenvalues and their corresponding eigenvectors. Now, we study **Gerschgorin's Theorem.**

Statement

All the eigenvalues λ's of the matrix A lie within the union of the circular disc specified by the following inequalities:

$$|\lambda - \lambda_{kk}| \le \sum_{\substack{j=1 \\ j \ne k}}^{n} |a_{kj}|$$

where each $k = 1, 2, \ldots, n$.

Proof

Let λ be any eigenvalue of $A_{n\times n}$ with corresponding eigenvector:

$$x = [x_1, x_2, x_3, \ldots, x_n]^T$$

then we can write $Ax = \lambda x$

or $$\sum_{j=1}^{n} a_{kj} \cdot x_j = \lambda x_k \qquad \ldots (1)$$

for $k = 1, 2, \ldots, n.$

or equivalently,

$$\sum_{\substack{j=1 \\ j\neq k}}^{n} a_{kj}\, x_j = (\lambda - a_{kk})\, x_k \qquad \ldots (2)$$

Since x is an eigenvector, it is non-zero. Suppose its kth component is the largest in absolute value:

$$|x_k| = \max \{|x_j|\} \qquad \ldots (3)$$

Let us divide both sides of (2) by x_k, $(x_k \neq 0)$:

$$\sum_{\substack{j=1 \\ j\neq k}}^{n} a_{kj} \frac{x_j}{x_k} = \lambda - a_{kk} \qquad \ldots (4)$$

If we now take the absolute value of both sides of (4), we get

$$|\lambda - a_{kk}| = \left| \sum_{\substack{j=1 \\ j\neq k}}^{n} a_{kj} \frac{x_j}{x_k} \right| \qquad \ldots (5)$$

By triangle-inequality ($|xy| \leq |x| \cdot |y|$), the R.H.S. of (5) satisfies the inequality:

$$\left| \sum_{\substack{j=1 \\ j\neq k}}^{n} a_{kj} \frac{x_j}{x_k} \right| \leq \sum_{\substack{j=1 \\ j\neq k}}^{n} |a_{kj}| \cdot \frac{|x_j|}{|x_{jk}|} \qquad \ldots (6)$$

Moreover from (3), it is apparent that

$$\frac{|x_j|}{|x_k|} \leq 1\,; \qquad j = 1, 2, \ldots, n$$

$$\left|\sum_{\substack{j=1\\j\neq k}}^{n} a_{kj}\frac{x_j}{x_k}\right| \leq \sum_{\substack{j=1\\j\neq k}}^{n} |a_{kj}|\cdot 1 \leq \left|\sum_{\substack{j=1\\j\neq k}}^{n} a_{kj}\right| \qquad \ldots (7)$$

Comparing (5) and (7), we get

$$|\lambda - \lambda_{kk}| = \sum_{\substack{j=1\\j\neq k}}^{n} a_{kj} \qquad \ldots (8)$$

$|\lambda - a_{kk}|$ is called disc, and

$|a_{kj}|$ is radius.

Let us apply the above theorem to obtain as much information as possible about the eigenvalues of matrices.

Examination of Equation (8)

For each k, the set of λ which satisfies (8) is a disc with center at a_{kk} and radius r_k, where

$$r_k = \sum_{\substack{j=1\\j\neq k}}^{n} |a_{kj}| \qquad \ldots (9)$$

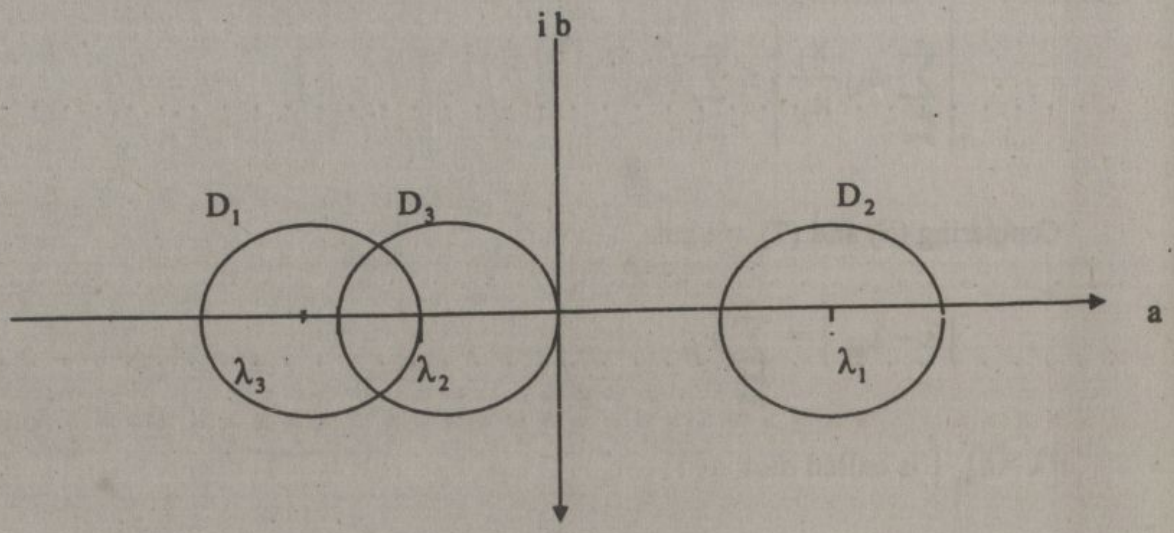
The term disc means a circle and its interior. Here the disc is in the complex plane (set of all complex numbers). The disc here means the Gerschgorin's disc. Thus, if we denote each of these n disc by D_k:

$$D_k = \{\lambda : |\lambda - a_{kk}| \leq r_k\};\ k = 1, 2, \ldots, n. \qquad \ldots (10)$$

then each (and therefore every) eigenvalue of A must lie in the union S of these discs:

$$S = \bigcup_{k=1}^{n} D_k \qquad \ldots (11)$$

Look at the following figure:



- Overlapping discs represent complex eigenvalues
- Isolated discs represent exactly different real eigenvalues.

Example 9 Given the following matrix:

$$A = \begin{bmatrix} -2 & 0 & -1 \\ 0 & 2 & 1 \\ 1 & 0 & -1 \end{bmatrix}$$

(a) Compute the eigenvalues of A.

(b) Use the Gerschgorin's theorem to obtain bounds on the magnitude of eigenvalues of the above matrix. .

Solution

(a) The eigenvalues are computed and are as follows:

$$\lambda_1 = 2$$

$$\lambda_2 = \frac{1}{2}\left(-3+\sqrt{3}\right)$$

$$\lambda_3 = \frac{1}{2}\left(-3-\sqrt{3}\right)$$

(b) **Bounds**

The Gerschgorin's discs are:

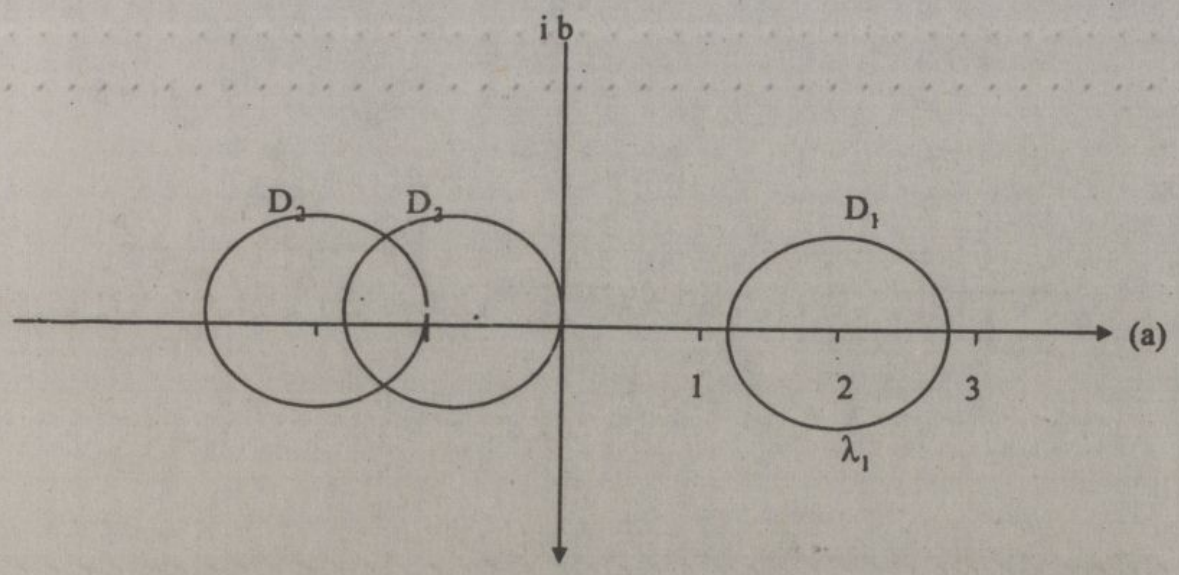
$$D_1 = \{\lambda : |\lambda+2| \le 1\};\ \lambda+2=1; \lambda=-1$$

$$D_2 = \{\lambda : |\lambda-2| \le 1\};\ \lambda-2=1; \lambda=3$$

$$D_3 = \{\lambda : |\lambda+1| \le 1\};\ \lambda+1=1; \lambda=0.$$

It is clear that λ lies between -1 and 3, i.e. $-1 \le \lambda \le 3$.

Figure:



From the diagram it is clear that the disc $|\lambda + 2| \le 1$ is isolated from the other contained in that disc. Moreover, since complex eigenvalues occur in pairs, we can assert that lone eigenvalue is real.

Example 10 Given the matrix:

$$A = \begin{bmatrix} 1 & 6 & 2 \\ 2 & -3 & -1 \\ 0 & 4 & 1 \end{bmatrix}$$

Using the Gerschgorin's theorem, find the range in which the eigenvalues lie.

Solution **Finding discs**

Row-wise:

$$D_1 = |\lambda - 1| \le |6| + |2| = 8$$

$$D_2 = |\lambda + 3| \le |2| + |-1| = 3$$

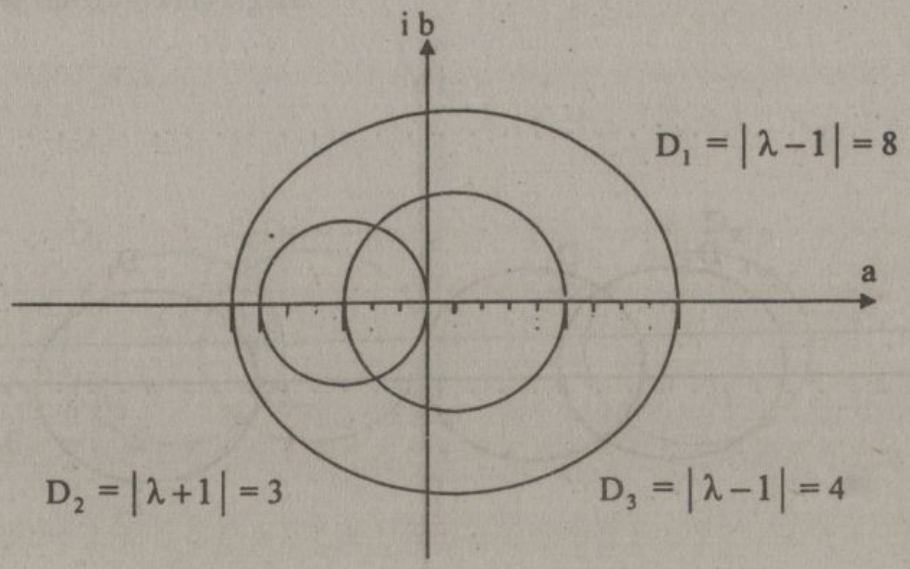
$$D_3 = |\lambda - 1| \le |4| = 4$$

$$\lambda = 1 + 8 = 9$$

$$\lambda = 3 - 3 = 0$$

$$\lambda = 4 + 1 = 5$$

Figure



Centre

$\lambda_1 = +1$; $\lambda_2 = -3$; $\lambda_3 = 1$

We can deduce that $0 \le \lambda \le 9$

Eigenvalues of the transpose of a matrix A are the same as those of A.

PROBLEMS

1. (a) (i) What are eigenvalues and eigenvectors of a matrix?

(ii) What is the characteristic polynomial of a matrix?

(iii) What is the spectrum of a matrix?

(iv) How can we determine if a matrix is singular by looking at its eigenvalues?

(v) What is an eigenvalue's multiplicity?

(vi) How can we compute the eigenvalues of a triangular matrix?

(vii) What are the eigenvalues and eigenvectors of the inverse of a matrix?

(viii) What can we say about the eigenvalues of a symmetric matrix?

(ix) What does Gerschgorin's theorem tell us about the eigenvlaues of a matrix?

(b) Name the various methods you have studied for computing eigenvalues and eigenvectors.

(c) Suppose that we have given two matrices, A and B. Their set of eigenvalues are:

i) A : 5, 8, –7

ii) B : 0.2, 1, –1

Identify the dominant eigenvlaue of each.

2. Compute the eigenvalues and eigenvectors of each of the following matrices using the general method:

(a) $\begin{bmatrix} -1 & 1 & 1 \\ -6 & 1 & 3 \\ -12 & -2 & 8 \end{bmatrix}$

(b) $\begin{bmatrix} 3 & 2 & 2 \\ 2 & 2 & 0 \\ 2 & 0 & 4 \end{bmatrix}$

(c) $\begin{bmatrix} 3 & -1 & 0 \\ -1 & 2 & -1 \\ 0 & -1 & 3 \end{bmatrix}$

(d) $\begin{bmatrix} 1 & 2 & -8 \\ -2 & 2 & -2 \\ 1 & -4 & 10 \end{bmatrix}$

(e) $\begin{bmatrix} -2 & 2 & 2 \\ 3 & -1 & 3 \\ 1 & 1 & -3 \end{bmatrix}$

(f) $\begin{bmatrix} 0 & 2 & -10 \\ -3 & 1 & -3 \\ 1 & -5 & 11 \end{bmatrix}$

(g) $\begin{bmatrix} -2 & 6 & -24 \\ 0 & -3 & 10 \\ 1 & -4 & 13 \end{bmatrix}$

(h) $\begin{bmatrix} 10 & -2 & 4 \\ -20 & 4 & -10 \\ -30 & 6 & -13 \end{bmatrix}$

(i) $\begin{bmatrix} 1 & 2 & 1 \\ 6 & -1 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

(j) $\begin{bmatrix} 5 & 8 & 16 \\ 4 & 1 & 8 \\ -4 & -4 & -11 \end{bmatrix}$

(k) $\begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 4 & 2 & 3 \end{bmatrix}$

(l) $\begin{bmatrix} -17 & 18 & -6 \\ -18 & 19 & -6 \\ -9 & 9 & -2 \end{bmatrix}$

(m) $\begin{bmatrix} 0 & 0 & -1 \\ 10 & 1 & 2 \\ 0 & 0 & 1 \end{bmatrix}$

(n) $\begin{bmatrix} 2 & 3 & 4 \\ 0 & 3 & 2 \\ 0 & 0 & -2 \end{bmatrix}$

(o) $\begin{bmatrix} -2 & 2 & 2 & 2 \\ -3 & 3 & 2 & 2 \\ -2 & 0 & 4 & 2 \\ -1 & 0 & 0 & 5 \end{bmatrix}$

(p) $\begin{bmatrix} 2 & 1 & -1 \\ 0 & -2 & -2 \\ 1 & 1 & 0 \end{bmatrix}$

(q) $\begin{bmatrix} 9 & -1 & 2 \\ 2 & 8 & 4 \\ 1 & 1 & 8 \end{bmatrix}$

(r) $\begin{bmatrix} -2 & 6 & -24 \\ 0 & -3 & 10 \\ 1 & -4 & 13 \end{bmatrix}$

3. (a) Explain Leverrier-Faddeev method to find the eigenvalue and eigenvector. Mention one disadvantage of this method under which it fails.

(b) Given the following matrices, compute their eigenvalues, eigenvectors and inverses:

(i) $\begin{bmatrix} 1 & 3 & 2 \\ -2 & 1 & 1 \\ 1 & -2 & -1 \end{bmatrix}$

(ii) $\begin{bmatrix} 12 & 6 & -6 \\ 6 & 12 & 2 \\ -6 & 2 & 16 \end{bmatrix}$

(iii) $\begin{bmatrix} 2 & 2 & 1 \\ -4 & 8 & 1 \\ -1 & -2 & 0 \end{bmatrix}$

(iv) $\begin{bmatrix} 1 & 2 & -8 \\ -2 & 2 & -2 \\ 1 & -4 & 10 \end{bmatrix}$

(v) $\begin{bmatrix} 0 & 2 & -10 \\ -3 & 1 & -3 \\ 1 & -5 & 11 \end{bmatrix}$

(vi) $\begin{bmatrix} -2 & 2 & 2 \\ 3 & -1 & 3 \\ 1 & 1 & -3 \end{bmatrix}$

4. (a) Describe power method to compute the largest eigenvalue and its corresponding eigenvector.

(b) Compute the dominant eigenvalues and corresponding eigenvectors of each of the following matrices (Use your initial vectors if not given):

(i) $\begin{bmatrix} 7 & 6 & -3 \\ -12 & -20 & 24 \\ -6 & -12 & 16 \end{bmatrix}$

(ii) $\begin{bmatrix} 4 & 1 & 0 \\ 1 & 20 & 1 \\ 0 & 1 & 4 \end{bmatrix}$

(iii) $\begin{bmatrix} 1.5 & 0 & 1 \\ -0.5 & 0.5 & -0.5 \\ -0.5 & 0 & 0 \end{bmatrix}$

Use $x^{(1)} = [1 \;\; 1 \;\; 1]^T$

(iv) $\begin{bmatrix} 1 & 0 & 2 \\ 0 & 1 & 3 \\ 1 & 1 & 1 \end{bmatrix}$

(v) $\begin{bmatrix} -2 & 1 & 0 \\ 1 & -2 & 1 \\ 0 & 1 & -2 \end{bmatrix}$

(vi) $\begin{bmatrix} 10 & -2 & 4 \\ -20 & 4 & -10 \\ -30 & 6 & -13 \end{bmatrix}$

(vii) $\begin{bmatrix} 4 & 2 & 1 \\ -4 & 10 & 1 \\ -1 & -2 & 2 \end{bmatrix}$

Use $x^{(1)} = [1 \;\; 1 \;\; 1]^T$

(viii) $\begin{bmatrix} 6 & 4 & -2 \\ 4 & 12 & -4 \\ -2 & -4 & 13 \end{bmatrix}$

(ix) $\begin{bmatrix} -2 & 2 & -1 \\ 7 & 3 & -1 \\ 4 & -4 & -2 \end{bmatrix}$

(x) $\begin{bmatrix} 5 & -2 & -4 \\ -2 & 2 & 2 \\ -4 & 2 & 5 \end{bmatrix}$

(xi) $\begin{bmatrix} 3 & 2 & 6 \\ -1 & 12 & 1 \\ 4 & 2 & 1 \end{bmatrix}$

(xii) $\begin{bmatrix} 1 & -3 & 2 \\ 4 & 4 & -1 \\ 6 & 3 & 5 \end{bmatrix}$

Use $x^{(1)} = [1 \;\; 1 \;\; 1]^T$

(xiii) $\begin{bmatrix} -1 & 0 & 0 \\ 2 & 1 & -2 \\ 0 & 0 & 5 \end{bmatrix}$

Use $x^{(1)} = [0 \;\; 0 \;\; 1]^T$

5. (a) What is the purpose of deflation?

(b) The following matrix:

$$\begin{bmatrix} 0 & 2 & -8 \\ -2 & 1 & -2 \\ 1 & -4 & 9 \end{bmatrix}$$

has $\lambda_1 = -1$ as an eigenvalue and its corresponding eigenvector $x^{(1)} = [1, \;\; 1.5, \;\; .5]^T$. Use the deflation method to compute the rest of the eigenvalues and eigenvectors.

(c) Given the following matrix:

$$\begin{bmatrix} 5 & 2 & 4 \\ -3 & 6 & 2 \\ 3 & -3 & 1 \end{bmatrix}$$

(i) Use the power method to calculate the largest eigenvalue of the matrix.

(ii) Use the deflation method to calculate the second largest eigenvalue of this matrix.

(d) Given the matrix:

$$\begin{bmatrix} 25 & 1 & 2 \\ 1 & 3 & 0 \\ 2 & 0 & -4 \end{bmatrix}$$

(i) Starting from the initial guess:

$$x^{(1)} = [1 \quad 0 \quad 0]^T$$

Find the largest eigenvalue and its corresponding eigenvector using power method.

(ii) Repeat the method with

$$x^{(1)} = [1 \quad 1 \quad 1]^T$$

(iii) Deflate the matrix A.

(e) Given the matrix:

$$\begin{bmatrix} 3 & 2 & -2 \\ -1 & 1 & 4 \\ 3 & 2 & -5 \end{bmatrix}$$

(i) Starting with the initial guess:

$$x^{(1)} = [1 \quad 1 \quad 1]^T,$$

find the largest eigenvalue and its corresponding eigenvector using power method.

(ii) Use the method of deflation to compute other eigenvalues.

(f) Consider the matrix:

$$\begin{bmatrix} 1 & 2 & 1 \\ -4 & 7 & 1 \\ -1 & -2 & -1 \end{bmatrix}$$

(i) Use the power method to calculate the largest eigenvalue. Take the initial vector, $x^{(1)} = [1 \quad 1 \quad 1]^T$.

(ii) Use the method of deflation to find the remaining eigenvalues:

6. Find the spectral radii of the following matrices:

(i) $\begin{bmatrix} 0 & -1 & -1 \\ 1 & -2 & 0 \\ 1 & 0 & -2 \end{bmatrix}$

(ii) $\begin{bmatrix} 7 & -2 & 1 \\ -2 & 10 & -2 \\ 1 & -2 & 7 \end{bmatrix}$

(iii) $\begin{bmatrix} 2 & 0 & -1 \\ 0 & 2 & 0 \\ -1 & 0 & 1 \end{bmatrix}$

(iv) $\begin{bmatrix} \frac{1}{3} & -\frac{1}{3} & \frac{1}{3} \\ -\frac{1}{3} & \frac{1}{6} & -\frac{1}{6} \\ \frac{1}{3} & -\frac{1}{6} & \frac{1}{6} \end{bmatrix}$

(v) $A = \begin{bmatrix} 1 & 2 & 1 \\ 6 & -1 & 0 \\ -1 & -2 & -1 \end{bmatrix}$

(vi) $A = \begin{bmatrix} 3 & 2 & 4 \\ 2 & 0 & 2 \\ 1 & 2 & 3 \end{bmatrix}$

7. Given the following matrix:

$$A = \begin{bmatrix} -2 & 6 & -24 \\ 0 & -3 & 10 \\ 1 & -4 & 13 \end{bmatrix}$$

Using Gerchgorin's theorem, find the range of eigenvalues.

8. Find the eigenvalues and spectral radii of the following triangular matrices:

(i) $\begin{bmatrix} 4 & -7 & 0 & 2 \\ 0 & 3 & 4 & 6 \\ 0 & 0 & 3 & -8 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

(ii) $\begin{bmatrix} 5 & -2 & 6 & -1 \\ 0 & 3 & -8 & 0 \\ 0 & 0 & 5 & 4 \\ 0 & 0 & 0 & 1 \end{bmatrix}$

(iii) $\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 0 & 2 & 3 & 4 & 5 \\ 0 & 0 & 1 & 2 & 3 \\ 0 & 0 & 0 & 2 & 3 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}$

Bibliography

1. Atkinson, Kendall, E., (1978), **An Introduction to Numerical Analysis**, John-Wiley and Sons, New York.
2. Atkinson, L.V., and Harvey, P.J., (1983), **An Introduction to Numerical Methods with Pascal**, Addison-Wesley, London.
3. Bajpai, A.C., Calius, I.M., and Fairley, J.A., (1975), **Numerical Methods for Engineers and Scientists**, Taylor and Francis, London.
4. Bhatti, S.A., (1977), **Fundamentals of Fortran Programming**, Star Book Depot, Urdu Bazar, Lahore.
5. Bhatti, S.A., (1988), **Learning Basic**, Star Book Depot, Urdu Bazar, Lahore.
6. Butler, R. and Kerr, E., (1967), **An Introduction to Numerical Methods**, Pitman, London.
7. Carnahan, B., Luther, H.A. and Wilkes, James, O., (1969), **Applied Numerical Methods**, John-Wiley, New York.
8. Chapra, Steven, C., and Canale, Raymond P., (2006), **Numerical Methods for Engineers**, 5th edition, McGraw-Hill, New York.
9. Churchhouse, R.F., (1987), **A First Course in Numerical Analysis**, University College Press, Cardiff.
10. Conte, S.D., and De Boor, C., (1972), **Elementary Numerical Analysis: An Algorithmic Approach**, McGraw-Hill, Kogakusha Ltd., Tokyo.
11. Faires, Burden, (1993), **Numerical Analysis**, PWS Publishing Co., Boston, USA.
12. Fröberg, Carl-Erik, (1974), Introduction to Numerical Analysis, Addison-Wesley, Reading, USA.
13. Gerald, Curtis, E., and Wheatley, Patrick, O., (1994), **Applied Numerical Analysis**, McGraw-Hill, New York.
14. Groves, W.E., (1966), **Brief Numerical Methods**, Prentice-Hall, New York.
15. Hildebrand, F.B., (1956). **Introduction to Numerical Analysis**, McGraw-Hill, New York.
16. Jacquez, John, A., (1970), **A First Course in Computing and Numerical Methods**, Addison-Wesley, Reading, USA.
17. Jacobs, D., (1977), **The State of the Art in Numerical Analysis**, Academic Press, New York.

18. James, M.L., Smith, G.M. and Wolford, J.C., (1993), **Applied Numerical Methods for Digital Computation**, Harper Coltins College publishers, New York.

19. Khabaza, I.M., (1965), **Numerical Analysis**, Pergamon Press, London.

20. Kellison, Stephen G., (1975), **Fundamentals of Numerical Analysis**, Richard D. Irwin, London.

21. Kue, F.F., (1965), **Numerical Methods and Computers**, Addison Wesley, Reading, USA.

22. Macon, N., (1963), **Numerical Analysis**, John-Wiley, New York.

23. Mathews, John H., (1992), **Numerical Methods for Mathematics, Science, and Engineering**, Prentice-Hall, London.

24. Ralston, A., (1965), **A First Course in Numerical Analysis**, McGraw-Hill, New York.

25. Ralston, A., and Rabinowitz, L., (1978), **A First Course in Numerical Analysis**, McGraw-Hill, New York.

26. Rojiani, K., B., (1997), **Programming in C with Numerical Methods for Engineers**, Prentice-Hall, New Jersey, USA.

27. Scraton, R.E., (1984), **Basic Numerical Methods**, Edward Arnold, London

28. Scraton, R.E., (1987), **Further Numerical Methods in Basic**, Edward Arnold, London.

29. Shoup, Terry E., (1984), **Applied Numerical Methods for the Micro-computer**, Prentice-Hall, New Jersey.

30. Smith W. Allen, (1979), **Elementary Numerical Analysis**, Harper and Row, New York.

31. Todd, John (1977), **Basic Numerical Mathematics, Vol. 1: Numerical Analysis**, Academic Press, New York.

32. Todd, John (1977), **Basic Numerical Mathematics, Vol. 2: Numerical Algebra**, Academic Press, New York.

33. Watson, W.A., Phillipson, T., and Dates, P.J., (1981), **Numerical Analysis**, Edward Arnold, London.

34. Wilkes, M.V., (1975), **Short Introduction to Numerical Analysis**, Canbridge University Press, London.

35. Wolf, M.A., (1972), **A First Course in Numerical Analysis**, Van Nostrand Reinhold, London.

Index

A

Absolute error, 6, 7
Acceleration of convergence, 230
Accumulated error, 4
Accuracy, 7
Adams-Bashforth method, 189
Adams-Moulton method, 194
Adjoint matrix, 270
Aitken's delta process, 230
Aitken's method for interpolation, 73
Algebraic equations, 217
Analytical methods, 167
Analytical solution, 120, 167
Automatic subdivision of intervals, 149

B

Backward substitution, 272
Backward difference operator, 40
Bairstow's method, 218
Bessel's interpolation formula, 64
Birga-Vieta, 256
Binary search, 242
Bisection method, 218, 242
Boundary conditions, 166
Boundary value problem, 166
Boole's rule, 132

C

Central difference formulas, 52
- Bessel's formula, 52, 64
- Everett's formula, 52, 65
- Gauss forward formula, 52, 65
- Gauss backward formula, 52, 65
- Stirling's formula, 52, 62

Central difference operator, 41
Characteristic equation, 324

Characteristic polynomial, 324
Choleski's method, 290
Chopping, 7
Complete pivoting, 280, 282
Continuing methods, 168
Convergence of the iterative scheme, 221, 225
Cramer's rule, 268, 269
Cube root, 237

D

Deflation, 257, 344
Derivatives, 93
Determinant, 269, 332
Determination of square root, 327
Differentiation, 93
Differences, 27
Difference operations, 35
Difference operators, 35
Difference table, 27
Differential equations, 165
Differential operators, 94
Direct methods, 268
Discretization error, 168
Divergent behaviour, 221
Dominant egenvalue, 325

E

Egenvalues, 324
Egenvectors, 324
Elimination methods, 272
Errors, 2
Error accumulation in addition, 9
Error accumulation in subtraction, 9
Error accumulation in multiplication, 10
Error accumulation in division, 12
Errors of powers and roots, 14
Error in function evaluation, 16
Error Analysis, 3
Error correction & detection, 30
Error estimation in interpolation, 80
Euler's method, 175
– improved, 175
– modified, 175

Everett's interpolation formula, 65
Exact polynomial, 29
Extrapolation, 51

F

False position, 249
Finite, 28
Finite calculus, 28
Finite differences, 27
Forward difference operator, 35
Forward stage, 273

G

Gaussian elimination, 268, 272
Gaussian interpolation formulas, 65
Gaussian quadrature formulas, 126
Gauss-Seidel method, 268. 298, 305
Gerschgorin's theorem, 351
Global error, 136, 175
Graphical methods, 167
Gregary-Newton forward formula, 52
Gregary-Newton backward formula, 57
Gross error, 3

H

Halley's formula, 266
Heun's method, 175
Hermite formula, 52
Higher-order derivatives, 95
Higher-order ODEs, 166
Horner's scheme, 258
Hotelling's deflation, 345, 348

I

Indirect methods, 268
Initial value problem, 166
Instability, 180
Integration, 125
Interpolation, 51
Inverse, 333
Inverse of a matrix, 285
Iteration, 218

Iterate, 219
Iterative methods, 75, 298
Iterative interpolation method, 73

J

Jacobi's iterative method, 268, 298, 299
Jury problem, 166

L

Lagrange formula, 69
Laplace, 126
Latent vector, 324
Leverrier-Faddev method, 324, 331
Linear equations, 166, 217
Local truncation error, 4, 138, 175
Localization of roots, 222
LU decomposition, 284

M

MacLaurin's series, 20
Matrix deflation, 257
Marching problem, 166
Mean operator, 43
Mechanical quadrature, 125
Milne-Simpson predictor-corrector, 169, 186
Multiple roots, 254
Multi-step methods, 168, 184

N

Nested polynomial, 171
Neville's formula, 75
Newton's backward difference formula, 56
Newton-Cotes formulas, 125
Newton's divided difference formula, 52
Newton's forward difference formula, 52
Newton-Raphson method, 218, 233
Non-linear equations, 166, 217
Numerical analysis, 1
Numerical cancellation, 17
Numerical differentiation, 93
Numerical integration, 125
erical methods, 1, 167

Numerical recipes, 2

O

One-step methods, 184
Order of convergence, 234
Order of equations, 166
Ordinary differential equations, 165

P

Partial differential equations, 165
Partial pivoting, 280, 281
Percentage error, 6, 7
Picard's method, 169
Pivot, 273
Pivotal strategy, 280
Polynomial, 51
Polynomial evaluation, 256
Power method, 324, 335
Precision, 6, 7
Predictor-Corrector methods, 169, 184
- Milne-Simpson method, 186
- Adams-Bashforth method, 189
- Adams-Moulton method, 194

Probable error, 8
Propagation error, 168
Positive definite matrix, 290

Q

Quadrature, 125

R

Reciprocal of a number, 239
Regula falsi method, 218
Relationships between operators, 43
Relative error, 6, 7
Repeated use of trapezoidal rule, 149
Remainder term, 5
Romberg integration, 152
Root, 226
Rounding errors, 3, 4, 168
Runge-Kutta methods, 177

S

Secant method, 218, 246
Shift operator, 42
Significant digits, 6
Simpson's 1/3rd-rule, 129
Simpson's 3/8th-rule, 131
Simple iterative procedure, 218
Single-step method, 168, 184
Simultaneous differential equations, 203
Spectral radius, 333
Spectrum, 325
Spider web convergence, 221
Sources of errors, 3
Stagnation, 250
Starting methods, 166
Staircase convergence, 221
Stirling's interpolation formula, 62
Subtractive cancellation, 17
Successive approximation method, 170
Synthetic division, 218

T

Taylor series method, 5, 19, 172
Trace, 332
Transcendental functions, 217
Trapezoidal rule, 127
Triangular decomposition, 268, 284, 290
Tridiagonal matrix, 268, 293
Truncation error, 3, 5

U

Unstable process, 93, 180
Upper triangular matrix, 275

W

Web convergence, 221
Weddle's rule, 132

Z

Zeros of polynomials, 256

Answers

Note: We have provided the answers of almost all problems. The reader in many cases may expect to obtain results which differ slightly from the answers given here, depending on the accuracy required and programming techniques or facilities employed.

Chapter 1

1.(a) AE = 0.1; RE = 0.000356

(b) AE = 7.7696×10^{-3}; RE = 7.4431×10^{-3}

(c) AE = 0.0478×10^{-3}; RE = 0.009×10^{-3}

(d) AE = 7.029×10^{-6}; RE = 0.016×10^{-3}

(e) AE = 0.070006; RE = 0.03238

(f) AE = 0.23×10^{-3}

(g) Do yourself.

2. (a) 0.1 (b) 0.028 (c) 2.0475

3. $9 \le x \le 11$; $5.5 \le y \le 6.5$; $2.5 \le x - y \le 5.5$; $14.5 \le x + y \le 17.5$

4. (a) $9.43 < uv < 13.23$; $0.465 < u/v < 0.655$

(b) $4.464 < z < 4.507$; Meaningful answer is 4.5 cm.

(c) RE = 0.48%

5. (a) AE = 0.17675; RE = 0.125

(b) 10; 1

(c) RE = 4.1%

(d) – 8.804 to – 8.801

6. (i) 1.5453×10^{-5}; 1.625×10^{-5}; (ii) 0.0049; 0.0275; (iii) 1.178; 0.5×10^{-3}; (iv) 3.63×10^{-4}; 1.135×10^{-3}

7. (a) –1.184 to –1.123; (b) –3.946 to –3.908; Meaningful answer is –3.9 correct to 1 dp; (c) Meaningful answer is 17.0; (d) 0.0141 to 0.0142; Meaningful answer for a is 0.014 correct to 3 dp

8. (a) i) 0.0582 ii) 0.0182 iii) 0.0100 iv) 0.00100 v) – 111.0999

(b) i) $\ln\left(1+\frac{1}{x}\right)$ ii) $\cos 2x$ iii) $\frac{1}{\sqrt{x^2+1}+x}$

iv) $1-\cos x = \frac{\sin^2 x}{1+\cos x}$

v) $\tan x - \sin x = \tan x\,(1-\cos x)$

$= \tan(2\sin^2(x/2)) = 2\tan x\,\mathrm{Sin}^2(x/2)$

9. (a) $f = 1 - \frac{x^2}{2} + \frac{x^4}{24}$; $f(1.5) = 0.086$

(b) $f(x) = x - \frac{x^2}{2}$; $f(1.2) = 0.18$; 0.0027

(c) i) Number of term = 7 ii) 1.359×10^{-3}

(d) i) $x + \frac{x^2}{2\times 2!} + \frac{x^3}{3\times 3!} + \frac{x^4}{4\times 4!} + \ldots + \frac{x^n}{n\times n!}$

ii) $R(x) = \frac{x^{n+1}}{(n+1)\,(n+1)!}$ iii) 7 terms

10. (a) $1 - \frac{1}{2}x + \frac{3}{8}x^2 - \frac{5}{16}x^3$; 0.9534; 2.73×10^{-5}

(b) i) $\left|\frac{1}{2n+1}x^{2n+1}\right| < \frac{1}{2}\times 10^{-6} = 5\times 10^{-7}$

when $x = 1$, $n > 500000$ and when $x = \frac{1}{2}$, $n = 15$

11.
```
double SolExp(double x)
{
  double expo, term;
  int i=1;
  expo = term + 1;
  while (fabs(term) > = le – 6)
  {
    term=term*x/i;
    expo=expo+term
    i++;
  }
```

```
    return expo;
}
```

12. (a)
```
double solPi=1
unsigned i, n=100;
float sign = 1;
for (i=1; i<n; i++)
{
  sign = – sign;
  solPi +=sign/2*i+1);
}
solPi*=4;
```

(b)
```
double solPi = 0;
unsigned i, n=100;
for (i=1; i<n; i++)
    solPi +=1.0/(4*i–3)/ (4*i–1);
solPi*=8;
```

(c) Do yourself

Chapter 2

1. (d) No, since the function is not a polynomial.

(f) It is zero and the third-order difference column is constant. Hence, the function represents an exact polynomial of degree three.

2. (a) 5th entry is in error. Correct value, f(3) = – 96.

(b) 6th entry is in error. Correct value, f(6) = 112.

(c) 4th entry is in error. Correct value, f(4) = 1.4108.

3. (a) 6th entry is in error. Correct value, f(6) = 72.

(b) 6th entry is in error. Correct value, z(6) = 0.598.

4. y(1.3) = 1.65 and z(1.3) = 7.2

5. (a) f = –1, 6, 9, 7 and 5.

(b) f = 0, 5, 7, 6 and 5.

(c) f = 1, 1, 13, 73 and 241.

6. (a) Error = – 0.18, f(2) = 2.35 – (– 0.18) = 2.53

(b) f(3.63) = 0.144518.

(c) $f(0.5) = 11.10$.

7. (a) $a = 1, b = 5$ and $c = 3$.

(b) $\delta^4 f = 0$.

(c) 0.0003

(d) (i) $\Delta f_0 = 16$, $\nabla f_{-1} = 0$, $\delta f_{\frac{3}{2}} = 66$, $\delta^2 f_1 = 50$, $\Delta^3 f_0 = 60$, $\nabla^3 f_2 = 36$, and $\delta^4 f_0 = 24$.

(e) $110 = \Delta f_0 = \nabla f_1 = \delta f_{\frac{1}{2}}$

$320 = \Delta f_1 = \nabla f_2 = \delta f_{\frac{3}{2}}$

$192 = \Delta^2 f_0 = \nabla^2 f_0 = \delta^2 f_1$

8. (a) 650 (b) 2.4101 (c) 20 (d) (i) $f(0.4) = -6.774$; (ii) $f(0.4) = 0.671$

(e) 0.678; $f(0.7) = 5.292$

10. (a) (i) $2n + 1, 2, 0$ (ii) $3n^2 + 9n + 4; 6n + 12; 6$ (iii) $3n^2 + n + 17; 6n + 4; 6$

(iv) $5n(n-1)(n-2)(n-3); 20n(n-1)(n-2); 60n(n-1)$.

(d) (i) $6(x-1)$ (ii) $6(x+1)/(x+2)^3$

(f) $12x^2 - 24x + 14$

(g) $2x(x+1)$

Chapter 3

1. (a) $f(1.23) = 1.37$ (b) $f(1.75) = 5.9$ (c) 0.3049 (d) 4.69 (e) 0.28

2. (i) $f(1.45) = 4.14$ (ii) $f(1.05) = 2.04$ (iii) 0.6101, 0.5984

3. (a) $f(2.5) = 25.4$

(b) i) $f(2.3) = 1.5166$ ii) $f(2.05) = 1.4318$ iii) $f(2.65) = 1.6282$

4. (i) 1.1775 (ii) 5.4803

5. (a) (i) 27.3548 (ii) 27.6578 (iii) 27.6396 (iv) 27.3555, 27.3540

(b) Same answer from all formulas; 0.1048

(c) 1.782 (d) 2.199 (e) 0.1495 (f) 1.186 (g) 0.129

6. (a) 10.5 (b) 8.4 (c) 14 (d) 49.46 (e) 89
(f) 3.625 (g) 0.5104, 0.2252 (h) 10.86

(i) $f(x) = \frac{1}{2}[x^3 - 7x^2 - 14x]$

$= f(1.5) = 4.3125$

(l) 0.9375

(m) $y = x^3 - 3x^2 + 5x - 5$

$y(4.5) = 46.875$

7. (a) .25 (b) 1.25 (c) 1.9402 (d) 0.031

8. (a) 1.2221 (b) $E = 1.12 \times 10^{-3}$; (c) $h = 0.0866$

9. (a) 0.85 (b) 0.385 (c) 14 (d) 1.2892 (e) 0.14104

10. (a) 0.8642; 5.713×10^{-6} (b) 0.28

11. (a) 6561 (b) 0.446198

12. (a) 2.1992 (b) -5.67×10^{-6}

13. (a) 0.149586 (b) –0.00001 (c) 0.07

Chapter 4

1. (b) $f_0' = 9.87$; $f_0'' = 10.90$; $f'(2.4) = 11.05$; $f''(2.4) = 11.37$

2. (a) 0.4997; analytical answer is 0.5; Error = 0.00029 (b) – 95.0

3. $y'(4.75) = 1.331$; $y''(4.75) = 0.326$

4. (a) 0.046; 1.408 and 6.70 (b) 345 and 112

5. (a) – 0.3647 and 0.3900

(b) $f'(0.8) = 0.79814$; $f''(0.8) = 0.75720$

$f'(0.85) = 0.83748$; $f''(0.85) = 0.81692$

6. (a) 0.045; 0.8095 and 6.7 (b) 0.46192, 0.39136 and 0.1425

(c) 34.5405, 34.6155, 0.075

7. (a) –1.0625; –1.375; 4.5

(b) 0.7494; 0.8053 and –1.5185; Error = –0.0078, –0.131 and 0.9055

(c) 1.796027

8. (i) -2.37×10^9 (ii) 2.63×10^9

(iii) -1.044×10^9; -0.94×10^9; -0.902×10^9; -1.024×10^9; -1.821×10^9.

9. (a) 4.0560, 4.0535 (b) 1.08, 4.04, 0.04

10. (a) 4.425 (b) 0.16159, 0.361, 2.435

11. (a) (i) 0.506, 0.0521 (ii) 2.309, 1.6105

(b) (i) 0.685, 0.827 (ii) 0.685, 0.824 (iii) 0.686, 0.827

(iv) 0.685, 0.827, 0.685, 0.827

Chapter 5

1. (a) $I_T = 52.46$ and $I_S = 52.12$; (b) $I_T = 6.3627$, $I_S = 6.3789$

(c) (i) $I_T = 7635$, $I_S = 7730$ (ii) $I_T = 89250$, $I_S = 89500$

2. (a) 16.678 and 16.699. The true answer is 16.778, which is nearer to Simpson's $\frac{1}{3}$rd rule

(b) $I_S = 0.43521$; $\frac{1}{\sqrt{2}} \tan^{-1}\left(\frac{1}{\sqrt{2}}\right) = 0.43521$. Both results are very close.

3. 0.694 and 0.693; exact answer = 0.693

$$-0.0002604 \geq E_T \geq -0.000326$$

$$-0.00000203 \geq E_S \geq -0.0000651$$

4. (i) (a) 10.56902 (b) 0.62792, 0.62797 (ii) 0.681 (iii) 0.9943, 1.0000, 0.006

(iv) 46.5, 46.49 (v) $I_T = 1.11006$, $I_S = 1.11058$, $I = 1.11073$

5. (a) $I_S = 0.7854$; $I_T = 0.7848$; Analytical answer = 0.7854

(b) Number of subdivisions in Trapezoidal rule are 1443 and in Simpson's rule are 26.

(c) Infinity in Trapezoidal rule and 118 in Simpson's rule.

(d) (i) $I_S = 0.2639$, $I_T = 0.2652$; (ii) $E_S = 0.0000326$ and $E_T = -002642$

(iii) 13 and 4

(e) 59

6. (a) 2.136 (b) 5.49 (c) 4.64 (d) 48.77, 56.724, 56.528

(e) $I_S = 2.32$ (f) $I_T = 0.5797$, Area = 3.643 and $I_S = 0.5672$, Area = 3.564

7. (a) $I_T = 0.2974$, $I_S = 1.9736$; $I_S + I_T = 2.277$

(b) $I_T = 4.5$, $I_S = 53.8$; $I_S + I_T = 53.8$

8. (a) 1.462657 (b) 0.785396 (c) 0.657669

9. (a) 0.8285 (b) 0.4570 (c) 0.6012 (d) 0.3861 (e) 0.109364

10. (a) 0.6012 (b) 0.4570 (c) 504.6933 (d) 0.657669 (e) –0.0207

11. (a) 0.8813; Exact = 0.881374 (b) 0.999886; Exact = 1.000000

(c) 39.225; 36.375

12. (c) 202

13. (a) (i) 0.6839397 (ii) 0.7313703 (iii) 0.7429841

(c) (i) 0.785398 (ii) $\frac{\pi}{4}$

Chapter 6

2. (a) $x + \frac{x^2}{2}$, 2.02 (b) 2.0202, 0.0002 (c) 002 (d) 0.1414

3. (b) Taylor series method cannot be used. Picard's method with three approximations:

$$y = 1 + 6x^{.5} + 4x^{1.5} + \frac{8}{5}x^{2.5}$$

$$y(0.1) = 3.0289;\ y(1.2) = 15.3547$$

(c) Taylor series method cannot be used because y' is infinite at $x = 0$. With three approximations Picard's method gives,

$$y = 1 + 2x^{\frac{1}{2}} + 2x + \frac{4}{3}x^{\frac{3}{2}}$$

$$y(1.5) = 8.899;\ y(1.2) = 15.3547$$

4. (a) $y = 1 - x + x^2 - \frac{1}{3}x^3$, where $h = x$.

(b) $y = 0.9097$ (c) $E = 0.00000833$; $h \le 0.05$ (d) 0.9097

5. (a) (i) $y = 1 + x(1 + x(1 + x(\frac{2}{3} + x(\frac{1}{6} + x(\frac{1}{40} + \frac{1}{360}x)))))$

(ii)

x	0	.1	.2	...	1.0
y	1.0	1.1107	1.2456	...	3.8611

(iii) $Y(1) = 3.8751$; Error $= 0.012$

(b) (i) $y = 1 + x + x^2 + \frac{2}{3}x^3 + \frac{1}{6}x^4 + \frac{1}{30}x^5 + \frac{1}{180}x^6$

(ii) $0 \le x \le 0.27$

x	0	.1	.2	...	1.0
y	1	1.1107	1.2456	...	3.873

(iii) Error $= |Y(1) - y(1)| = 0.0398$

(c) $y = 1 + x + \frac{1}{6}x^3 + \frac{1}{12}x^4 + \frac{1}{180}x^6$

x	y	$y'' = x\,y$
−0.2	0.79880	−0.15976
0	1.00000	0
0.2	1.20147	0.24029
0.4	1.41283	0.56513
0.6	1.64712	0.98827

(d) Exact solution, $y = 2e^x - 2x - 1$

7.

(a)

x	y	Exact
0	1	1
0.05	.9513	.9513
0.1	.9052	.9052
0.15	.8619	.8618
0.2	.8214	.8213

(c)

x	y
1.0	2.000
1.1	2.781
1.2	4.297

(d)

x	y
2.0	3.000
2.1	4.919
2.2	10.680

(e)

x	y
0	1.00000
0.2	0.00267
0.4	0.02136

8. $y = 1 + x^2\left(\frac{1}{2} + x\left(\frac{1}{3} + x\left(\frac{1}{8} + x\left(\frac{1}{15} + \frac{x}{48}\right)\right)\right)\right)$

x	y	f
0	1.000	0.0000
.1	1.0053	0.1105
.2	1.0229	0.2446
.3	1.0552	0.4066

Corrector, $y(0.4) = 1.1053$

9. Predictor, $y_4 = 5.9614$

Corrector, $y_4 = 6.9267$

Further values using corrector do not settle down; they go on increasing.

10.

	x	y	$f = 1 + y^2$
	0	0	1.0000
	0.2	0.2027	1.0411
	0.4	0.4228	1.1788
	0.6	0.6842	1.4681
Predictor	0.8	1.0239	2.0484
Corrector	0.8	1.0302	3.0615
Predictor	1.0	1.5394	3.3697
Corrector	1.0	1.5609	

11. $y(0.3) = 1.01499$

12. (a)

t	0	0.1	0.2	0.3
x	0.0000	0.097	0.2199	0.3447
y	1.0000	1.0266	0.9987	0.9955

(b) $y' = z;\ z' = x z + y^2;$

Initial conditions: $y(0) = 1;\ z(0) = 2$ with $h = 0.2$

$y(0.2) = 1.4289;\ z(0.2) = 2.3394$

13. (a) $y(.1) = 1.1142,\ h = .0029$

(b)

x	y
.2	2.0933
.4	2.1755
.6	2.2493

(e) 4.2748 (f) -0.70347

14. (e) $y(0.8) = 8.00$

15.

t	y
0.5	13.457
1.0	21.8278
⋮	⋮
10.0	47.8597
⋮	⋮
20.0	49.7349

16. (a) $y(.5) = -0.28326$, Error $= -0.00077$

(b) $y(.6) = 0.02919$

17. (a)

x	y
.4	–.8109652
.5	–.8195905

(b)

x	0	.2	.4	.6	.8	.1035
y	0	.0004	.0064	.0325	1.0	.2574

(c) 0.35181

18.

t_n	x_n	y_n
0.00	– 2.7000000	2.8000000
0.05	– 2.5521092	2.6742493
0.10	– 2.4078422	2.5570240
0.15	– 2.2662276	2.4484383
0.20	– 2.1261657	2.3487177

19. (a)

x	y	z
0.5	1.31959	– 0.39347
1.0	1.10364	0.36788

(b) $\frac{dy}{dx} = z = f_1(x, y, z)$

$$\frac{dz}{dx} = -2xz + 3y + x^2 + 2$$

$$= f_2(x, y, z)$$

with $y(x_0) = 1$

$z(x_0) = 2$

$h = 0.1$

(c) $\frac{dy}{dx} = z = f_1(x, y, z)$

$$\frac{dz}{dx} = -yz + 2x - y$$

$$= f_2(x, y, z)$$

with $y_0 = y(1) = 1$

$z_0 = z(1) = 1$

Chapter 7

1. (a) 1.3 (b) –1.6 and 0.3 (c) 0.6 and 1.5

2. (a) (i) and (ii) Do not satisfy (iii) 2.0945

(b) (i) and (ii) Do not satisfy (iii) and (iv) Converge to root 3.0983
(iv) is faster

(c) (i) and (iii) Converge to root 2.0983; (i) is faster

3. (a) (i) and (iii); (iii) is faster

(b) (ii) The root is 4.

(c) Convergence occurs in (i), (v), and (vi); (vi) is the faster with root 0.4534 after two iterations.

(d) (i) $f(x) = \ln(1 + 32)$

$x_5 = 1.90547$

(ii) $x_{13} = 1.2612$

$x_{13} = 1.2612$

(iii) Much faster; $x_3 = 1.2612$

(iv) $x_6 = -5.381$

$x_7 = -74.422$

No convergence

4. (a) – 1.7693 (b) 2.1038 (c) – 0.7781 (d) 0.4950 (e) 0.7391
(f) 0.5671 (g) 0.91 (h) 1.9346 (i) 4.2748 (k) 2.91
(l) – 6.44 (n) 2.17456

5. (a) 3.16 (b) 0.4472 (c) 1.618 (d) 2.080084

6. (a) 5.016; 5.0575 (b) 0.2592; 0.2591

7. (a) 1.8960 (b) 2.9429 (c) 0.6766 (d) 0.50 (e) 1.11416

8. (a) 2.9428 (b) 0.4950 (c) 1.7100 (d) –1.5214 (e) 1.2351

9. (a) 1.1347 (b) 2.9428 (c) 0.1419 (d) 0.6953 (e) 2.64575

10. (i) 0.50197 (ii) 0.65162 (iii) 0.65044 (iv) 0.6875 (v) 0.65161

11. (a) $p(2) = 65$; $p'(2) = 137$ (b) $p(-2) = 67$
(c) $p(-2) = 55$; $p'(-2) = -115$; $p''(-2) = 160$

12. (a) $p(3) = 17$, $p'(3) = 25$, $x_1 = 2$

(b) 3.1048, –1.0399, 1.4689 + 0.1062 i and 1.4689 – 0.1062 i

13. (a) 3, 3, –1 and – 2 (b) 1.0333, 1.0002, 1.0001 (c) 1.37

(d) (i) $x_1 = 1.5714,\ x_2 = 1.5009,\ x_3 = 1.5$

(ii) $x_3 = 1.5467,\ x_4 = 1.5492,\ x_5 = 1.5248$

14. (a) $f(x_{n+1}) = \dfrac{x_n\left(x_n^2 + 3a\right)}{3x_n^2 + a}$; $f(x_{n+1}) = \dfrac{15x_n + x_n^3}{5 + 3x_n^2}$;

$x_1 = 2.2353,\ x_2 = 2.2361,\ x_3 = 2.2361$

(b) $f(x_{n+1}) = \dfrac{2 + 2x_n + 2x_n^2 + x_n^3}{3 + 4x_n + 2x_n^2}$; $x_1 = -2.0130,\ x_2 = 2.0000,\ x_3 = -2.000$

Chapter 8

1. (a) $x_1 = -1,\ x_2 = -3,\ x_3 = 2$

(b) $x_1 = 0.6574,\ x_2 = 0.264,\ x_3 = 0.636$

(c) $x_1 = 1.353,\ x_2 = 2.412,\ x_3 = 3.706$

(d) $x_1 = 4,\ x_2 = 1,\ x_3 = 2$

(e) $a = -14.9,\ b = -29.5,\ c = 19.8$

(f) $x = -0.25,\ y = 2$, and $z = 0.75$

2. (a) $A^{-1} = \begin{bmatrix} 0.5 & 0.25 & 0.75 \\ 1 & 0 & -1 \\ -0.5 & 0.25 & 1.25 \end{bmatrix}$

(b) $x_1 = -2,\ x_2 = 0,\ x_3 = 5$

3. (a) $A^{-1} = \begin{bmatrix} 1.5 & 0 & -0.5 \\ 0.5 & -0.25 & 0.25 \\ 0 & -0.25 & 0.75 \end{bmatrix}$

$x_1 = -2,\ y = 3.25,\ z = 8.25$

(b) (i) $\begin{bmatrix} -.20 & -.76 & -2.36 & 1.08 \\ .40 & .72 & 2.92 & -.76 \\ -.20 & -.16 & -.76 & .28 \\ .80 & -0.16 & 4.84 & -1.52 \end{bmatrix}$

(ii) $\begin{bmatrix} 1 & 0 & -2 & 0 \\ -5 & 1 & 11 & -1 \\ 287 & 67 & -630 & 65 \\ -416 & 97 & 931 & -94 \end{bmatrix}$

(iii) $\begin{bmatrix} 25 & -41 & 16 & -6 \\ -16 & 27 & -11 & 4 \\ 16 & -27 & 13 & -5 \\ -6 & 10 & -5 & 2 \end{bmatrix}$

(iv) $\begin{bmatrix} -1 & -2.5 & 0.25 & 0.75 \\ -1.667 & -1.25 & 0.917 & 1.417 \\ 2.333 & 1.5 & -0.833 & -1.83 \\ 2.667 & -1.417 & -1.417 & -1.917 \end{bmatrix}$

4. (a) (i) $x_1 = \frac{22}{9}$, $x_2 = 3$, $x_3 = \frac{11}{9}$

(ii) $x_1 = 7.57$, $x_2 = -7.25$, $x_3 = -1.12$

(b) $x_1 = -.5$, $x_2 = 1.5$, $x_3 = 1$, $x_4 = -2$

(c) $x_1 = 1.5$, $x_2 = .5$, $x_3 = -1.25$, $x_4 = .25$

(d) $x_1 = 0.2$, $x_2 = 0.5$, $x_3 = -0.2$, $x_4 = 0.4$

(e) $x_1 = 3$, $x_2 = -1$, $x_3 = 0$, $x_4 = 2$

(f) $x_1 = 3$, $x_2 = -1$, $x_3 = 4$, $x_4 = 2$

(g) $x_1 = 1$, $x_2 = 1$, $x_3 = 1$, $x_4 = 1$

(h) $x_1 = 3$, $x_2 = 2$, $x_3 = 1$, $x_4 = 5$

(i) $x_1 = 5$, $x_2 = 4$, $x_3 = -7$, $x_4 = 1$

(j) $x_1 = 0$, $x_2 = 1$, $x_3 = -1$, $x_4 = 2$, $x_5 = -2$

(k) $x_1 = -7.233,\ x_2 = 1.133,\ x_3 = 2.433,\ x_4 = 4.5$

(l) $x_1 = 3,\ x_2 = 1,\ x_3 = -2,\ x_4 = 1$

5. (a) $A^{-1} = \begin{bmatrix} 1 & 1 & -4 \\ -3 & -2 & 11 \\ 2 & 1 & -6 \end{bmatrix}$

(b) $B^{-1} = \begin{bmatrix} -186 & 129 & 196 \\ 95 & -66 & -100 \\ -24 & 17 & 25 \end{bmatrix}$

(c) $x_1 = 3,\ x_2 = -4,\ x_3 = 2$

(d) $x_1 = 1,\ x_2 = 2,\ x_3 = 3$

(e) $x_1 = 3,\ x_2 = 2,\ x_3 = 1$

(f) $x_1 = 4.75,\ x_2 = -2.25,\ x_3 = 0.75$

6. (a) $A^{-1} = \begin{bmatrix} 0.609 & 0.219 & -0.344 \\ 0.219 & -0.563 & 0.313 \\ -0.344 & 0.313 & -0.063 \end{bmatrix}$

$x_1 = 2.34,\ x_2 = 0.69,\ x_3 = -0.94$

(b) $A^{-1} = \begin{bmatrix} 0.38 & -0.19 & -0.11 \\ -0.19 & 0.32 & 0.12 \\ -0.11 & 0.12 & -0.19 \end{bmatrix}$

$x_1 = 0.91,\ x_2 = 1.78,\ x_3 = 1.55$

7. (a) $A^{-1} = \begin{bmatrix} 68 & -41 & -17 & 10 \\ -41 & 25 & 10 & -6 \\ -17 & 10 & 5 & -3 \\ 10 & -6 & -3 & 2 \end{bmatrix}$

$a = b = c = d = 1.$

(b) $\begin{bmatrix} 13.5 & -6 & 2 & -1.5 \\ -6 & 3 & -2 & 1 \\ 2 & -2 & 10 & -3 \\ -1.5 & 1 & -3 & 1 \end{bmatrix}$

8. (a) $x_1 = x_2 = x_3 = x_4 = 1$

(b) $x_1 = 4,\ x_2 = 7,\ x_3 = 8,\ x_4 = 6$

(c) $x_1 = 3,\ x_2 = 5,\ x_3 = 6,\ x_4 = 6,\ x_5 = 5,\ x_6 = 3$

(d) 8.7058, 7.8230, 7.5864, 7.5224, 7.4913, 7.4618, 7.3558, 6.9616, 5.4904

9. (a) $x = 2.0,\ y = 1.0,\ z = 3.0$

(b) $x = 3.0,\ y = 2,\ z = 10$

(c) $x_1 = 2.013,\ x_2 = 0.957,\ x_3 = 1.039$

(d) No. Since the matrices are not diagonally dominant, interchanging the rows will not produce a diagonally dominant matrix.

(f) $x_1 = 1,\ x_2 = 2,\ x_3 = -1,\ x_4 = 1$

(g) $x_1 = 0.075,\ x_2 = 2.9625,\ x_3 = -1.1875,\ x_4 = -3.975$

10.(a) $x_1 = 0.8544,\ x_2 = -0.6001,\ x_3 = 1.0491$

(b) $x_1 = 1.322,\ x_2 = 0.139,\ x_3 = -3.484,\ x_4 = 4.079$

(c) $x_1 = 0.433,\ x_2 = 0.911,\ x_3 = 0.460,\ x_4 = -0.058,\ x_5 = -0.115,\ x_6 = 0.244$

(e) $x_1 = 2.048,\ x_2 = 0.921,\ x_3 = 1.118,\ x_4 = 0.931$

11.(a) $x = -0.011,\ y = 0.035,\ z = -0.014$

(b) $x = 0.27,\ y = 0.24,\ z = -0.55$

(c) $x = 0.660,\ y = 0.441,\ z = 1.093$

(d) 0.8890, −0.8126, 2.1419, 2.6497

12.(a) In 10 iterations, Jacobi's method gives the answer as: $x_1 = -2$, $x_2 = 1$ and $x_3 = -3$

(b) (i) $x_1 = .8166,\ x_2 = 1.9408,\ x_3 = 1.7956$

(ii) $x_1 = 2.0882,\ x_2 = 4.7534,\ x_3 = 8.2896$

(iii) $x_1 = 0.999995 \simeq 1,\ x_2 = 1.999995 \simeq 2,\ x_3 = 3.999995 \simeq 3$

(iv) $x_1 = 2.999871 \simeq 3,\ x_2 = 0.000065 \simeq 2,\ x_3 = 1.000048 \simeq 1$

13. $x_1 = 0.433,\ x_2 = 0.911,\ x_3 = 0.460,\ x_4 = -0.058,\ x_5 = -0.115,\ x_6 = 0.244$

Chapter 9

1. (a) and (b) Both work.

   (c) i) 8, ii) no, dominant eigenvalue.

2. (a) $\lambda_1 = 4;\quad x^{(1)} = [1\ \ 1\ \ 2]^T$

   $\lambda_2 = 2;\quad x^{(2)} = [2\ \ 3\ \ 5]^T$

   $\lambda_3 = 1;\quad x^{(3)} = [1\ \ 2\ \ 4]^T$

   (b) $\lambda_2 = 0;\ \lambda_2 = 3;\ \lambda_3 = 6$

   $x^{(1)} = [2\ \ -2\ \ 1]^T$

   $x^{(2)} = [1\ \ 2\ \ -2]^T$

   $x^{(3)} = [2\ \ 1\ \ 2]^T$

   (c) $\lambda_1 = 1;\ \lambda_2 = 3;\ \lambda_3 = 4$

   (d) $\lambda_1 = 0;\quad x^{(1)} = [2\ \ 3\ \ 1]^T$

   $\lambda_2 = 4;\quad x^{(2)} = [-2\ \ 1\ \ 1]^T$

   $\lambda_3 = 9;\quad x^{(3)} = [1\ \ 0\ \ -1]^T$

   (e) $\lambda_1 = 2;\quad x^{(1)} = [1\ \ 1.5\ \ 0.5]^T$

   $\lambda_2 = -4;\quad x^{(2)} = [2\ \ -3\ \ 1]^T$

   $\lambda_3 = -4;\quad x^{(3)} = [2\ \ -3\ \ 1]^T$

   (f) $\lambda_1 = 2;\quad x^{(1)} = [2\ \ 3\ \ 1]^T$

   $\lambda_2 = 4;\quad x^{(2)} = [-2\ \ 1\ \ 1]^T$

   $\lambda_3 = 10;\quad x^{(3)} = [1\ \ 0\ \ -1]^T$

(g) $\lambda_1 = -1;$ $x^{(1)} = [6 \quad 5 \quad 1]^T$

$\lambda_2 = 2;$ $x^{(2)} = [-3 \quad 2 \quad 1]^T$

$\lambda_3 = 7;$ $x^{(3)} = [-2 \quad 1 \quad 1]^T$

(h) $\lambda_1 = 0;$ $x^{(1)} = [1 \quad 5 \quad 0]^T$

$\lambda_2 = -1;$ $x^{(2)} = [0 \quad 2 \quad 1]^T$

$\lambda_3 = 2;$ $x^{(3)} = [1 \quad 0 \quad -2]^T$

(i) $\lambda_1 = 0;$ $x^{(1)} = [1 \quad 6 \quad 13]^T$

$\lambda_2 = -4;$ $x^{(2)} = [-1 \quad 2 \quad 1]^T$

$\lambda_3 = 3;$ $x^{(3)} = [2 \quad 3 \quad -2]^T$

(j) $\lambda_1 = 1;\ \lambda_2 = -3;\ \lambda_3 = -3$

(k) $\lambda_1 = 8;$ $x^{(1)} = [2 \quad 1 \quad 2]^T$

$\lambda_2 = -1;$ $x^{(2)} = [0 \quad 2 \quad -1]^T$

$\lambda_3 = -1;$ $x^{(3)} = [1 \quad 0 \quad -1]^T$

(l) $\lambda_1 = -2;\ \lambda_2 = 1;\ \lambda_3 = 1$

(m) $\lambda_1 = 0;\ \lambda_2 = \lambda_3 = 1$

(n) $\lambda_1 = 2;\ \lambda_2 = 3;\ \lambda_3 = -1$

(o) $\lambda_1 = 1;$ $x^{(1)} = [4 \quad 3 \quad 2 \quad 1]^T$

$\lambda_2 = 2;$ $x^{(2)} = [3 \quad 3 \quad 2 \quad 1]^T$

$\lambda_3 = 3;$ $x^{(3)} = [2 \quad 2 \quad 2 \quad 1]^T$

$\lambda_4 = 4;$ $x^{(4)} = [1 \quad 1 \quad 1 \quad 1]^T$

(p) $\lambda_1 = 0;$ $x^{(1)} = [1 \quad -1 \quad 1]^T$

$\lambda_2 = 1;$ $x^{(2)} = [5 \quad -2 \quad 3]^T$

$\lambda_3 = -1;$ $x^{(3)} = [1 \quad -2 \quad 1]^T$

(q) $\lambda_1 = 5;\ \lambda_2 = 10;\ \lambda_3 = 10$

3. (iv) $\lambda_1 = 0;\quad x^{(1)} = [2\ \ 3\ \ 1]^T$

$\lambda_2 = 4;\quad x^{(2)} = [-2\ \ 1\ \ 1]^T$

$\lambda_3 = 9;\quad x^{(3)} = [1\ \ 0\ \ -1]^T$

(v) $\lambda_1 = -2;\quad x^{(1)} = [2\ \ 3\ \ 1]^T$

$\lambda_2 = 4;\quad x^{(2)} = [-2\ \ 1\ \ 1]^T$

$\lambda_3 = 10;\quad x^{(3)} = [1\ \ 0\ \ -1]^T$

(vi) $\lambda_1 = 2;\quad x^{(1)} = [1\ \ 1.5\ \ .5]^T$

$\lambda_2 = -4;\quad x^{(2)} = [2\ \ -3\ \ 1]^T$

$\lambda_3 = -4;\quad x^{(3)} = [2\ \ -3\ \ 1]^T$

(vii) $\lambda_1 = -1;\quad x^{(1)} = [6\ \ 5\ \ 1]^T$

$\lambda_2 = 2;\quad x^{(2)} = [-3\ \ 2\ \ 1]^T$

$\lambda_3 = 7;\quad x^{(3)} = [-2\ \ 1\ \ 1]^T$

4. (a) Book work

(b) (i) $\lambda = 4;\quad x^{(1)} = [-3\ \ 4\ \ 2]^T$

(ii) $\lambda = 20.124;\quad x^{(1)} = [0.062\ \ 1.000\ \ 0.002]^T$

(iii) $\lambda = 1;\quad x^{(1)} = [1\ \ -0.5\ \ -0.5]^T$

(iv) $\lambda = 3.236;\quad x^{(1)} = [0.667\ \ 1\ \ 0.745]^T$

(v) $\lambda = -3.414;\quad x^{(1)} = [-0.707\ \ 1\ \ -0.707]^T$

(vi) $\lambda = 2;\quad x^{(1)} = [1\ \ 0\ \ -2]^T$

(vii) $\lambda = 8;$

(ix) $\lambda = 18;\quad x^{(1)} = [1\ \ 2\ \ -2]^T$

(x) $\lambda = 6;\quad x^{(1)} = [1\ \ 3\ \ -2]^T$

(xi) $\lambda = 10;$ $x^{(1)} = [2 \quad -1 \quad -2]^T$

(xiii) $\lambda = 7;$ $x^{(1)} = [9 \quad 2 \quad 30]^T$

(xiv) $\lambda = 5;$ $x^{(1)} = [0 \quad 0 \quad 1]^T$

5. (a) Bok work.

(b) $\lambda_2 = 3$ and $\lambda_3 = 8$

(c) (i) 7 (ii) 3

(e) 6; $[.7 \quad -0.3 \quad 1]^T$

(f) (i) 5 (ii) 2, 0

6. (i) $\delta(A) = 2$ (ii) $\delta(A) = 12$ (iii) $\delta(A) = 2$

(iv) $\delta(A) = 0.80$ (v) $\delta(A) = 4$ (iii) $\delta(A) = .8$

7. $|\lambda + 2| \leq 30$ is the last estimate.

8. (i) $\lambda_1 = 5;\ \lambda_2 = 3;\ \lambda_3 = 2;\ \delta(A) = 5$

(ii) $\lambda_1 = 4;\ \lambda_2 = 3;\ \lambda_3 = 1;\ \delta(A) = 4$

(iii) $\lambda_1 = 1;\ \lambda_2 = 2$