NATIONAL OPEN UNIVERSITY OF NIGERIA

SCHOOL OF SCIENCE AND TECHNOLOGY

COURSE CODE: PHY 314

COURSE TITLE: NUMERICAL COMPUTATIONS

PHY 314: Numerical Computations

Course Code	PHY 314
Course Title	Numerical Computations
Course Developer	Dr. A. B. Adeloye DEPARTMENT OF PHYSICS UNIVERSITY UNIVERITY OF LAGOS
Programme Leader	Dr. Ajibola S. O. National Open University of Nigeria Lagos

COURSE GUIDE

NATIONAL OPEN UNIVERSITY OF NIGERIA

Contents

Introduction

Numerical Analysis is an important part of Physics and Engineering. This is because most of the problems encountered in real life do not lend themselves to a solution in a closed form. In other words, we have to make do with approximate solutions. It is clear, therefore, that you need to be conversant with the various methods of approximate solution of problems, as well as the loss of information inherent in replacing the exact solution with an approximate one.

It is also quite clear that the fastest way of doing numerical computation is through the computer. It is imperative, then, that you understand one or more of the available programming languages. In this course, the programming language of interest is C++.

It is quite clear from the foregoing that numerical analysis is an interesting course, and we would expect you to apply yourself fully to the course, as a lot of your future work in the field of physics would warrant a sound knowledge of numerical analysis.

THE COURSE

PHY 309 (3 Credit Units)

This 2-unit course introduces you to numerical analysis. Unit 1 discusses the various types of errors and how they might be minimised.

Unit 2 is on curve-fitting. You would need to deduce some physical parameters from a given set of readings obtained perhaps in a laboratory. Various ways of linearising given formulas is given, preparatory to drawing a line of best fit from which the physical quantity is deduced.

Unit 3 is all about linear systems of simultaneous equations. You shall learn how to handle a large set of linear equations by writing them in the form of matrices. Such problems will then be solved with the methods applicable to matrices. You would also learn how to arrive at solutions through iterative methods.

Unit 4 discusses different methods of finding the roots of algebraic and transcendental equations.

In Unit 5, you will come across finite differences. You will be introduced to various kinds of differences, and how to detect the error in difference tables.

Numerical integration is the object of Unit 6. In this Unit, you shall learn how to integrate a function within a given set of limits (definite integrals).

Unit 7, the concluding part of the theory part of the course discusses the numerical solution of initial value problems of ordinary differential equations.

The C++ Programming aspect of the course is an introduction to program-writing in one of the most versatile programming languages.

We wish you success.

COURSE AIMS

The aim of this course is to teach you about the mechanics of the atomic and subatomic particles.

COURSE OBJECTIVES

After studying this course, you should be able to

- Understand the various types of errors and how to minimise them.
- Linearise a given expression in order to bring out a physical constant from the resultant relationship.
- Fit a curve to a given set of data.
- Solve a system of linear equations.
- Find the roots of a given algebraic or transcendental equation.
- Obtain the definite integral of a given function of a single variable.
- Work with finite difference schemes.
- Solve a first order initial value problems of ordinary differential equation.
- Solve higher order initial value problems of ordinary differential equations.
- Write C++ programs for solving the numerical problems.

WORKING THROUGH THE COURSE

Numerical methods provide a powerful way of solving almost any problem in physics, provided it has been properly formulated. It is our belief that the student would be motivated enough to put in a good effort in understanding the theoretical part of this course and be willing to learn to write programs in C++ language.

THE COURSE MATERIAL

You will be provided with the following materials:

Course Guide
Study Material containing study units

At the end of the course, you will find a list of recommended textbooks which are necessary as supplements to the course material. However, note that it is not compulsory for you to acquire or indeed read them.

STUDY UNITS for Numerical Analysis

The following study units are contained in this course:

Unit 1: Approximations and Errors in Numerical Computations
Unit 2: Approximations and Errors in Numerical Computations
Unit 3: Linear Systems of Equations
Unit 4: Roots of Algebraic and Transcendental Equations
Unit 5: Finite Differences and Interpolation
Unit 6: Numerical Integration
Unit 7: Initial Value Problems of Ordinary Differential Equations

TEXTBOOKS

Some reference books, which you may find useful, are given below:

1. Numerical Methods in Engineering and Science – Grewal, B. S.
2. Introductory Methods of Numerical Analysis – Sastry, S. S.
3. A friendly Introduction to Numerical Analysis – Bradie, B.

Assessment

There are two components of assessment for this course. The Tutor Marked Assignment (TMA), and the end of course examination.

Tutor Marked Assignment

The TMA is the continuous assessment component of your course. It accounts for 30% of the total score. You will be given 4 TMA's to answer. Three of these must be answered before you are allowed to sit for the end of course examination. The TMA's would be given to you by your facilitator and returned after they have been graded.

End of Course Examination

This examination concludes the assessment for the course. It constitutes 70% of the whole course. You will be informed of the time for the examination. It may or may not coincide with the university semester examination.

Summary

This course is designed to lay a foundation for you for further studies in Numerical Analysis. At the end of this course, you will be able to answer the following types of questions:

- What is the need for numerical analysis in Physics?
- What are the types of error that can be encountered in numerical work?

- What the ways of obtaining the line that best fits a set of laboratory data?
- What are the various ways of numerically solving a system of linear equations?
- What are the ways in which we can numerically find the roots of an equation?
- How do I integrate a function that does not lend itself to an analytical solution?
- How do I solve a first order ordinary differential equation?
- How do I tackle a higher order initial value problem of ordinary differential equation?
- What are the merits and demerits of some of the methods of numerical analysis?

We wish you success.

UNIT 1: Approximations and Errors in Numerical Computations

1.0 Introduction

Physics is an exact science. However, it is strictly impossible to achieve infinite accuracy in practice. You are quite aware that your apparatus or instrument is not perfect, neither is your eye nor your measuring ability. We then see that errors arise in everyday observations and measurements. The study of errors is very important in all areas of Science and Technology. This is necessitated by the fact that errors should not swamp our procedure enough to alter, significantly, conclusions that may be drawn from such observations or measurements.

Apart from the limitations of observation and measurement, there are some errors inherent in the problem itself. A good example in Quantum Mechanics is given by the Heisenberg Uncertainty Principle, which maintains that we cannot measure some pairs of quantities accurately simultaneously, for example, the position of a body and its momentum. Any attempt to measure either quantity accurately gives an infinite error in the other. Some other errors arise as a result of representing an infinite series with a truncated one. We shall talk a little bit more about this in a while.

2.0 Objectives

At the end of this unit, you would be able to:

- understand the importance of errors in numerical analysis.
- round a number to a certain number of significant figures
- know how to reduce the errors involved in your numerical work.
- understand arithmetic precision

3.0 Main Content

3.1 Accuracy of Numbers

3.1.1 Approximate Numbers

For the sake of numerical computation, all numbers can be classified under two broad headings: exact numbers and approximate numbers. As the name implies, the former comprises numbers that are fully represented by some digits. Examples include the integers, and rational numbers that can and have been completely written, e.g., 3.2158. Approximate numbers are those that are not fully specified by the digits representing them. As an example, we could write the rational number $\frac{3}{7}$ as 2.3333. You are quite aware that the actual number is not exactly 2.3333.

By this stage of your study, you must have worked with the rational numbers. These are numbers which can be written as a fraction of two integers. Although certain rational numbers are exact numbers, you have also come across a lot of rational numbers that cannot be written as exact numbers as in the example above. The irrational numbers are even more troublesome. An example of an irrational number is $\sqrt{2}$: such numbers cannot be written as the ratio of any two integers. There are two families of numbers that are unending: the ones that repeat certain sequences, and the ones that do not. For instance, 12.345454545 and 18.127849342. The order of preference in dealing with numbers in numerical computations is: natural numbers, rational numbers that have a finite string of digits, rational numbers that have unending strings of digits and irrational numbers.

3.1.2 Significant digits (figures)

We say a number is of *r* significant digits (figures) if *r* digits are used to express it. As an example, 1.612, 0.004812 and 3806000 all have four significant figures. You would notice that each of them could be written $x \times 10^n$ (with no loss of information), where x is of 4 (*r* = 4 in this case) digits, not starting or ending with zero and n is an integer, positive or negative.

The following rules will be of assistance to you. Make sure they become a part of you.

1. The leftmost non-zero digit is the most significant digit, e.g., in 0.001243, 1 is the most significant digit.
2. In the case where there is no decimal point, the rightmost non-zero digit is the least significant, e.g., 145630000, 3 is the least significant figure.
3. If there is a decimal point, the rightmost digit is the least significant, even if it is zero, e.g., in 235.34200, the last 0 is the least significant. The number is not 235.34201 or 235.34199.
4. All digits between the least significant and the most significant (inclusive) are significant, e.g., in the example under rule 1, 1→3 are significant. In the example in rule 2, 1 → 3 are significant.

Take another example: 0.00004 has one significant figure, while 984.13245 has 8 significant figures. It should be obvious to you why they have been classified this way.

There is an exception, however:
When a zero is obtained by rounding, for example, 329.5 is rounded to 3 significant figures. This becomes 330, the last zero being significant in this case. You can compare this with rule 2 above.

3.2.3 Rounding off

The irrational numbers are a perfect example of numbers with unending digits. Even in the case of rational numbers there can still be unending number of digits and in some other cases we may decide to reduce the number of digits by which a number is represented. This process is called **rounding off**.

Rules for Rounding off a number to n significant figures

(a) Discard all digits to the right of the *n*th digit
(b) If the discarded part of the number is
 (I) less than half a unit in the nth place leave the nth digit unchanged
 (II) greater than half a unit in the nth place, increase the nth digit by unity
 (III) exactly equal to half a unit in the nth place, leave the digit unchanged if it is even; increase by unity if otherwise.

Examples: Round the following numbers to 5 significant figures:
(i) 3.142857143 (ii) 6.32431925 (iii) 1.4123519

Solution: To 5 significant figures, the numbers are:
(i) 3.1428 (rules (a) and (b)(III) *n*th digit unchanged as it is even
(ii) 6.3243 (rules (a) and (b)(I) as the discarded part of the number is less than half a unit in the *n*th place.
(iii) 1.4124 (rules (a) and (b)(III) *n*th digit increased by unity as it is odd

Note:

A number rounded off to *n* significant figures is said to be correct to *n* significant places.

3.1.4 Arithmetic precision

As we have said before, it might be necessary to round off our numbers to make them useful for numerical computation, moreso as it would require an infinite computer memory to store an unending number. The precision of a number is an indication of the number of digits that have been used to express it. In scientific computing, it is the number of significant digits or numbers, while in financial systems, it is the number of decimal places. You are quite aware that most currencies in the world are quoted to two decimal places.

In our own case, a**rithmetic precision** (often referred to simply as precision) is the specified number of significant figures or digits to which the number of interest is to be rounded.

3.4 Errors

We said earlier, that we shall be revisiting the different types of errors. These are:

3.4.1 Rounding Errors

These are errors incurred by truncating a sequence of digits representing a number, as we saw in the case of representing the rational number $\frac{7}{3}$ by 2.3333, instead of 2.3333….., which is an unending number. Apart from being unable to write this number in an exact form by hand, our instruments of calculation, be it the calculator or the computer, can only handle a finite string of digits.

Rounding errors can be reduced if we change the calculation procedure in such a way as to avoid the subtraction of nearly equal numbers or division by a small number. It can also be reduced by retaining at least one more significant figure at each step than the one given in the data, and then rounding off at the last step.

3.4.2 Inherent Errors

As the name implies, these are errors that are inherent in the statement of the problem itself. This could be due to the limitations of the means of calculation, for instance, the calculator or the computer. This error could be reduced by using a higher precision of calculation.

3.4.3 Truncation Errors

If we truncate Taylor's series, which should be an infinite series, then some error is incurred. This is the error associated with truncating a sequence or by terminating an iterative process.

This kind of error also results when, for instance, we carry out numerical differentiation or integration, because we are replacing an infinitesimal process with a finite one. In either case, we would have required that the elemental value of the independent variable tend to zero in order to get the exact value.

3.3.4 Absolute Error, Relative Error and Percentage Error

The absolute error in a measurement is the absolute difference between the measured value and the actual value of the quantity. Thus, we can write

$$\text{Absolute error} = |\,\text{actual value} - \text{measured value}\,|$$

The ratio of the absolute error to the actual value is the relative error. We can therefore write the relative error as

$$\text{Absolute error} = \frac{|\,\text{actual value} - \text{measured value}\,|}{\text{actual value}}$$

The relative error taken to a percentage is the percentage error. Percentage error can therefore be written as

$$\text{Percentage error} = \frac{|\,\text{actual value} - \text{measured value}\,|}{\text{actual value}} \times 100$$

Examples

4.0 Conclusion

In this Unit you learnt that errors occur in measurement, because the imperfect observer makes use of imperfect measuring instruments. Some errors are inevitable as they are a part of the problem under investigation. Moreover, the instruments of calculation, such as the computer, can only handle a finite number of digits, as the memory is finite. You also learnt to write a certain number in a specified number of decimal points. You got to know how to round a number to a number of significant figures. Some ways of reducing some of these errors were also discussed.

5.0 Summary

In this Unit, you learnt the following:

- Errors are an integral part of life.
- How to round a number to a specific number of significant figures.
- The different types of error and how some of them may be reduced.

6.0 Tutor-Marked Assignment

1. Round the following to the number of significant figures indicated.

(a)	12.0234831	4 significant figures
(b)	295.10542	5 significant figures
(c)	0.0045829	3 significant figures

2. A student measured the length of a string of actual length 72.5 cm as 72.4 cm. Calculate the absolute error and the percentage error.

7.0 References/Further Readings

Solutions to Tutor Marked Assignment

1. Round the following to the number of significant figures indicated.

(a)	12.0234831	4 significant figures	= 12.02
(b)	295.10542	6 significant figures	= 295.105
(c)	0.0045829	3 significant figures	= 0.00458

2. A student measured the length of a string of actual length 72.5 cm as 72.4 cm. Calculate the absolute error and the percentage error.

Absolute error is $|72.5 - 72.4| = 0.01$.

The percentage error is $\frac{0.1}{72.5} \times 100 = 0.1379$

UNIT 2: Approximations and Errors in Numerical Computations

1.0 Introduction

In most experiments as a physicist, you would be expected to plot some graphs. This chapter explains in details, how you can interpret the equation governing a particular phenomenon, plot the appropriate graph with the data obtained, to illustrate the inherent physical features, and deduce the values of some physical quantities. The process of fitting a curve to a set of data is called curve-fitting. We shall now take a look at the possible cases that could arise in curve-fitting.

2.0 Objectives

At the end of this unit, you should be able to:

- Linearise a given equation in order to plot a linear graph from which some physical constants can be determined.
- Derive the equation for least squares linear fit.
- Derive the equation for the method of moving averages.
- Fit a linear graph to a set of data.

3.0 Main Content

3.1 Linear Graph

The law governing the physical phenomenon under investigation could be linear, of the form $y = mx + c$. It follows that a graph could be plotted of the points (x_i, y_i), i = 1, …, n, where n is the number of observations (or sets of data). We could obtain the line of best fit via any of a number of methods:

More on this.

3.2 Linearisation

A nonlinear relationship can be linearised and the resulting graph analysed to bring out the relationship between variables. We shall consider a few examples:

Case 1: $y = ae^x$.

(i) We could take the logarithm of both sides to base e:

$$\ln y = \ln(ae^x) = \ln a + \ln e^x = x + \ln a,$$

since $\ln e^x = x$. Thus, a plot of $\ln y$ against *x* gives a linear graph with slope unity and a *y*-intercept of ln *a*.

(ii) We could also have plotted *y* against e^x. The result is a linear graph through the origin, with slope equal to *a*.

Case 2: $T = 2\pi\sqrt{\dfrac{l}{g}}$

We can write this expression in three different ways:

(i) $\ln T = \ln(2\pi) + \dfrac{1}{2}\ln\left(\dfrac{l}{g}\right) = \ln(2\pi) + \dfrac{1}{2}(\ln l - \ln g)$.

Rearranging, we obtain,

$$\ln T = \frac{1}{2}\ln l + \left(\ln(2\pi) - \frac{1}{2}\ln g\right)$$

writing this in the form $y = mx + c$, we see that a plot of $\ln T$ against $\ln l$ gives a slope of 0.5 and a ln *T* intercept of $\left(\ln(2\pi) - \dfrac{1}{2}\ln g\right)$. Once the intercept is read of the graph, you can then calculate the value of *g*.

(ii) $T = \dfrac{2\pi}{\sqrt{g}}\sqrt{l}$

A plot of T versus $\sqrt{l}$ gives a linear graph through the origin (as the intercept is zero). The slope of the graph is $\dfrac{2\pi}{\sqrt{g}}$, from which the value of *g* can be recovered.

(iii) Squaring both sides,

$$T^2 = \frac{4\pi^2}{g} l$$

A plot of T^2 versus l gives a linear graph through the origin. The slope of the graph is $\dfrac{4\pi^2}{g}$, and the value of *g* can be obtained appropriately.

Case 3: $N = N_0 e^{-\lambda t}$

The student can show that a plot of $\ln N$ versus t will give a linear graph with slope $-\lambda$, and $\ln N$ intercept is $\ln N_0$.

What other functions of N and *t* could you plot in order to get λ and N_0?

Case 4: $\frac{1}{f} = \frac{1}{u} + \frac{1}{v}$

We rearrange the equation:

$$\frac{1}{v} = \frac{1}{f} - \frac{1}{u}$$

A plot of v^{-1} (y-axis) versus u^{-1} (x-axis) gives a slope of -1 and a vertical intercept of $\frac{1}{f}$.

Example

A student obtained the following reading with a mirror in the laboratory.

u	10	20	30	40	50
v	-7	-10	-14	-15	-17

Linearise the relationship $\frac{1}{v} = \frac{1}{f} - \frac{1}{u}$. Plot the graph of v^{-1} versus u^{-1} and draw the line of best fit. Hence, find the focal length of the mirror. All distances are in cm.

Solution

u	**v**	**1/u**	**1/v**
10	-7	0.1	-0.14286
20	-10	0.05	-0.1
30	-14	0.033333	-0.07143
40	-15	0.025	-0.06667
50	-17	0.02	-0.05882

The graph is plotted in Fig. 1.1.

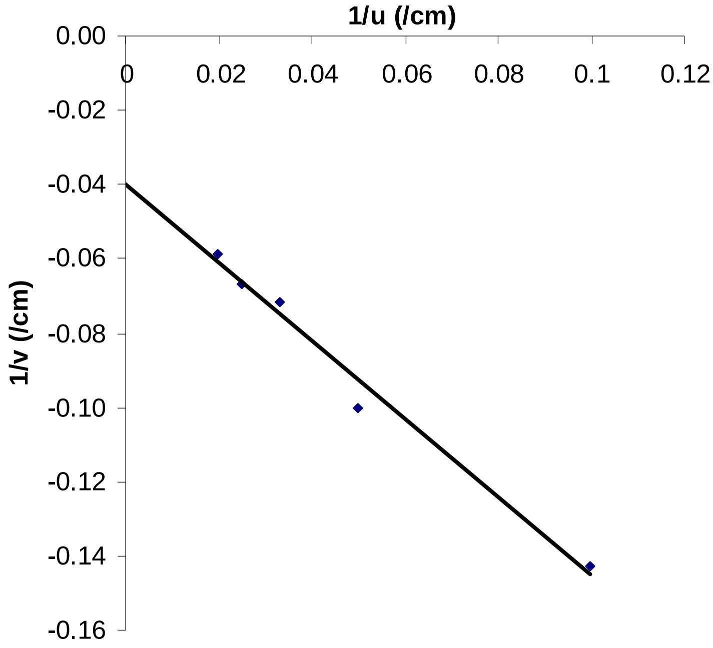


Fig. 1.1: Linear graph of the function $\frac{1}{v} = \frac{1}{f} - \frac{1}{u}$

The slope is -1.05 and the intercept -0.04. From $\frac{1}{v} = \frac{1}{f} - \frac{1}{u}$, we see that the intercept is $\frac{1}{f} = -0.04$, or $f = -\frac{1}{0.04} = -25$ cm.

3.3 Curve Fitting

What we did in Section 3.2, generally, was to plot the values of dependent variable against the corresponding values of the independent variable. With this done, we got the line of best fit. The latter could have been obtained by eye judgment. There are some other ways of deducing the relationship between the variables. We shall first consider the ones based on linear relationship, or the ones that can be somehow reduced to such relationships.

3.3.1 Method of Least Squares

Suppose x_i, $i = 1, \cdots, n$ are the points of the independent variable where the dependent variable having respective values y_i, $i = 1, \cdots, n$ is measured. Consider the graph below, where we have assumed a linear graph of equation $y = mx + c$. Then at each point x_i, $i = 1, \cdots, n$, $y_i = mx_i + c$.

The least square method entails minimizing the sum of the squares of the difference between the measured value and the one predicted by the assumed equation.

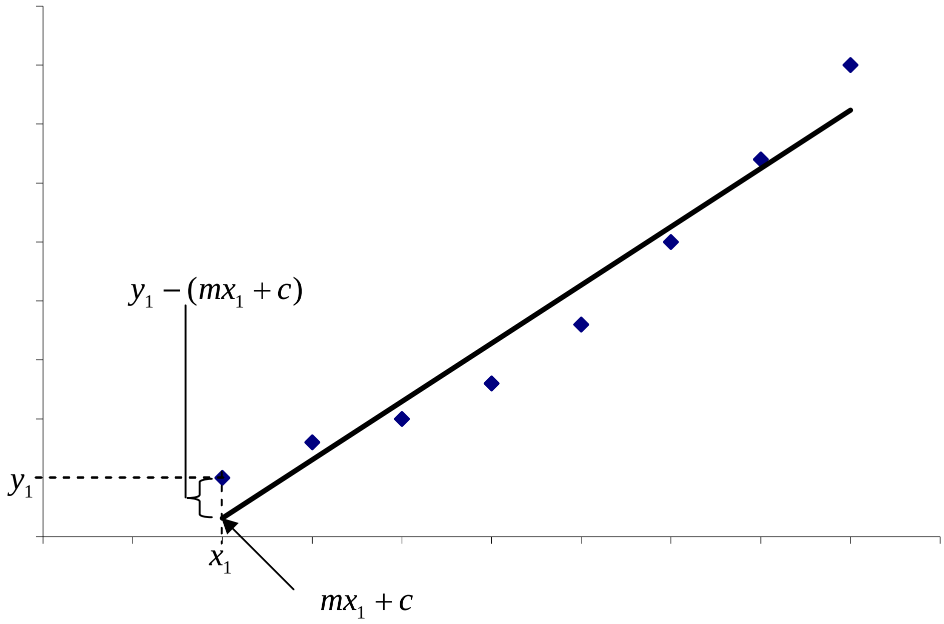


Fig. 1.2: Illustration of the error in representing a set of data with the line of best fit

$$S = \sum_{i=1}^{n} \left[y_i - (mx_i + c) \right]^2 \qquad 2.1$$

We have taken the square of the difference because taking the sum alone might give the impression that there is no error if the sum of positive differences is balanced by the sum of negative differences, just as in the case of the relevance of the variance of a set of data.

Now, S is a function of m and c, that is, $S = S(m,c)$. This is because we seek a line of best fit, which will be determined by an appropriate slope and a suitable intercept. In any case, x_i and y_i are not variables in this case, having been obtained in the laboratory, for instance.

You have been taught at one point or another that for a function of a single variable $f(x)$, the extrema are the points where $\frac{df}{dx} = 0$. However, for a function of more than one variable, partial derivatives are the relevant quantities. Thus, since $S = S(m,c)$, the condition for extrema is

$$\frac{\partial S}{\partial m} = 0 \text{ and } \frac{\partial S}{\partial c} = 0 \qquad 2.2$$

$$\frac{\partial S}{\partial m} = 2\sum_{i=1}^{n} [y_i - (mx_i + c)](-x_i) = 0 \qquad 2.3$$

$$\frac{\partial S}{\partial c} = 2\sum_{i=1}^{n}[y_i - (mx_i + c)](-1) = 0 \tag{2.4}$$

From equation 2.3,

$$\sum_{i=1}^{n} x_i y_i - m\sum_{i=1}^{n} x_i^{\,2} - \sum_{i=1}^{n} cx_i = 0 \tag{2.5}$$

and from equation 2.4,

$$\sum_{i=1}^{n} y_i - m\sum_{i=1}^{n} x_i - \sum_{i=1}^{n} c = 0 \tag{2.6}$$

It follows from the fact that $\bar{x}_i = \dfrac{\sum_{i=1}^{n} x_i}{n}$ and similar expressions, that equations 2.5 and 2.6 give, respectively,

$$\overline{xy} - m\overline{x^2} - c\bar{x} = 0 \tag{2.7}$$

$$\bar{y} - m\bar{x} - c = 0 \tag{2.8}$$

Multiplying equation 2.8 by $\bar{x}$ gives

$$\bar{x}\,\bar{y} - m\bar{x}^2 - c\bar{x} = 0 \tag{2.9}$$

Finally, from equations 2.7 and 2.9,

$$m = \frac{\overline{xy} - \bar{x}\,\bar{y}}{\overline{x^2} - \bar{x}^2} \tag{2.10}$$

and from equation 2.8,

$$c = \bar{y} - m\bar{x} \tag{2.11}$$

Example

A student obtained the following data in the laboratory. By making use of the method of least squares, find the relationship between *x* and *t*.

Thus, for the following set of readings:

t	5	12	19	26	33
x	23	28	32	38	41

The table can be extended to give

t	5	12	19	26	33	Σ=95	$\bar{t}$ =19
x	23	28	32	38	41	Σ=162	$\bar{x}$ =32.4
tx	115	336	608	988	1353	Σ=3400	$\overline{tx}$ =680
t^2	25	144	361	676	1089	Σ=2295	$\overline{x^2}$ =459

$$m = \frac{\overline{tx} - \bar{t}\,\bar{x}}{\overline{t^2} - \bar{t}^2} = \frac{680 - 19\times 32.4}{459 - 19^2} = 0.6571 \tag{2.12}$$

$$c = \bar{x} - m\bar{t} = 32.4 - 0.6571\times 19 = 19.9151 \tag{2.13}$$

Hence, the relationship between x and t is,

$$x = 0.6571t + 19.9151$$

3.3.2 Method of group averages

As the name implies, a set of data is divided into two groups, each of which is assumed to have a zero sum of residuals. Thus, given the equation

$$y = mx + c \qquad 2.14$$

we would like to fit a set of n observations as close as possible.

The error in the measured value of the variable and the value predicted by the equation is (as we have seen in Fig. …):

$$\varepsilon_i = y_i - (mx_i + c) \qquad 2.15$$

The fitted line requires two unknown quantities: m and c. Thus, two equations are needed. We would achieve these two equations by dividing the data into two, one of size l and the other of size n-l, where n is the total number of observations.

The assumption that the sum of errors for each group is zero, requires that

$$\sum_{i=1}^{l} [y_i - (mx_i + c)] = 0 \qquad 2.16$$

and

$$\sum_{l+1}^{n} [y_i - (mx_i + c)] = 0 \qquad 2.17$$

From equation 2.16,

$$\sum_{i=1}^{l} y_i = m \sum_{i=1}^{l} x_i + lc \qquad 2.18$$

and equation 2.17 yields

$$\sum_{i=l+1}^{n} y_i = m \sum_{i=l+1}^{n} x_i + (n-l)c \qquad 2.19$$

the latter equation being true since $n - l$ is the number of observations that fall into that group.

Dividing through by l and $n - l$, respectively, equation 2.18 gives

$$\frac{1}{l}\sum_{i=1}^{l} y_i = m\frac{1}{l}\sum_{i=1}^{l} x_i + c \qquad 2.20$$

and from equation 2.19,

$$\frac{1}{n-l}\sum_{i=l+1}^{n} y_i = m\frac{1}{n-l}\sum_{i=l+1}^{n} x_i + c \qquad 2.21$$

Thus,

$$\bar{y}_1 = m\bar{x}_1 + c$$
$$\bar{y}_2 = m\bar{x}_2 + c \qquad 2.22$$

Subtracting,

$$\bar{y}_1 - \bar{y}_2 = m(\bar{x}_1 - \bar{x}_2) \qquad 2.23$$

$$m = \frac{\bar{y}_1 - \bar{y}_2}{\bar{x}_1 - \bar{x}_2} \qquad 2.24$$

and

$$c = \bar{y}_1 - m\bar{x}_1 \qquad 2.25$$

Example

Let us solve the example in Section 3.3.1 using the method of group averages.

t	5	12	19	26	33
x	23	28	32	38	41

We shall divide the data into two groups, such as:

t	5	12	19
x	23	28	32

and

t	26	33
x	38	41

The tables can be extended to give, for Table 3:

t	5	12	19	Σ=36	$\bar{t}_1$=12
x	23	28	32	Σ=83	$\bar{x}_1$=27.666667

and for Table 4:

t	26	33	Σ=59	$\bar{t}_2$ =29.5
x	38	41	Σ=79	$\bar{x}_2$ =39.5

$$m = \frac{\bar{x}_1 - \bar{x}_2}{\bar{t}_1 - \bar{t}_2} = \frac{27.666667 - 39.5}{12 - 29.5} = 0.67619$$

and

$$c = \bar{x}_1 - m\bar{t}_1 = 27.666667 - (0.67619 \times 12)$$
$$= 19.552387$$

Thus, the equation of best fit is,

$$x = 0.67619t + 19.552387$$

4.0 Conclusion

In this Unit, you learnt how to linearise an expression in order to obtain some relevant information when written as a linear equation. You also derived the equations for two different methods of drawing the line of best fit. In addition, you applied these formulas to a set of data and was able to write the equation of best fit in each case.

5.0 Summary

In this Unit, you learnt:

- How to linearise a nonlinear expression in order to deduce some desired parameters.
- How to draw the line of best fit with the method of least squares.
- How to draw the line of best fit with the method of group averages.

6.0 Tutor Marked Assignment (TMA)

1. The current flowing in a particular R-C circuit is tabulated against the change in the time $t - t_0$, such that at time $t = t_0$, the current is 1.2 A. Using the least-squares method, find the slope and the intercept of the linear function relating the current *i* to the time *t*. Hence, determine the time-constant of the circuit.

t	2	2.2	2.4	2.6	2.8	3
i	0.20	0.16	0.13	0.11	0.09	0.07

2. Solve the problem in TMA 1 with the method of group averages by dividing into two groups of three data sets each.

t	2	2.2	2.4
i	0.20	0.16	0.13

and

t	2.6	2.8	3
i	0.11	0.09	0.07

3. A student performing the simple pendulum experiment obtained the following table, where t is the time for 50 oscillations.

l (cm)	50	45	40	35	30	25	20	15
t (s)	71	69	65	61	56	52	48	43

Find the acceleration due to gravity at the location of the experiment, using

(i) the method of least squares, and

(ii) the method of group averages.

7.0 References/Further Readings

Solutions to Tutor Marked Assignment

1. The current flowing in a particular R-C circuit is tabulated against the change in the time $t - t_0$, such that at time $t = t_0$, the current is 1.2 A. Using the least-squares method, find the slope and the intercept of the linear function relating the current *i* to the time *t*. Hence, determine the time-constant of the circuit.

t	2	2.2	2.4	2.6	2.8	3
i	0.20	0.16	0.13	0.11	0.09	0.07

Taking logs: $i = i_0 e^{-t/RC}$. $\log i = \log i_0 - \log(e^{-t/RC}) = \log i_0 + \dfrac{t}{RC}$. A plot of $\log i$ against *t* gives slope $\dfrac{1}{RC}$ and intercept $\log i_0$.

	t	I	tsquare	log I	tlogI	
	2.0	0.200000	4	-0.69897	-1.39794	-0.7022
	2.2	0.160000	4.84	-0.79588	-1.75094	-0.78902
	2.4	0.130000	5.76	-0.88606	-2.12654	-0.87584
	2.6	0.110000	6.76	-0.95861	-2.49238	-0.96266
	2.8	0.090000	7.84	-1.04576	-2.92812	-1.04948
	3.0	0.070000	9	-1.1549	-3.46471	-1.1363
Sum	15		38.2	-5.54017	-14.1606	
Average	2.5		6.3666667	-0.92336	-2.3601	
			Slope	-0.4431		
			Intercept	0.1844		

$$m = \frac{-2.3601 - (2.5 \times -0.92336)}{6.3666667 - 2.5^2} = -0.4431$$

$$c = \overline{\log l} - m\bar{t} = 0.1844$$

$$m = -\frac{1}{RC}, \text{ or } RC = -\frac{1}{m} = 2.2568 = \text{time constant of the circuit.}$$

2. Solve the problem in TMA 1 with the method of group averages by dividing into two groups of three data sets each.

t	2	2.2	2.4
i	0.20	0.16	0.13

and

t	2.6	2.8	3
i	0.11	0.09	0.07

Group 1

t	i	log i
2.0	0.20	-0.69897
2.2	0.16	-0.79588
2.4	0.13	-0.88606
6.6		-2.38091
2.2		-0.79364

Group 2

t	i	log i
2.6	0.11	-0.95861
2.8	0.09	-1.04576
3.0	0.07	-1.1549
8.4		-3.15927
2.8		-1.05309

$$m = \frac{\bar{y}_1 - \bar{y}_2}{\bar{x}_1 - \bar{x}_2} = \frac{-.79364 - (-1.05309)}{2.2 - 2.8} = -0.4324$$

$$c = \bar{y}_1 - m\bar{x}_1 = -0.79364 - (-0.4324 \times 2.2) = 0.1576$$

3. A student performing the simple pendulum experiment obtained the following table, where t is the time for 50 oscillations.

$l\,$(cm)	50	45	40	35	30	25	20	15
$t\,$(s)	71	69	65	61	56	52	48	43

Find the acceleration due to gravity at the location of the experiment, using

(iii) the method of least squares, and

(iv) the method of group averages.

Method of least squares (taking logs)

$\log T = \log\left(\frac{2\pi}{\sqrt{g}}\right) + \frac{1}{2}\log l$: A plot of $\log T$ against $\log l$ gives slope 0.5 and intercept $c = \log\frac{2\pi}{\sqrt{g}}$, from which the value of g is $\left(\frac{2\pi}{\log^{-1}(c)}\right)^2$.

l	t	log l	log T	(log l)*(log l)	(log l)*(log T)	
0.50	71	-0.30103	0.152288	0.090619058	-0.04584	0.2966771
0.45	69	-0.34679	0.139879	0.120261561	-0.04851	0.3165404
0.40	65	-0.39794	0.113943	0.158356251	-0.04534	0.3387458

0.35	61	-0.45593	0.086360	0.207873948	-0.03937	0.3639201
0.30	56	-0.52288	0.049218	0.273402182	-0.02574	0.3929817
0.25	52	-0.60206	0.017033	0.362476233	-0.01026	0.4273542
0.20	48	-0.69897	-0.017729	0.488559067	0.012392	0.4694229
0.15	43	-0.82391	-0.065502	0.678825613	0.053967	0.5236588
	Sum	-4.14951	0.475492	2.380373913	-0.1487	
	Average	-0.51869	0.059436	0.297546739	-0.01859	

slope	0.429391
intercept	0.282157
2 pi	6.284
log 2 pi	0.798236
log 2 pi -inter	0.516079
2(log 2pi- inter)	1.032159log g
	10.84g

Method of least squares (taking squares)

$T^2 = \frac{4\pi^2}{g} l$. A plot of T^2 against l gives a line through the origin with slope $m = \frac{4\pi^2}{g}$, from which $g = \frac{4\pi^2}{m}$:

L	t	L	Tsquare	lsquare	Tsquare l		
50	71	0.50	2.016400	0.2500	1.0082	2.05	0.5
45	69	0.45	1.904400	0.2025	0.85698	0.070136	0.45
40	65	0.40	1.690000	0.1600	0.676	0	0.4
35	61	0.35	1.488400	0.1225	0.52094	2.0164	0.35
30	56	0.30	1.254400	0.0900	0.37632	2.50932	0.3
25	52	0.25	1.081600	0.0625	0.2704	2.1661	0.25
20	48	0.20	0.921600	0.0400	0.18432	1.8264	0.2
15	43	0.15	0.739600	0.0225	0.11094	1.47766	0.15
	Sum	2.6	11.0964	0.9500	4.0041		
	Average	0.325	1.38705	0.11875	0.500513		

slope	3.788286	g	10.42
intercept	0.155857		

Method of group averages (taking logs)

Group 1

L	t	log l	log T
0.50	71	-0.3010	0.15229
0.45	69	-0.3468	0.13988
0.40	65	-0.3979	0.11394
0.35	61	-0.4559	0.08636

	Sum	-1.50169	0.49247
	Average	-0.37542	0.123118

Group 2

L	t	log l	log T
0.30	56	-0.5229	0.04922
0.25	52	-0.6021	0.01703
0.20	48	-0.6990	-0.0177
0.15	43	-0.8239	-0.0655
	Sum	-2.64782	-0.01698
	Average	-0.66196	-0.00425

slope	0.444496
intercept	0.289991
g	10.38

Method of group averages (taking squares)

Group 1

L	t	l	Tsquare
0.50	71	0.50	2.0164
0.45	69	0.45	1.9044
0.40	65	0.40	1.69
0.35	61	0.35	1.4884
	Sum	1.70	7.0992
	Average	0.43	1.7748

Group 2

L	t	l	Tsquare
0.30	56	0.30	1.2544
0.25	52	0.25	1.0816
0.20	48	0.20	0.9216
0.15	43	0.15	0.7396
	Sum	0.9	3.9972
	Average	0.225	0.9993

slope	3.8775
intercept	1.02051
g	10.18

UNIT 3: Linear Systems of Equations

1.0 Introduction

Perhaps in all areas of Physics, you would come across a system of linear equations. For example, you might want to know what proportions of two or more variables you would need to achieve some specific values of a desired composite product. This kind of problem could lead to a set of linear equations. This unit will equip you with the necessary tools to solve a system of linear equations. You shall come across direct methods as well as iterative ways of solving such problems.

2.0 Objectives

You should be able to do the following after studying this Unit:

- Write a system of linear equations in an augmented matrix form
- Solve a system of linear equations.

3.1 System of Linear Equations

It is necessary for us to set the stage by getting to know how to write the general set of simultaneous linear equations.

Let us consider a linear system of equations

$$\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \cdots a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots a_{2n}x_n = b_2 \\ . \\ . \\ . \\ a_{n1}x_1 + a_{n2}x_2 + \cdots a_{nn}x_n = b_n \end{array} \qquad 3.1$$

This can be written in the form

$$\begin{pmatrix} a_{11} & a_{12} & \cdot & \cdot & \cdot & a_{1n} \\ a_{21} & a_{22} & \cdot & \cdot & \cdot & a_{2n} \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & \cdot & \cdot & \cdot & \cdot \\ a_{n1} & a_{n2} & \cdot & \cdot & \cdot & a_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \cdot \\ \cdot \\ \cdot \\ x_n \end{pmatrix} = \begin{pmatrix} b_1 \\ b_2 \\ \cdot \\ \cdot \\ \cdot \\ b_n \end{pmatrix} \qquad 3.2$$

3.2 Gaussian Elimination

A recall of the solution of a system of two equations will help in introducing the Gaussian Elimination method.

For instance, let $(2,3)$ be a solution set (x, y). Then the following equations are in order.

$$2x + 3y = 13 \qquad 3.3$$
$$x - y = -1 \qquad 3.4$$

You might want to verify that these equations are consistent with the given solution set.

We could multiply equation 3.2 by -2 and add to equation 2.3. This yields

$$5y = 15 \qquad 3.5$$

Equivalently, $y = 3$. Substituting this value of y in either equation 3.3 or 3.4 gives $x = 2$.

The augmented matrix representing our system of two equations is

$$\left[\begin{array}{cc|c} 2 & 3 & 13 \\ 1 & -1 & -1 \end{array}\right]$$

By Gaussian elimination, we seek to make every entry below the main diagonal zero. This we achieve by reducing 1 to zero, making use of the first row.

$$\left[\begin{array}{cc|c} 2 & 3 & 13 \\ 1 & -1 & -1 \end{array}\right] \xrightarrow{(ii)'=(i)-2(ii)} \left[\begin{array}{cc|c} 2 & 3 & 13 \\ 0 & 5 & 15 \end{array}\right] \qquad 3.6$$

Thus,

$$5y = 15 \Rightarrow y = 3 \qquad 3.7$$

Substituting this in the first row gives

$$2x + 3(3) = 13 \qquad 3.8$$

from which we obtain $x = 2$.

The process of reducing every element below the main diagonal to zero (row echelon form) is called Gaussian Elimination. That of substituting obtained values to calculate other variables is called Back Substitution.

You can see that there is nothing new about Gaussian elimination. It is a process you have been carrying out all along, but which you never called this name.

The same process can be carried over to the case of a system of three equations.

Let $(1,2,-1)$ be a solution set.

Then, the equations below are valid:

$$\begin{aligned} 2x+y-z&=5 \\ x+3y+2z&=5 \\ 3x-2y-4z&=3 \end{aligned} \qquad 3.9$$

The augmented matrix is

$$\left[\begin{array}{ccc|c} 2 & 1 & -1 & 5 \\ 1 & 3 & 2 & 5 \\ 3 & -2 & -4 & 3 \end{array}\right]$$

This yields (by Gaussian elimination)

$$\left[\begin{array}{ccc|c} 2 & 1 & -1 & 5 \\ 1 & 3 & 2 & 5 \\ 3 & -2 & -4 & 3 \end{array}\right] \begin{array}{c} \\ \xrightarrow{(ii)'=(i)-2(ii)} \\ \xrightarrow{(iii)=(i)-(2/3)(iii)} \end{array} \left[\begin{array}{ccc|c} 2 & 1 & -1 & 5 \\ 0 & -5 & -5 & -5 \\ 0 & 7/3 & 5/3 & 3 \end{array}\right]$$

$$\begin{array}{c} \\ \\ \xrightarrow{(iii)''=(ii)'+(15/7)(iii)'} \end{array} \left[\begin{array}{ccc|c} 2 & 1 & -1 & 5 \\ 0 & -5 & -5 & -5 \\ 0 & 0 & -10 & 10 \end{array}\right] \qquad 3.10$$

Upon back substitution,

$$-10z=10 \text{ or } z=-1$$

$$z=-1;\; y+z=1 \Rightarrow y=2;\; 2x+y-z=5 \Rightarrow x=1$$

Traditionally, in Mathematics, it is usual to use indices such as $x_1, x_2,$ etc. instead of x, y, z. Do you have any idea why this is so? It is because if we stay with the alphabets, we shall soon run out of symbols. Bear in mind that not all the alphabets can be employed as variables; as an example, a, b, c are commonly used as constants. In addition, it makes it easy to associate the coefficients $a_{11}, a_{12},$ etc. with $x_1, x_2,$ etc. respectively. More importantly in numerical work, it makes programming easier. For instance for our system of three equations, we could use the more general notation:

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{array}\right] \xrightarrow[(iii)=(i)-(a_{11}\,/\,a_{31})(iii)]{(ii)'=(i)-(a_{11}\,/\,a_{12})(ii)} \left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22}' & a_{23}' & a_{24}' \\ 0 & a_{32}' & a_{33}' & a_{34}' \end{array}\right]$$

$$\xrightarrow{(iii)''=(ii)'+(a_{22}\,/\,a_{32})(iii)'} \left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22}' & a_{23}' & a_{24}' \\ 0 & 0 & a_{33}'' & a_{34}'' \end{array}\right] \qquad 3.11$$

We would like to sound a note of warning here. How do you set a_{21}' equal to zero? From the expression 3.11, $a_{21}' = 0 = a_{11} - \dfrac{a_{11}}{a_{21}} a_{21}$. In order to avoid having to deal with fractions which could lead to rounding errors, it is better to put this in the form:

$$(ii)' = a_{12}(i) - a_{11}(ii) \qquad 3.12$$

A better way of writing equation 3.11 is,

$$\left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \end{array}\right] \xrightarrow[(iii)=a_{11}(i)-a_{11}(iii)]{(ii)'=a_{12}(i)-a_{11}(ii)} \left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22}' & a_{23}' & a_{24}' \\ 0 & a_{32}' & a_{33}' & a_{34}' \end{array}\right]$$

$$\xrightarrow{(iii)''=a_{32}(ii)'+a_{22}(iii)'} \left[\begin{array}{ccc|c} a_{11} & a_{12} & a_{13} & a_{14} \\ 0 & a_{22}' & a_{23}' & a_{24}' \\ 0 & 0 & a_{33}'' & a_{34}'' \end{array}\right] \qquad 3.11$$

3.3 Gauss-Jordan Elimination

This entails eliminating in addition to the entries below the major diagonal, the entries above it, so that the main matrix is a diagonal matrix. In that case, the solution to the system is given by dividing the element in the augmented part of the matrix by the diagonal element for that row. In other words, the end product of Gauss-Jordan elimination looks like

$$\left[\begin{array}{ccc|c} a_{11} & 0 & 0 & a_{14}'' \\ 0 & a_{22}'''' & 0 & a_{24}''' \\ 0 & 0 & a_{33}'''' & a_{34}'''' \end{array}\right] \qquad 3.12$$

from which it follows that

$$x_1 = a_{14}''/a_{11} \qquad x_2 = a_{24}'''/a_{22}''' \qquad x_3 = a_{34}''''/a_{33}'''' \qquad 3.13$$

Example

We shall solve problem … using the Gauss-Jordan elimination. Luckily, we have already completed the Gaussian elimination part of this method. We continue from where we stopped.

$$\left[\begin{array}{ccc|c} 2 & 1 & -1 & 5 \\ 0 & -5 & -5 & -5 \\ 0 & 0 & -10 & 10 \end{array}\right] \xrightarrow[(ii)'=(iii)-2(ii)]{(i)'=(iii)-10(i)} \left[\begin{array}{ccc|c} -20 & 10 & 0 & -40 \\ 0 & -10 & 0 & 20 \\ 0 & 0 & -10 & 10 \end{array}\right]$$

$$\xrightarrow{(i)''=(ii)'+(i)'} \left[\begin{array}{ccc|c} -20 & 0 & 0 & -20 \\ 0 & -10 & 0 & 20 \\ 0 & 0 & -10 & 10 \end{array}\right] \qquad 3.14$$

It follows that,

$-20x = -20$ or $x = 1$; $-10y = 20$ or $y = 2$; and $-10z = 10$ or $z = -1$

3.4 LU Decomposition

Suppose we could write the matrix

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} l_{11} & 0 & 0 \\ l_{21} & l_{22} & 0 \\ l_{31} & l_{32} & l_{33} \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \qquad 3.15$$

This implies that

$$l_{11}u_{11} = a_{11},\ l_{11}u_{12} = a_{12},\ l_{11}u_{13} = a_{13} \qquad 3.16$$

$$a_{21} = l_{21}u_{11},\ a_{22} = l_{21}u_{12} + l_{22}u_{22},\ a_{23} = l_{21}u_{13} + l_{22}u_{23} \qquad 3.17$$

$$a_{31} = l_{31}u_{11},\ a_{32} = l_{31}u_{12} + l_{32}u_{22},\ a_{33} = l_{31}u_{13} + l_{32}u_{23} + l_{33}u_{33} \qquad 3.18$$

Without loss of generality, we could set the diagonal elements of the L matrix equal to 1. Then,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ l_{21} & 1 & 0 \\ l_{31} & l_{32} & 1 \end{bmatrix} \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ 0 & u_{22} & u_{23} \\ 0 & 0 & u_{33} \end{bmatrix} \qquad 3.19a$$

Multiplying out the right side of equation 3.19,

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix} = \begin{bmatrix} u_{11} & u_{12} & u_{13} \\ l_{21}u_{11} & l_{21}u_{12} + u_{22} & l_{21}u_{13} + u_{23} \\ l_{31}u_{11} & l_{31}u_{12} + l_{32}u_{22} & l_{31}u_{13} + l_{32}u_{23} + u_{33} \end{bmatrix} \qquad 3.19b$$

From the equality of matrices, this requires that,

$$u_{11} = a_{11} \qquad 3.20$$

$$u_{12} = a_{12} \tag{3.21}$$

$$u_{13} = a_{13} \tag{3.22}$$

$$a_{21} = l_{21}u_{11} \qquad \Rightarrow l_{21} = a_{21}/u_{11} = a_{21}/a_{11} \tag{3.23}$$

$$a_{31} = l_{31}u_{11} \qquad \Rightarrow l_{31} = a_{31}/u_{11} = a_{31}/a_{11} \tag{3.24}$$

$$a_{22} = l_{21}u_{12} + u_{22} \text{ , or } u_{22} = a_{22} - l_{21}u_{12} = a_{22} - \frac{a_{21}}{u_{11}}u_{12}$$

$$\Rightarrow u_{22} = a_{22} - \frac{a_{21}}{a_{11}}a_{12} \tag{3.25}$$

$$a_{23} = l_{21}u_{13} + u_{23}\text{, or } u_{23} = a_{23} - l_{21}u_{13} = a_{23} - \frac{a_{21}}{u_{11}}u_{13}$$

$$\Rightarrow u_{23} = a_{23} - \frac{a_{21}}{a_{11}}a_{13} \tag{3.26}$$

$$l_{32} = \frac{a_{32} - l_{31}u_{12}}{u_{22}} = \frac{1}{u_{22}}\left[a_{32} - \frac{a_{31}}{u_{11}}u_{12}\right]$$

$$\Rightarrow l_{32} = \frac{1}{u_{22}}\left[a_{32} - \frac{a_{31}}{a_{11}}a_{12}\right] \tag{3.27}$$

$$a_{32} = l_{31}u_{12} + l_{32}u_{22} \tag{3.28}$$

$$a_{33} = l_{31}u_{13} + l_{32}u_{23} + u_{33}$$

$$\Rightarrow u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23} \tag{3.29}$$

You can see that we have determined all the nine elements of the two matrices in terms of the elements of the original matrix.

Once we have obtained L and U, then we can write the original equation

$$\begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} \tag{3.30}$$

as

$$LU\mathbf{x} = \mathbf{y} \tag{3.31}$$

where **x** and **y** are column vectors.

We shall write $\mathbf{w} = U\mathbf{x}$
Then,

$$L\mathbf{w} = \mathbf{y} \tag{3.32}$$

Example
Solve the following system of equations using the method of LU decomposition.

$$\begin{aligned} 2x + y - z &= 5 \\ x + 3y + 2z &= 5 \\ 3x - 2y - 4z &= 3 \end{aligned} \qquad 3.33$$

The corresponding matrix is

$$\begin{bmatrix} 2 & 1 & -1 \\ 1 & 3 & 2 \\ 3 & -2 & -4 \end{bmatrix}$$

$$u_{11} = a_{11} = 2 \qquad 3.34$$

$$u_{12} = a_{12} = 1 \qquad 3.35$$

$$u_{13} = a_{13} = -1 \qquad 3.36$$

$$l_{21} = a_{21} / a_{11} = 1/2 \qquad 3.37$$

$$l_{31} = a_{31} / a_{11} = 3/2 \qquad 3.38$$

$$u_{22} = a_{22} - \frac{a_{21}}{a_{11}} a_{12} = 3 - \frac{1}{2}(1) = 5/2$$

$$3.39$$

$$u_{23} = a_{23} - \frac{a_{21}}{a_{11}} a_{13} = 2 - \frac{1}{2}(-1) = 2 + \frac{1}{2} = 5/2$$

$$l_{32} = \frac{1}{u_{22}} \left[a_{32} - \frac{a_{31}}{a_{11}} a_{12} \right] = \frac{1}{5/2} \left[-2 - \frac{3}{2}(1) \right] = -7/5 \qquad 3.40$$

$$u_{33} = a_{33} - l_{31} u_{13} - l_{32} u_{23} = -4 - (3/2)(-1) - (-7/5)(5/2) = 1 \qquad 3.41$$

Thus,

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 3/2 & -7/5 & 1 \end{bmatrix} \begin{bmatrix} 2 & 1 & -1 \\ 0 & 5/2 & 5/2 \\ 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 2 & 1 & -1 \\ 1 & 3 & 2 \\ 3 & -2 & -4 \end{bmatrix} \qquad 3.42$$

As you can see, we got the decomposition right, as the multiplication of the L and U gives the original matrix.

The original equation is equivalent to

$$LU\mathbf{x} = L\mathbf{w} = \mathbf{y} \qquad 3.43$$

$L\mathbf{w} = \mathbf{y}$ implies

$$\begin{bmatrix} 1 & 0 & 0 \\ 1/2 & 1 & 0 \\ 3/2 & -7/5 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} 5 \\ 5 \\ 3 \end{bmatrix} \qquad 3.44$$

Solving,

$$w_1 = \quad 5 \tag{3.45}$$

$$\frac{1}{2}w_1 + w_2 = 5 \text{ or } w_2 = 5 - \frac{1}{2}w_1 = 5 - \frac{1}{2}(5) = \frac{5}{2} \tag{3.46}$$

$$\frac{3}{2}w_1 - \frac{7}{5}w_2 + w_3 = 3, \text{ or } w_3 = 3 + \frac{7}{5}w_2 - \frac{3}{2}w_1 = 3 + \frac{7}{5}\left(\frac{5}{2}\right) - \frac{3}{2}(5) = -1 \tag{3.47}$$

$U\mathbf{x} = \mathbf{w}$ implies:

$$\begin{bmatrix} 2 & 1 & -1 \\ 0 & 5/2 & 5/2 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ 5/2 \\ -1 \end{bmatrix} \tag{3.48}$$

By back substitution,

$$x_3 = -1 \tag{3.49}$$

$$\frac{5}{2}x_2 + \frac{5}{2}x_3 = \frac{5}{2} \Rightarrow \frac{5}{2}x_2 = \frac{5}{2} - \frac{5}{2}x_3 = \frac{5}{2} - \frac{5}{2}(-1) = 5 \tag{3.50}$$

$$x_2 = 2 \tag{3.51}$$

$$2x_1 + x_2 - x_3 = 5 \tag{3.52}$$

$$x_1 = \frac{5 - x_2 + x_3}{2} = \frac{5 - 2 + (-1)}{2} = 1 \tag{3.53}$$

The solution set is therefore,

$$x_1 = 1,\ y = 2,\ z = -1. \tag{3.54}$$

3.5 Jacobi Iteration

Given the system of equations

$$a_1 x + b_1 y + c_1 z = d_1 \tag{3.55}$$

$$a_2 x + b_2 y + c_2 z = d_2 \tag{3.56}$$

$$a_3 x + b_3 y + c_3 z = d_3 \tag{3.57}$$

Solving for x, y and z, gives

$$x = \frac{1}{a_1}\left[d_1 - b_1 y - c_1 z\right] \tag{3.58}$$

$$y = \frac{1}{b_2}[d_2 - a_2 x - c_2 z] \tag{3.59}$$

$$z = \frac{1}{c_3}[d_3 - a_3 x - b_3 y] \tag{3.60}$$

It is easy to see that provided the diagonal elements are large relative to the other coefficients, the sequence of iteration would converge.

For initial values x_0, y_0 and z_0, the scheme would be as shown below:

$$x_1 = \frac{1}{a_1}\left[d_1 - b_1 y_0 - c_1 z_0\right] \qquad 3.61$$

$$y_1 = \frac{1}{b_2}[d_2 - a_2 x_0 - c_2 z_0] \qquad 3.62$$

$$z_1 = \frac{1}{c_3}[d_3 - a_3 x_0 - b_3 y_0] \qquad 3.63$$

We can now write, for $n = 0$ and above,

$$x_{n+1} = \frac{1}{a_1}\left[d_1 - b_1 y_n - c_1 z_n\right] \qquad 3.64$$

$$y_{n+1} = \frac{1}{b_2}[d_2 - a_2 x_n - c_2 z_n] \qquad 3.65$$

$$z_{n+1} = \frac{1}{c_3}[d_3 - a_3 x_n - b_3 y_n] \qquad 3.66$$

The sequence of iteration continues until there is convergence, in the sense that $| x_{n+1} - x_n |$, $| y_{n+1} - y_n |$ and $| z_{n+1} - z_n |$ are less than the prescribed tolerance.

Example

We shall solve the following system of equations using the Jacobi iteration method.

$$25x + y - z = 28 \qquad 3.67$$

$$x + 30y + 2z = 59 \qquad 3.68$$

$$3x - 2y - 20z = 19 \qquad 3.69$$

Equivalently,

$$x = \frac{28 - y + z}{25},\ y = \frac{59 - x - 2z}{30},\ z = \frac{3x - 2y - 19}{20} \qquad 3.70$$

Let us assume that the initial guess of solution is (0, 0, 0).
Then, the first set of values for the iteration is:

$$x_1 = \frac{28 - 0 + 0}{25} = \frac{28}{25} = 1.12 \qquad 3.71$$

$$y_1 = \frac{59 - 0 - 0}{30} = \frac{59}{30} = 1.96666667 \qquad 3.72$$

$$z_1 = \frac{0 - 0 - 19}{20} = -\frac{19}{20} = -0.95 \qquad 3.73$$

$$x = \frac{28 - 59/30 - 19/20}{25} = \frac{13850}{15000} = 1.00333333 \qquad 3.74$$

$$y = \frac{59 - \frac{28}{25} - 2\times(-\frac{19}{20})}{30} = \frac{5606}{1500} = \frac{2803}{1500} = 1.99266667 \qquad 3.75$$

$$z = \frac{3\times\frac{26}{25} - 2\times\frac{59}{30} - 19}{20} = -\frac{14860}{15000} = -0.97866667 \qquad 3.76$$

Table 3.1 shows the rest of the computation.

Table 3.1: Table for Jacobi iteration

n	x	y	z
1	1.12000000	1.96666667	-0.95000000
2	1.00333333	1.99266667	-0.97866667
3	1.00114667	1.99846667	-0.99876667
4	1.00011067	1.99987956	-0.99967467
5	1.00001783	1.99997462	-0.99997136
6	1.00000216	1.99999750	-0.99999479
7	1.00000031	1.99999958	-0.99999943
8	1.00000004	1.99999995	-0.99999991
9	1.00000001	1.99999999	-0.99999999
10	1.00000000	2.00000000	-1.00000000

3.6 Gauss-Seidal Iteration

You would recall that in each of the Jacobi iterations, we calculated the value of the variables using the old variables. The Gauss-Seidal iteration is a modification of this method, in which the value of x obtained in a particular iteration and the old value of *z* is put into the formula for y to obtain a new value for *y*. The new values of *x* and *y* are substituted into the equation for *z*.

Thus, given the system of equations

$$a_1 x + b_1 y + c_1 z = d_1 \qquad 3.77$$

$$a_2 x + b_2 y + c_2 z = d_2 \qquad 3.78$$

$$a_3 x + b_3 y + c_3 z = d_3 \qquad 3.79$$

with the initial condition x_0, y_0, z_0,

$$x_1 = \frac{1}{a_1}\left[d_1 - b_1 y_0 - c_1 z_0\right] \qquad 3.80$$

$$y_1 = \frac{1}{b_2}[d_2 - a_2 x_1 - c_2 z_0] \qquad 3.81$$

$$z_1 = \frac{1}{c_3}[d_3 - a_3 x_1 - b_3 y_1] \qquad 3.82$$

$$x_{n+1} = \frac{1}{a_1}\left[d_1 - b_1 y_n - c_1 z_n\right] \qquad 3.83$$

$$y_{n+1} = \frac{1}{b_2}[d_2 - a_2 x_{n+1} - c_2 z_n] \qquad 3.84$$

$$z_{n+1} = \frac{1}{c_3}[d_3 - a_3 x_{n+1} - b_3 y_{n+1}] \qquad 3.85$$

As in the case of the Jacobi iteration, the sequence of iteration continues until there is convergence, in the sense that $|\, x_{n+1} - x_n \,|$, $|\, y_{n+1} - y_n \,|$ and $|\, z_{n+1} - z_n \,|$ are less than the prescribed tolerance.

Example

We shall solve the following system of equations using the Gauss-Seidal iteration method. Assume (0,0,0) is the initial guess of solution.

$$25x + y - z = 28 \qquad 3.86$$

$$x + 30y + 2z = 59 \qquad 3.87$$

$$3x - 2y - 20z = 19 \qquad 3.88$$

$$x_1 = \frac{28 - y_0 + z_0}{25} \qquad 3.80$$

$$y_1 = \frac{59 - x_1 - 2z_0}{20} \qquad 3.81$$

$$z_1 = \frac{3x_1 - 2y_1 - 19}{30} \qquad 3.82$$

$$x_1 = \frac{28 - 0 + 0}{25} = \frac{28}{25} = 1.12 \qquad 3.80$$

$$y_1 = \frac{59 - \frac{28}{25} - 2 \times 0}{30} = \frac{1447}{750} = 1.929333333 \qquad 3.81$$

$$z_1 = \frac{3 \times \frac{28}{25} - 2 \times \frac{1447}{750} - 19}{20} = -\frac{365600}{375000} = -0.974933333 \qquad 3.82$$

You can verify the remaining calculations on Table 3.2.

Table 3.2: Table for Gauss-Seidal iteration

n	x	y	z
1	1.12000000	1.92933333	-0.97493333

2	1.00382933	1.99820124	-0.99924572
3	1.00010212	1.99994631	-0.99997931
4	1.00000298	1.99999852	-0.99999941
5	1.00000008	1.99999996	-0.99999998
6	1.00000000	2.00000000	-1.00000000

Observation: As expected, the Gauss-Seidal iteration converged faster than the Jacobi iteration.

4.0 Conclusion

In this Unit, you learnt various methods for solving a system of linear algebraic or transcendental equations using various methods: some were direct, while the others were iterative in nature. You also got to know the merits and the demerits of direct and iterative methods. You also found out that it is important, in elementary row operations, to avoid having to deal with fractions, so as to keep rounding errors minimal.

5.0 Summary

You learnt the following in this Unit:

- How to write a matrix in the form amenable for programming.
- How to numerically solve a set of linear equations.
- That the Gauss-Seidal iteration converges faster than the Jacobi iteration.
- In numerical work, for the sake of avoiding rounding errors, it is better to retain fractions for as long as possible.
- Iteration is advisable only if the main diagonal elements are large compared with the other entries of the equivalent matrix.

6.0 Tutor Marked Assignment

1. Solve the system of linear equations $x+y+z=-1$, $x+2y+2z=-4$, $9x+6y+z=7$ using the method of
   (i) Gaussian elimination
   (ii) Gauss-Jordan elimination
   (iii) LU decomposition
   (iv) Jacobi iteration
   (v) Gauss-Seidal iteration

2. Solve the system of linear equations $x+2y+2z=-2$, $2x+2y+z=-4$, $9x+6y+2z=-14$ using the method of
   (i) Gaussian elimination
   (ii) Gauss-Jordan elimination
   (iii) LU decomposition
   (iv) Jacobi iteration
   (v) Gauss-Seidal iteration

7.0 References/Further Reading

Solutions to Tutor Marked Assignment

1. Solve the system of linear equations $x+y+z=-1$, $x+2y+2z=-4$, $9x+6y+z=7$ using the method of

   (i) Gaussian elimination

Initial augmented matrix

$$\begin{array}{cccr} 1 & 1 & 1 & -1 \\ 1 & 2 & 2 & -4 \\ 9 & 6 & 1 & 7 \end{array}$$

First round of Gaussian elimination

$$\begin{array}{crrr} 1 & 1 & 1 & -1 \\ 0 & 1 & 1 & -3 \\ 0 & -3 & -8 & 16 \end{array}$$

Second round of Gaussian elimination

$$\begin{array}{crrr} 1 & 1 & 1 & -1 \\ 0 & 1 & 1 & -3 \\ 0 & 0 & -5 & 7 \end{array}$$

   (ii) Gauss-Jordan elimination

Last matrix for Gaussian elimination

$$\begin{array}{crrr} 1 & 1 & 1 & -1 \\ 0 & 1 & 1 & -3 \\ 0 & 0 & -5 & 7 \end{array}$$

First round of Jordan elimination

$$\begin{array}{crrr} 5 & 5 & 0 & 2 \\ 0 & 5 & 0 & -8 \\ 0 & 0 & -5 & 7 \end{array}$$

Second round of Jordan elimination

$$\begin{array}{rrrr} -25 & 0 & 0 & -50 \\ 0 & 5 & 0 & -8 \\ 0 & 0 & -5 & 7 \end{array}$$

   (iii) LU decomposition

$x+y+z=-1$

$x+2y+2z=-4$ 3.33

$9x+6y+z=7$

The corresponding matrix is

$$\begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 9 & 6 & 1 \end{bmatrix}$$

$$u_{11} = a_{11} = 1 \qquad 3.34$$

$$u_{12} = a_{12} = 1 \qquad 3.35$$

$$u_{13} = a_{13} = 1 \qquad 3.36$$

$$l_{21} = a_{21} / a_{11} = 1/1 \quad = 1 \qquad 3.37$$

$$l_{31} = a_{31} / a_{11} = 9/1 \quad = 9 \qquad 3.38$$

$$u_{22} = a_{22} - \frac{a_{21}}{a_{11}} a_{12} = 2 - (1)(1) = 1 \qquad 3.39$$

$$u_{23} = a_{23} - \frac{a_{21}}{a_{11}} a_{13} = 2 - (1)(1) = 1$$

$$l_{32} = \frac{1}{u_{22}} \left[a_{32} - \frac{a_{31}}{a_{11}} a_{12} \right] = \frac{1}{1} \left[6 - \frac{9}{1}(1) \right] = -3 \qquad 3.40$$

$$u_{33} = a_{33} - l_{31} u_{13} - l_{32} u_{23} = 1 - (9)(1) - (-3)(1) = -5 \qquad 3.41$$

Thus,

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 9 & -3 & 1 \end{bmatrix} \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 2 & 2 \\ 9 & 6 & 1 \end{bmatrix} \qquad 3.42$$

W got the decomposition right, as the multiplication of the L and U gives the original matrix.

The original equation is equivalent to $LU\mathbf{x} = L\mathbf{w} = \mathbf{y}$,
$L\mathbf{w} = \mathbf{y}$ implies

$$\begin{bmatrix} 1 & 0 & 0 \\ 1 & 1 & 0 \\ 9 & -3 & 1 \end{bmatrix} \begin{bmatrix} w_1 \\ w_2 \\ w_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -4 \\ 7 \end{bmatrix} \qquad 3.44$$

Solving,

$$w_1 = \quad -1 \qquad 3.45$$

$$w_1 + w_2 = -4 \text{ or } w_2 = -4 - w_1 = -4 - (-1) = -3 \qquad 3.46$$

$$9w_1 - 3w_2 + w_3 = 7 \text{, or } w_3 = 7 + 3w_2 - 9w_1 = 7 + 3(-3) - 9(-1) = 7 \qquad 3.47$$

$U\mathbf{x} = \mathbf{w}$ implies:

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -5 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -3 \\ 7 \end{bmatrix}$$ 3.48

By back substitution,

$x_3 = -7/5 = -1.4$ 3.49

$x_2 + x_3 = -3 \Rightarrow x_2 = -3 - x_3 = -3 - (-7/5)$ 3.50

$x_2 = -8/5 = -1.6$ 3.51

$x_1 + x_2 + x_3 = -1$ 3.52

$x_1 = -1 - x_2 - x_3 = -1 - (-8/5) - (-7/5) = \frac{10}{5} = 2$ 3.53

The solution set is therefore,

$x_1 = 2$, $y = -1.6$, $z = -1.4$. 3.54

Notice that, where necessary, we reverted to fractions to avoid incurring rounding errors.

2. Solve the system of equations $25x + 2y - z = 26$, $3x - 20y + 2z = -15$, $x + 4y + 15z$ using:
   (i) Jacobi iteration
   (ii) Gauss-Seidal iteration

   Assume a starting set of values $x_0 = y_0 = z_0 = 0$ and a tolerance of $|x_{i+1} - x_i| \le 5\times10^{-6}$, $|y_{i+1} - y_i| \le 5\times10^{-6}$, $|z_{i+1} - z_i| \le 5\times10^{-6}$.

   (i) Jacobi iteration

```
1.040000 0.750000 1.333333
1.033333 1.039333 1.064000
0.999413 1.011400 0.987289
0.998580 0.998641 0.996999
0.999989 0.999487 1.000457
1.000059 1.000044 1.000138
1.000002 1.000023 0.999984
0.999998 0.999999 0.999994
1.000000 0.999999 1.000001
1.000000 1.000000 1.000000
```

   (ii) Gauss-Seidal iteration

```
1.040000 0.906000 1.022400
1.008416 1.003502 0.998505
0.999660 0.999799 1.000076
1.000019 1.000010 0.999996
0.999999 0.999999 1.000000
1.000000 1.000000 1.000000
```

Observation: The Gauss-Seidal iteration scheme converged faster than the Jacobi iteration, as was expected.

3. Solve the system of linear equations $x+2y+2z=-2$, $2x+2y+z=-4$, $9x+6y+2z=-14$ using the method of

(iv) Gaussian elimination

Initial augmented matrix

1	2	2	-2
2	2	1	-4
9	6	2	-14

First round of Gaussian elimination

1	2	2	-2
0	-2	-3	0
0	-12	-16	4

Second round of Gaussian elimination

1	2	2	-2
0	-2	-3	0
0	0	-4	-8

Answers

x	0
y	-3
z	2

(v) Gauss-Jordan elimination

Last matrix for Gaussian elimination

1	2	2	-2
0	-2	-3	0
0	0	-4	-8

First round of Jordan elimination

4	8	0	-24
0	-8	0	24
0	0	-4	-8

Second round of Jordan elimination

32	0	0	0
0	-8	0	24
0	0	-4	-8

(vi) LU decomposition

$$\begin{aligned} x+y+z&=-1\\ x+2y+2z&=-4\\ 9x+6y+z&=7 \end{aligned} \qquad 3.33$$

The corresponding matrix is

$$\begin{bmatrix} 1 & 1 & 1\\ 1 & 2 & 2\\ 9 & 6 & 1 \end{bmatrix}$$

$$u_{11} = a_{11} = 1 \qquad 3.34$$

$$u_{12} = a_{12} = 1 \qquad 3.35$$

$$u_{13} = a_{13} = 1 \qquad 3.36$$

$$l_{21} = a_{21}/a_{11} = 1/1 \quad = 1 \qquad 3.37$$

$$l_{31} = a_{31}/a_{11} = 9/1 \quad = 9 \qquad 3.38$$

$$u_{22} = a_{22} - \frac{a_{21}}{a_{11}}a_{12} = 2-(1)(1) = 1 \qquad 3.39$$

$$u_{23} = a_{23} - \frac{a_{21}}{a_{11}}a_{13} = 2-(1)(1) = 1$$

$$l_{32} = \frac{1}{u_{22}}\left[a_{32} - \frac{a_{31}}{a_{11}}a_{12}\right] = \frac{1}{1}\left[6 - \frac{9}{1}(1)\right] = -3 \qquad 3.40$$

$$u_{33} = a_{33} - l_{31}u_{13} - l_{32}u_{23} = 1-(9)(1)-(-3)(1) = -5 \qquad 3.41$$

Thus,

$$\begin{bmatrix} 1 & 0 & 0\\ 1 & 1 & 0\\ 9 & -3 & 1 \end{bmatrix}\begin{bmatrix} 1 & 1 & 1\\ 0 & 1 & 1\\ 0 & 0 & -5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1\\ 1 & 2 & 2\\ 9 & 6 & 1 \end{bmatrix} \qquad 3.42$$

W got the decomposition right, as the multiplication of the L and U gives the original matrix.

The original equation is equivalent to $LU\mathbf{x} = L\mathbf{w} = \mathbf{y}$,
$L\mathbf{w} = \mathbf{y}$ implies

$$\begin{bmatrix} 1 & 0 & 0\\ 1 & 1 & 0\\ 9 & -3 & 1 \end{bmatrix}\begin{bmatrix} w_1\\ w_2\\ w_3 \end{bmatrix} = \begin{bmatrix} -1\\ -4\\ 7 \end{bmatrix} \qquad 3.44$$

Solving,

$w_1 = \quad -1$ 3.45

$w_1 + w_2 = -4$ or $w_2 = -4 - w_1 = -4 - (-1) = -3$ 3.46

$9w_1 - 3w_2 + w_3 = 7$, or $w_3 = 7 + 3w_2 - 9w_1 = 7 + 3(-3) - 9(-1) = 7$ 3.47

$U\mathbf{x} = \mathbf{w}$ implies:

$$\begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 0 & 0 & -5 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -3 \\ 7 \end{bmatrix} \qquad 3.48$$

By back substitution,

$x_3 = -7/5 = -1.4$ 3.49

$x_2 + x_3 = -3 \Rightarrow x_2 = -3 - x_3 = -3 - (-7/5)$ 3.50

$x_2 = -8/5 = -1.6$ 3.51

$x_1 + x_2 + x_3 = -1$ 3.52

$x_1 = -1 - x_2 - x_3 = -1 - (-8/5) - (-7/5) = \dfrac{10}{5} = 2$ 3.53

The solution set is therefore,

$x_1 = 2$, $y = -1.6$, $z = -1.4$. 3.54

Notice that, where necessary, we reverted to fractions to avoid incurring rounding errors.

4. Solve the system of equations $25x + 2y - z = 26$, $3x - 20y + 2z = -15$, $x + 4y + 15z$ using:

(i) Jacobi iteration

(ii) Gauss-Seidal iteration

Assume a starting set of values $x_0 = y_0 = z_0 = 0$ and a

(i) Jacobi iteration tolerance of $|x_{i+1} - x_i| \le 10^{-7}$, $|y_{i+1} - y_i| \le 10^{-7}$, $|z_{i+1} - z_i| \le 10^{-7}$.

x	y	z
0.571429	-0.300000	0.800000
0.878571	-0.722857	0.805714
0.910816	-0.907714	0.913429
0.962490	-0.937833	0.980922
0.988746	-0.975586	0.982635
0.992054	-0.991511	0.992485
0.996710	-0.994481	0.998194
0.998961	-0.997845	0.998451
0.999293	-0.999221	0.999346
0.999711	-0.999510	0.999830

(ii) Gauss-Seidal iteration of $|x_{i+1} - x_i| \le 5\times10^{-6}$, $|y_{i+1} - y_i| \le 5\times10^{-6}$, $|z_{i+1} - z_i| \le 5\times10^{-6}$.

x	y	z
0.571429	-0.642857	0.942857
0.954082	-0.966735	0.995878
0.996152	-0.997279	0.999681
0.999692	-0.999783	0.999975
0.999976	-0.999983	0.999998
0.999998	-0.999999	1.000000
1.000000	-1.000000	1.000000

Unit 4:Roots of Algebraic and Transcendental Equations

1.0 Introduction

In Physics, as well as in many other scientific fields, there is always the need to find the root of an equation. You have no doubt been tackling such problems from high school days. However, up till now, you have been able to handle simple cases that a calculator could be employed to do. In this Unit, you shall learn how to handle the more complicated cases of roots of algebraic and transcendental equations.

2.0 Objectives

By the time you are through with this Unit, you should be able to:

- Find the root of an equation or equivalently the zero of a function.
- You would also be able to compare the various methods of obtaining the zero of a function.

3.0 Main Content

3.1 Introduction

You are probably quite familiar with the concept of the function of a continuous variable $f(x)$, continuous over a certain interval of the independent variable *x*. If we equate $f(x)$ to zero, we obtain the equation $f(x) = 0$. You might even see the process as that of equating two different functions $f_1(x)$ and $f_2(x)$, where the latter is identically zero.

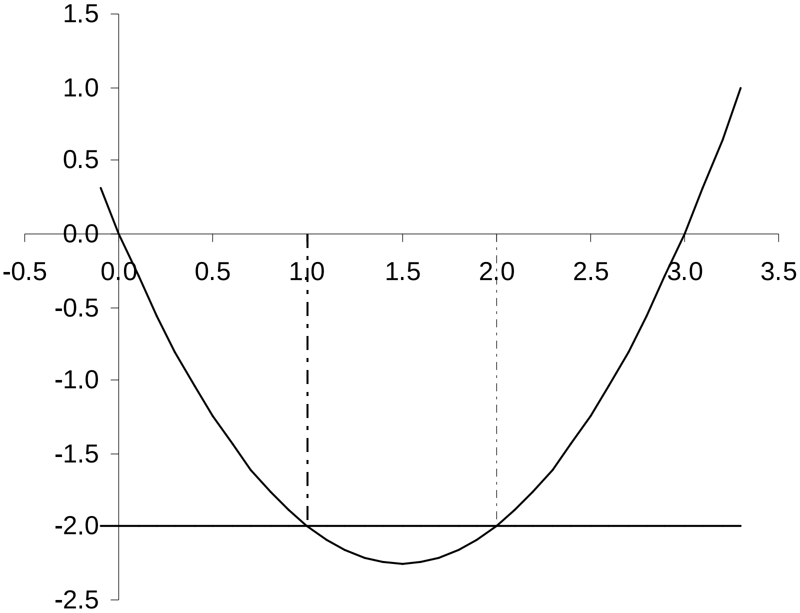


Fig. 4.1

Figure 4.1 shows the graph of $f_1(x) = x^2 - 3x$. The x-axis can be seen as the function $f_2(x) = 0$. Equating the two functions gives $f_1(x) = x^2 - 3x = 0 = f_2(x)$. The resulting equation, $x^2 - 3x = 0$, has two solutions $x = 0$ and 3 (the two solutions are indicated in Figure 4.1). Let us 'slide' $f_2(x)$ down to $f_2(x) = -2$, the lower horizontal line. The equation becomes $x^2 - 3x = -2$. This is perhaps one of the commonest quadratic equations you ever came across. The solutions are: 1.0 and 2.0. You can check this out on Fig. … as well. Shifting $f_2(x)$ lower to -2.5 would ensure that the resulting equation has no real solutions as the curve would not intersect the line.

The equations we have dealt with so far have been such that could easily be solved using analytical methods. It should be obvious to you that such equations should form a small subset of a much larger family of equations, the solutions of most of which do not readily lend themselves to analytical methods, especially as the power of the polynomial being equated to zero becomes large. Equating a polynomial to zero gives an algebraic equation. A transcendental function is a function that 'transcends' the normal laws of algebra as it cannot be expressed as a sequence of the algebraic operations of addition/subtraction, multiplication/division, an example being the square root of another function. Other examples include logarithmic, trigonometric, exponential functions and their inverses. If an equation involves the transcendental expressions, such as exponentials, trigonometric, logarithmic functions, the equation is said to be a transcendental equation.

We shall assume that the function whose roots we desire, $f(x)$, is a function of x, whose zeros (or the roots of the resulting equation) lie on the real axis. That is, the roots of the equation $f(x) = 0$ are real numbers. There are a number of methods of finding the roots. We shall now take some of these.

3.2 Bisection Method

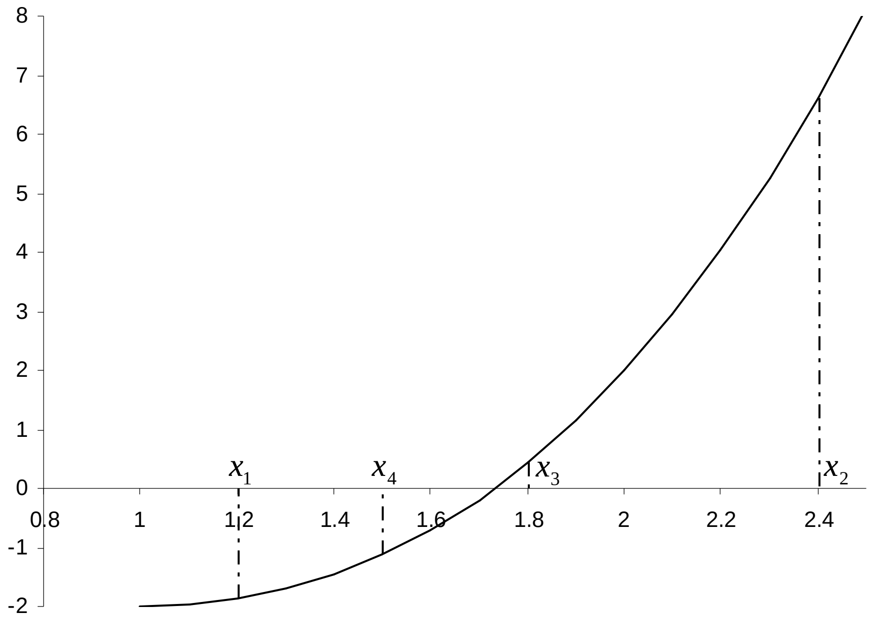


Fig.

As the name implies, we obtain the points x_1 and x_2, such that $f(x_2)f(x_1) < 0$, meaning that the value of f has opposite signs at the two points, which points to the fact that a root exists between x_1 and x_2. We approximate this root by the average of the two, i.e., $(x_1 + x_2)/2$. Let this be x_3. Then we evaluate $f(x_3)$. x_3 is then combined with x_1 or x_2, depending on the one at which the sign of the function is opposite that of $f(x_3)$. This gives x_4. This process is repeated until $f(x)$ attains the prescribed tolerance. We have illustrated this in Fig … for the root of the equation $x^3 - 3x = 0$, given that the root lies between x_1= 1.2 and x_2= 2.4. Then, $x_3 = (x_1 + x_2)/2$ = 1.8. $f(x_3) > 0$, so we combine it with x_1 to arrive at $x_4 = (x_1 + x_3)/2$, and so on.

The convergence of the Bisection method is slow and steady

3.2.1 Merits of the Bisection Method

1. As you can see, the root bisection method always converges. This is because you would get closer and closer to the root as the distance between the two points of interest is halved each time.
2. You can also keep a tab on the error. If the root lies between the points *a* and *b*, there will be a sequence:

$b_n - a_n = \frac{1}{2}(b_{n-1} - a_{n-1}) = \frac{1}{4}(b_{n-2} - a_{n-2}) = ... = \frac{1}{2^{n-1}}(b_1 - a_1)$. But you would recall that $b_1 = b$ and $a_1 = a$. Thus, $b_n - a_n = \frac{b-a}{2^{n-1}}$. On the other hand, we note that the first iteration point x_3 is at least as close to the root as half the interval $b_1 - a_1$, i.e., $|x_3 - x| \le \frac{b_1 - a_1}{2}$. Similarly, for the *n*th iteration $|x_n - x| \le \frac{b_n - a_n}{2}$. But $b_n - a_n = \frac{b-a}{2^{n-1}}$. Hence, $|x_n - x| \le \frac{1}{2}(b_n - a_n) \le \frac{1}{2}\frac{b-a}{2^{n-1}} = \frac{b-a}{2^n}$. We conclude that $|x_n - x| \le \frac{1}{2}\frac{b-a}{2^{n-1}} = \frac{b-a}{2^n}$, and this gives us an idea of the maximum error in our estimate of the root.

3.2.2 Demerits of the Bisection Method

1. The convergence is generally slow.
2. You might actually be approaching a singularity, for example, while dealing with functions that are not continuous between the two initial points. A classical example is the function $f(x) = \frac{1}{x}$, negative for $x < 0$ and positive for $x > 0$. As you start out with the bisection method with a point on the right of 0 and another on the left of 0, you are under the impression that there should be a root in-between. If the function is continuous between the initial guesses, this problem is eliminated.
3. The bisection method will not work if the function is tangential to the *x*-axis at the desired root. For example, $f(x) = x^2$ is tangential to the *x*-axis at the point $x = 0$ which is the root of the equation $x^2 = 0$. The function is positive on either side of $x = 0$, so you would not even try to get it in the first place, as the bisection method imposes the condition that the signs on either side be different.
4. If one of the initial points is close to the root, you would need many iterations to arrive at the root.
5. It does not work for repeated roots. If there are multiple roots within the interval given, the scheme narrows down on only one of the roots.
6. It does not work for repeated roots.

Example: Find a zero of the function $f(x) = 2x^3 - 3x^2 - 2x + 3$ between the points 1.4 and 1.7, using the bisection method. Take the tolerance to be $|x_{j+1} - x_j| \le 10^{-5}$.

Solution

$$f(1.4) = -0.192$$
$$f(1.7) = 0.756$$
$$x_3 = \frac{1.4 + 1.7}{2} = 1.55$$

$$f(1.55) = 1.4025 \times 10^{-1}$$
$$x_4 = \frac{1.55 + 1.4}{2} = 1.475$$
$$f(1.475) = -0.0588$$
$$x_5 = \frac{1.55 + 1.475}{2} = 1.5125$$

You can confirm that Table 4.1 is indeed true.

Table 4.1: Table for Bisection method

n	x	$f(x)$
1	1.55	0.14025
2	1.475	-5.88E-02
3	1.5125	3.22E-02
4	1.49375	-1.54E-02
5	1.503125	7.87E-03
6	1.498437	-3.89E-03
7	1.500781	1.96E-03
8	1.499609	-9.76E-04
9	1.500195	4.89E-04
10	1.499902	-2.44E-04
11	1.500049	1.22E-04
12	1.499976	-6.10E-05

3.3 Newton-Raphson Method

Consider Taylor Series

$$f(x + \Delta x) = f(x) + f'(x)\Delta x + \frac{f''(x)(\Delta x)^2}{2!} + \cdots \qquad 4.1$$

To a first order approximation, we can neglect second order and higher order terms. In that case, if $f(x + \Delta x) = 0$, then we truncate equation 4.1, leaving only the first two terms on the right. Then,

$$f(x + \Delta x) \approx f(x) + f'(x)\Delta x = 0 \qquad 4.2$$

or

$$\Delta x = \frac{-f(x)}{f'(x)}, \qquad 4.3$$

so that with an initial guess of x_0, we obtain a better approximation $x_0 + \Delta x$, i.e.,

$$x_1 = x_0 + \Delta x = x_0 - \frac{f(x_0)}{f'(x_0)} \qquad 4.4$$

It is quite clear that the function $f(x)$ must be differentiable for you to be able apply the Newton-Raphson method.

More generally,

$$x_{i+1} = x_i + \Delta x = x_i - \frac{f(x_i)}{f'(x_i)} \qquad 4.5$$

With an initial guess of x_0, we can then get a sequence x_1, x_2, …, which we expect to converge to the root of the equation.

We can rearrange equation 4.5 to obtain,

$$f'(x_i) = \frac{0 - f(x_i)}{x_{i+1} - x_i} \qquad 4.6$$

meaning that Newton-Raphson method is equivalent to taking the slope of the function $f(x)$ at the ith iterative point, and the next approximation is the point where the slope intersects the x axis. See the Fig 4.1:

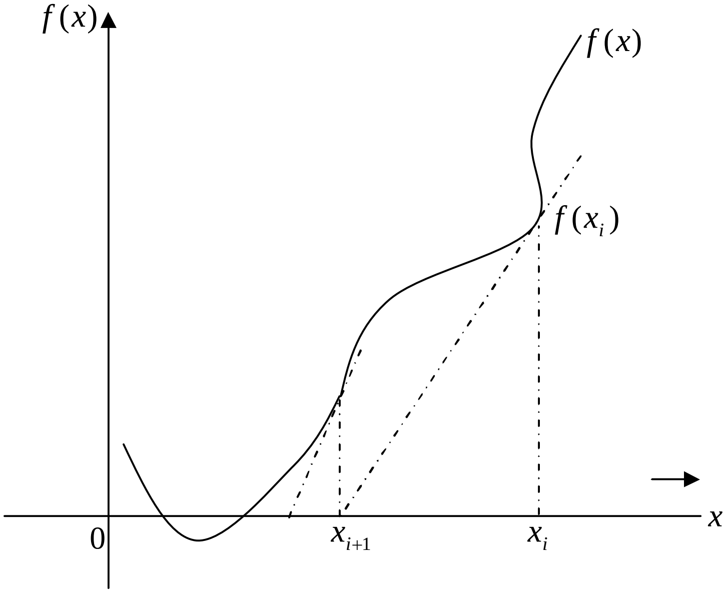


Fig. 4.1: Graph showing the gradient relationship of Newton-Raphson method

3.3.1 Merits of the Newton-Raphson Method

1. The Newton-Raphson method has a fast rate of convergence.
2. It can identify repeated roots, since it does not explicitly look for changes in the sign of $f(x)$.
3. It can find complex roots of polynomials if you started with a complex initial guess.

3.3.2 Demerits of the Newton-Raphson Method

1. It requires that we compute both $f(x)$ and $f'(x)$, which makes the scheme taxing.
2. Some functions might not be so easy to differentiate. In that case, it might be useful to take an approximate differential, $\frac{f(x+\Delta x)-f(x)}{\Delta x}$.
3. It is quite sensitive to initial condition and may diverge for the wrong choice of initial point.
4. It will not work if $f'(x)=0$. Also, if the differential is sufficiently close to zero, the sequence may diverges away from the root, or converge very slowly.
5. If the derivative changes signs at a test point, the sequence may oscillate around a point that may not even be the root.
6. It cannot detect repeated roots.

Example: Find the zeros of the function $f(x)=2x^3-3x^2-2x+3$ using the Newton-Raphson method, starting with x = 1.4. Take the tolerance to be $|x_{j+1}-x_j|\leq 10^{-5}$.

Solution

$$f(x)=2x^3-3x^2-2x+3$$

$$f'(x)=6x^2-6x-2$$

$$x_0=1.4$$

$$x_1=x_0-\frac{f(x_0)}{f'(x_0)}=\frac{x_0f'(x_0)-f(x_0)}{f'(x_0)}$$

$$=\frac{6x_0{}^3-6x_0{}^2-2x_0-2x_0{}^3+3x_0{}^2+2x_0-3}{6x_0{}^2-6x_0-2}$$

$$=\frac{4x_0{}^3-3x_0{}^2-3}{6x_0{}^2-6x_0-2}$$

$$=\frac{4(1.4)^3-3(1.4)^2-3}{6(1.4)^2-6(1.4)-2}$$

$$=1.5412$$

$x_1=1.5412$, $|x_1-x_0|=0.1412$

$x_2=1.5035$, $|x_2-x_1|=0.0377$

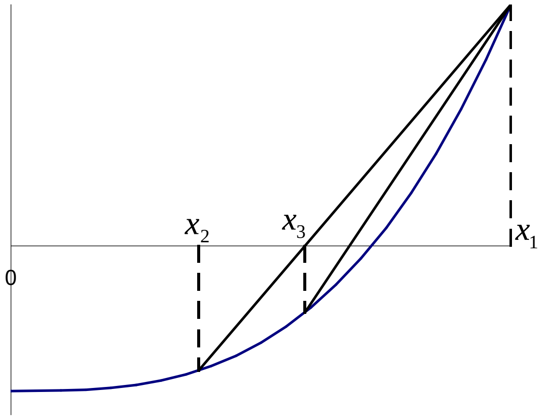
$x_3=1.5$, $|x_3-x_2|=0.0035$

$x_4=1.5$, $|x_4-x_3|=0$

3.4 Regula-falsi method

A regula-falsi or a method of false position assumes a test value for the solution of the equation.

You would recall that with the root-bisection method, we knew that a root existed between x_1 and x_2 if the function was smooth and $f(x_1)f(x_2) < 0$. Let us again choose these two points as in the case of root-bisection.



Then, for an arbitrary *x* and the corresponding *y*,

$$\frac{y - f(x_1)}{x - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1} \qquad 4.9$$

gives the equation of the chord joining the points $(x_1, f(x_1))$ and $(x_2, f(x_2))$.

Setting $y = 0$, that is, where the chord crosses the x-axis,

$$x_3 = x_1 - f(x_1)\frac{x_2 - x_1}{f(x_2) - f(x_1)} \qquad 4.10$$

Then, we evaluate $f(x_3)$. Just as in the case of root-bisection, if the sign is opposite that of $f(x_1)$, then a root lies in-between x_1 and x_3. Then, we replace x_2 by x_3 in equation 4.10. In just the same way, if the root lies between x_1 and x_3, we replace x_2 by x_1. We shall repeat this procedure until we are as close to the root as desired.

Example

Find the root of the equation $f(x) = 2x^3 - 3x^2 - 2x + 3$ between *x* = 1.4 and 1.7 by the regula-falsi method.

$f(1.4) = -0.192$, $f(1.7) = 0.756$

A solution lies between *x* = 1.4 and 1.7. Let $x_1 = 1.4$ and $x_2 = 1.7$. Then,

$$x_3 = x_1 - f(x_1)\frac{x_2 - x_1}{f(x_2) - f(x_1)} = 1.4 - (-0.192)\frac{1.7 - 1.4}{0.756 - (-0.192)}$$

= 1.4607595

$f(1.4607595) = -0.088983$

The root lies between 1.46076 and 1.7. Let $x_1 = 1.46076$ and $x_2 = 1.7$.

$$x_4 = x_1 - f(x_1)\frac{x_2 - x_1}{f(x_2) - f(x_1)} = 1.4607595 - (-0.088983)\frac{1.7 - 1.46076}{0.756 - (-0.088983)}$$

= 1.485953

Table 4.2 gives the remaining iterations.

Table 4.2: Table for Regula-falsi method

n	x	$f(x)$
1	1.460759	-0.088983
2	1.485953	-0.033938
3	1.495149	-0.011985
4	1.498346	-0.004118
5	1.499439	-0.001401
6	1.499810	-0.000475
7	1.499936	-0.000161
8	1.499978	-0.000055

3.5 Secant Method

In the case of the secant method, it is not necessary that the root lie between the two initial points. As such, the condition $f(x_1)f(x_2) < 0$ is not needed. Following the same analysis with the case of the regula-falsi method,

$$\frac{y - f(x_1)}{x - x_1} = \frac{f(x_2) - f(x_1)}{x_2 - x_1} \qquad 4.11$$

Setting $y = 0$ gives

$$x_3 = x_2 - f(x_2)\frac{x_2 - x_1}{f(x_2) - f(x_1)} \qquad 4.12$$

Thus, having found x_n, we can obtain x_{n+1} as,

$$x_{n+1} = x_n - f(x_n)\frac{x_n - x_{n-1}}{f(x_n) - f(x_{n-1})},\ n = 2, 3, \ldots \qquad 4.13$$

By inspection, if $f(x_n) - f(x_{n-1}) = 0$, the sequence does not converge, because the formula fails to work for x_{n+1}. The regula-falsi scheme does not have this problem as the associated sequence always converges.

Example

Find the root of the equation $f(x) = 2x^3 - 3x^2 - 2x + 3$ between x = 1.4 and 1.7 by the regula-falsi method.

$x_1 = 1.4,\ x_2 = 1.7$

$f(1.4) = -0.192$, $f(1.7) = 0.756$

A solution lies between x = 1.4 and 1.7. Let $x_1 = 1.4$ and $x_2 = 1.7$. Then,

$$x_3 = x_1 - f(x_1)\frac{x_2 - x_1}{f(x_2) - f(x_1)} = 1.4 - (-0.192)\frac{1.7 - 1.4}{0.756 - (-0.192)}$$

= 1.460759

$f(x_3) = -0.088983$

$$x_4 = x_3 - f(x_3)\frac{x_3 - x_2}{f(x_3) - f(x_2)} = 1.460759 - (-0.088983) \times \frac{1.460759 - 1.7}{-0.088983 - 0.756}$$

= 1.485953

You can continue with this scheme. Table 4.3 shows the other values obtained from the operation.

Table 4.3: Table for Secant Method

n	x	$f(x)$
1	1.460759	-0.088983
2	1.485953	-0.033938
3	1.501487	0.003730
4	1.499949	-0.000129
5	1.500000	0.000000

4.0 Conclusion

In this Unit, you learnt to find the zeros of an algebraic or transcendental function. We explored a number of methods, and outlined their merits and demerits. We were also able to estimate the maximum error in the bisection method.

5.0 Summary

In this Unit, you learnt:

- to find the zeros of an algebraic or transcendental function using several methods.
- the merits and demerits of the methods.
- to the maximum error that can be incurred in using the bisection method.

6.0 Tutor Marked Assignment

1. Find the upper bound of the error you are likely to incur in using the bisection method in finding the root of an equation if the two starting points are 1.4 and 2.5 and you needed 8 steps to achieve the required tolerance.
2. Find a root of the equation $2x^3 - 3x^2 - 2x - 0.5$ using the following methods (tolerance …..):
   (i) Root bisection [starting points1.9 and 2.1 (tolerance $|f(x)| \le 0.001$)].
   (ii) Newton-Raphson starting point 2.0
   (iii) Regula-falsi [starting points1.9 and 2.1].
   (iv) Secant [starting points1.9 and 2.1].

3. Find a root of the equation $x - 2\sin x$ using

   (i) The bisection method, given that the root is between 1.5 and 3, with tolerance $| f(x) | \le 0.02$.

   (ii) Newton-Raphson method, with the starting point 1.35, with tolerance $| f(x) | \le 10^{-6}$.

   (iii) Regula-falsi [starting points1.5 and 3.0].

   (iv) Secant [starting points1.5 and 3.0].

7.0 References/Further Reading

Solutions to Tutor Marked Assignment

1. Find the upper bound of the error you are likely to incur in using the bisection method in finding the root of an equation if the two starting points are 1.4 and 2.5 and you needed 8 steps to achieve the required tolerance.

$$| x_n - x |\le \frac{2.5-1.4}{2^8} = 4.297\times10^{-3}$$

2. Find a root of the equation $2x^3 - 3x^2 - 2x - 0.5$ using the following methods (tolerance …..):

(v) Root bisection [starting points1.9 and 2.1 (tolerance $| f(x) |\le 0.001$)].

Iteration No.	x_i	$f(x_i)$
1	2	-0.5
2	2.05	2.27E-02
3	2.025	-0.24434
4	2.0375	-0.11224
5	2.04375	-4.51E-02
6	2.046875	-1.13E-02
7	2.048437	5.72E-03
8	2.047656	-2.78E-03
9	2.048047	1.47E-03
10	2.047851	-6.58E-04

(vi) Newton-Raphson starting point 2.0

Iteration No.	x_i	$f(x_i)$
1	2.05	0.05
2	2.04792	0.002084
3	2.04791	3.81E-06

(vii) Regula-falsi [starting points1.9 and 2.1].

Iteration No.	x_i	$f(x_i)$
1	2.040918	-0.075613
2	2.047610	-0.003287
3	2.047899	-0.000142

(viii) Secant [starting points1.9 and 2.1].

Iteration No.	x_i	$f(x_i)$
1	2.568354	8.457968
2	1.912709	-1.30566
3	2.000387	-0.49613
4	2.054121	6.79E-02
5	2.047653	-2.81E-03
6	2.047911	-1.49E-05

2. Find a root of the equation $x - 2\sin x$ using

(v) The bisection method, given that the root is between 1.5 and 3, with tolerance $|f(x)| \leq 0.02$.

Iteration No.	x_i	$f(x_i)$
1	2.25	0.69385361
2	1.875	-0.0331716
3	2.0625	0.29944043
4	1.96875	0.12503812
5	1.921875	0.04387042
6	1.8984375	0.00482936
7	1.886719	-0.0143012

(vi) Newton-Raphson method, with the starting point 1.35, with tolerance $|f(x)| \leq 10^{-6}$.

Iteration No.	x_i	$f(x_i)$
1	2.420215	1.099376
2	1.980780	0.146526
3	1.899250	0.006165
4	1.895502	0.000013
5	1.895494	0.000000

(vii) Regula-falsi [starting points1.5 and 3.0].

Iteration No.	x_i	$f(x_i)$
1	1.731106	-0.243250
2	1.835347	-0.095074
3	1.874712	-0.033632

4	1.888467	-0.011464
5	1.893136	-0.003858
6	1.894705	-0.001292
7	1.895230	-0.000432

(viii) Secant [starting points1.5 and 3.0].

Iteration No.	x_i	$f(x_i)$
1	1.731106	-0.243250
2	1.835347	-0.095074
3	1.902230	0.011077
4	1.895251	-0.000399
5	1.895493	-0.000002

UNIT 5: FINITE DIFFERENCES AND INTERPOLATION

1.0 Introduction

Given the function $f(x)$ we can evaluate the values of f at different x, thereby representing a continuous function with a set of discrete data. On the other hand, it could be that we have a set of data and we would like to see if they could have been got from a polynomial or if indeed we could represent the points by a polynomial. Finite differences would help us in this regard. With the aid of finite differences, we shall then derive Newton’s forward and Newton’s backward interpolation formulas.

2.0 Objectives

At the end of this Unit, you would be able to:

- Deduce a polynomial from its difference table.
- Derive Newton’s forward and Newton’s backward interpolation formulas.
- Fit a polynomial to a given a set of data
- Interpolate and extrapolate with Newton’s forward difference
- Interpolate and extrapolate with Newton’s backward difference

3.0 Main Content

3.1 Finite Differences

We proceed by defining the finite difference

i. First Forward difference:

$$f_{i+1} - f_i = \Delta f_i \tag{5.1}$$

ii. First Backward difference:

$$f_{i-1} - f_i = \nabla f_i \tag{5.2}$$

iii. First Central difference:

$$\frac{\left(f_{i+1} - f_{i-1}\right)}{2} = \delta_{i/2} \tag{5.3}$$

The table for forward difference would look like Table 5.1. What do you notice about this table? You can see that y_0 and the differences related to it appear on the first line slanting down to the right.

Table 5.1: Forward difference Table

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$
x_0	y_0			
		Δy_0		
x_1	y_1		$\Delta^2 y_0$	
		Δy_1		$\Delta^3 y_0$
x_2	y_2		$\Delta^2 y_1$	
		Δy_2		$\Delta^3 y_1$
x_3	y_3		$\Delta^2 y_2$	
		Δy_3		
x_4	y_4			

You can see that differences with similar subscripts form a line slanting downward to the right from the top.

Table 5.2 is the backward difference table.

Table 5.2: Backward difference table

x	y	∇y	$\nabla^2 y$	$\nabla^3 y$
x_0	y_0			
		∇y_1		
x_1	y_1		$\nabla^2 y_2$	
		∇y_2		$\nabla^3 y_3$
x_2	y_2		$\nabla^2 y_3$	
		∇y_3		$\nabla^3 y_4$
x_3	y_3		$\nabla^2 y_4$	
		∇y_4		
x_4	y_4			

Can you spot what makes this table unique? Differences with similar subscripts form a line slanting upward to the right from the bottom.

Note that for forward difference, $\Delta^2 y_0 = \Delta y_1 - \Delta y_0$, or generally,

$$\Delta^2 y_n = \Delta y_{n+1} - \Delta y_n \qquad 5.4$$

and for backward difference,

$$\nabla^2 y_n = \nabla y_n - \nabla y_{n-1} \tag{5.5}$$

Of course, we can also get a table for central differences, Table 5.3.

Table 5.3: Central difference table

x	y	$\delta\, y$	$\delta^{\,2} y$	$\delta^{\,3} y$
x_0	y_0			
		$\delta\, y_{1/2}$		
x_1	y_1		$\Delta^2 y_1$	
		$\delta\, y_{3/2}$		$\Delta^3 y_{3/2}$
x_2	y_2		$\Delta^2 y_2$	
		$\delta\, y_{5/2}$		$\Delta^3 y_{5/2}$
x_3	y_3		$\Delta^2 y_3$	
		$\delta\, y_{7/2}$		
x_4	y_4			

Do you notice that like subscripts appear on the same row.

3.1.1 Forward Differences

Suppose the given function is $f(x) = x^2 + 2x + 3$, then we can evaluate f at $x = 0, 1, 2, \cdots, 6$, and then with the aid of forward difference, arrive at Table 5.4:

Table 5.4: Forward difference table for $y = x^2 + 2x + 3$

x	y	Δy	$\Delta^2 y$
0	3		
		3	
1	6		2
		5	
2	11		2
		7	
3	18		2
		9	
4	27		2
		11	
5	38		2
		13	
6	51		

The second forward difference produces a constant value of 2.

A similar operation carried out on the function $f(x) = 2x$ will produce a constant difference after only one forward difference.

It follows that the number of forward differences needed to achieve a constant value of difference is the degree of the polynomial, and the constant value in the second forward difference is the second differential of the function.

Hence,

$$\frac{d^2 f}{dx^2} = 2$$

Integrating,

$$\frac{df}{dx} = 2x + c_1$$

and finally,

$$f(x) = x^2 + c_1 x + c_2$$

The values of the constants c_1 and c_2 will be determined from the values of f at different values of x.

$$f(0) = c_2 = 3$$

$$f(1) = 1 + c_1 + 3 = 4 + c_1 = 6$$

Thus, $c_1 = 2$.

The function, therefore, is

$$f(x) = x^2 + 2x + 3.$$

This was the same function we started with. Of course, if what we started with was just the table, we could then have obtained the polynomial the way we did.

We could extrapolate for values of x not given on the table, such as for x = -2.0 or 7.0 or interpolate for values such as x = 3.5 and 4.2.

3.1.2 Error in Finite Difference Table

Consider Table 5.5 for forward difference table into which we have introduced an error ε through x_4.

Table 5.5: Forward difference table with error

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$
x_0	y_0				
		Δy_0			
x_1	y_1		$\Delta^2 y_0$		
		Δy_1		$\Delta^3 y_0$	
x_2	y_2		$\Delta^2 y_1$		$\Delta^4 y_0 + \varepsilon$
		Δy_2		$\Delta^3 y_1 + \varepsilon$	
x_3	y_3		$\Delta^2 y_2 + \varepsilon$		$\Delta^4 y_1 - 4\varepsilon$
		$\Delta y_3 + \varepsilon$		$\Delta^3 y_2 - 3\varepsilon$	
x_4	$y_4 + \varepsilon$		$\Delta^2 y_3 - 2\varepsilon$		$\Delta^3 y_2 + 6\varepsilon$
		$\Delta y_4 - \varepsilon$		$\Delta^3 y_3 + 3\varepsilon$	
x_5	y_5		$\Delta^2 y_4 + \varepsilon$		$\Delta^3 y_3 - 4\varepsilon$
		Δy_5		$\Delta^3 y_4 - \varepsilon$	
x_6	y_6		$\Delta^2 y_5$		$\Delta^3 y_4 + \varepsilon$
		Δy_6		$\Delta^3 y_5$	
x_7	y_7		$\Delta^2 y_6$		
		Δy_7			
x_8	y_8				

The higher the degree of the difference, the more the error involved. Moreover, you would notice that the error terms are the binomial coefficient of $(1-\varepsilon)^n$, where n is the order of the difference. Thus, for degree 1, it is $-\varepsilon$. For degree 2, it is $(1-\varepsilon)^2 = 1-2\varepsilon+\varepsilon^2$. For $(1-\varepsilon)^3 = 1-3\varepsilon+3\varepsilon^2+\varepsilon^3$. But can you notice one thing? The errors in each difference column cancel out. You shall need this property later.

Example
Find the wrong entry in the following table, given that they represent a cubic polynomial.

x	0	1	2	3	4	5	6	7	8
y	-2	4	34	106	238	448	754	1174	1726

Solution
The forward difference table is as shown below on the left part of Table 5.6. The right part of the table would have resulted if there had been no error.

Table 5.6: Forward difference table with error (a) and without (b)

(a)
From the given Table

(b)
What the table would have looked like had there been no error

0	-2					0	-2			
		6						6		
1	4		24			1	4		24	
		30		18				30		18
2	34		42			2	34		42	
		72		20				72		18
3	106		62			3	106		60	
		134		12				132		18
4	240		74			4	238		78	
		208		24				210		18
5	448		98			5	448		96	
		306		16				306		18
6	754		114			6	754		114	
		420		18				420		18
7	1174		132			7	1174		132	
		552						552		
8	1726					8	1726			

We recall that the third difference should have been a constant. This constant we can determine by remembering that you were told the sum of errors in a single difference column cancel out. Thus, the sum of the entries in the column representing the third forward difference remains the same as it would have been had there been no error. This sum is 108. We divide this by 6 to arrive at 18. Each entry in that column should have been 18. We notice that the shaded entries in the table can be traced backwards to the entry 240 in the values of y. This is the entry in error. Moreover,

$y_5 + \varepsilon = 240$, $\Delta^3 y_1 + \varepsilon = 20$, $\Delta^3 y_2 - 3\varepsilon = 12$. But $\Delta^3 y_1 = \Delta^3 y_2$, implying that

$$20 - \varepsilon = 12 + 3\varepsilon$$

Solving for ε, $4\varepsilon = 8$ and $\varepsilon = 2$. Thus, $y_5 + 2 = 240$, giving $y_5 = 238$. You can now see that the table on the left of Table … should have been the correct table if there had been no error.

3.2 Interpolation

3.2.1 Newton forward interpolation formula

At times, we would like to represent a set of values (x_i, y_i) with a function, enabling us, among other things, to be able to interpolate or extrapolate values that are not in the given set.

Let the interpolating function be a polynomial given by $y(x)$. Then, we can write the polynomial as,

$$y_n(x) = a_0 + a_1(x - x_0) + a_2(x - x_0)(x - x_1) + a_3(x - x_0)(x - x_1)(x - x_2) + \dots + a_n(x - x_0)(x - x_1)\dots(x - x_{n-1}) \qquad 5.6$$

$y_n(x)$ must be equal to the tabulated values of y. Thus, we require that:

$$y_0(\,y \text{ at } x = x_0\,) = a_0 \qquad 5.7$$

$$y_1(\,y \text{ at } x = x_1) = a_0 + a_1(x_1 - x_0)$$

which implies

$$a_1 = \frac{y_1 - a_0}{x_1 - x_0} = \frac{y_1 - y_0}{x_1 - x_0} = \frac{\Delta y_0}{h} \qquad 5.8$$

$$\begin{aligned} y_2\,(\,y \text{ at } x = x_2\,) &= a_0 + a_1(x_2 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) \\ &= a_0 + a_1(x_2 - x_1 + x_1 - x_0) + a_2(x_2 - x_0)(x_2 - x_1) \\ &= a_0 + a_1(x_1 - x_0) + a_1(x_2 - x_1) + 2h^2 a_2 \ \text{(since } x_2 - x_0 = 2h\text{)} \\ &= y_1 + \frac{\Delta y_0}{h}h + 2h^2 a_2 \end{aligned}$$

$$y_2 - y_1 = \Delta y_1 = \Delta y_0 + 2h^2 a_2$$

from which

$$a_2 = \frac{\Delta y_1 - \Delta y_0}{2h^2} = \frac{\Delta^2 y_0}{2!h^2} \qquad 5.9$$

Similarly,

$$a_3 = \frac{\Delta^2 y_0}{3!h^3} \qquad 5.10$$

Putting these values in equation 5.6 gives

$$y(x) = y_0 + \frac{\Delta y_0}{h}(x - x_0) + \frac{\Delta^2 y_0}{2!h^2}(x - x_0)(x - x_1) + \frac{\Delta^3 y_0}{3!h^2}(x - x_0)(x - x_1)(x - x_2) + \dots \qquad 5.11$$

Now, let $x = x_0 + rh$. Then,

$$x - x_0 = rh,\ x - x_1 = x - x_0 + x_0 - x_1 = rh - h = (r-1)h$$

$$x - x_2 = x - x_1 + x_1 - x_2 = (r-1)h - h = (r-2)h$$

Hence, from equation 5.11,

$$y(x) = y(x_0 + rh) = y_0 + r\Delta y_0 + \frac{r(r-1)}{2!}\Delta^2 y_0 + \frac{r(r-1)(r-2)}{3!}\Delta^3 y_0 + ...$$

$$+ \ldots \frac{r(r-1)...(r-(n-1))}{n!}\Delta^n y_0 + ... \qquad 5.12$$

This is Newton's forward interpolation formula.

Note: Newton's forward interpolation formula is for

(i) interpolating the values of y near the beginning of a set of tabulated values, and

(ii) extrapolating values of y a little to the left of y_0

3.2.2 Newton's Backward Interpolation Formula

Let us choose $y_n(x)$ in the form,

$$y_n(x) = a_0 + a_1(x - x_n) + a_2(x - x_n)(x - x_{n-1})$$

$$+ ... + a_n(x - x_n)(x - x_{n-1})...(x - x_1) \qquad 5.13$$

$y_n(x)$ must be equal to the tabulated values of y. Thus, we require that:

$$y_n \;\; (y \text{ at } x = x_n) = a_0 \qquad 5.14$$

$$y_{n-1}(y \text{ at } x = x_{n-1}) \quad = a_0 + a_1(x_{n-1} - x_n)$$

$\Rightarrow$

$$a_1 = \frac{y_{n-1} - a_0}{x_{n-1} - x_n} = \frac{y_{n-1} - y_n}{x_{n-1} - x_n} = \frac{\nabla y_0}{h} \qquad 5.15$$

$$y_{n-2}(y \text{ at } x = x_{n-2}) \quad = a_0 + a_1(x_{n-2} - x_n) + a_2(x_{n-2} - x_n)(x_{n-2} - x_{n-1})$$

$$= a_0 + a_1(-2h) + a_2(-2h)(-h)$$

$$y_{n-2} \quad = y_n - 2h\frac{\nabla y_n}{h} + 2h^2 a_2$$

$$= y_n - 2(y_n - y_{n-1}) + 2h^2 a_2$$

$$= 2y_{n-1} - y_n + 2h^2 a_2$$

We can then write

$$a_2 = \frac{1}{2h^2}(y_{n-2} + y_n - 2y_{n-1})$$

But $\nabla^2 y_n = \nabla y_n - \nabla y_{n-1} = (y_n - y_{n-1}) - (y_{n-1} - y_{n-2}) = y_{n-2} + y_n - 2y_{n-1}$

Hence,

$$a_2 = \frac{1}{2h^2}\nabla^2 y_n \qquad 5.16$$

Similarly,

$$a_3 = \frac{1}{3\,h^3}\nabla^3 y_n \qquad 5.17$$

$$a_n = \frac{1}{n h^n} \nabla^n y_n \qquad 5.18$$

Putting these values in equation 5.13 yields,

$$y_n(x) = y_n + \frac{(x-x_n)}{h}\nabla y_n + \frac{(x-x_n)(x-x_{n-1})}{2h^2}\nabla^2 y_n + \ldots$$

$$+\ldots+\frac{(x-x_m)(x-x_{m-1})\ldots(x-x_1)}{mh^m}\nabla^m y_m + \ldots$$

Setting $x = x_n + rh$, $x - x_n = rh$, $x - x_{n-1} = x - x_n + x_n - x_{n-1} = rh + h = (r+1)h$. Similarly, $x - x_{n-2} = x - x_{n-1} + x_{n-1} - x_{n-2} =$ $(r+1)h + h = (r+2)h$. Thus, $x - x_1 = [r + (n-1)]h$.

$$y(x) = y(x_0 + rh) = y_n + r\nabla y_n + \frac{r(r+1)}{2!}\nabla^2 y_n + \frac{r(r+1)(r+2)}{3!}\nabla^3 y_n + \ldots$$

$$+ \ldots \ \frac{r(r+1)\ldots(r+(n-1))}{n!}\nabla^n y_n + \ldots$$

This is the Newton's backward interpolation formula.

Note: Newton's backward interpolation formula is for

(i) interpolating the values of *y* near the end of a set of tabulated values, and

(ii) extrapolating values of *y* a little to the right of y_n.

Example

Find the cubic polynomial that fits the following table.

x	1	2	3	4
y	3	9	27	63

Solution

The forward difference table gives:

x	*y*	Δ	Δ^2	Δ^3
1	3			
		6		
2	9		12	
		18		6
3	27		18	
		36		
4	63			

The step-size, *h*, is 1. Let $x = x_0 + rh$, with $x_0 = 1$.

$$y(x) = y(x_0 + rh) = y_0 + r\Delta y_0 + \frac{r(r-1)}{2!}\Delta^2 y_0 + \frac{r(r-1)(r-2)}{3!}\Delta^3 y_0 + ...$$
$$+ \dots \frac{r(r-1)...(r-(n-1))}{n!}\Delta^n y_0 + ...$$

$r = x - x_0 = \; x - 1$.

Then,

$$y(x) = 3 + (x-1)\times 6 + \frac{(x-1)(x-2)}{2!}\times 12 + \frac{(x-1)(x-2)(x-3)}{3!}\times 6 + ...$$
$$= \; 3 + 6x - 6 + 6(x^2 - 3x + 2) + (x^2 - 3x + 2)(x-3) + ...$$
$$= \; 6x - 3 + 6x^2 - 18x + 12 + x^3 - 3x^2 + 2x - 3x^2 + 9x - 6 + ...$$
$$= \; x^3 - x + 3$$

Check: Find the value of y when $x = 3$:

$$y(3) = 3^3 - 3 + 3 = 27$$

You could also get the value of y when x is 0.95, being a little to the left of $x_0 = 1$.

$$y(3) = .95^3 - .95 + 3 = 2.907375$$

Let us solve the same problem with Newton's backward formula.

x	y	∇y	$\nabla^2 y$	$\nabla^3 y$
1	3			
		-6		
2	9		12	
		-18		-6
3	27		18	
		-36		
4	63			

$r = x - x_n \; = \; x - 4$, since $h = 1$.

$$y(x) = y_n + r\nabla y_n + \frac{r(r+1)}{2!}\nabla^2 y_n + \frac{r(r+1)(r+2)}{3!}\nabla^3 y_n + ...$$
$$+ \dots \frac{r(r+1)...(r+(n-1))}{n!}\nabla^n y_n + ...$$
$$= \; 63 + (x-4)\times 36 + \frac{(x-4)(x-3)}{2}\times 18 + \frac{(x-4)(x-3)(x-2)}{6}\times 6 + ...$$
$$= \; 63 + 36x - 144 + 9\times(x^2 - 7x + 12) + (x-4)(x^2 - 5x + 6)$$
$$= \; 63 + 36x - 144 + 9x^2 - 63x + 108 + x^3 - 5x^2 + 6x - 4x^2 + 20x - 24$$

$= x^3 - x + 3$

Check: Find the value of y when x = 2:

$$y(3) = 2^3 - 2 + 3 = 9$$

You could also have found $y(3.9)$, x = 3.9 being a point to the left of x = 4:

$$y(3.9) = (3.9)^3 - 3.9 + 3 = 58.419$$

4.0 Conclusion

In this Unit, you have learnt how to carry out the three different difference schemes. You have also learnt how to deduce a polynomial from tabulated data. Moreover, you can now detect what and where an error has been introduced into a difference table. You also derived Newton's forward and backward interpolation formulas. From the interpolation formulas, you were able get interpolating functions.

5.0 Summary

In this Unit, you leant to do the following:

- Carry out any of the three difference schemes.
- Derive the polynomial that fits a set of tabulated data.
- Derive Newton's forward interpolation formula.
- Derive Newton's backward interpolation formula.
- With the aid of Newton's forward or backward formula, obtain a function that takes the values in a set of tabulated data.

6.0 Tutor Marked Assignment

1. Carry out the forward, backward, and the central difference schemes on the set of data provided below:

1	2	3	4	5	6	7
1	12	47	118	237	416	667

2. Starting with the function $8x^3 - 8x^2 - 2x - 12$, draw up a difference table. Deduce the equation that fits the data, starting from the table alone.

3. We have deliberately inserted an error in the data in the table below. If the data represents a cubic polynomial, find which of the entries is in error.

0	1	2	3	4	5	6	7	8	9
-12	-14	16	126	366	778	1416	2326	3556	5154

4. Find the quartic polynomial that fits the following table.
   (i) Using the Newton's forward interpolation formula.
   (ii) Using the Newton's backward interpolation formula.

7.0 References/Further Reading

Solutions to Tutor Marked Assignment

1. Carry out the forward, backward, and the central difference schemes on the set of data provided below:

1	2	3	4	5	6	7
1	12	47	118	237	416	667

Solution

Forward difference:

1	1			
		11		
2	12		24	
		35		12
3	47		36	
		71		12
4	118		48	
		119		12
5	237		60	
		179		12
6	416		72	
		251		
7	667			

Backward difference:

1	1			
		-11		
2	12		24	
		-35		-12
3	47		36	
		-71		-12
4	118		48	
		-119		-12
5	237		60	
		-179		-12
6	416		72	
		-251		
7	667			

2. Starting with the function $8x^3 - 8x^2 - 2x - 12$, draw up a difference table. Deduce the equation that fits the data, starting from the table alone.

Solution

0	-12	
		-2

1	-14		32	
		30		48
2	16		80	
		110		48
3	126		128	
		238		48
4	364		176	
		414		
5	778			

The degree of the polynomial is 3.

$$\frac{d^3y}{dx^3} = 48 = y''$$

Hence, $y = 8x^3 + c\frac{x^2}{2} + dx + e$

Substituting in turn three different values of x yields c, d, e, respectively –16, –2 and –12.

The polynomial is then $y = 8x^3 - 8x^2 - 2x - 12$.

3. We have deliberately inserted an error in the data in the table below. If the data represents a cubic polynomial, find which of the entries is in error.

0	1	2	3	4	5	6	7	8	9
-12	-14	16	126	366	778	1416	2326	3556	5154

Solution

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$
0	-12			
		-2		
1	-14		32	
		30		48
2	16		80	
		110		50
3	126		130	
		240		42
4	366		172	
		412		54
5	778		226	
		638		46
6	1416		272	
		910		48
7	2326		320	
		1230		48
8	3556		368	
		1598		
9	5154			

The third difference should have been a constant. The sum of errors in a single difference column cancel out. Thus, the sum of the entries in the column representing the third forward difference remains the same as it would have been had there been no error. This sum is 336. We divide this by 7 to arrive at 48. Each entry in that column should have been 48. We notice that the shaded entries in the table can be traced backwards to the entry 336 in the values of y. This is the entry in error. Moreover,

$y_5 + \varepsilon = 366$, $\Delta^3 y_1 + \varepsilon = 50$, $\Delta^3 y_2 - 3\varepsilon = 42$. But $\Delta^3 y_1 = \Delta^3 y_2$, implying that

$$50 - \varepsilon = 42 + 3\varepsilon$$

Solving for ε, $4\varepsilon = 8$ and $\varepsilon = 2$. Thus, $y_5 + 2 = 366$, giving $y_5 = 364$. You can now see that the table below should have been the correct table if there had been no error.

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$
0	-12			
		-2		
1	-14		32	
		30		48
2	16		80	
		110		48
3	126		128	

		238		48
4	364		176	
		414		48
5	778		224	
		638		48
6	1416		272	
		910		48
7	2326		320	
		1230		48
8	3556		368	
		1598		
9	5154			

4. Find the quartic polynomial that fits the following table.

(iii) Using the Newton's forward interpolation formula.

(iv) Using the Newton's backward interpolation formula.

x	0	2	4	6	8
y	8	17	230	1230	3972

Solution

The forward difference table gives:

x	y	Δy	$\Delta^2 y$	$\Delta^3 y$	$\Delta^4 y$
0	8				
		9			
2	17		204		
		213		583	
4	230		787		372
		1000		955	
6	1230		1742		
		2742			
8	3972				

The step-size, *h*, is 2. Let $x = x_0 + rh$, with

$x_0 = 0$, *h* = 2. Hence, $r = (x - x_0)/h = \dfrac{x}{2}$

$$y(x) = y(x_0 + rh) = y_0 + r\Delta y_0 + \frac{r(r-1)}{2!}\Delta^2 y_0 + \frac{r(r-1)(r-2)}{3!}\Delta^3 y_0 + ...$$

$$+ \ldots \ \frac{r(r-1)...(r-(n-1))}{n!}\Delta^n y_0 + ...$$

Then,

$$y(x)=8+\frac{x}{2}\times 9+\frac{\frac{x}{2}(\frac{x}{2}-1)}{2!}\times 204+\frac{\frac{x}{2}(\frac{x}{2}-1)(\frac{x}{2}-2)}{3!}\times 583$$

$$+\frac{\frac{x}{2}(\frac{x}{2}-1)(\frac{x}{2}-2)(\frac{x}{2}-3)}{4!}\times 372+\ldots$$

Simplifying this expression,

$$y(x)=\frac{31}{32}x^4+\frac{25}{48}x^3-\frac{19}{4}x^2+\frac{25}{6}x+8$$

Solving the same problem with Newton's backward formula.

x	y	∇y	$\nabla^2 y$	$\nabla^3 y$	$\nabla^4 y$
0	8				
		-9			
2	17		204		
		-213		-583	
4	230		787		372
		-1000		-955	
6	1230		1742		
		-2742			
8	3972				

$r\ =\ \frac{x-x_n}{h}=\frac{x-8}{2}=\frac{x}{2}-4$, since $h=2$.

$$y(x)=y_n+r\nabla y_n+\frac{r(r+1)}{2!}\nabla^2 y_n+\frac{r(r+1)(r+2)}{3!}\nabla^3 y_n+\ldots$$

$$+\ \ldots\ \frac{r(r+1)\ldots(r+(n-1))}{n!}\nabla^n y_n+\ldots$$

$$=\ 3972+(\frac{x}{2}-4)\times 2742+\frac{(\frac{x}{2}-4)(\frac{x}{2}-3)}{2}\times 1742+\frac{(\frac{x}{2}-4)(\frac{x}{2}-3)(\frac{x}{2}-2)}{6}\times 955+$$

$$+\frac{(\frac{x}{2}-4)(\frac{x}{2}-3)(\frac{x}{2}-2)(\frac{x}{2}-1)}{6}\times 372+\ldots$$

Simplifying, we yet again arrive at

$$y(x)=\frac{31}{32}x^4+\frac{25}{48}x^3-\frac{19}{4}x^2+\frac{25}{6}x+8$$

Unit 6: Numerical Integration

1.0 Introduction

No doubt, before you could get to this stage of your studies, you integrated quite a number of function analytically. Perhaps you were told at the onset that the process of analytical integration arose from discretising the function, that is, ‘slicing’ up the function into vertical bars as shown in Fig. 5.1 and then adding up the areas of the bars in the limit as the slivers become infinitesimally narrow. Numerical integration goes back to this idea, and represents a continuous function by a discrete set of points as this is the way the program compiler can handle data. Numerical integration is called quadrature when the function is a function of a single variable. In this unit, you shall learn several methods of integrating a function numerically.

2.0 Objectives

At the end of this Unit, you should be able to:

- Numerically integrate a given function of a single variable between a given set of limits.
- Know the merits and demerits of various numerical integration schemes.
- Deduce the error involved in approximating an analytical integral with a numerical integral.

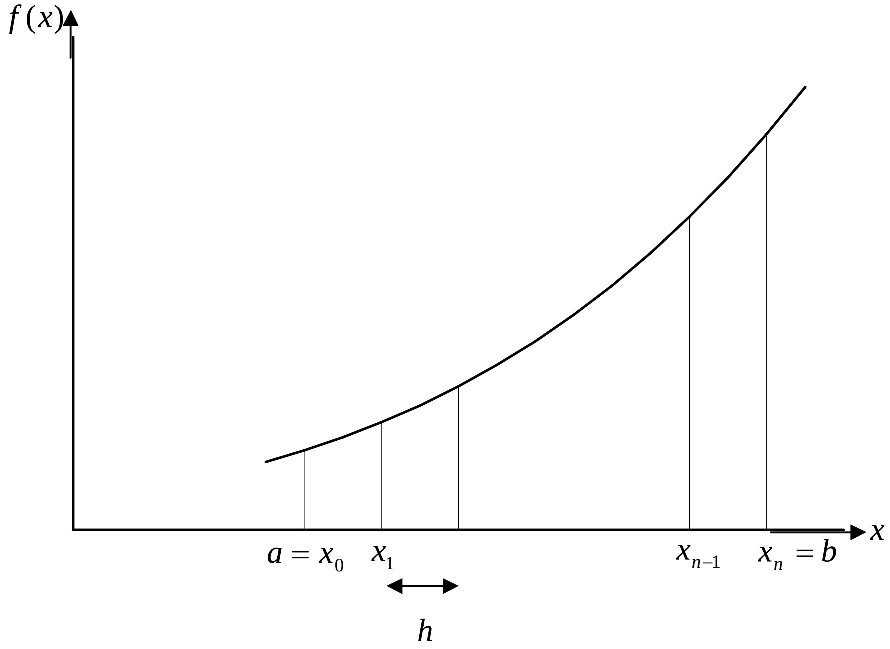


Fig. 5.1: Discretisation of the interval of integration

3.1 The Newton-Coates Quadrature Formula

As you can see, numerical integration as the process of finding the value of a definite integral,

$$I = \int_a^b f(x)\,dx \qquad 6.1$$

with $a \le x \le b$ (Fig. 5.1). An approximate value of the integral is obtained by replacing the function by an interpolating polynomial. Thus, different formulas for numerical integration would result for different interpolating formulas. In our own case, we shall be making use of take Newton's forward difference formula.

We shall divide the interval $[a,b]$ into n equal subintervals: $a = x_0 < x_1 < ... < x_n = b$, such that $x_{j+1} = x_j + h$, where the interval, $h = (b-a)/h$. Hence, we can write $x_1 = x_0 + h$, $x_2 = x_1 + h = (x_0 + h) + h = x_0 + 2h$. It follows therefore, that $x_r = x_0 + rh$. The integral becomes,

$$I = \int_{x_0}^{x_n} f(x)\,dx \qquad 6.2$$

We can write $x = x_0 + qh$ and $dx = hdq$. $x_n = x_0 + nh$.

Let us make a change of variable from x to q: $x = x_0 + qh$. Then, $q = (x - x_0)/h$. It follows, therefore, that when $x = x_0$, $q = 0$; when $x = x_n = x_0 + nh$, $q = (x_n - x_0)/h = nh/h = n$.

The integral becomes

$$I=\int_{x_0}^{x_0+nh} f(x_0+qh)(hdq)=h\int_0^n f(x_0+qh)\,dq \qquad 6.3$$

Let us approximate $f(x)=f(x_0+q\,h)$ by the Newton's forward difference formula. Then, from equation 6.3, and setting $y=f(x)$,

$$I=h\int_0^n\left[y_0+q\Delta y_0+\frac{q(q-1)}{2}\Delta^2 y_0+\frac{q(q-1)(q-2)}{6}\Delta^3 y_0+...\right]dq \qquad 6.4$$

Integrating and putting the limits of integration,

$$I=\int_0^n\left[y_0+q\Delta y_0+\frac{q(q-1)}{2}\Delta^2 y_0+\frac{q(q-1)(q-2)}{6}\Delta^3 y_0+...\right]dq$$

$$=nh\left[y_0+\frac{n}{2}\Delta y_0+\frac{n(2n-3)}{12}\Delta^2 y_0+\frac{n(n-2)^2}{24}\Delta^3 y_0+\left(\frac{n^4}{5}-\frac{3n^2}{2}+\frac{11n^2}{3}-3n\right)\frac{\Delta^4 y_0}{4!}+...\right] \qquad 6.5$$

This is the Newton-Coates quadrature formula.

By setting *n* equal to 1, 2, 3, …, we obtain different integration formulas.

3.2 The Trapezoidal Rule

Suppose we set n equal to 1, and take the curve between two consecutive points as linear. Thus, we terminate the sequence on the right in equation 6.5 at the linear term as the higher difference terms ($\Delta^2 y_0$, $\Delta^3 y_0$, etc.) would be zero. Then,

$$\int_{x_0}^{x_0+h} f(x)\,dx=h\left(y_0+\frac{1}{2}\Delta y_0\right)=\frac{h}{2}\left(y_0+\frac{1}{2}(y_1-y_0)\right)=\frac{h}{2}(y_0+y_1) \qquad 6.6$$

Similarly,

$$\int_{x_0+h}^{x_0+2h} f(x)\,dx=h\left(y_1+\frac{1}{2}\Delta y_1\right)=\frac{h}{2}\left(y_1+\frac{1}{2}(y_2-y_1)\right)=\frac{h}{2}(y_1+y_2) \qquad 6.7$$

.

.

.

$$\int_{x_0+(n-1)h}^{x_0+nh} f(x)\,dx=\frac{h}{2}(y_{n-1}+y_n) \qquad 6.8$$

Do these equations remind you of an old formula for finding the area of a triangle? Yes, each of them is the area of a trapezium, hence the procedure is known as the trapezoidal rule.

Adding all these integrals,

$$\int_{x_0}^{x_0+nh} f(x)\,dx=\frac{h}{2}(y_0+y_1)+\frac{h}{2}(y_1+y_2)+...+\frac{h}{2}(y_{n-1}+y_n) \qquad 6.9$$

$$=\frac{h}{2}[(y_0+y_n)+2(y_1+y_2+...+y_{n-1})]$$

3.3 Simpson's one-third rule

We set n = 2 in equation 6.5, and assume the function is quadratic between two consecutive intervals. Then,

$$\int_{x_0}^{x_0+2h} f(x)\,dx = 2h\left(y_0 + \Delta y_0 + \frac{1}{6}\Delta^2 y_0\right) \qquad 6.10$$

$$= 2h\left(y_0 + (y_1 - y_0) + \frac{1}{6}(\Delta y_1 - \Delta y_0)\right)$$

$$= h\left(y_0 + (y_1 - y_0) + \frac{1}{6}[(y_2 - y_1) - (y_1 - y_0)]\right)$$

$$= h\left(y_0 + (y_1 - y_0) + \frac{1}{6}[y_2 - 2y_1 + y_0]\right)$$

$$= h(6y_0 + 6(y_1 - y_0) + [y_2 - 2y_1 + y_0])$$

$$= 2h\frac{y_0 + 4y_1 + y_2}{3}$$

$$= = \frac{h}{3}(y_0 + 4y_1 + y_2) \qquad 6.11$$

Similarly,

$$\int_{x_0+2h}^{x_0+4h} f(x)\,dx = \frac{h}{3}(y_2 + 4y_3 + y_4) \qquad 6.12$$

.

.

.

$$\int_{x_0+(n-2)h}^{x_0+nh} f(x)\,dx = \frac{h}{3}(y_{n-2} + 4y_{n-1} + y_n) \qquad 6.12$$

Adding all these integrals, with the proviso that n be even (this condition is necessary as we need two consecutive intervals, x_k to x_{k+1} to x_{k+2}),

$$\int_{x_0}^{x_0+nh} f(x)\,dx = \frac{h}{3}[(y_0 + y_n) + 4(y_1 + y_2 + \ldots + y_{n-1}) + 2(y_2 + y_4 + \ldots + y_{n-2})]$$

With the aid of the summation symbol,

$$\int_{x_0}^{x_0+nh} f(x)\,dx = \frac{h}{3}[(y_0 + y_n) + 4\sum_{i=1,\, i\,\text{odd}}^{n-1} y_i + 2\sum_{i=2,\, i\,\text{even}}^{\text{n-2}} y_i] \qquad 6.13$$

This is Simpson's one-third rule.

3.4 Simpson's three-eighth rule

In this case, we set n equal to 3 in equation 6.5 and take the curve over each interval as a polynomial of order 3.

$$\int_{x_0}^{x_0+3h} f(x)\,dx = 3h\left(y_0 + \frac{3}{2}\Delta y_0 + \frac{3}{2}\Delta^2 y_0 + \frac{1}{8}\Delta^3 y_0\right) \qquad 6.14$$

The student can show that,

$$\int_{x_0}^{x_0+3h} f(x)\,dx = \frac{3h}{8}(y_0 + 3y_1 + 3y_2 + y_3) \qquad 6.15$$

Similarly,

$$\int_{x_0+3h}^{x_0+5h} f(x)\,dx = \frac{3h}{8}(y_3 + 3y_4 + 3y_5 + y_6) \qquad 6.16$$

.
.
.

Adding all these integrals, with the proviso that *n* be a multiple of 3,

$$\int_{x_0}^{x_0+nh} f(x)\,dx = \frac{3h}{8}[(y_0 + y_n) + 3(y_1 + y_2 + ... + y_{n-1}) + 2(y_3 + y_6 + ... + y_{n-3})] \qquad 6.17$$

This is Simpson's three-eighth rule.

Exercise: Integrate the following function of x with respect to x using the Trapezoidal rule, Simpson's one-third rule and Simpson's three-eighth rule.

$3x^2 + 5x - 1$; $1 \le x \le 4$; step size 0.5.

Compare your results with the exact value of the integral.

Solution:

$$x_0 = 0,\ x_1 = \frac{1}{2},\ x_2 = 1,\ x_3 = \frac{3}{2},\ x_4 = 2,\ x_5 = \frac{5}{2},\ x_6 = 3$$

$$y_0 = 1,\ y_1 = 2.75,\ y_2 = 5,\ y_3 = 7.75,\ y_4 = 11,\ y_5 = 14.75,\ y_6 = 19$$

i. Trapezoidal rule

$$\text{Integral } = \frac{h}{2}\left(y_0 + y_6 + 2\sum_{i=1}^{5} y_i \right) = 25.625$$

ii. Simpson's $\frac{1}{3}$ rule

$$\text{Integral } = \frac{h}{3}\left(y_0 + y_6 + 4\sum_{i=1, i\,odd}^{5} y_i + 2\sum_{i=2,\, i\ even}^{4} y_i \right) = 25.5$$

The exact integral is 25.5.

The Simpson's one-third rule is a second order approximation to the integral. Since the function is quadratic, an accurate result is obtained.

(iii) Simpson's three-eighth rule

$$x_0 = 0,\ x_1 = \frac{1}{2},\ x_2 = 1,\ x_3 = \frac{3}{2},\ x_4 = 2,\ x_5 = \frac{5}{2},\ x_6 = 3$$

$$y_0 = 1,\ y_1 = 2.75,\ y_2 = 5,\ y_3 = 7.75,\ y_4 = 11,\ y_5 = 14.75,\ y_6 = 19$$

Integral = $\frac{3h}{8}[(y_0 + y_6) + 3(y_1 + y_2 + y_4 + y_5) + 2y_3]$

= $\frac{3}{8}\frac{1}{2}[(1+19) + 3(2.75 + 5 + 11 + 14.75) + 2(7.75)]$

= 25.5

3.5 Errors in the Quadrature formulas

We approximated the function $f(x)$ with the polynomial $P(x)$. The error involved in the approximation is

$$E = \int_a^b f(x)\,dx - \int_a^b P(x)\,dx \qquad 6.18$$

The Taylor series expansion of $f(x)$ about x_0 is, where $h = x - x_0$,

$$f(x) = f(x_0) + hf'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots \qquad 6.19$$

3.5.1 Error in the Trapezoidal rule

$$\int_{x_0}^{x_0+h} f(x)\,dx = \int_{x_0}^{x_0+h} \left[f(x_0) + hf'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots \right] dx \qquad 6.20$$

In the first interval, $[x_0, x_1]$, trapezoidal rule gives an area $\frac{1}{2}(f(x_0) + f(x_1))$. The integral (integrating term by term) on the right side of equation 6.20 gives

$$hf(x_0) + \frac{h^2}{2} f'(x_0) + \frac{h^3}{3 \cdot 2!} f''(x_0) + \ldots \qquad 6.21$$

When $x = x_1$, $f(x) = f(x_1)$. Thus, from equation 6.19,

$$f(x_1) = f(x_0) + hf'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots \qquad 6.22$$

Thus,

$$\frac{h}{2}(f(x_0) + f(x_1)) = \frac{h}{2}\left[f(x_0) + \left\{ f(x_0) + hf'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots \right\} \right]$$

$$= \frac{h}{2}\left[2f(x_0) + hf'(x_0) + \frac{h^2}{2!} f''(x_0) + \ldots \right] \qquad 6.23$$

The error in the first interval is therefore (5.21 minus 5.23):

$$\left[hf(x_0) + \frac{h^2}{2} f'(x_0) + \frac{h^3}{3 \cdot 2!} f''(x_0) + \ldots \right] - \left[hf(x_0) + \frac{h^2}{2} f'(x_0) + \frac{h^3}{2 \cdot 2!} f''(x_0) + \ldots \right]$$

$$= \left[\frac{1}{3 \cdot 2} - \frac{1}{2 \cdot 2!} \right] h^3 f''(x_0) + \ldots = -\frac{h^3}{12} f''(x_0) + \ldots \qquad 6.24$$

In the interval $[x_1, x_2]$, the error in the two values is $-\frac{h^3}{12}f''(x_1)+...$, etc.

The total error, therefore, is

$$E = -\frac{h^3}{12}[f''(x_0) + f''(x_1) + f''(x_2) + ... + f''(x_{n-1})] \qquad 6.25$$

If the largest value of the sequence of the second differentials at different discrete values of x is $f''(\hat{x})$, then we can write,

$$E < -\frac{nh^3}{12}f''(\hat{x}) = -\frac{(b-a)h^2}{12}f''(\hat{x}) \qquad 6.26$$

since $n = (b-a)/h$

3.5.2 Error in the Simpson's one-third rule

$$\int_{x_0}^{x_0+2h} f(x)\,dx = \int_{x_0}^{x_0+2h}\left[f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + ...\right]dx \qquad 6.27$$

In the first interval, $[x_0, x_1]$, Simpson's one-third rule gives an area $\frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2))$. The integral on the right side of equation 6.27 gives

$$2hf(x_0) + \frac{4h^2}{2!}f'(x_0) + \frac{8h^3}{3!}f''(x_0) + ... \qquad 6.28$$

When $x = x_1$, $f(x) = f(x_1)$. Thus, from equation 6.19,

$$f(x_1) = f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + \frac{h^3}{3!}f'''(x_0) + ... \qquad 6.29$$

Setting $x = x_0 + 2h$, $f(x) = f(x_2)$ and

$$f(x_2) = f(x_0) + 2hf'(x_0) + \frac{4h^2}{2!}f''(x_0) + \frac{8h^3}{3!}f'''(x_0) + ... \qquad 6.30$$

Putting equations 6.29 and 6.30 into equation 6.27 and equating to the approximate integral in the interval x_0 to $x_0 + 2h$,

$$\begin{aligned}
&\frac{h}{3}(f(x_0) + 4f(x_1) + f(x_2)) \\
&= \frac{h}{3}\left[f(x_0) + 4\left(f(x_0) + hf'(x_0) + \frac{h^2}{2!}f''(x_0) + \frac{h^3}{3!}f'''(x_0) + ...\right)\right. \\
&\qquad \left. + \left(f(x_0) + 2hf'(x_0) + \frac{4h^2}{2!}f''(x_0) + \frac{8h^3}{3!}f'''(x_0) + ...\right)\right] \\
&= 2hf(x_0) + \frac{4h^2}{2!}f'(x_0) + \frac{8h^3}{3\times 2!}f''(x_0) + \frac{5h^5}{18}f^{(iv)}(0) + ... \qquad 6.31
\end{aligned}$$

The error in the interval $[x_0, x_2]$ is therefore 5.28 minus 5.31:

$$\left[2hf(x_0)+\frac{4h^2}{2!}f'(x_0)+\frac{8h^3}{3!}f''(x_0)+...\right]$$

$$-\left[2hf(x_0)+\frac{4h^2}{2!}f'(x_0)+\frac{8h^3}{3\times 2!}f''(x_0)+\frac{5h^5}{18}f^{(iv)}(0)+...\right]$$

$$=\left[\frac{5}{18}-\frac{4}{15}\right]h^5 f^{(iv)}(x_0)+...= -\frac{h^5}{90}f^{(iv)}(x_0)+... \qquad 6.32$$

In the interval $[x_2, x_4]$, the error in the two values is $-\frac{h^5}{90}f^{(iv)}(x_0)+...$, etc.

The total error, therefore, is

$$E=-\frac{h^5}{90}[f^{(iv)}(x_0)+f^{(iv)}(x_1)+f^{(iv)}(x_2)+...] \qquad 6.33$$

If the largest value of the sequence of the second differentials at different discrete values of x is $f^{(iv)}(\hat{x})$, then we can write,

$$E<-\frac{nh^5}{90}f^{(iv)}(\hat{x})= -\frac{(b-a)h^5}{2\times 90}f^{(iv)}(\hat{x})= -\frac{(b-a)h^5}{190}f^{(iv)}(\hat{x}) \qquad 6.34$$

since $h=(b-a)/2n$.

3.6 Romberg's method

Yet again, we refer to the integral,

$$I=\int_a^b f(x)\,dx$$

You would recall that the error in trapezoidal rule in a subinterval h is

$$E=-\frac{(b-a)h^2}{12}f''(x)$$

Thus, for subinterval of width h_1, the error in the integral is

$$E_1=-\frac{(b-a)h_1^{\ 2}}{12}f''(x) \qquad 6.35$$

For subinterval of width h_2, the error in the integral is,

$$E_2=-\frac{(b-a)h_2^{\ 2}}{12}f''(\bar{x}) \qquad 6.36$$

We expect that $f''(x)$ and $f''(\bar{x})$ would be almost equal. Dividing equation 6.35 by equation 6.36,

$$\frac{E_1}{E_2}=\frac{h_1^{\ 2}}{h_2^{\ 2}} \qquad 6.37$$

It follows that $E_2=\frac{h_2^{\ 2}}{h_1^{\ 2}}E_1$, so that $E_2-E_1=\frac{h_2^{\ 2}}{h_1^{\ 2}}E_1-E_1=E_1\left(\frac{h_2^{\ 2}}{h_1^{\ 2}}-1\right)$

$$\frac{E_1}{E_2 - E_1} = \frac{E_1}{E_1\left(\frac{h_2{}^2}{h_1{}^2} - 1\right)} = \frac{h_1{}^2}{h_2{}^2 - h_1{}^2} \qquad 6.38$$

We also note that adding the error to the estimate gives the correct integral *I*.

$$I = I_1 + E_1 = I_2 + E_2 \qquad 6.39$$

from which

$$E_2 - E_1 = I_1 - I_2 \qquad 6.40$$

From $\frac{E_1}{E_2 - E_1} = \frac{h_1{}^2}{h_2{}^2 - h_1{}^2}$,

$$E_1 = \frac{h_1{}^2}{h_2{}^2 - h_1{}^2}(E_2 - E_1) = \frac{h_1{}^2}{h_2{}^2 - h_1{}^2}(I_1 - I_2) \qquad 6.41$$

We can therefore write,

$$I = I_1 + E_1 = I_1 + \frac{h_1{}^2}{h_2{}^2 - h_1{}^2}(I_1 - I_2)$$

$$= \frac{I_1(h_2{}^2 - h_1{}^2) + h_1{}^2(I_1 - I_2)}{h_2{}^2 - h_1{}^2} = \frac{I_1 h_2{}^2 - I_2 h_1{}^2}{h_2{}^2 - h_1{}^2} \qquad 6.42$$

This is a better approximation to the integral, *I*. Why, do you think?

Let us take a situation where $h_1 = h$ and $h_2 = \frac{1}{2}h$. Then, equation 6.42 gives,

$$I = \frac{I_1(h/2)^2 - I_2 h^2}{(h/2)^2 - h^2} = \frac{I_1(h^2/4) - I_2 h^2}{(h^2/4) - h^2} \qquad 6.43$$

Multiplying through by 4, and denoting I by $I(h, h/2)$

$$I(h, h/2) = \frac{4I_2 - I_1}{3} \qquad 6.44$$

But $I_1 = I(h)$ and $I_2 = I(h/2)$

We can therefore write,

$$I(h, h/2) = [4I(h/2) - I(h)]/3 \qquad 6.45$$

We can develop the scheme below by applying equation 6.45 to the estimates of the integral over successively halved intervals.

$I(h)$			
	$I(h, h/2)$		
$I(h/2)$		$I(h, h/2, h/4)$	
	$I(h/2, h/4)$		$I(h, h/2, h/4, h/8)$
$I(h/4)$		$I(h/2, h/4, h/8)$	
	$I(h/4, h/8)$		

$I(h/8)$			

We continue the table until successive values converge. This gives a better result than could have been obtained with the trapezoidal rule.

Example

Let us once again solve the problem $\int_0^3 (x^2 + 3x + 1)dx.$

Solution:

Let us choose h = 1.0, 0.5 and 0.25. Then, the following table obtains.

$I(h) = 26$	
	$I(h, h/2)$ = (4×25.625-26)/3 = 25.5
$I(h/2) = 25.625$	
	$I(h/2, h/4)$ = (4×25.53125-25.625)/3 = 25.5
$I(h/4) = 25.53125$	

25.5 is a better approximation to the integral than the trapezoidal method. Indeed, in this case, it is the exact integral.

4.0 Conclusion

In this Unit, you derived the Newton-Coates quadrature formula. Also, you learnt how to carry out, with several methods, the numerical integration of a function between a given limit of integration. Having found the error in the quadrature formula for the different integration methods, you were able to link up with the Romberg method of numerical integration.

5.0 Summary

In this Unit, you were able to:

- Derive the Newton-Coates quadrature formula, and, consequently, the different formulas for integrating a function between limits.
- Estimate the error in the quadrature formula for different numerical integration methods

6.0 Tutor Marked Assignment

1. Integrate the function $x(t) = \frac{1}{2}t^2 + \frac{5}{2}t + 2$, $0 \le t \le 0.6$, with six intervals, using the following methods:
   (i) Trapezoidal rule
   (ii) Simpson's one-third rule
   (iii) Simpson's three-eighth rule

2. Evaluate the integral $\int_0^{\pi/2} x \sin x \, dx$ (where x is in radians) with a step-size of $\Delta x = \pi/16$, using

(i) Trapezoidal rule

(ii) Simpson's one-third rule

(iii) Simpson's three-eighth rule

7.0 References/Further Reading

Solutions to Tutor Marked Assignment

1. Integrate the function $x(t) = \frac{1}{2}t^2 + \frac{5}{2}t + 2$, $0 \le t \le 0.6$, with six intervals, using the following methods:
   (i) Trapezoidal rule
   (ii) Simpson's one-third rule

   Formula for Trapezoidal rule:

$$\int_a^b f(x)dx = \frac{\Delta x}{2}\left(y_0 + 2\sum_{i=1}^{n-1} y_i + y_n \right)$$

   Formula for Simpson's one-third rule:

$$\int_a^b f(x)dx = \frac{\Delta x}{3}\left(y_0 + 4\sum_{\substack{i=1\\ i,odd}}^{n-2} y_i + 2\sum_{\substack{i=2\\ i,even}}^{n-2} y_i + y_n \right)$$

			Trapezo	Simpson	
Step	0.1				
x	$f(x)$				
0	2	**First value of $f(x)$ =**	2	2	
0.1	2.255				
0.2	2.52			33.7	**4 times sum of odd**
0.3	2.795	**Sum of intermediate values of $f(x)$**	28.05		Simpson's 1-3 rule
0.4	3.08	(Trapezoidal rule)		11.2	**2 times sum of even**
0.5	3.375				
0.6	3.68	**Last value of $f(x)$ =**	3.68	3.68	
			33.73	50.58	
	Results	**Answer**	**1.6865**	**1.686**	

Analytical solution:

$$\int_0^{0.6}\left(2 + 2.5t + \frac{1}{2}t^2 \right)dt = \left. 2t + 2.5\frac{t^2}{2} + \frac{t^3}{6} \right|_0^{0.6} = 1.686$$

(iii) Simpson's three-eighth rule

Integral = $\frac{3h}{8}[(y_0 + y_6) + 3(y_1 + y_2 + y_4 + y_5) + 2y_3]$

$= \frac{3(0.1)}{8}[(2 + 3.68) + 3(2.255 + 2.52 + 3.08 + 3.375) + 2(2.795)]$

= 1.686

2. Evaluate the integral $\int_0^{\pi/2} x \sin x \, dx$ (where x is in radians) with a step-size of $\Delta x = \pi/16$, using

(i) Trapezoidal rule

(ii) Simpson's one-third rule

Working with radians

			Trapezoidal	Simpson's 1-3	
x	**xsinx**				
0	0	**First value of** $f(x)$ **=**	0	0	
0.19635	0.038306				
0.392699	0.150279			10.11957735	**4 times sum of odd**
0.589049	0.327258	**Sum of intermediate values of** $f(x)$	8.6479081		
0.785398	0.55536			3.588119468	**2 times sum of even**
0.981748	0.816293				
1.178097	1.08842	**Last value of** $f(x)$ **=**	1.5707963	1.570796328	
1.374447	1.348037				
1.570796	1.570796				
			10.218704	15.27849315	
		Answer	**1.0032188**	**0.99997483**	

Unit 7: Initial Value Problems of Ordinary Differential Equations

1.0 Introduction

Ordinary differential equations abound in Physics. This is because we often have to deal with a rate of change of function of a single variable. It could be a time-rate of change, say velocity or acceleration, or it could be a spatial rate of change as you would expect from the variation of temperature over a metallic bar heated at one end at any particular fixed instant of time. Unlike analytic differentiation of a function, which is most times achievable, the larger number of functions do not lend themselves to analytical integration. We therefore have to resort to numerical integration when confronted with such functions. In this Unit, you shall learn how to numerically integrate a function of a single variable.

2.0 Objectives

By the end of this Unit, you should be able to:

- Write an *n*th order ordinary differential equation in terms of *n* first order ordinary differential equations.
- Solve a first order ordinary differential equation.
- Solve a system of first order ordinary differential equations.

3.1 Reduction of a higher order ODE to a system of first order ODE

Every ordinary differential equation can be put in the form

$$\frac{dy}{dx} = f(y, x) = y' \qquad 7.1$$

or a system of such equations. As an example, take the equation of simple harmonic oscillation,

$$x'' + \omega^2 x = 0 \qquad 7.2$$

where ω is the angular frequency of oscillation.

Let

$$z = x' \qquad 7.3$$

Then,

$$z' = -\omega^2 x \qquad 7.4$$

The last two equations form a system of ordinary differential equations. Likewise an *n*th order ordinary differential equation can be written as a set of *n* ordinary differential equations. Thus, it suffices to solve the ordinary differential equation $\frac{dy}{dx} = f(y,x)$.

Example

The Henon-Heiles system of equations leads to chaotic motion. We can reduce the two second-order differential equations to four first order ordinary differential equations. We can then solve the equations with the methods to be learnt later in this Unit.

The Henon-Heile's Hamiltonian is,

$$H = \frac{p_1{}^2 + p_2{}^2}{2} + \frac{q_1{}^2 + q_2{}^2}{2} + q_1{}^2 q_2 - \frac{1}{3} q_2{}^3$$

The resulting equations are,

$$\frac{d^2 q_1}{dt^2} = -(q_1 + 2q_1 q_2)$$
$$\frac{d^2 q_2}{dt^2} = -(q_2 + 2q_1{}^2 - q_2{}^2) = -2(1 + 2q_1)$$

Each of these equations has been broken up into two first order ordinary differential equations:

$$\frac{dq_1}{dt} = p_1$$
$$\frac{d}{dt} p_1 = -(q_1 + 2q_1 q_2) = -2(1 + 2q_1)$$

$$\frac{dq_2}{dt} = p_2$$
$$\frac{d}{dt} p_2 = -(q_2 + 2q_1{}^2 - q_2{}^2) = -2(1 + 2q_1)$$

3.2 Methods of Solving First Order Ordinary Differential Equations

We shall now take a look at the various methods of solving a first order ordinary differential equation.

3.2.1 Picard's Method

Given the ordinary differential equation

$$\frac{dy}{dx} = f(x,y) \qquad 7.5$$

we can write

$$dy = f(x, y)\,dx \qquad 7.6$$

Integrating both sides,

$$\int_{y_0}^{y} dy = \int_{x_0}^{x} f(x, y)\,dx \qquad 7.7$$

Then,

$$y = y_0 + \int_{x_0}^{x} f(x, y)\,dx \qquad 7.8$$

We take, as a first approximation to the solution $y(x)$, the value of y when $x = x_0$, that is, y_0. Then,

$$y_1 = y_0 + \int_{x_0}^{x} f(x, y_0)\,dx \qquad 7.9$$

The next approximation to y, that is, y_2, is obtained by substituting y_1 under the integral.

$$y_2 = y_0 + \int_{x_0}^{x} f(x, y_1)\,dx \qquad 7.10$$

Thus, we obtain a sequence of approximations to y which would converge to the solution of the ordinary differential equation provided the function $f(x, y)$ is bounded in a region about (x_0, y_0) and satisfies the Lipschitz condition:

$$| f(x, y) - f(x, \hat{y}) | \leq M \, | y - \hat{y} | \qquad 7.11$$

where M is a constant.

Obviously, a drawback to this method is that most times, the function has to be a simple function that can be easily integrated. As we have discussed before, only a limited class of functions satisfies this condition.

3.2.2 Euler Method

We discretize the ordinary differential equation 7.1 as

$$\frac{y_{j+1} - y_j}{\Delta x} = f(y_j, x_j) \qquad 7.12$$

From which it follows that

$$y_{j+1} = y_j + \Delta x \cdot f(y_j, x_j) \qquad 7.13$$

This method is self-starting, but is so low in accuracy that it is rarely ever used in serious computational work.

Example: With the aid of the Euler method, calculate $y(0.8)$, given the differential equation

$\frac{dy}{dx} = x + y$; $y(0) = 0$; with $h = 0.2$

Solution:

$y_{j+1} = y_j + h \cdot f(y_j, x_j)$

$j = 0$; $\quad y_0 = 0, x_0 = 0$; $f(y_0, x_0) = 0 + 0 = 0$;

$$y_1 = y_0 + h \cdot f(y_0, x_0) = 0 + 0.2 * 0 = 0$$

$j = 1;$

$$y_1 = 0, x_1 = 0.2; \; f(y_1, x_1) = 0 + 0.2 = 0.2;$$
$$y_2 = y_1 + h \cdot f(y_1, x_1) = 0 + 0.2 * 0.2 = 0.04$$

$j = 2;$

$$y_2 = 0.04, x_2 = 0.4; \; f(y_2, x_2) = 0.04 + 0.4 = 0.44;$$
$$y_3 = y_2 + h \cdot f(y_2, x_2) = 0.04 + 0.2 * 0.44 = 0.128$$

3.2.3 Modified Euler Method

We could write equation 7.1 as

$$\frac{dy}{dx} = f(x, y)$$
$$dy = f(x, y)dx$$

Integrating,

$$y_1 - y_0 = \int_{x_0}^{x_1} f(x, y)dx$$

Rearranging and generalizing,,

$$y_{j+1} = y_j + \int_{x_j}^{x_{j+1}} f(x, y)dx$$

With the aid of the trapezoidal rule, we can write the last equation as

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1})] \qquad 7.14$$

Indeed, it is best to write equation 7.14 as

$$y_{j+1}^{(i+1)} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1}^{(i)})] \qquad 7.15$$

This is the modified Euler method. It is an implicit scheme.

The starting value $y_0^{(1)}$ is obtained by an implicit formula, e.g., the Euler formula. Thus, the scheme would look like (for *j* = 0),

$$i = 0 \quad y_1^{(1)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(0)})]$$
$$i = 1 \quad y_1^{(2)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(1)})]$$

This is continued until convergence is achieved.

Example

Using the modified Euler method, find $y(0.2)$, if $y' - (x + y) = 0$, given that $y(0) = 1$. Take a step length of 0.1 and the tolerance as $| y_i^{(k)} - y_i^{(k-1)} | \leq 0.0001$.

Solution

$$x_0 = 0, \; y_0 = 1$$

Using Euler’s formula,

$$y_1^{(0)} = y_0 + hf(x_0, y_0) = 1 + 0.1 \times (0+1) = 1.1$$

We now apply the modified Euler formula.

$$i = 0 \quad y_1^{(1)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(0)})]$$

$$= 1 + 0.05[(0+1) + (0.1 + 1.1)] = 1.11$$

$$i = 1 \quad y_1^{(2)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(1)})]$$

$$= 1 + 0.05[(0+1) + (0.1 + 1.11)] = 1.1105$$

$$| y_1^{(2)} - y_1^{(1)} | = \left| 1.1105 - 1.11 = 0.0005 \right|$$

$$i = 2 \quad y_1^{(3)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(2)})]$$

$$= 1 + 0.05[(0+1) + (0.1 + 1.1105)] = 1.110525$$

$$| y_1^{(3)} - y_1^{(2)} | = \left| 1.110525 - 1.1105 \right| = 0.000025$$

With the tolerance satisfied, we can now proceed to get y_2, that is, $y(0.2)$

Using Euler’s formula,

$$y_2^{(0)} = y_1 + hf(x_1, y_1) = 1.110525 + 0.1 \times (0 + 1.110525) = 1.23155$$

We now apply the modified Euler formula.

$$i = 0 \quad y_2^{(1)} = y_1 + \frac{h}{2}[f(x_1, y_1) + f(x_2, y_2^{(0)})]$$

$$= 1.1105 + 0.05[(0.1 + 1.1105) + (0.2 + 1.23155)] = 1.242603$$

$$i = 1 \quad y_2^{(2)} = y_1 + \frac{h}{2}[f(x_1, y_1) + f(x_2, y_2^{(1)})]$$

$$= 1.1105 + 0.05[(0.1 + 1.1105) + (0.2 + 1.242603)] = 1.243155$$

$$| y_2^{(2)} - y_2^{(1)} | = \left| 1.243155 - 1.242603 \right| = 0.000552$$

$$i = 2 \quad y_2^{(3)} = y_1 + \frac{h}{2}[f(x_1, y_1) + f(x_2, y_2^{(2)})]$$

$$= 1.1105 + 0.05[(0.1 + 1.1105) + (0.2 + 1.243155)] = 1.243183$$

$$| y_2^{(3)} - y_2^{(2)} | = \left| 1.243183 - 1.243155 \right| = 0.000028$$

3.2.4 Runge-Kutta Methods

We recall that Taylor’s series is given as

$$f(x + \Delta x) = f(x) + f'(x)\Delta x + \frac{f''(x)(\Delta x)^2}{2!} + \cdots$$

This we can write as (if we set $f(x + \Delta x) = y_1,\ f(x) = y_0$) and $\Delta x = h$)

$$y_1 = y_0 + hf'(x) + \frac{f''(x)h^2}{2!} + \ldots$$

A first order approximation to the series is

$$y_1 = y_0 + hf'(x) \qquad 7.16$$

This is the **Runge-Kutta first order method**, which you would also notice is the Euler method.

On the other hand, we recall equation 7.14,

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_{j+1})]$$

Writing $y_{j+1} = y_j + hf(x_j, y_j)$ in equation 7.14 (the modified Euler formula),

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_j + hf(x_j, y_j))]$$

Let us set

$$hf(x_j, y_j) = k_1$$

and

$$hf(x_{j+1}, y_j + hf(x_j, y_j)) = hf(x_{j+1}, y_j + k_1) = k_2$$

Then, we can write

$$y_{j+1} = y_j + \frac{1}{2}[k_1 + k_2] \qquad 7.17$$

This is the **second-order Runge-Kutta** formula.

Example

Find the value of y at x = 0.2 if $y' + 2y = 0$; $y(0) = 1$, step-length 0.2.

Solution

$$y_{j+1} = y_j + \frac{h}{2}[f(x_j, y_j) + f(x_{j+1}, y_j + hf(x_j, y_j))]$$

$$y_1 = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_0 + hf(x_0, y_0))]$$

Let us set

$$hf(x_0, y_0) = k_1 = 0.2(-2y_0) = 0.2[-2(1)] = -0.4$$

and

$$hf(x_1, y_0 + k_1) = k_2 = 0.2[-2(1 - 0.4)] = -0.24$$

Then, we can write

$$y_{j+1} = y_j + \frac{1}{2}[k_1 + k_2]$$

$$y_1 = y_0 + \frac{1}{2}[k_1 + k_2] = 1 + \frac{1}{2}[-0.4 - 0.24]$$

Hence,

$$y(0.2) = 0.68$$

The formula for the **third-order Runge Kutta method** is

$$y_{j+1} = y_j + \frac{1}{6}(k_1 + 4k_2 + k_3) \qquad 7.18$$

where

$$k_1 = hf(x_j, y_j)$$

$$k_2 = hf\left(x_j + \frac{h}{2}, y_0 + \frac{k_1}{2}\right)$$

$$k_3 = hf(x_j + h, y_j - k_1 + 2k_2)$$

Example

Using the third-order Runge-Kutta method, find the value of y when $x = 0.2$, given that $y' = x - y$, $y(0) = 2$, with step length 0.1.

Solution

$$j = 0$$

$$k_1 = hf(x_0, y_0) = 0.1 \times (0 - 2) = -0.2$$

$$k_2 = hf\left(x_0 + \frac{h}{2}, y_0 + \frac{k_1}{2}\right)$$

$$= 0.1 \times \left(\left[0 + \frac{0.1}{2}\right] - \left[1 - \frac{0.2}{2}\right]\right) = 0.1 \times (.05 - 0.9) = -0.085$$

$$k_3 = hf(x_0 + h, y_0 - k_1 + 2k_2)$$

$$= 0.1 \times f(0.1,\ 2 - (-0.2) + 2 \times (-0.085)) = 0.1 \times f(0.1,\ 2.03) = -0.193$$

$$y_1 = y_0 + \frac{1}{6}(k_1 + 4k_2 + k_3) = 2 + \frac{1}{6}[-0.2 + 4 \times (-0.085) + (-0.193)] = 1.877833$$

$$j = 1$$

$$k_1 = hf(x_1, y_1) = 0.1 \times (0.1 - 1.877833) = -0.177783$$

$$k_2 = hf\left(x_1 + \frac{h}{2}, y_1 + \frac{k_1}{2}\right)$$

$$= 0.1 \times \left(\left[0.1 + \frac{0.1}{2}\right] - \left[1.877833 - \frac{0.177783}{2}\right]\right) = -0.163894$$

$$k_3 = hf(x_1 + h, y_1 - k_1 + 2k_2)$$

$$= 0.1 \times f(0.2, 1.877833 - (-0.177783) + 2 \times (-0.163894)) = 0.1 \times f(0.2,\ 1.727828) = -0.152783$$

$$y_2 = y_1 + \frac{1}{6}(k_1 + 4k_2 + k_3)$$

$$= 1.877833 + \frac{1}{6}[-0.177783 + 4 \times (-0.163894) + (-0.152783)]$$

$$= 1.713476$$

Fourth Order Runge-Kutta Method

The formula is , where *h* is the step-length,

$$y_{j+1} = y_j + h\left[\frac{1}{6} f(y_j, x_j) + \frac{1}{3} f(y*_{j+1/2}, x_{j+1/2}) + \frac{1}{3} f(y**_{j+1/2}, x_{j+1/2}) + \frac{1}{6} f(y*_{j+1}, x_{j+1})\right] \qquad 7.19$$

The computation follows the order

(i) $f(y_j, x_j)$ 7.20

(ii) $x_{j+1/2} = x_j + \frac{h}{2}$ 7.21

(iii) $y*_{j+1/2} = y_j + \frac{h}{2} f(y_j, x_j)$ 7.22

(iv) $y**_{j+1/2} = y_j + \frac{h}{2} f(y*_{j+1/2}, x_j)$ 7.23

(v) $y*_{j+1} = y_j + h \cdot f(y**_{j+1/2}, x_j)$ 7.24

(vi) Evaluate y_{j+1} with equation 7.19.

Another, equivalent, computation scheme is as follows:

(i) $f(y_j, x_j)$ 7.25

(ii) $k_1 = hf(x_0, y_0)$ 7.26

(iii) $k_2 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1\right)$ 7.27

(iv) $k_3 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2\right)$ 7.28

(v) $k_4 = hf(x_0 + h, y_0 + k_3)$ 7.29

(vi) $k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$ 7.30

(vii) Evaluate $y_{j+1} = y_j + k$

Example: Solve the following ordinary differential equation using the Runge-Kutta Fourth order method.

$\frac{dy}{dx} = y + x$; $y(0) = 1$. Find y at $x = 0.2$

Solution:

$x_0 = 0,\ y_0 = 1,\ h = 0.2,\ f(x_0, y_0) = 1$

$$k_1 = hf(x_0, y_0) = 0.2 \times 1 = 0.2$$

$$k_2 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1\right) = 0.2 \times f(0.1, 1.1) = 0.2400$$

$$k_3 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2\right) = 0.2 \times f(0.1, 1.12) = 0.2440$$

$$k_4 = hf\left(x_0 + h, y_0 + k_3\right) = 0.2 \times f(0.2, 1.244) = 0.2888$$

$$\therefore k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$$

$$= \frac{1}{6}(0.2 + 0.48 + 0.488 + 0.2888) = 0.2428$$

Hence, $y_1 = y_0 + k = 1.2428$

The fourth-order Runge-Kutta method is the most accurate of the Runge-Kutta methods.

3.3 Fourth-Order Runge-Kutta Scheme for a System of Three Equations

We shall solve the Lorenz system of equation with the fourth-order Runge-Kutta method. The equations are:

$$\frac{dx}{dt} = 10(y - x) = f_1(x, y, z)$$

$$\frac{dy}{dt} = (100x - y - xz) = f_2(x, y, z)$$

$$\frac{dz}{dt} = (xy - 2z) = f_3(x, y, z)$$

Let us make use of the set of equations given in 7.19-7.24.

There will be three k_1's, one each for three variables, three k_2's and so on.

$$k_{1x} = hf_1(t_0, x_0, y_0, z_0)$$

$$k_{1y} = hf_2(t_0, x_0, y_0, z_0)$$

$$k_{1z} = hf_3(t_0, x_0, y_0, z_0)$$

$$k_{2x} = hf_1\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{1x}, y_0 + \frac{1}{2}k_{1y}, z_0 + \frac{1}{2}k_{1z}\right)$$

$$k_{2y} = hf_2\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{1x}, y_0 + \frac{1}{2}k_{1y}, z_0 + \frac{1}{2}k_{1z}\right)$$

$$k_{2z} = hf_3\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{1x}, y_0 + \frac{1}{2}k_{1y}, z_0 + \frac{1}{2}k_{1z}\right)$$

$$k_{3x} = hf_1\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{2x}, y_0 + \frac{1}{2}k_{2y}, z_0 + \frac{1}{2}k_{2z}\right)$$
$$k_{3y} = hf_2\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{2x}, y_0 + \frac{1}{2}k_{2y}, z_0 + \frac{1}{2}k_{2z}\right)$$
$$k_{3z} = hf_3\left(t_0 + \frac{1}{2}h, x_0 + \frac{1}{2}k_{2x}, y_0 + \frac{1}{2}k_{2y}, z_0 + \frac{1}{2}k_{2z}\right)$$

$$k_{4x} = hf_1\left(t_0 + h, x_0 + k_{3x}, y_0 + k_{3y}, z_0 + k_{3z}\right)$$
$$k_{4x} = hf_1\left(t_0 + h, x_0 + k_{3x}, y_0 + k_{3y}, z_0 + k_{3z}\right)$$
$$k_{4x} = hf_1\left(t_0 + h, x_0 + k_{3x}, y_0 + k_{3y}, z_0 + k_{3z}\right)$$

$$k_x = \frac{1}{6}(k_{1x} + 2k_{2x} + 2k_{3x} + k_{4x})$$
$$k_y = \frac{1}{6}(k_{1y} + 2k_{2y} + 2k_{3y} + k_{4y})$$
$$k_z = \frac{1}{6}(k_{1z} + 2k_{2z} + 2k_{3z} + k_{4z})$$

Hence, $x_1 = x_0 + k_x$
$y_1 = y_0 + k_y$
$z_1 = z_0 + k_z$

4.0 Conclusion

In this Unit, you got to know how to reduce an *n*th ordinary differential equation to *n* first order differential equations. In particular, you were able to see how a pair of second-order ordinary differential equations were reduced to four first-order differential equations. You also learnt various methods of solving a first order ordinary differential equation.

5.0 Summary

In this Unit, you learnt how to:

- Reduce an *n*th order ordinary differential equation to *n* first order ordinary differential equations.
- Numerically solve a first order ordinary differential equation.
- Numerically solve a system of first order ordinary differential equations.

6.0 Tutor Marked Assignment

1. Given that $\frac{dy}{dx} = xy^2$; $y(0) = -1$, evaluate $y(0.2)$ (step length 0.2), using the
   (i) Modified Euler method.
   (ii) Fourth order Runge-Kutta method.

2. With a step length of 0.1, find the value of y at $x = 0.2$ given the ordinary differential equation: $\frac{dy}{dx} - y + x = 0$; $y(0) = 0$.

(i) Second-order Runge-Kutta method

(ii) Fourth-order Runge-Kutta method.

7.0 References/Further Reading

Solutions to Tutor Marked Assignment

1. Given that $\frac{dy}{dx} = xy^2$; $y(0) = -1$, evaluate $y(0.2)$ (step length 0.2), using the

(iii) Modified Euler method.

(iv) Fourth order Runge-Kutta method.

Solution

$x_0 = 0$, $y_0 = -1$

(i) First using Euler's formula,

$y_1^{(0)} = y_0 + hf(x_0, y_0) = y_0 + x_0 y_0{}^2 = -1 + 0(-1)^2 = -1$

We can now apply the modified Euler formula.

$i = 0 \quad y_1^{(1)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(0)})]$

$= -1 + 0.1[(0 \times (-1)^2 + (0.2 \times (-1)^2)] = -0.98$

$i = 1 \quad y_1^{(2)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(1)})]$

$= -1 + 0.1[(0 \times (-1)^2 + (0.2(-0.98)^2)] = -0.980792$

$| y_1^{(2)} - y_1^{(1)} | = \left| \text{- 0.980792 - (-0.98)} \right| = .000792$

$i = 2 \quad y_1^{(3)} = y_0 + \frac{h}{2}[f(x_0, y_0) + f(x_1, y_1^{(2)})]$

$= -1 + 0.1[(0 \times (-1)^2 + (0.2(-0.980792)^2] = -0.980761 = 1.110525$

$| y_1^{(3)} - y_1^{(2)} | = \left| \text{- 0.980761 - (-0.980792)} \right| = 0.000031$

Hence, $y(0.2) = 0.980761$.

(v) fourth-order Runge-Kutta method.

$x_0 = 0,\ y_0 = -1,\ h = 0.2,\ f(x, y) = xy^2$

$k_1 = hf(x_0, y_0)$

$k_2 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_1\right)$

$k_3 = hf\left(x_0 + \frac{1}{2}h, y_0 + \frac{1}{2}k_2\right)$

$k_4 = hf\left(x_0 + h, y_0 + k_3\right)$

$\therefore k = \frac{1}{6}(k_1 + 2k_2 + 2k_3 + k_4)$

Hence, $y_1 = y_0 + k$

$x(0)$	0	-1	$y(0)$
Step-size	0.2		
k_1	0	-1	0
k_2	0.1	-1	0.02
k_3	0.1	-0.99	0.0196
k_4	0.2	-0.9804	0.03845
k			0.01961
y_1			-0.98039

2. With a step length of 0.1, find the value of y at $x = 0.2$ given the ordinary differential equation: $\dfrac{dy}{dx} - y + x = 0$; $y(0) = 0$.

(iii) Second-order Runge-Kutta method

(iv) Fourth-order Runge-Kutta method.

Solution

(i) Second-order Runge-Kutta method

$$k_1 = hf(x_j, y_j) = 0.1(-x_0 + y_0) = 0.1(0+0) = 0$$

$$k_2 = hf(x_{j+1}, y_j + k_1) = 0.1(-x_1, y_0 + 0) = 0.1(-0.1+0) = -0.01$$

$$y_1 = y_0 + \frac{1}{2}[k_1 + k_2] = y_0 + \frac{1}{2}[k_1 + k_2] = 0 + \frac{1}{2}[0 - 0.01] = -0.005$$

$$k_1 = hf(x_j, y_j) = 0.1(-x_1 + y_1) = 0.1(-0.1 - 0.005) = -0.0105$$

$$k_2 = hf(x_{j+1}, y_j + k_1) = 0.1(-0.2, (-0.005 - 0.0105))$$

$$= 0.1(-0.2 - 0.0155) = -0.02155$$

$$y_2 = y_1 + \frac{1}{2}[k_1 + k_2] = y_1 + \frac{1}{2}[k_1 + k_2]$$

$$= -0.005 + \frac{1}{2}[-0.0105 - 0.02155] = -0.021025$$

(ii) Fourth-order Runge-Kutta method.

$x_0 = 0$	$y_0 = 0$		$x_1 = 0.1$	$y_1 = -0.00517$
k_1	0		k_1	-0.0105171
k_2	-0.005		k_2	-0.0160429
k_3	-0.00525		k_3	-0.0163192
k_4	-0.01053		k_4	-0.022149
k	-0.00517		k	-0.0162317
y_1	-0.00517		y_2	-0.0214026

Elements of C++ Programming

In this chapter we take a look at C++ programming as a tool for numerical analysis. This should not be taken as a substitute for a good book on C++ programming. Indeed, space would only permit us to treat just what it would take you to do write scientific programs.

As is usual with most books on C++ programming language, it would be in order to start with a simple program, the ‘Hello World.’

```
#include <iostream>
using namespace std;

int main ()
{
// Program to write ‘Hello World’ on the screen.

cout << “Hello World”;
return (0);
}
```

We shall now examine this program with a view to familiarizing you with the simplest program in C++ language.

#include <iostream>
A line that begins with # is a directive for the preprocessor. Including this file, which is the iostream standard file. This line is necessary as we shall be making use of input or output (in this particular case, the standard output stream, *cout*). The symbol << is the insertion operator. In the program, the insertion operator inserts the variable “Hello World” into the output stream *cout.*

using namespace std;
namespace contains all the elements of the standard C++ library. This expression enables us to use the elements of the standard C++ library.

int main ()
This is the statement that begins the definition of the main function. All C++ programs are executed beginning from this statement. Thus, it is essential that every C++ program has a main function.

After the *int main ()* statement, the bracket opens with ‘{‘ signifying the beginning of the codes within the main function. This ends with an ‘}’ after the *return (0);* statement.

// Program to write ‘Hello World’ on the screen
Any statement that begins with two slashes (/) is taken as a comment by the compiler. Comments are used to make some ‘thought sense’ of a program. You would be surprised

a program you wrote a few weeks back might not make any sense anymore if you never put enough comments.

```
cout << ”Hello World”;
```

Note that apart from the #include statement and int main (), every statement in this program ends with the semi-colon.

The basic ideas of C++ programming can be listed under the following broad headings:
Declaration Statements
Array Dimensioning
Input / Output
Arithmetic / Logical Expressions
Looping
Subroutines and functions

We shall however discuss first the variables and data types. There are several types: integer, floating point and string.

An identifier is required by every variable. This distinguishes it from other variables. An identifier contains one or more digits, letters and single underscore characters. Usually, it begins with a letter, although it might begin with an underscore sign, where it does not clash with those reserved for the compiler.

Basic Data Types
It is necessary at this point to mention that the byte (4 bits) is the unit of representation in C++. A bit is the smallest. This can store a single character or a small integer. Integers could be signed or unsigned. A byte can store an integer between 0 and 255 if it is an unsigned integer. For a signed integer, it can store between –128 and 127, both limits inclusive in both cases.

Table 1 shows each data type, its size and the range of data that it can take. The size and range are for a 32-bit system.

Table 1: Data types in C++

Name	Description	Size (bytes)	Range
Char	A character or small integer	1	Signed: -128 to 127 Unsigned: 0 to 255
short int	A short integer	2	Signed: -32768 to 32767 Unsigned: 0 to 65535
Int	An integer	4	Signed: -2147483648 to 2147483648 Unsigned: 0 to 4294967295
long int	A long integer	4	Signed: -2147483648 to

			2147483648 Unsigned: 0 to 4294967295
Float	A floating point number	4	$+/-3.4\times10^{+/-38}$
double	A double precision floating point number	8	$+/-1.7\times10^{+/-308}$
long double	A long double precision floating point number	8	$+/-1.7\times10^{+/-308}$
wchar_t	A wide character	2 or 4	1 wide character
Bool	A Boolean value. It takes true or false	1	True or false

A variable has to be declared to be used in C++. This is achieved by simply stating the type of variable it is. For example, int, long, char, short, long (long int), short (short int), float, bool, long double, or wchar_t. This is followed by the variable name. For example,
int number;
or
float age_goat;

Variables of the same type could be declared using the same statement, e.g.,
long number, year;

Moreover, the default is signed. For an unsigned variable, we would need to declare it so. For example,
unsigned distance;

An exception is char. Char has no default; as such, you have to declare it signed or unsigned.

Variable names are case-sensitive, meaning that Happy is not the same variable as happy.

Strings
These are non-numeric values that are longer than a single character. It is not a fundamental type. This necessitates including the header file <string> along with <iostream>.

The Span of a Variable
A variable in a program could be local or global. Local variables are declared within a block or a function. Their scope is limited to the block or the function usually delineated by { … }. Outside the block or the function, the variables are of no relevance. As an example, variable chalk has relevance throughout the function main (). So does variable chalk_dust, but chalk_dust has relevance only within the function minute.

int main ()

float chalk;
{

```
// Program to demonstrate span of variable

int minute ()
   {
// Span of chalk_dust is only within the function minute.
float chalk_dust;
cout <<chalk_dust;
    }
return (0);
}
```

Initialising a Variable

We can fix the initial value of a variable after it might have been appropriately declared, as in:

```
        int counter;
        counter = 5;
```

On the other hand, we could also set the initial value of a variable as we declare its type, as in:

```
        int counter = 5;
```

Another way of initializing a variable is by writing the initial value in parenthesis:

```
        short counter (5);
```

Strings

These are variables that store non-numeric values longer than a character.

To be able to make use of strings, the programmer would need to include the standard header file <string>:

```
#include <iostream>
#include <string>
using namespace std;

int main ()
{
string My_Name;
My_Name = Johnson;
cout << My_Name;
return 0;
}
```

Constants

A constant, as the name implies, has a fixed value.

Constants can be further divided into three categories: Literals, Defined constants and Declared constants.

Literals state the specific values within a program. These can be further divided into three: integer numerals, floating point numbers, Boolean literals and character and string literals.

Integer numerals identify integer decimal, octal (base 8) or hexadecimal (base 16) values. The last two are expressed, respectively, by putting a suffix 0 and 0x. Thus, decimal 750 is equivalent to 01356, and 0x2ee in hexadecimal. Integer numerals are by default integers (int), but we could still declare them unsigned, long or unsigned long by appending the appropriate letter (l, u or ul), where it is immaterial whether the letter is upper or lower case. For example, 750ul.

Floating point numbers are numbers with decimals, and could be with or without exponents. Examples include 4.1239, 6.64e-34. Floating point literals are of type *double* by default, but we can still express a floating point literal as float or long double. In this case, respectively, we append *f* or *l*. For instance, 4.1239f. The appended symbol could be lower or upper case.

Boolean Literals have only two values: true and false. Their type is bool. For example, Bool Decision.

Character and String Literals are non-numerical constants. Single characters are enclosed within single quotes, e.g., ‘t’. A string is expressed within double quotes, “Hello World” for example.

Declared Constants are constants the user declares. After the declaration, the values of the constants remain unchanged as their values cannot be modified. For example,
const int Number.

Defined Constants are constants the user might need quite often in a program. A good example is the number pi. Thus, we could define pi as follows:
#define pi = 3.142

As is usual with all the lines starting with the hatch sign (#), this is a command for the preprocessor.

OPERATORS

Assignment operator

This is the operator that assigns a value on the right of the equality sign to the variable on the left. Thus,
int a = 3;
float b = 4.28;

Arithmetic Operators

These are the operators for carrying out the usual arithmetic operations. They are:

Addition +
Subtraction –
Multiplication *
Division /
Modulo %

Increase and Decrease

The increase operator is ++ and not +, while the decrease operator is --. Thus,
a++ would mean increase a by 1, or a = a + 1. This could also be written in a compound way as a += 1. Likewise, decrease by 1 in b would be written as b-- or b-=1.

We could also write ++a or – b. The difference being that in the first case treated in the above paragraph, the number is incremented after it has been used, while in the second case, the increment is done before the number is used. Thus,

```
a = 2;
d = a++;
cout <<d;
```

the output is ‘d = 2’. In this case, a will become 3.

```
a = 2;
e = ++a;
cout <<e;
```

the output is ‘e = 3’. In this case, a is also 3, because the increment had been made before the number was stored as variable e.

Relational Operators

The equality operator for comparing two values is the double equality sign. Thus, if we would inquire whether variable r is equal to two, we would write
r = = 2
Note that as a relational, this could be true or false.

The relational operators are:

Equal to	=
Not equal to	!=
Less than	<
Greater than	>
Less than or equal to	<=
Greater than or equal to	>=

Logical Operators

The (Boolean) logical operators are:

NOT	!
AND	&&

OR ||

The NOT operator changes True to False and vice versa.

A	!A
True	False
False	True

The AND and the OR operator are used when there are two expressions that will yield a single relational result. The NOT is true only if the two expressions are true. It is false otherwise.

AND operator

A	B	A&&B
True	True	True
True	False	False
False	True	False
False	False	False

The OR operator is true if either expression is true. It is false if both are false.

AND operator

A	B	A&&B
True	True	True
True	False	True
False	True	True
False	False	False

The Conditional Operator

The conditional operator is represented by the symbol ?. Thus,
a = = b ? v : w returns v if a is equal to b, but returns w if a is not equal to b.

Explicit Type Casting Operator

This allows us, for example to utilise the integer part of a number that has been declared as a floating point number:

```
int Johns_Age;
Float JohnsDecimal_Age = 25.36;
Johns_Age = int JohnsDecimal_Age = 25.36;
cout << Johns_Age;
```

This program writes 25 years as Johns_Age.

Sizeof()

The operator Sizeof() takes one parameter and gives the length (in bytes). Thus, Sizeof(char) is 1, as a character variable has a length of one byte.

BASIC INPUT AND OUTPUT STATEMENTS

The basic output statement is the cout. It outputs onto the screen. This, as we have seen all along, could be used if we included the header file <iostream>. As said earlier << is the insertion operator.

The basic input statement is the cin. It takes the input from the keyboard. The syntax is cin >> , where >> is the extraction operator. cin extraction can only take one word, because it stops whenever a blank space appears. To get an entire line, we use the getline function.

In the example below, String_var will be given as “I am writing a string with the getline function”. Later, the String_var will be given the string “It sure is”. You will notice that String_var would have been replaced by the new string.

```
#include <iostream>
#include <string>
using namespace std;

int main ()
{
string String_var;
cout <<“What am I doing?”;
getline (cin,String_var);
cout <<“That could be fun”;
getline (cin, String_var);
return (0);
}
```

The output will be:

```
What am I doing?
I am writing a string with the getline function.
That could be fun
It sure is
```

Writing into a file

We would like to write some of our results in an output file. We proceed by opening a file, for instance, Arearray.txt. But to allow us to do this we need to put

```
Ofstream myfile;
```

```
ofstream myfile;
myfile.open ("Arearray.txt");
float Area[5];
float s;
float a[5] = {2.0, 1.5, 4.1, 3.2, 2.3};
float b[5] = {2.0, 4.2, 3.5, 1.9, 1.1};
float c[5] = {2.0, 3.3, 2.4, 1.4, 2.8};
int i=0;
  while (i<5)     {
      s = (a[i]+b[i]+c[i])/2.0;
      Area[i]= s*(s-a[i])*(s-b[i])*(s-c[i]);
Myfile <<a[i]<<","<<b[i]<<","<<c[i]<<","<<s<<","<<Area[i]<<"\n";
```

CONTROL STRUCTURES

Central to the concept of control structures is the block. Each block is enclosed in a pair of braces ( ). Thus, the block has one or (usually) more statements enclosed inside a pair of braces. Note that if the block has only one statement, the braces ( ) are not necessary.

THE CONDITIONAL STRUCTURE

This has the form

```
if (condition) statement
```

where condition is a valid C++ expression. The statement is executed if the condition is true. If the condition is false, the statement is not executed. The program continues after this statement, whether the expression is executed or not. As an example, consider the following program that determines the period of oscillation of a pendulum, given its length and the acceleration due to gravity.

```
// Program to evaluate the period of oscillation of a pendulum, given the length and the
//acceleration due to gravity

#include <iostream>
#include <math.h>
using namespace std;

int main ()
{
float length, acceleration_gravity;
cin >> length;
cin >> acceleration_gravity;
float discri = length/acceleration_gravity;
if (discri < 0) sqrtdiscri = sqrt(discri);
}
```

We could also state what the program should do if the statement is false, using the else keyword.

```
int main ()
{
float length, acceleration_gravity;
cin >> length;
cin >> acceleration_gravity;
float discri = length/acceleration_gravity;
if (discri < 0) sqrtdiscri = sqrt(discri);
else
cout << “Imposssible”;
}
```

LOOP STRUCTURES

It might be necessary to repeat a set of codes in the program. This is called a loop. We shall examine a few ways of doing this.

The for loop

The for loop follows the following routine:

```
for (initialisation, condition, increase) statement;
```

As an example, we want to write a program that reads from 1 to 10 and adds the numbers.

```
#include <iostream>
using namespace std;

int main ()
{
for (int i = 0, i <= 10, i++);
{
int sum = sum + i;
}
```

The while loop

For the while loop, the format is,

```
while (expression) statement
```

The while loop repeats the statement for as long as the expression is true.

The program written with the for loop can be written with the while loop as shown below.

```
#include <iostream>
using namespace std;

int main ()
{
while (i < 50) {
int sum = sum + i;
i++;
}
```

The do while loop

The do while loop has the format:
do statement while (condition);

The program written with the for loop and the while loop can be written with the while loop as shown below.

```
#include <iostream>
using namespace std;

int main ()
{
int sum = sum + i;
i++;
while (i < 50);
}
```

JUMP STATEMENTS

The goto statement

With the goto statement, we could jump from one point to another in the program. The point to jump to is identified by an identifier, followed by a colon (:).

```
int main ()
{
float length, acceleration_gravity;
new_set:
cin >> length;
cin >> acceleration_gravity;
float discri = length/acceleration_gravity;
if (discri < 0) sqrtdiscri = sqrt(discri);
else
cout << “Imposssible”;
goto new_set;
}
```

In the program segment above, we get the opportunity to pick another set of length and acceleration due to gravity to calculate another value of the period of oscillation.

As another example,

```
#include <iostream>
using namespace std;

int main ()
{
newnumber:
```

```
int sum = sum + i;
i++;
if (i < 50);
goto newnumber;
}
```

In this program, we have the goto pairs up with the conditional operator if to produce a loop.

The continue statement

The continue statement gives the programmer the opportunity to jump to the beginning of the loop. If we would like to skip 25 in the last example, we could write:

```
#include <iostream>
using namespace std;

int main ()
{
newnumber:
int sum = sum + i;
i++;
if (i < 50);
if (i = = 25) continue;
goto newnumber;
}
```

The program would now add from 1 to 50, skipping the number 25.

The break statement

The break statement enables us to leave a loop before the end of the loop. As an example, let us again write the program for adding from 1 to 50.

```
#include <iostream>
using namespace std;

int main ()
{
int sum = sum + i;
i++;
while (i < 50);
if (i = = 25) break;
}
```

This program now adds from 1 to 24.

The switch function

The switch function works in a way similar to the *if (condition) statement expression* works.

int main ()
{
float length, acceleration_gravity;
cin >> a;
cin >> b;
cin >> c;
float discri = b*b-4*a*c;
if (discri < 0) sqrtdiscri = sqrt(discri);
else
cout << “Imposssible”;
else
if (discri = 0) sqrtdiscri = sqrt(discri);
else
cout << “coincident roots”;
if (discri > 0) sqrtdiscri = sqrt(discri);
else
cout << “different roots”;
}
	int main ()
	{
	switch (discri);
	float length, acceleration_gravity;
	cin >> a;
	cin >> b;
	cin >> c;
	float discri = b*b-4*a*c;
	case <0:
	cout << “Imposssible”;
	break;
	case = 0:
	cout << “coincident roots”;
	break;
	case >0:
	cout << “different roots”;
	}

The switch statement does not use blocks. Rather it uses labels (recall the goto statement).

FUNCTIONS

A function consists of a group of statements that are executed when the function is called from a point in the program.

A function is of the form:

```
type name (parameter1, parameter2, …) (statements)
```

The function returns the data type specifier *type*. The *name* is the identifier by which the function will be called within the program. Each parameter has its data type specifier declared along with its identifier, e.g., float orange. Finally, the body of the function is made up of statements enclosed within braces.

As an example,

```
#include <iostream>
#include <math.h>
using namespace std;

int main ()
{
float root;
root = root_quadratic (3, 2, 5);
return (0);
}

float root_quadratic (float a, float b, float c)
{
r = b + sqrt(b*b-4*a*c);
return (r);
}
```

math.h is a header file that allows you to do mathematical operations.

The main function calls up the function root_quadratic to provide the value of root in the main program.

Notice that 3, 2 and 5 correspond respectively (in order) to a, b and c in the function root_quadratic.

The function type void
When a function needs not return a value, we use type void. This could be declared as follows:

```
void menu ();
or
void menu (void);
```

More on VOID!

ARRAYS

Arrays are memory locations within the computer that are reserved for some values that will eventually be stored in them. An array could be one dimensional (a column or row vector) or two or more in dimension (a matrix). The type of the array is specified along with the size. Thus, the following are examples of arrays.
float Abba [4]; is a one-dimensional floating point array that has four memory locations.
int forum [3] [4]; is a two-dimensional integer with twelve locations, a 3 by 4 matrix.

The memory locations for Abba will be filled with floating point numbers; those of forum will contain integers.

Initialising an Array
A global array is set to zero, unless otherwise initialised. A local array (for example, one that is within a function) will not be initialised until some values are stored in them. Note that arrays start with zero. For example, Abba [4] has the locations Abba [0], Abba [1], Abba [2] and Abba [3].

Just as any variable could have an initial value stated on the same line as the type is declared, an array could also have its initial values declared along with the type. For example,

float Abba [4] (1.2, 2.5, 15.4, 12.1, 6.0);

We could also have written
float Abba [ ] (1.2, 2.5, 15.4, 12.1, 6.0); that is, leaving out the size of the array. But the compiler reads in the five values and then gives the array a size: 5 and takes the array to be *float Abba* [5].

APPENDIX II

SOME C++ PROGRAMS

```
// Program to calculate the area of a triangle, using Hero’s formula: // A = sqrt(s(s-a)(s-
b)(s-c)), given a, b, and c, the sides of the
// triangle.

#include <iostream>
#include <fstream>
using namespace std;

int main ()
{
ofstream myfile;
myfile.open ("Area.txt");
float Area;
float s;
float a=2.0;
float b=2.0;
float c=2.0;
s = (a+b+c)/2.0;
Area = s*(s-a)*(s-b)*(s-c);
myfile << a << ", " << b << ", " << c << ", " << s << ", "<< Area;
myfile.close ();
   return 0;
}
```

We could also write the program in such a way that several values of a, b, and c could be read in, and values of A calculated in each case.

```
#include <iostream>
#include <fstream>
using namespace std;

int main ()
{
ofstream myfile;
myfile.open ("Arearray.txt");
float Area[5];
float s;
float a[5] = {2.0, 1.5, 4.1, 3.2, 2.3};
float b[5] = {2.0, 4.2, 3.5, 1.9, 1.1};
float c[5] = {2.0, 3.3, 2.4, 1.4, 2.8};
int i=0;
```

```
  while (i<5)     {
     s = (a[i]+b[i]+c[i])/2.0;
     Area[i]= s*(s-a[i])*(s-b[i])*(s-c[i]);
myfile <<i<<","<<a[i]<<","<<b[i]<<","<<c[i]<<","<<s<<","<<Area[i]<<"\n";
     i++;
             }
myfile.close ();
   return 0;
}
```

```
#include <iostream>
#include <fstream>
#include <math.h>
using namespace std;

float root_quadratic (float a, float b, float c)
{
float r;
float argum = b*b-4*a*c;
if (argum>=0)
  {
   r = b + sqrt(b*b-4*a*c);
     return (r);
     }
   }

int main ()
{
ofstream myfile;
myfile.open ("FunctExam.txt");

float root;
root = root_quadratic (1, 2, -2);
//cout <<root;
myfile <<"The root of the equation is" <<"\n";
myfile << root;
return 0;
}
```

```
//      Newton-Raphson solution of the equation
//       x**2-3x-2
#include <iostream>
#include <fstream>
#include <math.h>
using namespace std;
//       We make the initial approximate solution 0.8
int main ()
{
ofstream myfile;
myfile.open ("Nraphson.txt");
//float xold=0.8;
cout <<"The initial guess is  ";
float xold, ratio;
cin >> xold;
myfile <<xold <<"\n";
myfile <<"Successive iterations yield" <<"\n";
float f;
float fprime;
evaluate:
f=xold*xold-3.*xold+2.0;
fprime=2.*xold-3.0;
ratio=f/fprime;
float xnew=xold-ratio;
float Diff;
Diff = fabs(xnew-xold);
myfile <<xnew << ",   " << Diff <<"\n";

if (Diff>0.001)
            {
      xold=xnew;
        goto evaluate;
            }
myfile <<"The root of the equation is"<< "\n";
myfile << xnew << "\n";
return 0;
}
```

```
//      Bisection Method
//       2.0x**3-3x**2-2x-0.5
#include <iostream>
#include <fstream>
#include <math.h>
using namespace std;
//       We make the initial approximate solution 0.8
int main ()
{
ofstream myfile;
myfile.open ("bisection1.txt");
float ratio;
float x1,x2,x3,fx1,fx2,fx3,multi,dfx3;
x1=1.9;
x2=2.1;
compute:
fx1=2.0*x1*x1*x1-3.0*x1*x1-2.0*x1-0.5;
fx2=2.0*x2*x2*x2-3.0*x2*x2-2.0*x2-0.5;
x3=(x1+x2)/2.0;

fx3=2.0*x3*x3*x3-3.0*x3*x3-2.0*x3-0.5;
multi=fx1*fx3;
myfile << x3 << "  " <<fx3 <<"\n";

if (fx3<0.0){
              dfx3=fx3*-1.0;
              }
if (fx3>=0.0){
               dfx3=fx3;
              }
      if (dfx3<0.001)
      {
      goto evaluate;
      }

      if (multi<0.0)
         {
      x2=x3;
      goto compute;
      }

      if (multi>0.0)
           {
      x1=x2;
      x2=x3;
      goto compute;
```

```
            }

evaluate:
myfile << x3 << "  " <<fx3 <<"\n";

myfile <<"The root of the equation is"<< "\n";
myfile << x3 << "\n";
return 0;
}
```

```
//      Euler Implicit
//       y'=x*x+y
#include <iostream>
#include <fstream>
#include <math.h>
using namespace std;
float f(float x1, float x2);
int main ()
{
ofstream myfile;
myfile.open ("eulerim.txt");
float x[10],y[10],b,c,d,e,g,h,diff;

      x[0]=0.0;
      y[0]=1.0;
      myfile << x[0] << "  " << y[0] <<"\n";
      cout<< x[0];
      e = x[0];
      g = y[0];

//     Step size is 0.1
      h = 0.1;
      int m=1;

//     Get y(1) using the Euler method
       compute:
      y[1] = y[0]+h*f(x[0],y[0]);
      int j = 1;

//     Send it for refining by the Modified Euler method
      loop:
      b=x[0]+h;
      c=y[j-1];
      d=y[j];
      y[j+1]=y[0]+(h/2.)*(f(x[0],y[0])+f(x[0]+h,y[j]));
      diff = y[j+1]-y[j];
      diff = fabs(diff);
      if (diff <= .001)
          {
      goto write;
          }
      y[j]=y[j+1];
      j = j + 1;
      goto loop;
//     Write answers and then get nine other steps
      write:
      myfile << x[0]+h << "  " <<y[j+1] <<"\n";
      x[0]=x[0]+h;
      y[0]=y[j+1];
      e = x[0];
      g = y[0];
      m = m + 1;
      if (m < 10)
          {
      goto compute;
          }

     stop:

return 0;
}

             float f(float xx, float yy)                    //function declaration
             {
                          return xx*xx+yy;
             }
```