Introduction to Numerical Computing

Numerical method is an approach to find an approximate solution to complex mathematical problems using only simple arithmetic operations namely addition, subtraction, multiplication, division and functional evaluations.

- Numerical methods are usually employed for problem for which an analytical result either cannot be obtained or would require too much work.
- *Numerical analysis* treats the subject matter of error analysis extensively. On the other hand, in *Numerical methods*, focus is given more towards the techniques of solving non-analytical problems rather than on the errors that accompany these methods.
- Numerical computations invariably involve a large number of arithmetic calculations and, therefore, require fast and efficient computing devices. The microelectronics revolution and the subsequent development of high power, low cost, personal computers have had a profound impact on the application of numerical computing methods to solve scientific problems.
- The traditional numerical computing methods usually deals with the following topics:
  - Finding roots of equations.
  - Solving system of linear algebraic equations
  - Interpolation & Regression Analysis
  - Numerical Integration
  - Numerical Differentiation
  - Solving of Differential Equation
  - Boundary value Problems
  - Solution of matrix problems

Numeric Data

Numerical computing may involve two types of data: *discrete data* and *continuous data*.

- Data that are obtained by counting are called discrete data. Examples of discrete data are total number of students in a class, total number of items in a box etc.
- Data that are obtained through measurement are called continuous data. Example of continuous data is the temperature of a patient as measured by a thermometer.

Process of Numerical Computing

Numerical computing involves formulation of mathematical models of physical problems that can be solved using basic arithmetic operations. The process of numerical computing can be roughly divided into the following four phases which are illustrated in Fig-1.1 [Page-4, Balagurusamy]:

1. Formulation of a mathematical model
2. Construction of an appropriate numerical method
3. Implementation of the method to obtain a solution
4. Validation of the solution

- The formulation of a suitable mathematical model is critical to the solution of the problem. A mathematical model can be broadly defined as a formulation of certain mathematical equation that expresses the essential features of a physical system or process. Models may range from a simple algebraic equation to a complex set of differential equations. [Fig-1.2, Balagurusamy, Page-5]
- Once a mathematical model is available, our first step would be to try to obtain an explicit analytical solution. In most cases, the mathematical models may not be amenable to analytical solutions or they may not be solved efficiently using analytical techniques. In such cases, we have to construct appropriate numerical methods to solve mathematical models. For a given problem, there might be several alternative numerical methods. We must consider different factors or trade-offs before selecting a particular method- such as type of equation, type of computer available, accuracy, speed of execution, and programming and maintenance efforts required.
- The third phase of the numerical computing process is the implementation of the method selected. This phase is concerned with the following three tasks:
    i. design of an algorithm
    ii. writing of a program
    iii. executing it on a computer to obtain the results.
- Once we are able to obtain the results, the next step is the validation of the process. Validation means the verification of the results to see that it is within the desired limits of the accuracy. If it is not, then we must go back and check each of the following
    i. mathematical model itself
    ii. numerical method selected
    iii. computational algorithm used to implement the method.

  This may mean modification of the model, selection of an alternative numerical method or improving the algorithm (or a combination of them). Once a modification is introduced, the cycle begins again. [Fig-1.3, Balagurusamy, Page-7]

Characteristics of Numerical Computing

Numerical methods exhibit certain computational characteristic during their implementation. It is important to consider these characteristics while choosing a particular method for implementation. The characteristics that are critical to the success of implementation are:

1. Accuracy
2. Rate of convergence
3. Numerical stability
4. Efficiency

Accuracy: Every method of numerical computing introduces errors. Different type of errors like truncation errors or round off errors occurs which affect the accuracy of the results. The results we obtain must be sufficiently accurate to serve the purpose for which the mathematical model was built.

Rate of convergence: Many numerical methods are based on the idea of an iterative process. This process involves generation of a sequence of approximations with the hope that the process will converge to the required solution. Certain methods converge faster than others. Some methods may not converge at all. It is, therefore, important to test for convergence before a method is used.

Numerical Stability: Errors introduced into computation, from whatever source, propagate in different ways. In some cases these errors tend to grow exponentially, with disaster computational results. A computing process that exhibits such exponential error growth is said to be numerically unstable. We must choose methods that are not only fast but also stable.

- Numerical instability may also arise due to *ill-conditioned* problems. There are many problems which are inherently sensitive to round off errors and other uncertainties. Thus, we must distinguish between sensitivity of the methods and sensitivity inherent in problems. When the problem is ill-conditioned, there is nothing we can do to make a method to become numerically stable.

Efficiency: Efficiency in numerical method means the amount of effort required by both human and computer to implement the method. A method that requires less of computing time and less of programming effort and yet achieves the desired accuracy is always preferred.

New trends in Numerical computing

In recent years, the increasing power of computer hardware has affected the approach of numerical computing in several ways. It has forced scientists and engineers to search for algorithms that computationally fast and efficient. An important new trend is the construction of algorithms to take advantage of specialized computer hardware such as vector computers and parallel computers. Another trend is the use of sophisticated interactive graphics, in which the user can view the results graphically and advice the computer, graphically, on how to proceed further. Object oriented numerical computing is gaining importance due to the popularity of languages like C++ and java.

Representation of Numbers

All modern computers are designed to use binary digits to represent numbers and other information. The memory is usually organized into strings of bits called *words*.

The largest number a computer can store depends on its word length. For example, the largest binary number a 16 bit word can hold is 16 bits of 1. This binary number is equivalent to a decimal value of 65535. The following relation gives the largest decimal number that can be stores in a computer:

Largest number = $2^n - 1$

where n is the word length in bits. Thus, we see that the greater number of bits, the larger the number that may be stored.

Integer representation

Decimal numbers are first converted into the binary equivalent and then represented in either integer or floating-point form.

For integers, the decimal or binary point is always fixed to the right of the least significant digit and therefore, fraction are not included. The magnitude of the number is restricted to $2^n - 1$ where n is the word length in bits.

Again, negative numbers are stored by using the 2's complement. This is achieved by taking the 1's complement of the binary representation of the positive number and then adding 1 to it. [Example 3.8, Balagurusamy, Page-50]

If we reserve one bit to represent the sign of the number, called *sign bit*, we have only n-1 bits to represent the number. Thus a 16bit word can contain numbers -2^{15} to $2^{15} - 1$ (i,e. –32768 to 32767). Generally, if the sign bit is 1, the number is negative and if the sign bit is 0, the number is positive.

Floating point Representation

Floating-point numbers are stored and processed differently. The entire memory location is divided into three fields called sign , exponent and mantissa as shown in Fig 3.1 [page-51, Balagurusamy]. Typically, floating numbers use a field width of 32 bits where 24 bits are used for the mantissa, 7 bits for the exponent and 1 bit for sign.

For example, 35.7812 can be expressed 0.357812 * 10^2. Similarly, the number 987654321 can be expressed as 0.987654 * 10^9. By writing a large number in exponential form, we may lose some digits. If x is a real number, its floating-point form representation is $x = f * 10^E$ where the number f is called mantissa and E is the exponent.

Example- 3.10: Page 52, Balagurusamy.

- The shifting of decimal points to the left of the most significant digit is called normalization and the numbers represented in normalized form are known as normalized floating point numbers.
- Mantissa should satisfy the following conditions:
  For positive numbers: less than 1.0 but greater than or equal to 0.1.
  For negative numbers: greater than -1.0 but less than or equal to -0.1.
  That is $0.1 \leq |f| < 1$.