National University -FAST

CS 325
Numerical Computing

OR
Numerical Methods
Numerical Analysis

Introduction to Numerical Computing Methods

- WHAT IS NUMERICAL **COMPUTING ?**
- WHY DO WE NEED THEM?

Numerical Computing :

- is study of Algorithms that are used to obtain numerical (approximate) solutions of a mathematical problem.

- is concerned with how to solve a problem numerically, i.e., how to develop a sequence of numerical calculations to get a satisfactory answer.

$$\left.\begin{aligned} x^9 - 2x^2 + 5 = 0 \\ x = e^{-x} \end{aligned}\right\}$$

Why do we need them?

1. No analytical solution exists,
2. An analytical solution is difficult to obtain

Course Outline:

1- Error analysis:

2- Solution(Root) of equations in one variable:

3-Interpolation and Polynomial approximation:

4-Numerical differentiation and Integration:

5-Differential Equations:

6-Direct Method for solving linear system:

7-Iterative Techniques for solving linear system:

8-Difference Operator analysis:

Text Book: Numerical Analysis , Burden and Faires , 9th Ed

Number Representation and Accuracy

- NUMBER REPRESENTATION
- NORMALIZED FLOATING POINT REPRESENTATION
- SIGNIFICANT DIGITS
- BITS AND BYTE
- ACCURACY AND PRECISION
- SINGLE AND DOUBLE PRECISION
- ALGORITHM AND FLOW CHART

- ROUNDING AND CHOPPING
- ABSOLUTE , RELATIVE AND PERCENTAGE ERROR
- LOSS OF SIGNIFICANCE

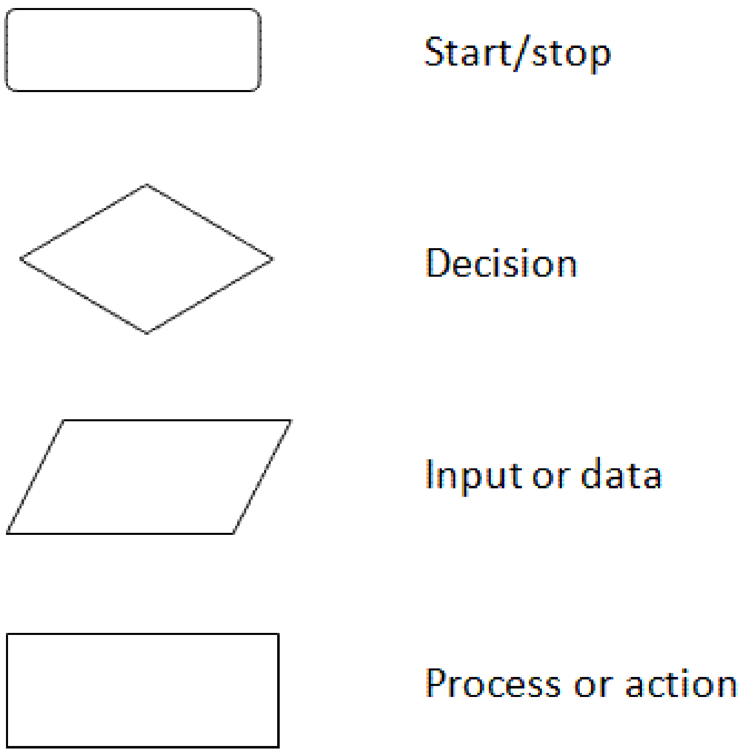
READING ASSIGNMENT:

Algorithm:

- To write a logical step-by-step method to solve the problem is called algorithm, in other words,
- An algorithm includes calculations, reasoning and data processing.

Flow chart :

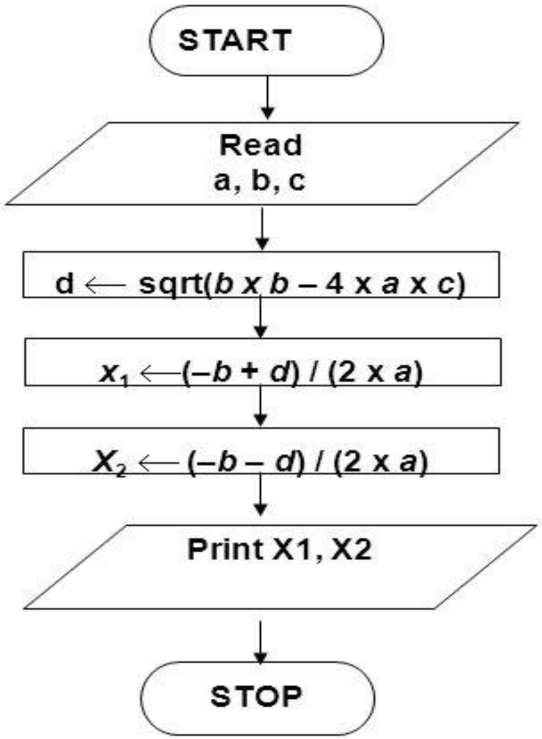
A flowchart is the graphical or pictorial representation of an algorithm with the help of different symbols



Example:

Problem: Write Algorithm and Flowchart to find solution of Quadratic equation

- **Algorithm:**
- Step 1: Start
- Step 2: Read a, b, c
- Step 3: $d \leftarrow \text{sqrt}(\ b \times b - 4 \times a \times c\ )$
- Step 4: $x1 \leftarrow (-b + d) / (2 \times a)$
- Step 5: $x2 \leftarrow (-b - d) / (2 \times a)$
- Step 6: Print $x1$, $x2$
- Step 7: Stop



Representing Real Numbers

You are familiar with the decimal system:

$$312.45 = 3\times10^{2} + 1\times10^{1} + 2\times10^{0} + 4\times10^{-1} + 5\times10^{-2}$$

Decimal System: Base = 10 , Digits (0,1,...,9)

Standard Representations:

$$\begin{array}{ccc} \pm & 3\;1\;2 & . & 4\;5 \\ \text{sign} & \text{integral part} & & \text{fraction part} \end{array}$$

Normalized Floating Point Representation

$$\pm \quad \underline{d.\ f_1\ \ f_2\ \ f_3\ \ f_4} \times 10^{\pm n}$$

sign mantissa exponent

$d \neq 0, \qquad \pm n : \text{signed exponent}$

- Scientific Notation: Exactly one non-zero digit appears before decimal point.
- Advantage: Efficient in representing very small or very large numbers.

Binary System:

Binary System: Base = 2, Digits {0,1}

$$\pm \quad \underbrace{1.\ f_1\ \ f_2\ \ f_3\ \ f_4}_{\text{mantissa}} \times 2^{\pm n}$$

sign mantissa signed exponent

$$(1.101)_2 = (1 + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3})_{10} = (1.625)_{10}$$

$$(1.1)_{10} = (1.000110011001100...)_2$$

You can never represent 1.1 exactly in binary system.

Significant digits are those digits that can be used with confidence.

Rules:

- **Non zero numbers are always significant**

  1.23 45.6 6,7263

- **In between zeros are always significant**

  1.005 70206

- **Leading zeros are never significant**

  0.0055 0.0302

- **Trailing zeros are some time significant**

  70,000 70,000. 1,030 1030.0000

IEEE 754 Floating-Point Standard

Single and double precision

Single Precision (32 bit)

23 bits used for significant digits

8 bit used for store exponent

1 bit used for to store sign (+,-)

Double precision: (64 bit)

52 for significant digits ,

11 bit for exponent

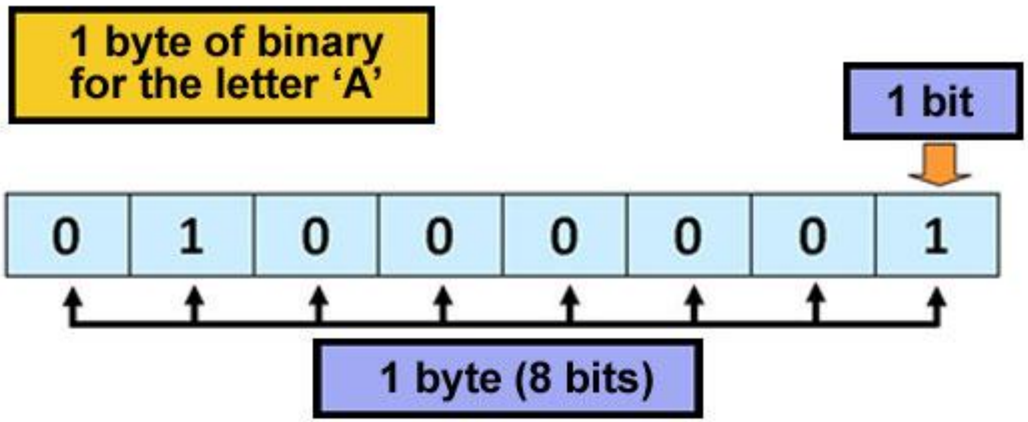
1 bit for sign

Bits and byte :

The **byte** is a unit of digital information that most commonly consists of eight **bits**.

Historically, the **byte** was the number of **bits** used to encode a single character of text in a **computer** and for this reason it is the smallest addressable unit of memory in many **computer** architectures.

ASCII table :



Bits and byte :

The **byte** is a unit of digital information that most commonly consists of eight **bits**.

bits used to encode a single character of text in a **computer**

How to Convert Bits and Bytes:

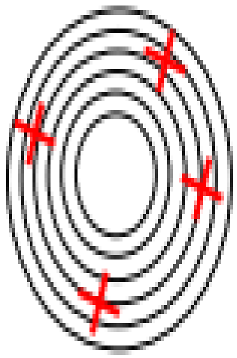
- 8 bits = 1 byte
- 1,024 bytes = 1 kilobyte
- 1,024 kilobytes = 1 megabyte
- 1,024 megabytes = 1 gigabyte
- 1,024 gigabytes = 1 terabyte

Accuracy and Precision

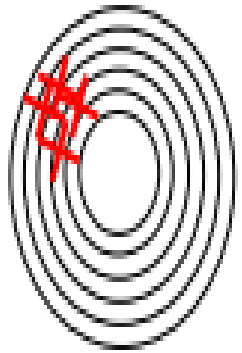
- Accuracy is related to the closeness to the true value.

- Precision is related to the closeness to other estimated values.

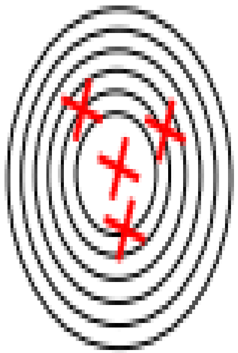
Nether Precise NOR accurate:



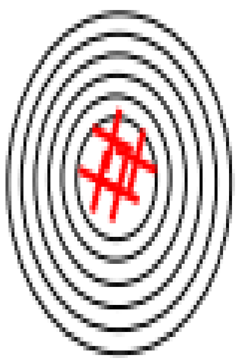
Precise, but NOT accurate:



Accurate but NOT precise:



Precise AND accurate



Rounding and Chopping

Rounding: Replace the number by the nearest

machine number. OR

its impossible to represent all real numbers exactly on machine with finite

Chopping: Throw all or drop the extra digits.

Error: is difference between an approximation of number used in computation and its exact value

OR Error = True value – approximate value

Example: Round vs Chop

$\sqrt{2}$ =1.414213562373095048801168872

∏ = 3.141592653589793238462643383

∏round =3.1416

∏chop =3.1415

ERROR Analysis:

Truncation Error:

are when an iterative method is terminated

OR mathematical procedure is approximated and

approximate solution differs from exact solution

Discretization Error :

are committed when a solution

of discrete problem does not coincide with solution

of continuous problem

Error in CM – True Error

Can be computed if the true value is known:

Absolute Error:

$$AE = \left| \text{true value} - \text{approximation} \right|$$

Absolute Relative Error:

$$ARE = \left| \frac{\text{true value} - \text{approximation}}{\text{true value}} \right|$$

Error in CM – Estimated Error

When the true value is not known:

$$\text{Estimated Absolute Error}$$

$$AE = \left| \text{current estimate} - \text{previous estimate} \right|$$

$$\text{Estimated Absolute Relative Error}$$

$$ARE = \left| \frac{\text{current estimate} - \text{previous estimate}}{\text{current estimate}} \right|$$

Loss of significance:

occurs in numerical calculations when too many significant digits cancel

In some cases, the relative error involved in arithmetic calculations can grow significantly large. This often involves subtracting two almost equal numbers.

For example, consider the case $y = x - \sin(x)$. Let us have a computing system which works with ten decimal digits. Then

$$
\begin{aligned}
x &= 0.66666\,66667 \times 10^{-1} \\
\sin x &= 0.66617\,29492 \times 10^{-1} \\
x - \sin x &= 0.00049\,37175 \times 10^{-1} \\
&= 0.49371\,75000 \times 10^{-4}
\end{aligned}
$$

Thus the number of significant digits was reduced by three! Three **spurious zeros** were added by the computer to the last three decimal places, but these are not significant digits. The correct value is $0.49371\,74327 \times 10^{-4}$.

Loss of precision theorem

Exactly how many significant binary digits are lost in the subtraction $x - y$ when x is close to y?

Let x and y be normalized floating-point numbers with $x > y > 0$. If $2^{-p} \leq 1 - y/x \leq 2^{-q}$ for some positive integers p and q, then at most p and at least q significant binary digits are lost in the subtraction $x - y$.

Example.

How many significant bits are lost in the subtraction $x - y = 37.593621 - 37.584216$?

We have

$$1 - \frac{y}{x} = 0.0002501754$$

This lies between $2^{-12} = 0.000244$ and $2^{-11} = 0.000488$. Hence, at least 11 but at most 12 bits are lost.

Avoiding loss of significance:

i. Rationalizing

Consider the function

$$f(x) = \sqrt{(x^2+1)} - 1$$

We see that near zero, there is a potential loss of significance.

However, the function can be rewritten in the form

$$f(x) = (\sqrt{(x^2+1)} - 1)\left(\frac{\sqrt{(x^2+1)}+1}{\sqrt{(x^2+1)}+1}\right)$$

$$= \frac{x^2}{\sqrt{(x^2+1)}+1}$$

ii. Using series expansion

Consider the function

$$f(x) = x - \sin x$$

whose values are required near $x = 0$. We can avoid the loss of significance Taylor series for $\sin x$

$$\sin x = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} - \ldots$$

For x near zero, the series converges quite rapidly.

We can now rewrite the function f as

$$f(x) = x - \left(x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} - \ldots \right) = \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} - \ldots$$

iii. Using trigonometric identities

As a simple example, consider the function

$$f(x) = \cos^2(x) - \sin^2(x)$$

There will be loss of significance at $x = \pi/4$.

The problem can be solved by the simple substitution

$$\cos^2(x) - \sin^2(x) = cos(2x)$$

Example:

Consider using the Taylor series approximation for e^x to evaluate e^{-5}:

$$e^{-5} = 1 + \frac{(-5)}{1!} + \frac{(-5)^2}{2!} + \frac{(-5)^3}{3!} + \frac{(-5)^4}{4!} + \dots$$

Degree	Term	Sum	Degree	Term	Sum
0	1.000	1.000	13	-0.1960	-0.04230
1	-5.000	-4.000	14	0.7001E-1	0.02771
2	12.50	8.500	15	-0.2334E-1	0.004370
3	-20.83	-12.33	16	0.7293E-2	0.01166
4	26.04	13.71	17	-0.2145E-2	0.009518
5	-26.04	-12.33	18	0.5958E-3	0.01011
6	21.70	9.370	19	-0.1568E-3	0.009957
7	-15.50	-6.130	20	0.3920E-4	0.009996
8	9.688	3.558	21	-0.9333E-5	0.009987
9	-5.382	-1.824	22	0.2121E-5	0.009989
10	2.691	0.8670	23	-0.4611 E-6	0.009989
11	-1.223	-0.3560	24	0.9607 E-7	0.009989
12	0.5097	0.1537	25	-0.1921 E-7	0.009989

Table. Calculation of $e^{-5} = 0.006738$ using four-digit decimal arithmetic

There are loss-of-significance errors in the calculation of the sum.
To avoid the loss of significance is simple in this case:

$$e^{-5} = \frac{1}{e^5} = \frac{1}{\text{series for } e^5}$$

and form e^5 with a series not involving cancellation of positive and negative terms.

Motivation:

To introduce modern approximation techniques; to explain how, why, and when they

can be expected to work; and to provide a foundation for further study of numerical

analysis and scientific computing.