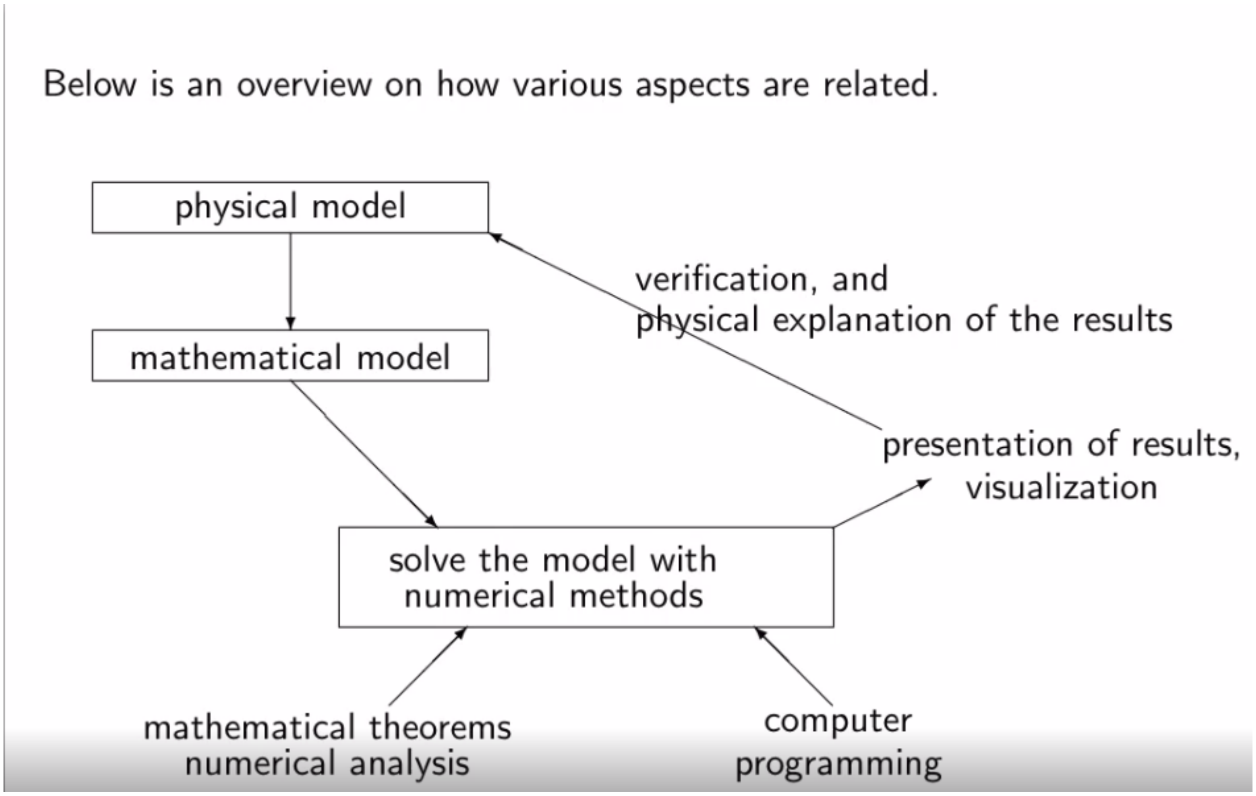
Chapter 1: Computer Arithmetic

Contents

1 Introduction to Numerical Computation

1.1 Numerical Methods

They are **algorithms** that compute **approximations** to functions, their derivatives, their integrations, and solutions to various equations etc. Such algorithms could be programmed on a computer.



Numerical methods are not about numbers. It is about mathematical ideas and insight. A little idea can go a long way. Some classical problems:

- Development of algoritms
- Implementation
- Some analysis, including error-estimates, convergence, stability, etc.

Matlab

2 Representation of numbers in different bases

There have been several different ways of representing numbers:

- 10: decimal, daily use
- 2: binary, computer use
- 8: octal
- 16: hexadecimal, ancient China
- 20: vigesimal, ancient France (can be seen in french language)
- 60: sexagesimal, used by Babylonians

In principle we can use any number β as the base. Writing such a number with a decimal point gives us an integer part and fraciton part.

$$
\begin{aligned}
&\Big(\overbrace{a_n a_{n-1} \cdots a_1 a_0}^{\text{integer part}} . \overbrace{b_1 b_2 b_3 \cdots}^{\text{fractional part}}\Big)_\beta \\
= \quad & a_n\beta^n + a_{n-1}\beta^{n-1} + \cdots + a_1\beta + a_0 \qquad \text{(integer part)} \\
& +b_1\beta^{-1} + b_2\beta^{-2} + b_3\beta^{-3} + \cdots \qquad \text{(fractonal part)}
\end{aligned}
$$

This above formula allows us to convert a number in any base β into base 10.

2.1 Examples

2.1.1 Octal to Decimal

We have $(45.12)_8$.

$$4 * 8^1 + 5 * 8^0 + 1 * 8^{-1} + 2 * 8^{-2} = (37.15625)_{10}$$

2.1.2 Octal to Binary

$$
\begin{aligned}
(1)_8 &= (1)_2 \\
(2)_8 &= (10)_2 \\
(3)_8 &= (11)_2 \\
(4)_8 &= (100)_2 \\
(5)_8 &= (101)_2 \\
(6)_8 &= (110)_2 \\
(7)_8 &= (111)_2 \\
(10)_8 &= (1000)_2
\end{aligned}
$$

Since 8 is a power of 2 it makes it very easy to convert from octal to binary:

$$(5034)_8 = (101000011100)_2$$

Each digit in octal gets converted to 3 digits in binary. And vice versa...

You can also go back from binary to octal:

$$(110010111001)_2 = (6271)_8$$

2.1.3 Decimal to Binary

Write $(12.45)_{10}$ in binary. (base 2) This is particularly interesting because we use decimal and computer uses binary. This conversion takes two steps. First we convert the integer part into binary.

Procedure: Divide the integer by 2 and store the remainder of each step until integer is zero. (Euclid's algorithm.)

$$
\begin{array}{r|ll}
 & & (\textit{remainder}) \\
2 & \underline{12} & 0 \\
2 & \underline{6} & 0 \\
2 & \underline{3} & 1 \\
2 & \underline{1} & 1 \\
 & 0 &
\end{array}
\quad \Rightarrow (12)_{10} = (1100)_2
$$

Now to convert the fractional part to binary you multiply by 2 and store the integer part for the result.

$$
\begin{array}{l|ll}
0.45 & \times & 2 \\
\underline{0}.9 & \times & 2 \\
\underline{1}.8 & \times & 2 \\
\underline{1}.6 & \times & 2 \\
\underline{1}.2 & \times & 2 \\
\underline{0}.4 & \times & 2 \\
\underline{0}.8 & \times & 2 \\
\underline{1}.6 & \times & 2 \\
\cdots & &
\end{array}
\quad \Rightarrow (0.45)_{10} = (0.01110011001100\cdots)_2.
$$

Note that the fractional part here is not finite. Putting them together:

$$(12.45)_{10} = (1100.01110011001100...)_2$$

So how do we store this kind of a number in a computer?

3 Floating Point Representation

4 Loss of Significance

5 Review of Taylor Series

6 Finite Difference Approximation