Lecturer

- Parq, Jae-Hyeon
  - Researcher & Lecturer in College of Natural Sciences, SNU
  - E-mail: parkq2@snu.ac.kr
  - Office: 25-1 316A

Main References

- Wen Shen,
  An Introduction to Numerical Computation

- Suli,
  Lecture Notes on Finite Element Methods for Partial Differential Equations

- Liu & Quek,
  The Finite Element Method (2nd Edition)

Lecture Plan

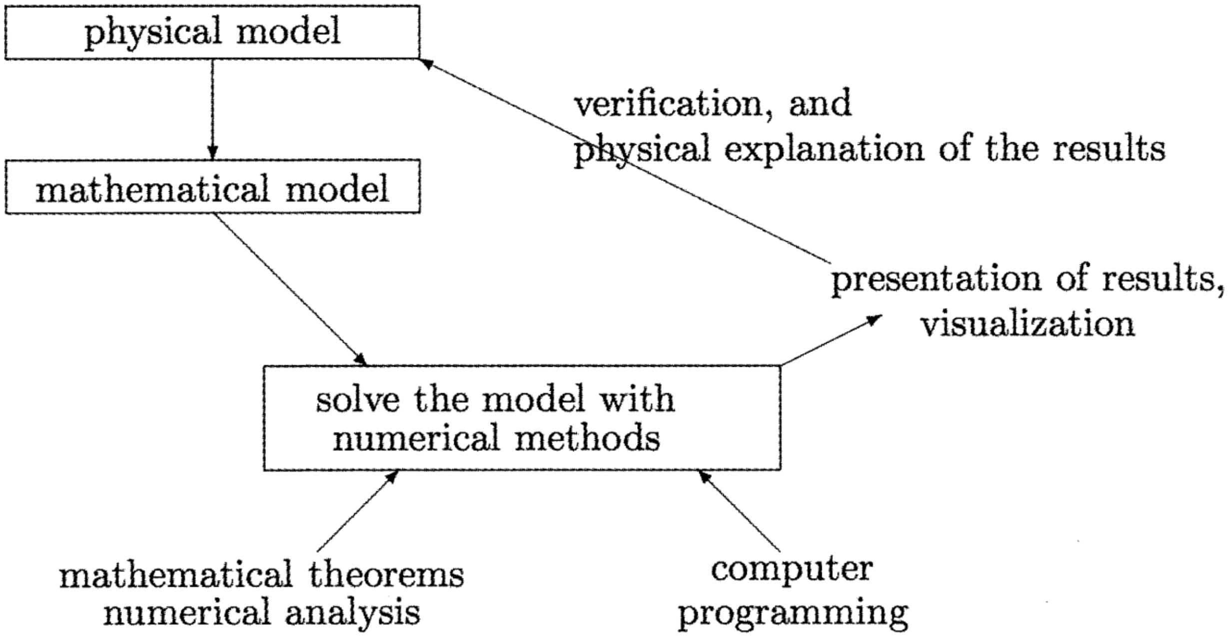
- Computer arithmetic
- Interpolation
- Differentiation & Integration
- Solving non-linear equations, Least Squares
- Linear systems
- ODE - IVP
- Midterm exam.
- ODE - BVP
- PDE - general
- FDM
- FEM
- BEM/SEM
- Final exam.
- LBM

Computer Arithmetic

IPCST
Seoul National University

Numerical Methods

- They are algorithms that compute approximate solutions to a number of problems for which exact solutions are not available.
  - Wen Shen



Representation of Numbers

- Bases used in human history
  - 10
  - 2 (computer)
  - 8
  - 16 (ancient China)
  - 20 (ancient France)
  - 60 (Babylon)
- In base β,

$$\left(\overbrace{a_n a_{n-1} \cdots a_1 a_0}^{\text{integer part}} . \overbrace{b_1 b_2 b_3 \cdots}^{\text{fractional part}}\right)_\beta$$

$$= a_n\beta^n + a_{n-1}\beta^{n-1} + \cdots + a_1\beta + a_0 \qquad \text{(integer part)}$$

$$+ b_1\beta^{-1} + b_2\beta^{-2} + b_3\beta^{-3} + \cdots \qquad \text{(fractonal part)}$$

Floating-Point Representation

$$x = \pm r \times \beta^e, \qquad 1 \le r < \beta$$

$\beta = 2$ *in computer arithmetic*

- Single-precision (32 bits)

$$-126 \le e \le 127$$

  – About 10^{-38}~10^{38}, 7 or 8 decimal digits
  (sign: 1 bit, exponent: 8 bits, mantissa: 23 bits)

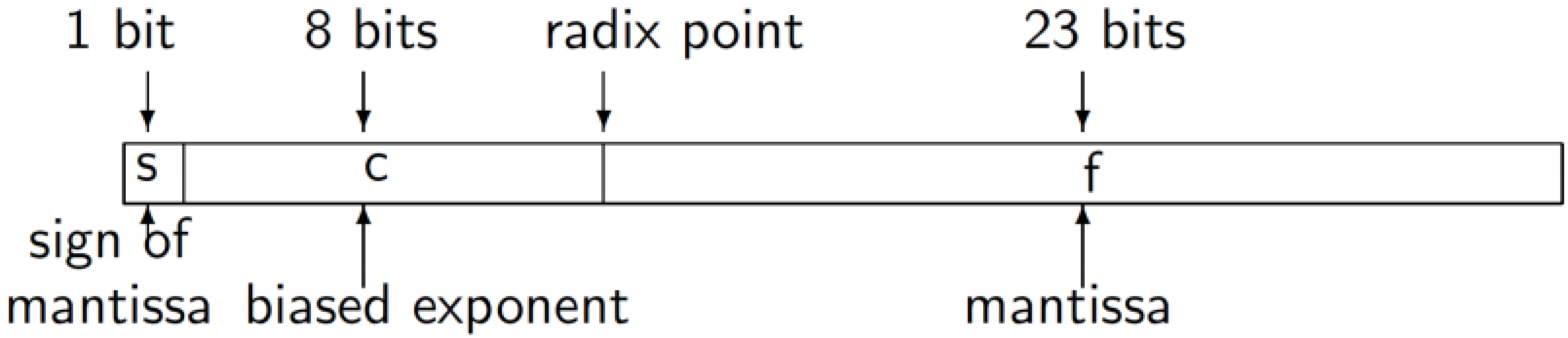
- Double-precision (64 bits)

$$-1022 \le e \le 1023$$

  – About 10^{-308}~10^{308}, 15 or 16 decimal digits
  (sign: 1 bit, exponent: 11 bits, mantissa: 52 bits)

Floating-Point Representation

- Information to be stored:
  - The sign
  - The exponent e
  - The value $r \left(= 1 + f = 1 + n/2^b\right)$
    - $b = 23$ for single precision, $0 \leq n < 2^b$ (integer)



32-bit single precision bit layout (Figure from Wen Shen)

Floating-Point Representation

- Let’s say $-m \leq e \leq M$

- Maximum absolute value = $\left(2 - 2^{-b}\right) \times 2^{M}$
- Minimum absolute value = 2^{-m}

- Inf: infinity ($e = M + 1$ & $f = 0$)
  - Cf.) NaN: not a number ($e = M + 1$ & $f > 0$)
- 0: zero ($e = -m - 1$)

- (Arithmetic) overflow: $|x| > \left(2 - 2^{-b}\right) \times 2^{M}$ (→ Inf)
- (Arithmetic) underflow: $|x| < 2^{-m}$ (→ 0)

Floating-Point Representation

- The floating-point representation $\mathrm{fl}(x)$

$$\mathrm{fl}(x) = x \cdot (1 + \delta)$$

- Relative error $\delta = \frac{\mathrm{fl}(x) - x}{x}$
- Absolute error $= \mathrm{fl}(x) - x = \delta \cdot x$

$$|\delta| \leq \varepsilon$$

  - In case of round-off, $|\delta| \leq \varepsilon/2$

- **Machine epsilon** $\varepsilon = 2^{-b}$

Floating-Point Representation

$$\alpha \leq \gamma < \beta$$

$$\alpha = \left(1 + \frac{n}{2^b}\right) \times 2^e$$

$$\beta = \left(1 + \frac{n+1}{2^b}\right) \times 2^e$$

round-off!

$$x$$

$$x = \alpha \text{ or } \beta$$

Floating-Point Representation

- Error propagation
  - Errors can be accumulated through multiple arithmetic operations.
  - Example 1.4 in Wen Shen

$$\mathrm{fl}(x) = x(1+\delta_x), \qquad \mathrm{fl}(y) = y(1+\delta_y)$$

$$\begin{aligned}
\mathrm{fl}(z) &= \mathrm{fl}\left(\mathrm{fl}(x) + \mathrm{fl}(y)\right) \\
&= \left(x(1+\delta_x) + y(1+\delta_y)\right)(1+\delta_z) \\
&= (x+y) + x\cdot(\delta_x+\delta_z) + y\cdot(\delta_y+\delta_z) + (x\delta_x\delta_z + y\delta_y\delta_z) \\
&\approx (x+y) + x\cdot(\delta_x+\delta_z) + y\cdot(\delta_y+\delta_z)
\end{aligned}$$

$$\text{relative error} = \frac{\mathrm{fl}(z)-(x+y)}{x+y} = \underbrace{\frac{x\delta_x + y\delta_y}{x+y}}_{\text{propagated err}} + \underbrace{\delta_z}_{\text{round off err}}$$

Floating-Point Representation

- Error propagation
  - Errors can be accumulated through multiple arithmetic operations.
  - Example 1.4 in Wen Shen

$$\mathrm{fl}(x) = x(1+\delta_x), \qquad \mathrm{fl}(y) = y(1+\delta_y)$$

$$\begin{aligned}
\mathrm{fl}(z) &= (x+y) + x\cdot(\delta_x+\delta_z) + y\cdot(\delta_y+\delta_z) + (x\delta_x\delta_z + y\delta_y\delta_z) \\
&\approx (x+y) + x\cdot(\delta_x+\delta_z) + y\cdot(\delta_y+\delta_z)
\end{aligned}$$

$$\begin{aligned}
\text{absolute error} &= \mathrm{fl}(z) - (x+y) = x\cdot(\delta_x+\delta_z) + y\cdot(\delta_y+\delta_z) \\
&= \underbrace{\underbrace{x\cdot\delta_x}_{\substack{\text{abs. err.}\\ \text{for } x}} + \underbrace{y\cdot\delta_y}_{\substack{\text{abs. err.}\\ \text{for } y}}}_{\text{propagated error}} + \underbrace{(x+y)\cdot\delta_z}_{\text{round off err}}
\end{aligned}$$

Loss of Significance

- Loss of significant digits typically happens in subtraction of two very close numbers

$$\begin{array}{r} 1.2345678 \\ -1.2344444 \\ 0.0001234 \end{array}$$

4 significant digits disappear.

Loss of Significance

- Example 1.5 in Wen Shen
  - Find roots of $x^2 - 40x + 2 = 0$. Use 4 significant digits.

$$r_{1,2} = \frac{1}{2a}\left(-b \pm \sqrt{b^2 - 4ac}\right)$$

$$x_{1,2} = 20 \pm \sqrt{398} \approx 20.00 \pm 19.95$$

$$x_1 \approx 20 + 19.95 = 39.95$$

$$x_2 \approx 20 - 19.95 = 0.05 \longrightarrow \text{Loss of 3 significant digits}$$

- Change the algorithm to get x_2

$$x_2 = \frac{c}{ax_1} = \frac{2}{1 \cdot 39.95} \approx 0.05006$$

Loss of Significance

- Example 1.6 in Wen Shen

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

  - Numerical computation of this function may lead to loss of significance. Find a remedy.
    - For large x, $\sqrt{x^2+2x}$ and $x+1$ are very close to each other. → loss of significant digits in subtraction
    - Change the form of the function

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

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

Choice of Language

- MATLAB
- Python (with NumPy & MatPlotLib)
- Fortran
- C
- C++

(+ any graph tools)

Investigation

- Find the floating-point representation applied in your programming language

References

- Wen Shen,
  An Introduction to Numerical Computation

- C. Moler,
  Numerical Computing with MATLAB

- Wikipedia