數值分析

Chapter 1
Introduction

授課教師：劉耀先
國立陽明交通大學 機械工程學系 EE464
yhliu@nctu.edu.tw
110學年度第一學期

Outline

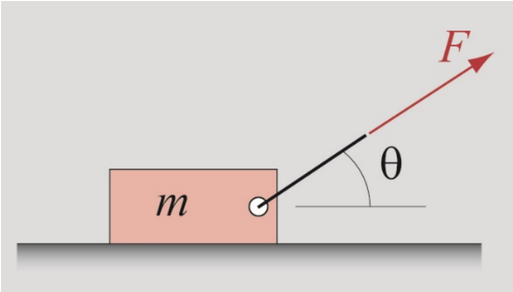
- Basic of computer programming

- Basic of numerical methods
  - Background (1.1)
  - Representation of numbers on a computers (1.2)
  - Errors in numerical solutions, round-off errors, and truncation errors (1.3)
  - Computers and programming (1.4)

1.1 Background

- Numerical methods are mathematical techniques used for solving mathematical problems that cannot be solved or are difficult to solve analytically
  - Analytical solution: exact answer in the form of mathematical expression
  - Numerical solution: approximate numerical value

- Example: Motion of a block on a surface with friction

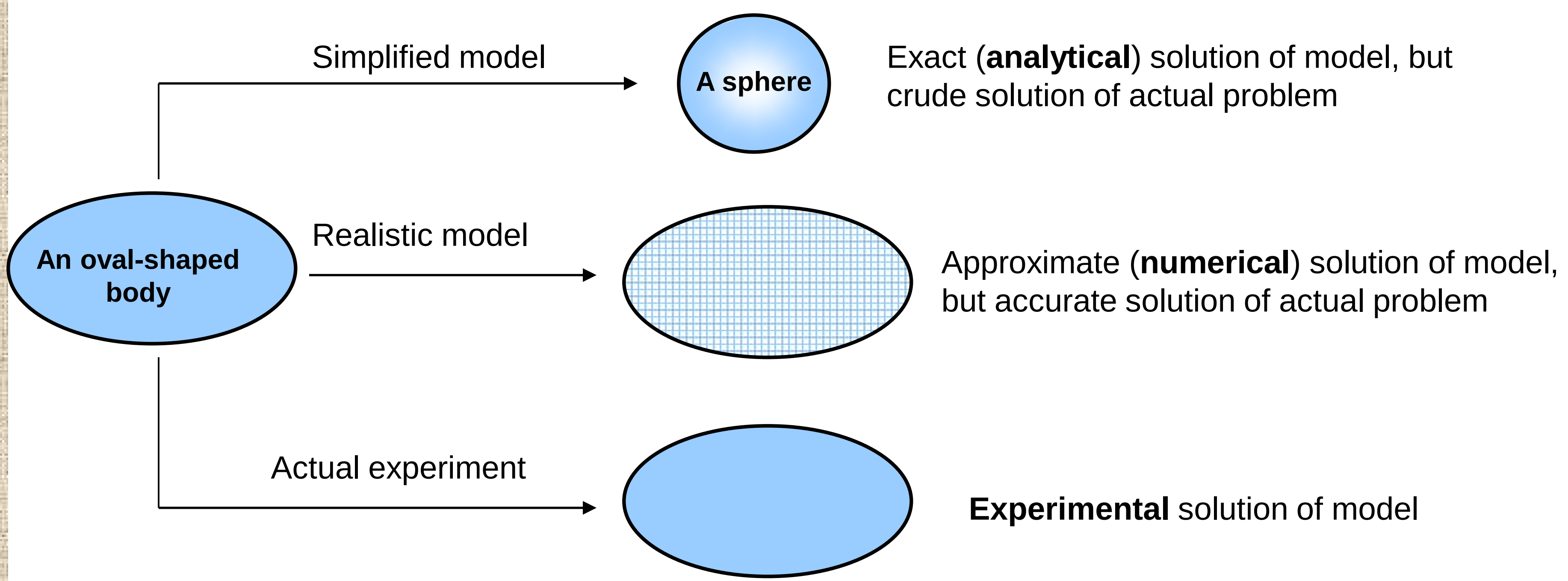


$$F = \frac{\mu m g}{\cos\theta + \mu \sin\theta}$$

μ: friction coefficient

- This equation cannot be solved analytically for θ, but an approximate numerical solution can be determined for specified accuracy

Experimental vs. Numerical vs. Analytical



Limitations of Experiment:

- Space, equipment, time, and money
- Lengthy, risky, and incomplete (simultaneous velocity, pressure, and temperature measurement?)
- Fundamental limitations of ground-base experimental facilities (in terms of size, flow rate, temperature limit…etc)

1.2 Representation of numbers on a computer

- Decimal and binary representation

10^4	10^3	10^2	10^1	10^0		10^{-1}	10^{-2}	10^{-3}	10^{-4}
6	0	7	2	4	.	3	1	2	5

$$6\times10^4+0\times10^3+7\times10^2+2\times10^1+4\times10^0+3\times10^{-1}+1\times10^{-2}+2\times10^{-3}+5\times10^{-4}=60{,}724.3125$$

Figure 1-2: representation of the numbers 60,724.3125 in the decimal system

Base 10	Base 2			
	2^3	2^2	2^1	2^0
1	0	0	0	1
2	0	0	1	0
3	0	0	1	1
4	0	1	0	0
5	0	1	0	1
6	0	1	1	0
7	0	1	1	1
8	1	0	0	0
9	1	0	0	1
10	1	0	1	0

Figure 1-3: representation of numbers in decimal and binary forms

Representation of numbers on a computer

- Binary representation of 19.625

2^4	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}
↓	↓	↓	↓	↓		↓	↓	↓
1	0	0	1	1	.	1	0	1

$$1\times 2^4 + 0\times 2^3 + 0\times 2^2 + 1\times 2^1 + 1\times 2^0 + 1\times 2^{-1} + 0\times 2^{-2} + 1\times 2^{-3}$$

$$1\times 16 + 0\times 8 + 0\times 4 + 1\times 2 + 1\times 1 + 1\times 0.5 + 0\times 0.25 + 1\times 0.125 = 19.625$$

- Binary representation of 60,724.3125

2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^9	2^8	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0		2^{-1}	2^{-2}	2^{-3}	2^{-4}
↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓	↓		↓	↓	↓	↓
1	1	1	0	1	1	0	1	0	0	1	1	0	1	0	0	.	0	1	0	1

$$1\times 2^{15} + 1\times 2^{14} + 1\times 2^{13} + 0\times 2^{12} + 1\times 2^{11} + 1\times 2^{10} + 0\times 2^9 + 1\times 2^8 + 0\times 2^7 + 0\times 2^6 + 1\times 2^5$$

$$+ 1\times 2^4 + 0\times 2^3 + 1\times 2^2 + 0\times 2^1 + 0\times 2^0 + 0\times 2^{-1} + 1\times 2^{-2} + 0\times 2^{-3} + 1\times 2^{-4} = 60{,}724.3125$$

Floating point representation

- Decimal floating point representation (scientific notation)
- Mantissa and Exponent

  $6519.23 = 6.51923 \times 10^{3}$
- Storing a number in computer memory
  - Byte = 8 bits
  - Single precision (32 bits = 4 bytes)
  - Double precision (64 bits = 8 bytes)
  - Number stored in computer memory = Sign + Exponent + Mantissa

Number stored in computer memory

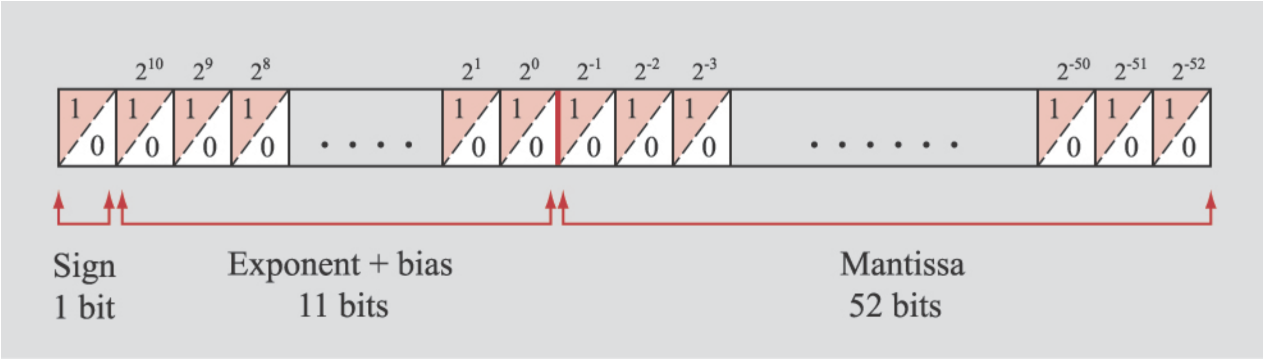


Figure 1-6: Storing in double precision a number written in binary floating point representation

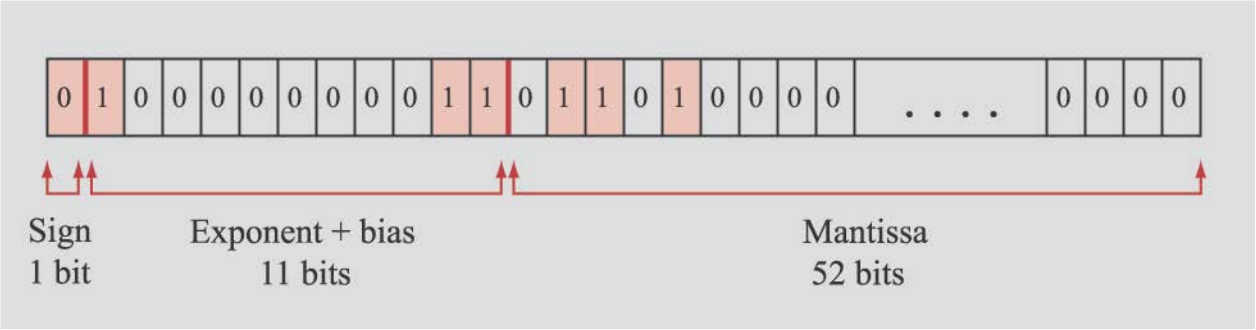
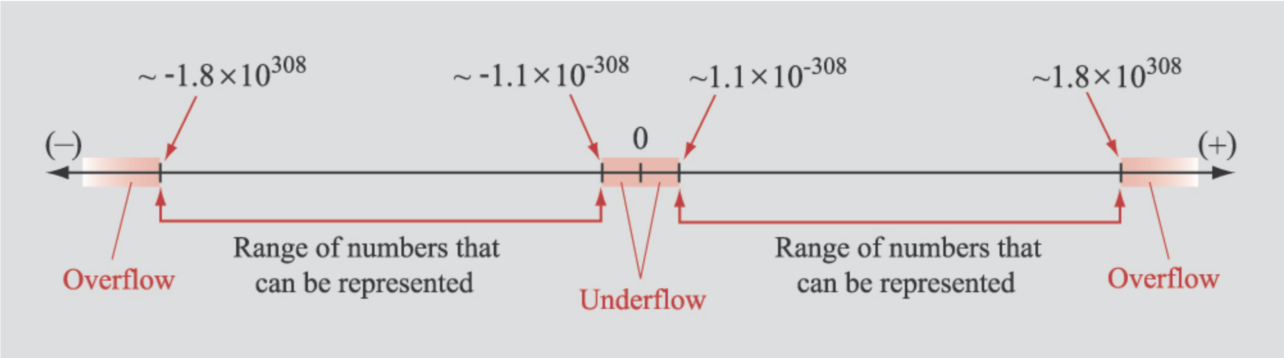


Figure 1-7: Storing the number 22.5 in double precision

Range of numbers can be represented in double precision

- Since a finite number of bits is used, not every number can be accurately written in binary form
- The smallest positive number in double precision: 2^{-1023} ($=1.1\times10^{-308}$)
- The largest positive number in double precision: 2^{1024} ($=1.8\times10^{308}$)

- **Underflow**: define a number in the gap between zero and the smallest number that can be stored on the computer
- **Overflow**: define a larger number than the largest positive number that can be expressed in double precision



1.3 Errors in numerical solutions

- Numerical solutions can be very accurate but in general are not exact
- Small error in one step can grow larger to the final result

- **Round-off errors**: because of the way that digital computers store numbers and execute numerical operations
- **Truncation errors**: introduced by the numerical method that is used for the solution (numerical methods use *approximations* for solving problems)
- **Total errors**: the difference between the true solution and approximate numerical solution

Round-off errors

- It is because of the way that digital computers store numbers and execute numerical operations
- A number can be shortened either by **chopping** off, or discarding, the extra digits or by **rounding**
- Round-off errors are likely to occur when two nearly identical numbers are subtracted from each other
- Example: find the solution of x^2-100.0001x+0.01=0
- The solutions are 100 and 0.0001

```
format long
a = 1; b = -100.0001; c = 0.01;
R = sqrt(b^2-4*a*c);
x1 = (-b+R)/(2*a)
x2 = (-b-R)/(2*a)
```

- The problem can be solved by changing the solution to a different form

Truncation errors

- Truncation errors occur when the numerical methods used for solving a mathematical problem use an approximate mathematical procedure
- Truncation error is dependent on the **specific numerical method** or **algorithm** used to solve a problem
- Example: sin(π/6)=?
- Using Taylor's series expansion:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \frac{x^9}{9!} - \frac{x^{11}}{11!} + \ldots$$

$$\sin(\frac{\pi}{6}) = \frac{\pi}{6} = 0.5235988 \qquad Error^{Trun} = 0.5 - 0.5235988 = -0.0235988$$

$$\sin(\frac{\pi}{6}) = \frac{\pi}{6} - \frac{(\pi/6)^3}{3!} = 0.4996742 \qquad Error^{Trun} = 0.5 - 0.4996742 = 0.0003258$$

Total Error

- Numerical solution always includes round-off errors and, depending on the numerical method, can also include truncation errors

- Total Error = Round-off error + Truncation Error
- Definition: True error = True solution – Numerical Solution
- True Relative Error = $\left| \dfrac{\text{True solution} - \text{Numerical Solution}}{\text{True solution}} \right|$

- Since the true errors cannot, in most cases, be calculated, other means are used for estimating the accuracy of a numerical solution

1.4 Computers and Programming

- Algorithm: a step-by-step procedure for calculating or implementing numerical method

- Example: Calculate the solution of $ax^2+bx+c = 0$
- (1) Calculate $D = b^2-4ac$
- (2) if $D>0$, calculate the two roots
- (3) if $D=0$, calculate the root and display the message "the equation has a single root."
- (4) if $D<0$, display the message "the equation has no real roots."