

Lecture Notes

Kyle Chui

2022-01-04

1 Lecture 1

The goal of this class is to solve *mathematical* problems with the help of *computers*.

1.1 Chapter Summaries

1. Computations in computers

- How to store (real) numbers in the computer

Note. If the number has finitely many digits, then it is simple. What about numbers with infinitely many digits, i.e. $\frac{1}{3}$? We have to truncate or round, and store an approximation with finitely many digits.

- How to perform computations

Note. From regular math, we know that $\frac{1}{3} + \frac{1}{3} = \frac{2}{3}$. However, in the computer, due to errors, we have $\frac{1}{3} \oplus \frac{1}{3} = ? \oplus ? = ??$.

- Errors

2.
 - Find roots of $f(x) = 0$ using bisection, Newton's method, ...
 - Convergence
 - Convergence order (how fast it converges)

3. Polynomial interpolation

- Approximate a function $f(x)$ by a polynomial $P(x)$, where $f(x_i) = P(x_i)$ for finitely many x_i
- *Accuracy* of the polynomial approximations

4. Numerical differentiations and numerical integrations

- Using the approximations from chapter 3, we can approximate using

$$f'(x^*) \approx P'(x^*) = \sum_{i=0}^k f(x_k) c_k,$$

and

$$\int_a^b f(x) dx \approx \int_a^b P(x) dx = \sum_{i=1}^k f(\bar{x}_k) \bar{c}_k.$$

- Error analysis

6.7. Solving linear systems of equations

- Direct methods: Gaussian elimination (computationally expensive)
- Iterative methods: (faster and cheaper)
- Solution stability

1.2 Round-off errors and computer arithmetics

There are three kinds of errors:

- Modeling Error: Occurs when we convert a problem from the real world into the mathematical world.
- Method Error: Occurs when we try to solve the mathematical problem numerically.
- Round-off error: Occurs when the computer gives an incorrect result with the correct algorithm (comes from storage and computation).

1.2.1 Storage

Infinite digit real numbers are *stored* as finite digit numbers, using the normalized decimal form of real numbers.

Definition. *Normalized decimal form of a real number*

For any $y \in \mathbb{R}$, we may write

$$y = \pm 0.d_1 d_2 d_3 \dots d_k d_{k+1} \dots \cdot 10^n,$$

where $0 < d_1 \leq 9$, $0 \leq d_i \leq 9$, n are integers. For the particular case where $y = 0$, we write $y = 0.0 \cdot 10^0$.

Definition. *Normalized machine numbers (Floating-point form)*

Any machine number y can be written as

$$y = \pm 0.d_1 d_2 \dots d_k \cdot 10^n,$$

where $0 < d_1 \leq 9$, $0 \leq d_i \leq 9$, n are integers.

We can think of the storage process as mapping normalized real numbers to normalized machine numbers. We do this via rounding or truncating.

Consider some $y \in \mathbb{R} \setminus \{0\}$.

- Truncating (k -digit truncation of $y = \pm d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n$) Simply omit the digits from d_{k+1} and onwards, in other words

$$f_\ell(\pm d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n) = \pm d_1 d_2 \dots d_k \cdot 10^n.$$

Thus we have $y \approx f_\ell(y)$.

- Rounding (k -digit rounding of $y = \pm d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n$)

If $d_{k+1} < 5$, then we drop $d_{k+1} d_{k+2} \dots$ (same with truncating)

If $d_{k+1} \geq 5$, then add 1 to d_k and drop $d_{k+1} d_{k+2} \dots$

$$f_\ell(\pm d_1 d_2 d_3 \dots d_k d_{k+1} d_{k+2} \dots \cdot 10^n) = \pm \delta_1 \delta_2 \dots \delta_k \cdot 10^m.$$