BME 369 Numerical Methods

Lecture 1: Fundamentals of Numerical Methods

Definition of Numerical Methods

In Engineering, we apply mathematical models to our problems. To have a solution, we can either use analytical or numerical methods.

Analytical Methods: We have an exact solution. However, it may not be computationally feasible to obtain it. (Remember: Calculus, Linear Algebra and Differential Equations)

Numerical Methods: Since we may not always have an analytical solution and even if it exists, it may not be computationally feasible to calculate, we need to apply some approximations and have a "close-enough" solution. These methods are called Numerical Methods.

Some Examples: Integration, Differentiation, Optimization, etc…

Mathematical Modeling

We use a Mathematical Model to understand and solve a real-world problem. Real-world problem may be related to:

- Disease
- Energy
- Financial Industry
- Transportation
- Environmental Issue

Trade-Offs

To solve these real-world problems, we make approximations and we obtain "close-enough" results compared to analytical solution if it had been existed.

That means: we have error in our solutions and we know it!

Pros (+)	Cons (-)
- A practical solution with reasonable accuracy - Clear – Easy to follow (such as iterative solutions) - Generality	- A complex model - In general, high computational effort is required - Modeling Effort

Introduction to Error Analysis

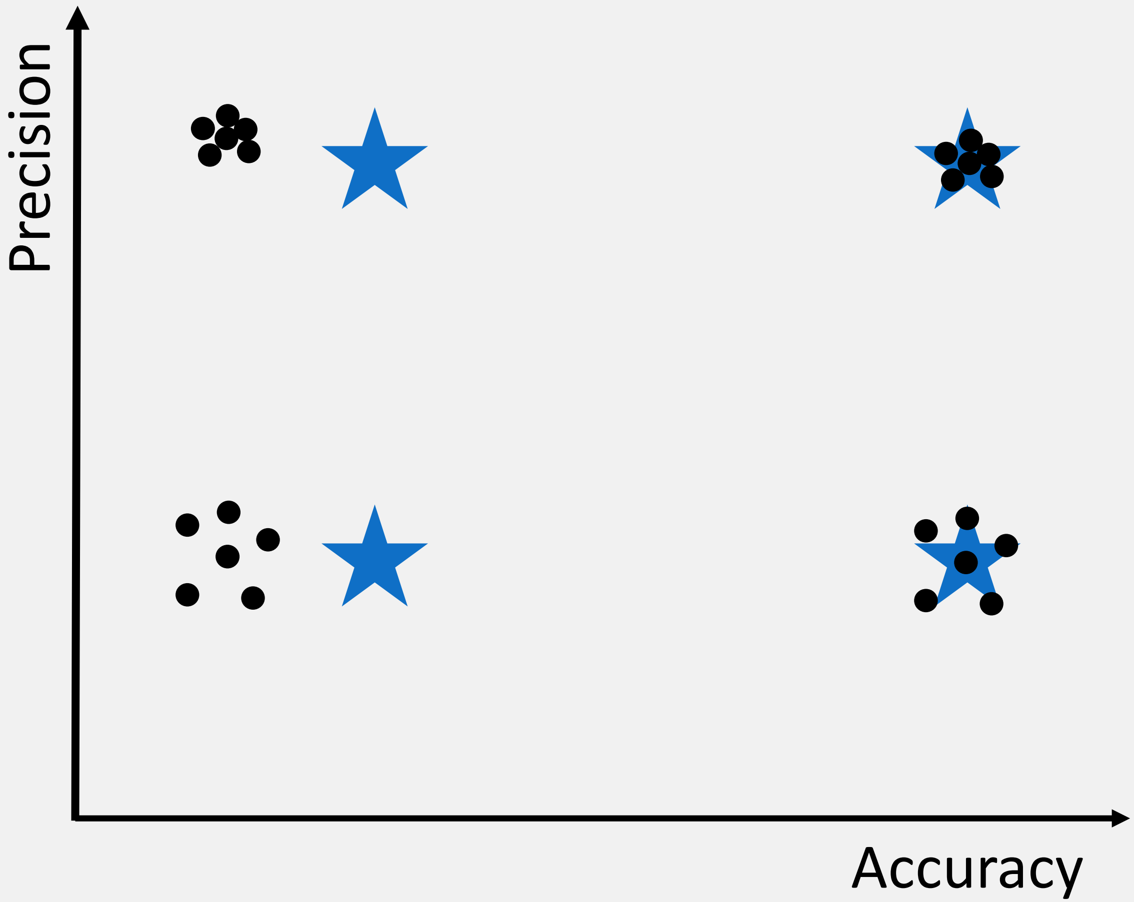
How do we define “good-enough” or “reasonable accuracy”?

Sources of Error

Modeling Error	Numerical Error

- Human Mistakes
- Formulation Error
- Data Noise
	- Truncation error
	- Round-off error
	    - Quantization error
	    - Numerical Manipulations

Accuracy vs. Precision



Formal Definitions of Error

True Value = Approximation + Error

True Error = True Value – Approximation

Based on these two definitions, we can define the following:

$$\varepsilon_t = \frac{True\ Error}{True\ Value} \times 100$$

ε_t : True Percent Relative Error

Formal Definitions of Error

Unfortunately, we do not always know the true value. In fact, we have true value if the problem has an analytical solution. However, we still want to have an error measure to judge our accuracy:

$$\varepsilon_a = \frac{Approximate\ Error}{Approximate\ Value} \times 100$$

ε_a : Percent Approximate Error

Without knowing the true value, how do we calculate the approximate error?

Iterative Methods

For many cases, we use iterative methods and we define our approximate error based on the difference between the previous and current iterations:

$$\varepsilon_a = \frac{Current\ Approximation - Previous\ Approximation}{Current\ Approximation} \times 100$$

We can also assume that the true value can be computed for an initial point, hence we can calculate approximate error for each iteration.

Iterative Methods

<u>Pseudo Code:</u>

$$\textit{while } \varepsilon_A^{(i-1)} < \varepsilon_S \,\&\, i < max.iter$$

$$(sol^i, val^i) = f\left(val^{(i-1)}\right)$$

$$\varepsilon_A^i = \left|\frac{sol^i - sol^{(i-1)}}{sol^i}\right| \times 100$$

$$i = i + 1$$

where, ε_S is the stopping criteria (threshold) and $max.iter$ is the maximum number of iterations to guarantee convergence.

Iterative vs. Direct Method

Direct Method	Iterative Method
• If a mistake is made in one of the steps, the end result will be most likely wrong.	• An answer exists even from step 1. Algorithm can correct errors in further steps.
• Directly find the final answer	• Gradually finds the final answer
• An example: Least Squares	• An example: Gradient Descend Algorithm

Types of Error

1) Quantization Error: Quantization error is the error that is introduced by transforming an analog signal to digital. (signal processing)
2) Numerical Manipulations: Common arithmetic operations such as addition, subtraction, multiplication and division.

Number Representation:

- Integers
- Fixed-Point Representation
- Floating Point Representation

Digital Data

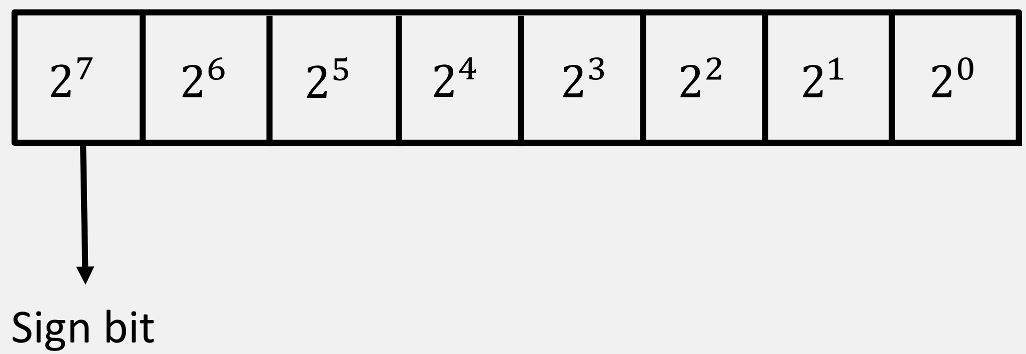
Computers store and manipulate data in binary form. That means, we have bits which are either 0 or 1.

Quantum Computing: Quantum computing studies theoretical computation systems (quantum computers) that make direct use of quantum-mechanical phenomena, such as superposition and entanglement to perform operations on data. (*Wikipedia*)

In today's computers, a single bit is always in one of the two states (0 or 1).

Quantum computation uses qubits which can be in one, zero or any quantum superposition of those two qubit states.

Binary Representation of Integers



Maximum number as unsigned 8 bit integer is 255.

Minimum number as unsigned 8 bit integer is 0.

Fixed-Point Representation

What happens when we have floating point numbers?

We can make use of fixed-point representation.

In an 8 bit representation, assume we have an imaginary “binary point” which separates the integer and fractional parts.

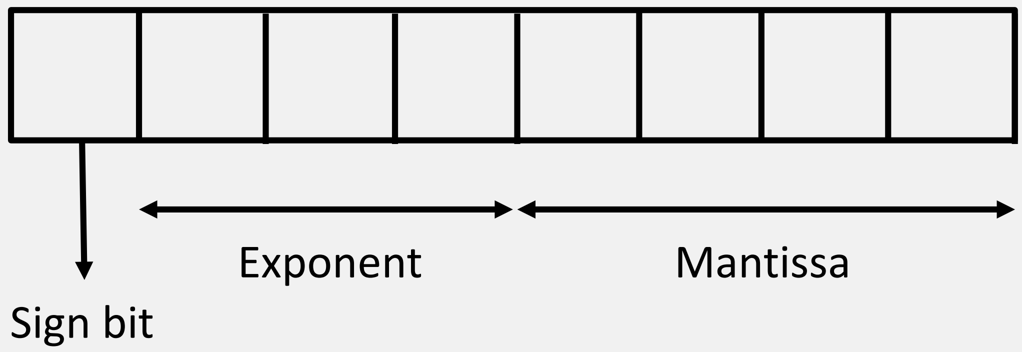
2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}

Floating Point Representation

In this approach, the number is expressed as a fractional part called a mantissa or significand, and an integral part called an exponent or characteristic as in: [4]

$$\text{number} = mb^e$$

where m is the mantissa, b is the base of the number system that is used, and e is the exponent.



Well-Posed Problem (Wiki.)

A problem is defined as well-posed if:

- a solution exists,
- the solution is unique,
- the solution behavior changes continuously with the initial conditions.

Then, it is straightforward to define ill-posed problems:

The problems that are not well-posed are termed ill-posed ☺