Numerical Analysis

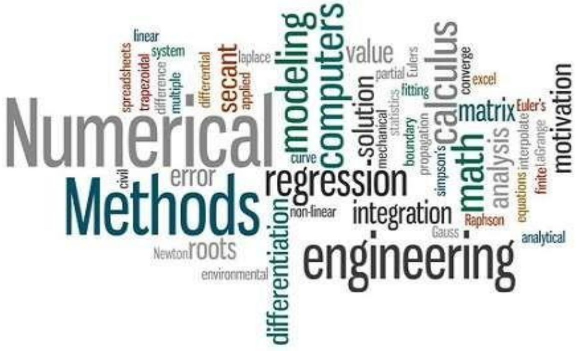
Dr. Farman Ali

Assistant Professor

DEPARTMENT OF SOFTWARE

SEJONG UNIVERSITY

Week 1



Objectives

This is an introductory course which covers the main topics of numerical analysis.

Before starting this course, students must have knowledge about

- Calculus

At the end of this course the students will learn

- The concept of root finding for nonlinear equations
- Interpolation and approximation of function by simpler computational building blocks (polynomials)
- Numerical differentiation and numerical quadrature and integration.
- Numerical solutions of ordinary differential equations

The knowledge of numerical analysis will help students to handle problems in different computation tasks.

Course Syllabus

- **Introduction:** Numerical analysis and backround, definitions of computer number systems, floating point representation, representation of numbers in different bases, and round-off errors.

- **Root Finding:** Bisection method, fixed-point iteration, Newton's method, the secant method and their error analysis, and order of convergence (Newton’s method and Secant method).

- **Direct Methods for Solving Linear Systems:** Gaussian elimination, LU decomposition, pivoting strategies, and PA=LU-factorization,….. .

- **Polynomial:** Polynomial interpolation, piecewise linear interpolation, divided differences interpolation, cubic spline interpolation, and curve fitting in interpolation (Application: Regression).

- **Integration:** Numerical differentiation, numerical integration, and composite numerical integration.

- **Ordinary Differential Equations:** Euler's Method, higher-order Taylor method, and Runge-Kutta methods….

Books| Guidance | Grading

Text books

- Introductory methods of Numerical Analysis (Author: S. S. Sastry)
- Numerical Analysis (Author: L. Ridgway Scott)
- Numerical Analysis (Author: Timothy Sauer)

Additional Guidance

- Lecture materials, assignments will be based on contents taken from recommended books and internet.
- Quizzes will be based on the material delivered during the lecture.
- All the classes will be offline
- The lecture material will be formatted in the form of PPT slides.
- PPT slides will be used during lecture to discuss topics and solve equations. Slides will be provided before the lecture time.

Grading

Midterm Exam (%)	Final (%)	Assignments and quizzes (%)	Attendance (%)
30	35	25	10

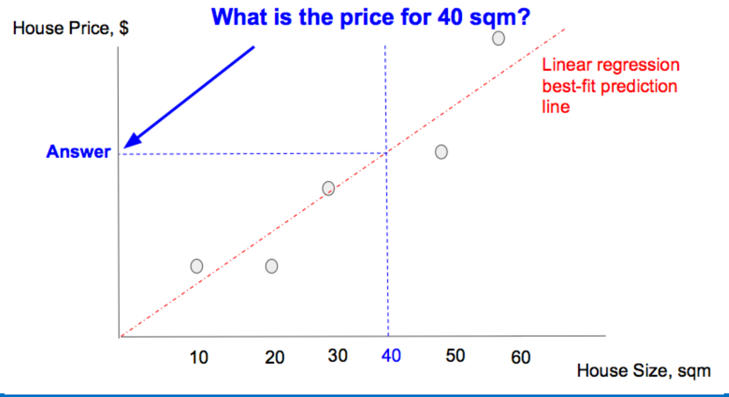
Introduction

- Numerical analysis | Numerical methods | Numerical Computation is a mathematics course for engineers and scientists. It is designed to provide mathematical procedures for determining approximate solutions of certain problems that arises in science and engineering.

Why numerical analysis is used in Science and Engineering?

- Engineers use mathematical modelling which includes various equations and data to describe and predict the behavior of systems.
- Accurate approximation (lower error)
- Optimized -> faster algorithm (computation time)

Example



Analytical and Numerical Methods

- Analytical:

$$\int_a^b f\,(x)dx = [anti\ Derivative]$$

- Applied a set of logical steps to solve the problem that are proven to find the exact answer to the problem. For example,

$$X + 1 = 0 \qquad\qquad X = -1$$

(what if finding that exact answer takes too long or is impossible?)

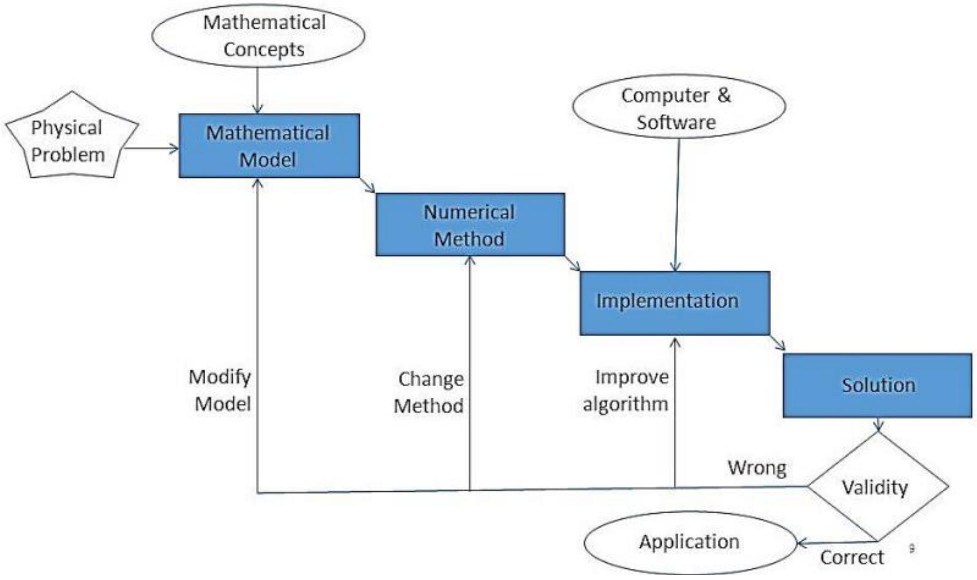
That is why numerical analysis come in

- Numerical method: Plugging in inputs, checking how close we are to the solution, adjusting our input and repeating this process until get the approximate answer.
- Using Numerical Method, we rarely reach an exact answer, we can get really close to the exact answer much quicker than solving analytical.

Computation time increase if allowable error decrease

How numerical algorithms can be applied ?

- Develop a mathematical problem with your skills and the requirement.
- Come up with a numerical algorithm.
- Implement the algorithm.
- Run, debug, test the code.
- Visualize and interpret the result.
- Validate the result.



How numerical analysis can be applied ?

- Mathematical models are a central piece of science and engineering.

- Some models have closed-form solutions, therefore they can be solved analytically. Many models can not be solved analytically or the analytic solution is too costly to be practical.

- All models can be solved computationally and the result may not be the exact answer but it can be useful.

Root Findings (Solution of nonlinear Equations)

Some simple equations can be solved analytically:

$$x^2 + 4x + 3 = 0$$

Analyticsolution roots $= \dfrac{-4 \pm \sqrt{4^2 - 4(1)(3)}}{2(1)}$

$x = -1$ and $x = -3$

$$ax^2 + bx + c = 0$$

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Many other equations have no analytical solution:

$$x^9 - 2x^2 + 5 = 0$$

$$x = e^{-x}$$

No analytic solution

Root Findings (Methods for solving nonlinear equation)

- Bisection method
- Newton's method
- Secant method

Solution of systems of linear equations

$$x_1 + x_2 = 3$$

$$x_1 + 2x_2 = 5$$

we can solve it as :

$$x_1 = 3 - x_2, \quad 3 - x_2 + 2x_2 = 5$$

$$x_2 = 2, \quad x_1 = 3 - 2 = 1$$

What to do if we have

1000 equations in 1000 unknowns.

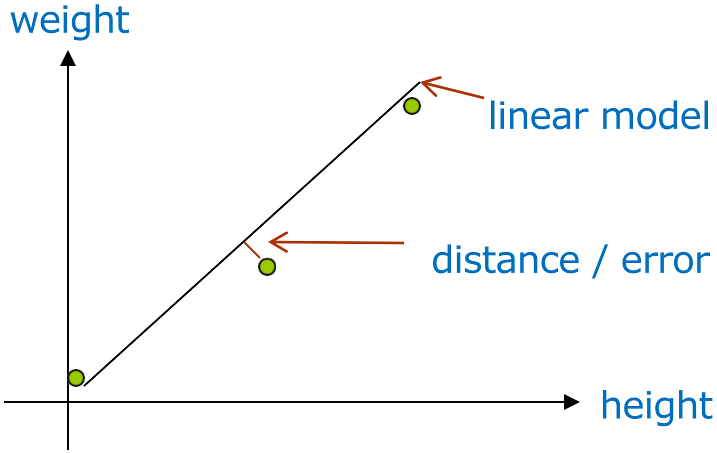
Direct Methods for Solving Linear Systems:

- Gaussian elimination
- LU decomposition
- Pivoting strategies
- LU-factorization
- Forward substitution
- Crout factorization

Curve Fitting

Given a set of data:

height	x	65	69	73
weight	y	105	130	140



Select a curve that best fits the data. One choice is to find the curve so that the sum of the square of the error is minimized.

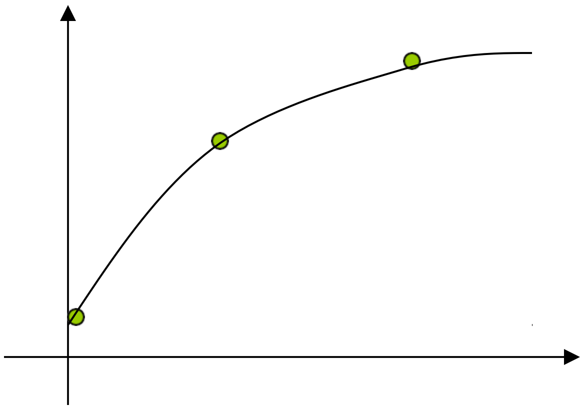
Too much data

Numerical analysis and Background

Interpolation

Given a set of data:

x_i	0	1	2
y_i	0.5	10.3	15.3



Find a polynomial $P(x)$ whose graph passes through all tabulated points.

$$y_i = P(x_i) \quad \text{if } x_i \text{ is in the table}$$

Numerical analysis and Background

Interpolation

- Polynomial interpolation
- Piecewise linear interpolation
- Divided differences interpolation
- Cubic spline interpolation
- Curve fitting in interpolation (Application: Regression).

Integration

Some functions can be integrated analytically:

$$\int_1^3 x\,dx = \frac{1}{2}x^2\Big|_1^3 = \frac{9}{2}-\frac{1}{2} = 4$$

But many function have no analytical solutions

$$\int_0^a e^{-x^2}\,dx = ?$$

Integration

- Numerical differentiation
- Numerical integration
- Composite numerical integration

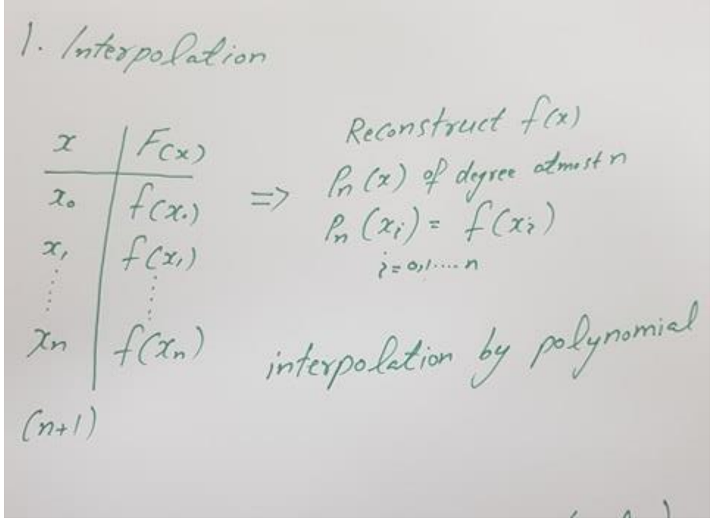
Numerical analysis Background

➢ Example of complex problems

Our main focus should be

1. **Accuracy** → How accurate our numerical algorithm ? What is amount of error in each step ?
2. **Efficiency** → We should focus on the efficiency of numerical algorithm
3. **Stability** → Is the developed method is stable or not.

✶ We need numerical algorithms for the above three.



1. Interpolation

x	$F(x)$
x_0	$f(x_0)$
x_1	$f(x_1)$
$\vdots$	$\vdots$
x_n	$f(x_n)$

$(n+1)$

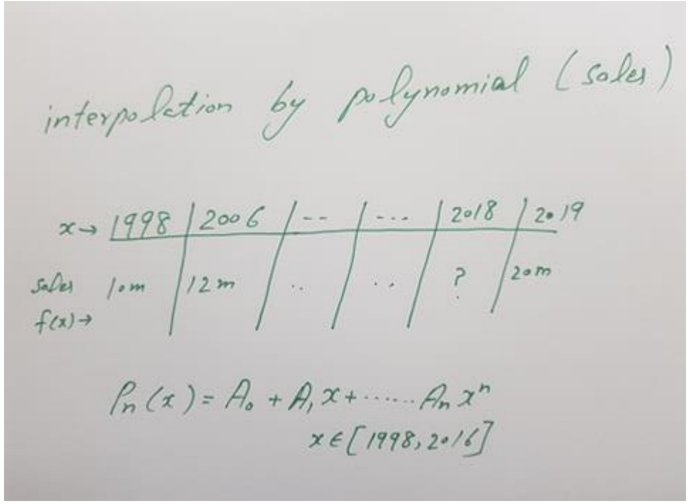
$\Rightarrow$ Reconstruct $f(x)$

$P_n(x)$ of degree atmost n

$P_n(x_i) = f(x_i)$

$i = 0, 1, \ldots n$

interpolation by polynomial



interpolation by polynomial (sales)

$x \rightarrow$	1998	2006	--	...	2018	2019
sales $f(x) \rightarrow$	10m	12m	..	..	?	20m

$$P_n(x) = A_0 + A_1 x + \cdots\cdots A_n x^n$$

$$x \in [1998, 2016]$$

Real world applications

- ➢ The World’s Largest Matrix Computation: Google’s PageRank is an eigenvector of a matrix of order about 3 billion

- ➢ Airlines use optimization algorithms to decide ticket prices, airplane and crew assignments and fuel needs.

Real world applications

- Car companies can improve the crash safety by using computer simulations of car crashes. These simulations are essentially solving partial differential equations numerically.

- Hedge funds (private investment funds) use tools from all fields of numerical analysis to calculate the value of stocks and derivatives.

Real world applications

- Modelling in industry: Aerospace

- Weather prediction

Representation of Real Number

Floating Point Representation

Real Number

Real Numbers: A real number is any positive or negative number

Real Numbers include:

Whole Numbers (like 1,2,3,4, etc)

Rational Numbers (like 3/4, 0.125, 0.333..., 1.1, etc)

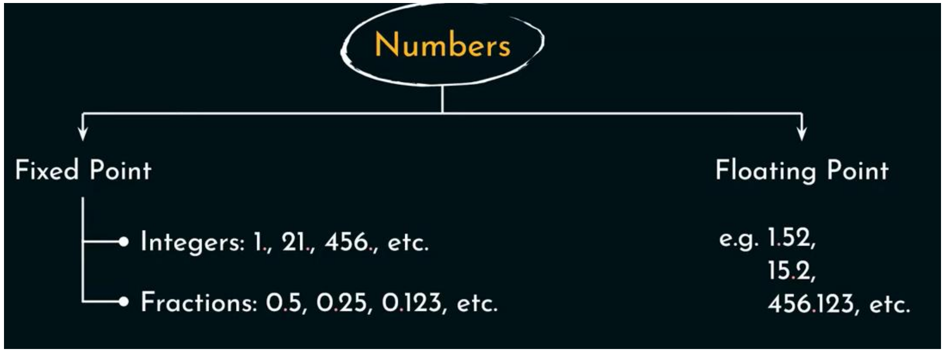
Irrational Numbers (like π, √3, etc)

14.75 15 3148 22.9 99/100

100.159 2/7 Π -27 √3

But digital computers cannot understand characters other than 0 and 1.

Real Number



Numbers

Fixed Point

- Integers: 1., 21., 456., etc.
- Fractions: 0.5, 0.25, 0.123, etc.

Floating Point

e.g. 1.52,
15.2,
456.123, etc.

You are familiar with the decimal Number system:

$$312.45 = 3\times10^{2} + 1\times10^{1} + 2\times10^{0} + 4\times10^{-1} + 5\times10^{-2}$$

Decimal System: Base = 10 , Digits (0,1,…,9)

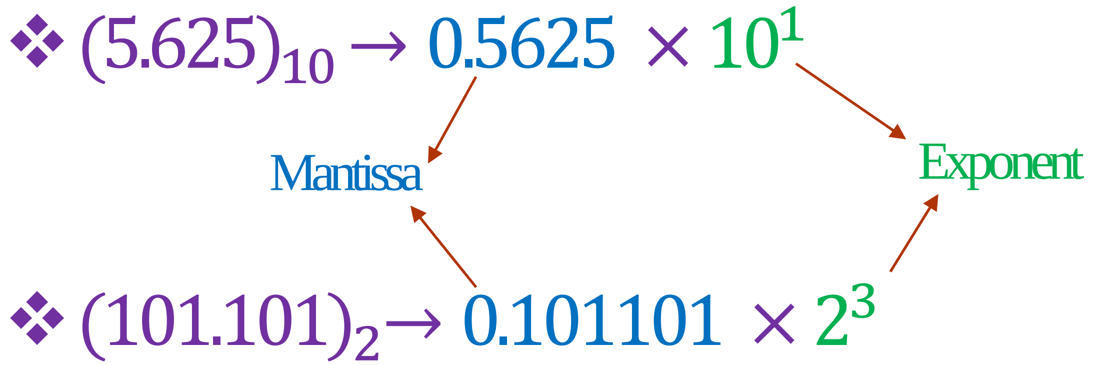
Floating point representation: real numbers converted into binary form where no decimal point is exists.

Floating point representation

Point is present after 3rd bit

- $(5.625)_{10} \rightarrow (101.101)_2 \longrightarrow$ early day stored in the computer memory like [101101,3]

This method was not impractical.



- $(5.625)_{10} \rightarrow 0.5625 \times 10^1$

Mantissa

Exponent

- $(101.101)_2 \rightarrow 0.101101 \times 2^3$

This information will be saved in a fixed bit memory space.

S	Exponent	Mantissa

Before storing, we need to know normalization because the representation of mantissa and exponent can be done in different ways.

Floating point representation

Need of Normalization:

- $(101.101)_2 \rightarrow 0.101101 \times 2^3$
- $(101.101)_2 \rightarrow 1.01101 \times 2^2$
- $(101.101)_2 \rightarrow 0.0101101 \times 2^4$
- $(101.101)_2 \rightarrow 101101 \times 2^{-3}$

There two types of normalization:

❖ Explicit Normalization: Move the radix point to the LHS of the most significant '1' in the bit sequence.

$$(101.101)_2 \rightarrow 0.101101 \times 2^3$$

❖ Implicit Normalization: Move the radix point to the RHS of the most significant '1' in the bit sequence.

$$(101.101)_2 \rightarrow 1.01101 \times 2^2$$

Implicit normalization is better with respect to precision

Floating point representation

Explicit Normalization

- $(5.625)_{10} \rightarrow (101.101)_2$
- $(101.101)_2 \rightarrow 0.101101 \times 2^3$
- S=0
- Exponent=3+8=11 $(1011)_2$
- Mantissa=101101

10 bits memory

S (1bit)	E(4 bits)	M (5 bits)
0	1011	10110

The value which we store here is incorrect

Implicit Normalization

- $(5.625)_{10} \rightarrow (101.101)_2$
- $(101.101)_2 \rightarrow 1.01101 \times 2^2$
- S=0
- Exponent=2+8=10 $(1010)_2$
- Mantissa=01101

10 bits memory

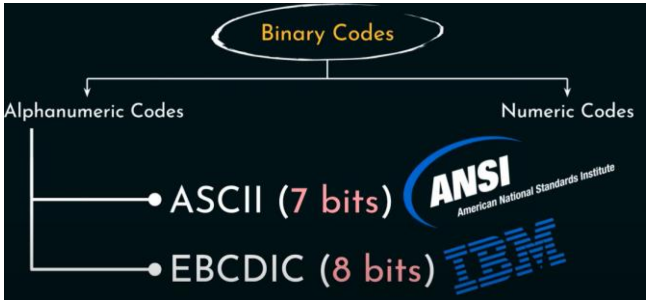
S (1bit)	E(4 bits)	M (5 bits)
0	1010	01101

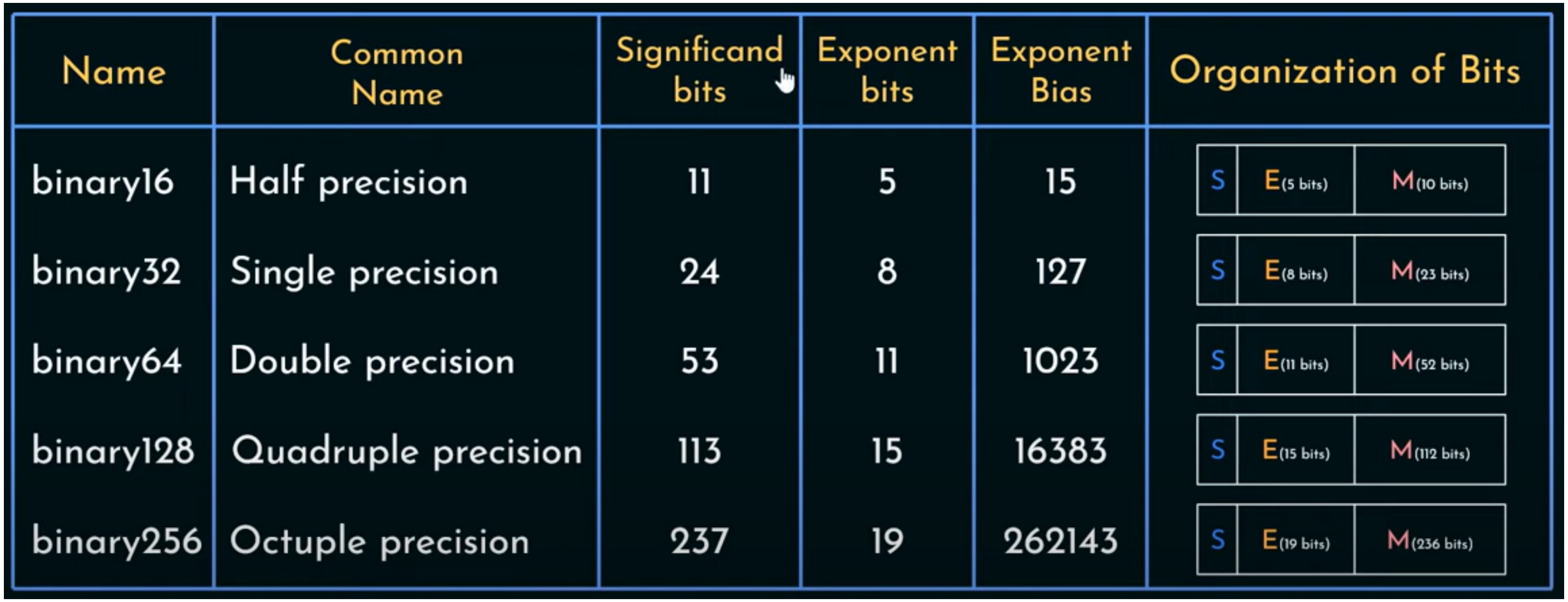
The value which we store here is precisely the value which we want to store

Floating point representation

IEEE Standard for floating-point arithmetic (IEEE 754)

- IEEE is responsible for standardization of floating-point numbers
- IEEE 754-1985
- IEEE 754-2008
- IEEE 754-2019





Name	Common Name	Significand bits	Exponent bits	Exponent Bias	Organization of Bits
binary16	Half precision	11	5	15	S | E(5 bits) | M(10 bits)
binary32	Single precision	24	8	127	S | E(8 bits) | M(23 bits)
binary64	Double precision	53	11	1023	S | E(11 bits) | M(52 bits)
binary128	Quadruple precision	113	15	16383	S | E(15 bits) | M(112 bits)
binary256	Octuple precision	237	19	262143	S | E(19 bits) | M(236 bits)

Floating point representation

Standard form is a scientific notation of representing numbers as a **base** number and an **exponent**.

Using this notation:

The decimal number **8674.26** can be represented as

8.67426×10^3, with mantissa = **8.67426**, base = **10** and exponent = **3**

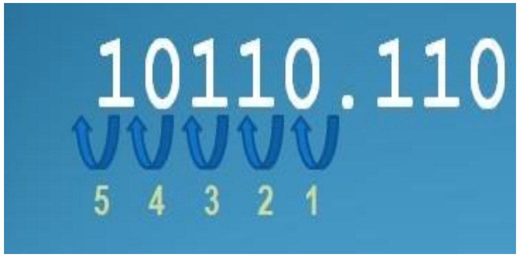
Any number can be represented in any number base in the form **$m \times b^n$**

m= mantissa
b= base or radix of the number system
n= exponent

Floating point representation

What is the exponent of 10110.110?

The exponent is 5, because the decimal point has to be moved 5 places to get it to the left hand side.



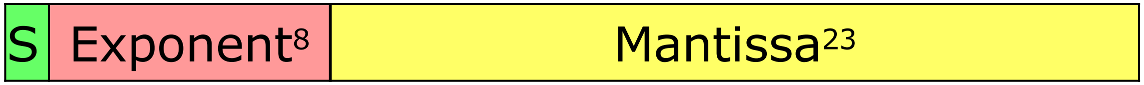
The exponent would be represented as 0101 in binary

Floating point representation

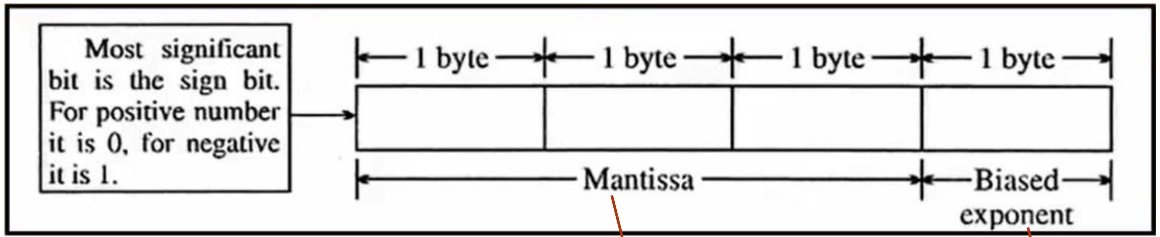
IEEE Floating Point Standard

- Single Precision (32-bit representation)

  1-bit Sign + 8-bit Exponent + 23-bit Fraction



S | Exponent[8] | Mantissa[23]

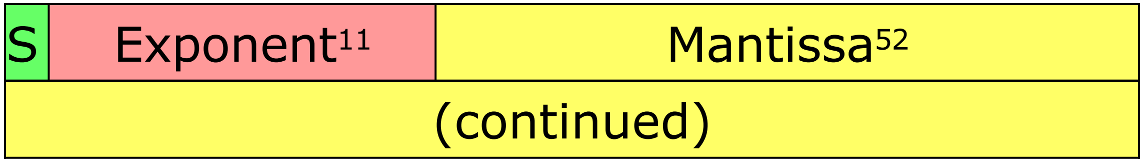

Most significant bit is the sign bit. For positive number it is 0, for negative it is 1.
1 byte
1 byte
1 byte
1 byte
Mantissa
Biased exponent

number without point

number of places that the point must be moved in the original number

Double Precision (64-bit representation)

1-bit Sign + 11-bit Exponent + 52-bit Fraction



S | Exponent[11] | Mantissa[52]

(continued)

Floating point representation

Step 1: single precision real number representation

- Non-normalized form: Find the binary equivalent of the given number by the conventional method.

Ex: 25.75_{10}

11001.11

2	25
2	12-1
2	6 - 0
2	3 - 0
	1 - 1

0.75x2=1.50 1

0.50x2=1.00 1

Step 2: represent it in formalized form

$1.100111\ X\ 2^4$

mantissa part is ready

Step 3: remove the first number and add zeros to the right hand side to get the full mantissa part until it becomes in 24 bits (3bytes form

Sign (1 bit) -> 0 for +ve

100 1110 0000 0000 0000 0000

Floating point representation

Step 4: find biased exponent part (add 127 with the exponent and find the binary equivalent in 8 bits (1) form)

Ex: 25.75_{10}

n= 4

127+4=131

$1.100111\ X\ 2^4$

2	131
2	65-1
2	32-1
2	16-0
2	8-0
2	4-0
2	2-0
	1-0

10000011

Sign (1 bit) -> 0 for +ve

Mantissa: 100 1110 0000 0000 0000 0000

Biased exponent:1000 0011

0 1000 0011 100 1110 0000 0000 0000 0000

Floating point representation

Store 110.0011001 in floating point representation, using 8 bits for the mantissa and 4 bits for the exponent.

1	1	0	0	0	1	1	0	~~0~~	~~1~~

Mantissa

0	0	1	1

Exponent

- The mantissa only holds 8 bits and so cannot store the last two bits
- These two bits cannot be stored in the system, and so they are forgotten.
- The number stored in the system is 110.00110 which is less accurate that its initial value.

- If the size of the mantissa is increased then the accuracy of the number held is increased.

1	1	0	0	0	1	1	0	0	1

Mantissa (10 bits)

Representation of Real Number

Numbers Systems

Numbers Systems

- **Number systems** are the technique to represent numbers in the computer system architecture, every value that you are saving into or getting from computer memory has a defined number system.

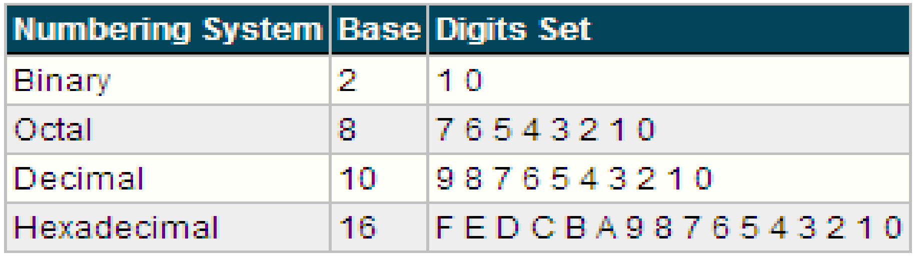
Number system can be categorized

Decimal number system (10 digits)

Binary number system (2 digits)

Octal number system (8 digits)

Hexadecimal Number System (16 digits)



Numbering System	Base	Digits Set
Binary	2	1 0
Octal	8	7 6 5 4 3 2 1 0
Decimal	10	9 8 7 6 5 4 3 2 1 0
Hexadecimal	16	F E D C B A 9 8 7 6 5 4 3 2 1 0

Numbers Systems

Conversion of Number System

Decimal	Binary	Octal	Hexa-Decimal
0	0	0	0
1	1	1	1
2	10	2	2
3	11	3	3
4	100	4	4
5	101	5	5
6	110	6	6
7	111	7	7
8	1000	10	8
9	1001	11	9

Decimal	Binary	Octal	Hexa-Decimal
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Numbers Systems

Binary Number System

- Binary to Decimal Conversion Techniques:
  - Multiply each bit by 2^n, where n is the "weight" of the bit.
  - The weight is the position of the bit, starting from 0 on the right.
  - Add the results.

$$101011_2 = 43_{10}$$

- Binary to Octal Conversion Techniques:
  - Group binary digits in a 3 bits, starting on right side
  - Convert to octal digits.
  - Example

$$1011010111_2 = 1327_8$$

- Binary to Hexa-decimal Conversion Techniques:
  - Group binary digits in a 4 bits, starting on right side.
  - Convert to hexa-decimal digits.
  - Example

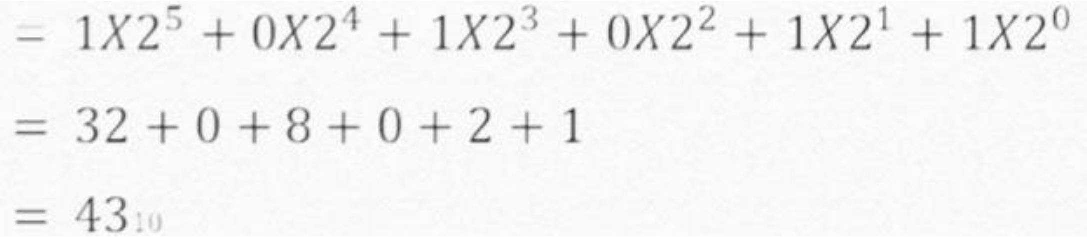
$$1010111011_2 = 2BB_{16}$$

Numbers Systems

Binary Number System

Binary to Decimal Conversion

101011=43_{10}



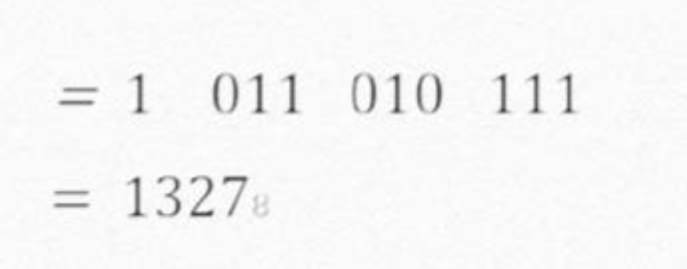
$$= 1X2^5 + 0X2^4 + 1X2^3 + 0X2^2 + 1X2^1 + 1X2^0$$

$$= 32 + 0 + 8 + 0 + 2 + 1$$

$$= 43_{10}$$

Binary to Octal Conversion

1011010111= 1327_8

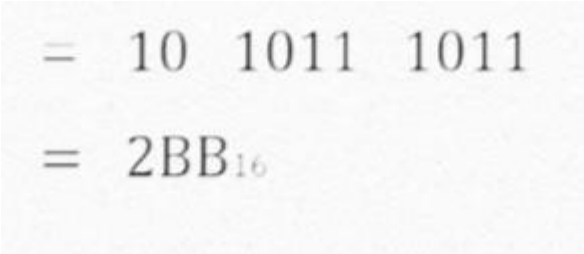


$$= 1 \quad 011 \quad 010 \quad 111$$

$$= 1327_8$$

Binary to Hexa-decimal Conversion

1010111011=$2BB_{16}$



$$= 10 \quad 1011 \quad 1011$$

$$= 2BB_{16}$$

Thank You !