MA2252 Introduction to Computing

Lecture 9

Representation of Numbers

Sharad Kumar Keshari

School of Computing and Mathematical Sciences

University of Leicester

Learning outcomes

At the end of lecture, students will be able to understand

- Base-N numbers
- Binary numbers
- Floating Point Numbers

Motivation

Sometimes, MATLAB gives unexpected outputs. Consider these examples:

- exp(i∗pi)+1==0 → false in MATLAB
- sin(pi), cos(pi/2)
- 5+8e-17>5

  $8e-17 = 8 \times 10^{-17}$
- 1 - 3∗(4/3 - 1)
- 2^-1073==0, 2^-1075==0
- sqrt(1e-16 + 1) - 1 → $\sqrt{10^{-16}+1} - 1 > 0$

The goal of this lecture is to understand the reason behind these strange results.

Base-N numbers

Base: Number of unique digits to represent a number.

Example:

Decimal number (Base-10 number) → 10 unique digits

1. Uses digits from 0 to 9. So, base is 10.
2. Decimal number can be expanded in powers of 10.

   Examples:

   $582 = 5 \times 10^2 + 8 \times 10 + 2 \times 10^0,$

   500 80 2

   $25.36 = 2 \times 10^1 + 5 \times 10^0 + 3 \times 10^{-1} + 6 \times 10^{-2}.$

   20 5 0.3 0.06

Base-N numbers (contd.)

Other examples: 1011011101

- **Binary number:** Uses only two digits: 0 and 1.
  Base 2
- **Octal number:** Uses digits from 0 to 7. 01234567
  Base 8 → 8 digits
- **Hexadecimal number:** Uses 16 digits: 0-9 followed by A-F or a-f.
  Base 16
  10 + 6 = 16

Binary numbers

- Binary numbers are represented by only 0s and 1s. So, the base is 2.
- A digit in binary number is called a **bit**.
- **Example:** The number 1001000110.

  This number can be converted to a decimal number as follows:

11010

$60 = 32 + 16 + 8 + 4 + 0\times2^1 + 0\times2^0$

$2^6 = 64$

decimal ← 60 ⟶ binary

$$1\times2^9 + 0\times2^8 + 0\times2^7 + 1\times2^6 + 0\times2^5 + 0\times2^4 + 0\times2^3 + 1\times2^2 + 1\times2^1 + 0\times2^0 = 582$$

512 0 0 64 0 0 0 4 2 0 = 582

111100

Binary numbers (contd.)

computers treat operations like switches/gates

Pros

OR, AND, NOT

- Easier for a computer to store and work with.

add, multiply, divide

- A computer can do all arithmetic operations with them.

$$\begin{array}{r} 111 \\ +101 \\ \hline 1100 \end{array}$$

- A 32-bit computer can represent upto 2^{23} binary numbers.

0 or 1

10, 11

$2 = 1 \times 2^1 + 0 \times 2^0$

$2 = 10 \to \cancel{20}$

32 digit binary number

$+(\ )\times 2^1 + (\ )\times 2^0$

-5.78

$1 \quad 10 \quad 11$

Cons

$1 \quad 2 \quad 3 \quad 4$

- Harder for humans to do binary algebra.

gap between two consecutive number

- Limited **range** and **precision** to perform all mathematical calculations.

1.005

32 bit computer

2^{23} values

Floating Point Numbers

MATLAB uses **floating point** numbers or **floats** to achieve the desired range and precision required to perform mathematical calculations.

Note: Floating point numbers are always **rational**. MATLAB approximates irrational numbers (e.g. π) by rational numbers.

Types of float:

- single precision (32 bits)
- double precision (64 bits)

$\underbrace{1.3333\ldots\ldots}_{\text{repeating}}$ | $\underbrace{1.5479}$ → rational

irrational ← non-terminating & non-repeating

Single precision float

$0 \le e \le 255$

$2^8 = 256$

In the **IEEE754** standard for single precision, a float which is represented as

2^{23} values range

$$n = -1^s 2^{e-127}(1+f)$$

range

can take large no. of values

$0 \times 2^{()} + \square \times 2^{()} + \square \times \dots + \square \times 2^0$

Here, s, e and f are sign indicator, exponent and fraction respectively.

$0 \le x < 1$

domain $\rightarrow f \rightarrow$ range

0 or 1s

domain

32 bits are allocated as follows:

- s has 1 bit so it takes values of 0 or 1.
- e has 8 bits so it can take 2^8 values.
- f has 23 bits so it can take 2^{23} values. $0 \le f < 1$.

$-1^0 = 1$

$-1^1 = -1$

fraction 32

0 or 1

Example:

Convert the number

1 10000011 11000000000000000000000 (IEEE754)

into decimal (base 10) number.

$s = 1$

$e = 1\times 2^7 + 0 + 0 + 0 + 0 + 0 + 1\times 2^1 + 1\times 2^0$

$= 128 + 2 + 1$

$= 131$

$f = 1\times 2^{-1} + 1\times 2^{-2} + \ldots 0 + + 0\times 2^{-22} + 0\times 2^{-23}$

$= 0.5 + 0.25 = 0.75$

$n = (-1)^1 2^{131-127} \times (1 + 0.75) =$

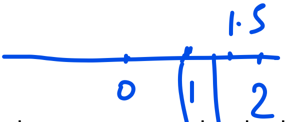
Floating Point Numbers (contd.)

Solution:

$s = 1$,
e=$1 \text{x} 2^7 + 0 + 0 + 0 + 0 + 0 + 1 \text{x} 2^1 + 1 \text{x} 2^0 = 131$,
f=$1 \text{x} 2^{-1} + 1 \text{x} 2^{-2} + \cdots = 0.75$.

$$n = -1^1 2^{131-127}(1 + 0.75) = -2^4 \text{x} 1.75 = -28.$$

Gaps between numbers



$\frac{1+2}{2} = 1.5$

Not every decimal (base 10) number can be represented by double precision float. This causes gaps between numbers in MATLAB.

$\frac{1+1.5}{2}$

To find the gap, use MATLAB's eps function. eps(x) gives the gap between number x and next representable number.

real no.s are continuous

5 ( ) 6 → no number

Example:
eps(5)=8.881784197001252e-16.

gap

MATLAB takes discrete values

The gap increases as numbers get large because the factor 2^{e-127} grows in size.

Special cases

1. e=0 (00000000(base 2))
   In this case, the float is calculated using

$$\underbrace{n = -1^s 2^{-126} f}_{\text{subnormal numbers}}$$

$\lim_{n \to 0} \frac{1}{n} \to \infty$

2. e=255 (11111111(base 2)) $\to \frac{0}{0}, \frac{Inf}{Inf}$
   - $f \neq 0$: n=NaN (Not a Number)
   - f=0 and s=0: n=Inf
   - f=0 and s=1: n=-Inf

$\frac{1}{0} \to$ undefined

Single precision float interesting facts

$\to 2^7 \times 1 + 2^6 \times 1 + \cdots$

- Largest defined number $1\times 2^{-1} + 1\times 2^{-22} + 1\times 2^{-23}$
  0 11111110 11111111111111111111111 $(1+f)$
  (3.402823466385289e+38)

- Smallest defined positive 'normal' number
  0 00000001 00000000000000000000000
  (1.175494350822288e-38)
  (2^-126)

Note: Use MATLAB functions realmax('single') and realmin('single') to find the above results.

Floating Point Numbers (contd.)

- Smallest defined subnormal number
  0 00000000 00000000000000000001
  (1.401298464324817e-45)
  (2^-149)

Note: Use MATLAB single() function to represent a number in single precision.

Double precision float

MATLAB uses default double precision of **IEEE754** standard. A double precision is represented as

$$n = -1^s 2^{e-1023}(1+f)$$

Again, s, e and f are sign indicator, exponent and fraction respectively.

64 bits are allocated as follows:

- s has 1 bit so it takes values of 0 or 1.
- e has 11 bits so it can take 2^{11} values.
- f has 52 bits so it can take 2^{52} values. $0 \leq f < 1$.

Floating Point Numbers (contd.)

The special cases for double precision float are similar to single-precision float.

Double precision float interesting facts

- Largest defined number
  1.797693134862316e+308 (type realmax in command window)
- Smallest defined positive 'normal' number
  2.225073858507201e-308 (type realmin in command window)
- Smallest defined subnormal number
  4.940656458412465e-324
  (2^-1074)

Activity

Typing 1e309 in the command window gives the output

1. 1.0000e+309
2. Inf
3. NaN
4. Some error

To answer, please go to mentimeter link in the chat.

Overflow and Underflow

- Any number larger than the largest defined floating point number causes overflow. MATLAB assigns the result to Inf.
- Any number smaller than the smallest defined floating point number causes underflow. MATLAB assigns the result to 0.

End of Lecture 9

Please provide your feedback here