MONASH University

MONASH
ENGINEERING
ENG1060

DATA TYPES AND LIMITATIONS

Presented by Tony Vo

Slides by Tony Vo

GROUP OF EIGHT AUSTRALIA

THE DETAILS ARE IMPORTANT

- MATLAB is a powerful tool for engineering computing but it has limitations
- Engineers need to know the advantages and limitations of their tools
  - Knowing what MATLAB does well and not so well
  - Choosing the right tools for the job
- Therefore it is important to know the following in MATLAB
  - Data types available
  - How numbers are represented and stored
  - Error metrics

MATLAB DATA TYPES

- We have used these data types

  double → A = 1.5;

  complex → A = 1.5 + 1.5i;

  char → A = 'my text';

  logical → A = ([1 2 3] > 2);

  function_handles → A = @sin

- Other data types (see MATLAB help)
  - Single precision floating point: single
  - Integers: int8, int16, int32, int64
  - Unsigned integers: uint8, uint16, uint32, uint64

BINARY NUMBERS

MONASH University

- Humans count in base-10 (decimal) while computers count in base-2 (binary)
  - Digits in binary numbers are either 0 or 1

- Example:

  Decimal of 19: $1*10^1 + 9*10^0 = 19$

  Binary of 19: $1*2^4 + 0*2^3 + 0*2^2 + 1*2^1 + 1*2^0 = 10011$

- All data on the computer are stored in binary
  - This means 19 is actually stored as a 5-bit binary number 10011 in the computer

BINARY ARITHMETIC

MONASH University

- Calculations in computers are carried out using binary numbers

- Example:

  Binary of 5 is 101 ← (4 + 0 + 1)

  Binary of 7 is 111 ← (4 + 2 + 1)

  Binary addition:

```
  101
 +111
 ----
 1100
```

Just like base-10 addition, with 1 as the largest digit and carries are always 1

1100 in base-10 is 12: $1*2^3 + 1*2^2 + 0*2^1 + 0*2^0 = 8 + 4 + 0 + 0 = 12$

BINARY NUMBERS: BUILT-IN FUNCTIONS

MONASH University

- MATLAB provides built-in functions that convert between binary (base-2) numbers and decimal (base-10) numbers

- Binary to decimal
  - Syntax: number = bin2dec('string')

    dec_num = bin2dec('111')

- Decimal to binary
  - Syntax: string = dec2bin(number)

    bin_num = dec2bin(7)

STORING NUMBERS

MONASH University

- What happens when a number's magnitude (absolute value) is too large?
- Example: Let's assume that there is a limit of 5 bits

  A = 29; B = 7; C = A + B;

  - Decimal: 29 + 7 = 36
  - Binary: 11101 + 00111 = 100100 ← 6 bits!
  - The decimal number of 36 requires 6 bits, but the computer can only store 5 bits
- So what happens?

OVERFLOW AND UNDERFLOW

- Overflow and underflow occur if there aren't enough bits to store the number
- Overflow occurs when numbers become too large
  - That is, the number is larger than the largest representable number
  - E.g. 1×10^{400}
- Underflow is the reverse of overflow, where numbers become too small
  - That is, the number is smaller than the smallest representable number
  - E.g. 1×10^{-400}

OVERFLOW AND UNDERFLOW

- By default, MATLAB stores numbers as doubles
  - Double precision floating point: 64 bits of memory

- MATLAB uses the *IEEE Standard 754* for double precision

Bit	Usage
63	Sign (0 is +ve, 1 is –ve)
62 to 52	Exponent (biased by 1023)
51 to 0	**f** in Binary number 1.**f** (Mantissa)

SEEEEEEE EEEEMMMM MMMMMMMM MMMMMMMM MMMMMMMM MMMMMMMM MMMMMMMM MMMMMMMM

- IEEE 754 can also represent +Inf, -Inf and NaN

REALMIN AND REALMAX

- The commands "realmin" and "realmax" returns the smallest and largest positive normalised numbers, respectively

- If abs(number) > realmax
  - MATLAB will return Inf or –Inf to indicate overflow

- If abs(number) < realmin
  - Precision of calculations will be reduced

- If abs(number) << realmin
  - underflow occurs and MATLAB returns 0

OVERFLOW AND UNDERFLOW: EXAMPLES

- Overflow occurs when numbers become too large
- Underflow is the reverse of overflow, where numbers become too small

```
>> % overflow
>> 1000^1000
ans =
   Inf

>> % overflow
>> -1000^1000
ans =
  -Inf
```

```
>> % underflow
>> 1000^-1000
ans =
     0

>> % Notice that going below real min does not
immediately cause underflow
>> realmin / 4
ans =
  5.5627e-309
```

ROUND-OFF ERRORS

- Round-off errors are caused by the fact that real numbers are not represented perfectly in a computer
  - How many digits or bits do you need for π?

- Example:

```
>> 11*(15/11) - 15
ans =
  -1.7764e-15….
```

- With repeated calculations, such as those in loops, round-off errors can quickly add up over time

```
>> 10000000000000000*(11*(15/11) - 15)
ans =
 -177.6357
```

DISTANCE BETWEEN NUMBERS

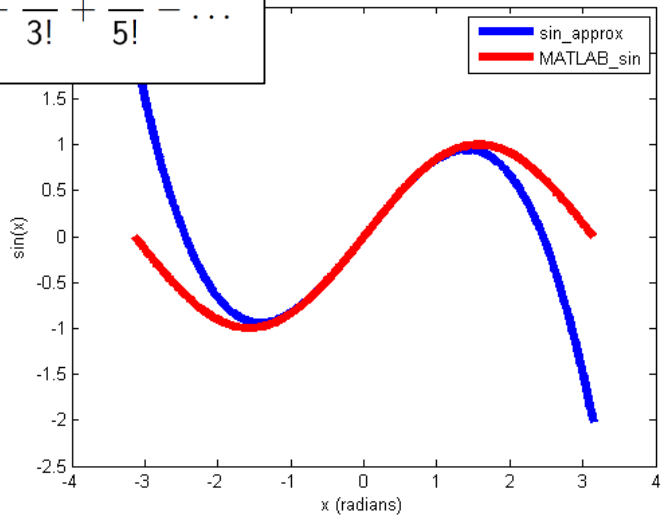
MONASH University

- There are gaps in MATLAB's number line due to the limited number of bits are used to represent numbers
- The function eps(num) returns the epsilon of num
  - Epsilon is the distance between num and the next positive floating point number
  - Defaults to eps(1) if num is not specified
- This means that a number between 1 and 1+eps(1) will not be represented properly
  - Be aware that MATLAB may not display enough digits after the decimal place to show the effects of 1 + eps(1)

TRUNCATION ERROR (NOT ROUND-OFF ERROR)

- Occurs because of inaccurate math model
  - Not the fault of MATLAB or the computer

$$\sin(x) = \sum_{n=0}^{\infty} \frac{(-1)^n}{(2n+1)!} x^{2n+1} = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \dots$$



MEASURING ERRORS: METRICS

- Error metrics are used to measure the error between an approximation and the true result

- Relative error
  - Rel_err = (approx – true) / true

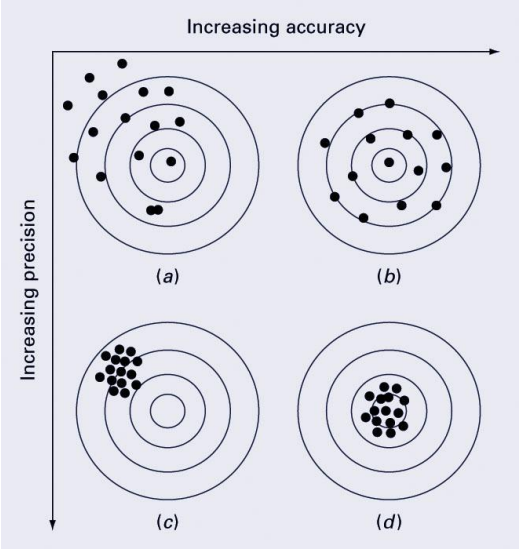
- Percentage error
  - Percent_err = Rel_err * 100

- Absolute error
  - Abs_err = abs(approx – true)

MEASURING ERROR: ACCURACY AND PRECISION

- Accuracy refers to how closely a computed or measured value agrees with the true value
  - This can be evaluated using the mean

- Precision refers to how closely individual computed or measured values agree with each other
  - Evaluated using the standard deviation



SUMMARY

- Computers store information through binary
- All numbers cannot be represented due to limited bits
- Very small and large numbers may undergo underflow and overflow, respectively
- Types of errors

- Why would you want to use single precision instead of double precision?