數值分析

Chapter 0
Introductory to MATLAB

授課教師：劉耀先

國立陽明交通大學 機械工程學系 EE464

yhliu@nctu.edu.tw

110 學年度第一學期

Basic

- MATLAB = MATrix + LABoratory
- Used for mathematical computation, modeling, simulation, data analysis, visualization, graphics, algorithm development
- MATLAB program has (1) tools (2) optional toolbox

- Eight windows can be used: Command window, Figure window, Editor window, Help window…
- Command window:
- >> 2 + 3   executed by pressing "Enter"
- Semicolon (;) to stop the **output** generated by command
- Symbol (%) comment – will not execute
- >> 1+2+3 …

  >> + 4 + 5

Elementary arithmetic operation with scalars

- 5+3; 5-3; 5*3; 5/3; 3/5 = 5\3; 5^3 = 5^3

- Order of precedence
- 5+3*2 = 5+(3*2) = 11 ; (5+3)*2 = 16

- Numerical values can be assigned to variables => letters
  - Mathematical expressions, functions, commands, …

```
>> a=2 ; b = 4; c = a + b
```

- UpperCase ≠ LowerCase (大小寫不相同)
- Override the variable

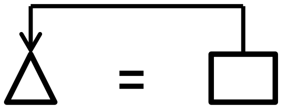
```
>> d = 5; >>d = 4;
```

- Exercise: $\dfrac{18}{5(7)} + 5(9^2)$

Operators

- Assignment operator (replacement operator) : “**=**“

  >> x = 3; x = x + 2; x = x + 2;



  >> 6 = x; x + 2 = 20; (Invalid operation)

  >> x = 5 + y;

- Predefined constants
  - Inf = ∞   NaN   ans

  $pi = \pi$ $i = j = \sqrt{-1}$

- Complex numbers

  >> c1 = 1 - 2i; c2 = 1 + 2i

  >> c1+c2 = ?

Elementary math built-in functions

- sqrt(x) = square root of x
- abs(x) = absolute value of x
- exp(x) = exponential e^x log (x) = natural log; log10(x) = $\log_{10}x$
- sin(x) => x in radians   sind(x) => x in degrees

  >>sin(pi/6) = 0.5; sind(30) = 0.5
- round(x) => round to the nearest integer

  >>round(17/5)

Display format

- default setting: short (four digits)

- format short >> 290/7 = 41.4286
  - Fixed point with four decimal digits for 0.001<=number<=1000
  - Otherwise display format short e

- format long >> 290/7 = 41.42857142857143
  - Fixed point with 14 decimal digits for 0.01<=number<=100
  - Otherwise display format long e

- format short e – scientific notation with four decimal digits
- format long e – scientific notation with 14 decimal digits

- format short g – best of 5-digit fixed or floating point
- format long g – best of 15-digit fixed or floating point

Arrays

- Vector : a row or column
- Matrix : rows and columns

- (1) Variable = [number number number … number]

  Var = [1 3 5 8 10 13] row

  Var = [1;3;5;8;10;13] column

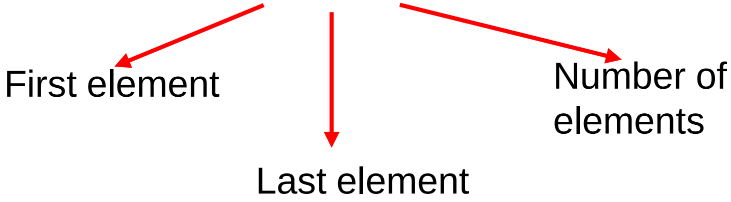
- (2) a series of numbers with constant spacing

  Variable = m:q:n m: first element, n: last element, q:spacing

  x = 1:2:13 or x = [1:2:13]

- (3) “**linspace**” command = > variable = linspace(x_i, x_f, n)

  Var = linspace (0, 8, 6)



First element

Last element

Number of elements

Matrix

- Creating a two-dimensional array
- Variable = [1st row elements; 2nd row elements; …;last row elements]

- A = [5 6 8; 10 2 1; 21 0 3]
  - All rows must have the same number of elements

- >> m=2; n=3; B = [m, n, 2; m*2, 3, 1; cos(Pi/3), sqrt(100), 1]

- Note:
- All variables in MATLAB are arrays
- The variable is defined by the input when the variable is assigned, no need to define the size of the array before the elements are assigned
- A existed variable can be changed to any other size (adding/deleting elements)

Array addressing

- Elements in an array can be addressed individually or in subgroups

- ve = [35 46 23 17 15]   =>   ve(4) = 17; ve(1) = 35

- ve = [34 46 1; 23 17 2]   => ve(1,2) = 46; ve(2,3) = 2

  ve(k, p) refers to the element in row k and column p

- Re-assigning the value of element   => ve(2,3) = 5

- Mathematical calculation
- ve(1,2) + ve(2,3)

Array addressing

- Using a colon “**:**” in addressing arrays: address a range of elements in a vector or a matrix

- Ex: va(m:n) refers to elements m through n of the vector va

- A(:, n)   elements in all the rows of column n
- A(n, :)   elements in all the columns of row n

- A(:, m:n)   elements in all the rows between column m and n
- A(m:n, :)   elements in all the columns between row m and n

- A(m:n, p:q)   elements in row m through n, column p through q

Built-in functions for arrays

- length(A) - returns the number of elements in vector A
- size(A) - returns the size of array A

- >> A = [1 2 3 4 5; 6 7 8 9 10] >>size(A) = 2 5

- zeros(m,n) - creates a matrix with a size of m × n, elements are 0
- ones(m,n) - creates a matrix with a size of m × n, elements are 1

- >> B = zeros(4,3)

- max(A) and min(A) - return the maximal and minimal value in A
- >> max(A); min(A)

- Matrix transpose using 【’】 >>A =[1 2 3] >>B=A’

Strings

- An array of characteristics (typed with the single quote   ‘AAA’)
- Strings can include letters, digits, symbols, and spaces
- Example: ‘abcd’   ‘3%fr2’   ‘{po34d:cd’
- Color of the string: maroon → purple
- Strings are used in output commands to display text messages, in formatting commands of plots, and as input arguments of some functions
- Strings can also be assigned to variables
- >> B = ‘My name is John’

Mathematical operations with arrays

- The mathematical operations in MATLAB can be divided into three categories:
- (1) scalar (1×1 array) single element
- (2) arrays – linear algebra
- (3) element-by-element operations with arrays

- >> VA = [1 2 3]; VB = [4 5 6]; VC = VA + VB
- >> VA – 8   8 is subtracted from each element of VA

- Multiplication of arrays
- A(m,n) B(n,q) => C = A*B => C(m,q)
- >> A=[2 -1; 8 3; 6 7]; B = [4 9 1 -3; -5 2 4 6] >> C = A*B

Mathematical operations with arrays

- Array division: solution of a system of linear equations (Chapter 3.8)

- Element-by-element operation using period "**.**"

- Multiplication **.*** division **./** **.\** exponentiation **.^**

- a = [a1 a2 a3 a4] b = [b1 b2 b3 b4]
- a.*b = [a1b1 a2b2 a3b3 a4b4]
- a./b = [a1/b1 a2/b2 a3/b3 a4/b4]

- Example: A = [1 2 3; 4 5 6] B = [1 2 3; 4 5 6]
- C = A.*B C = A./B

Plotting

- Two-dimensional plots: plot(x,y)   plot(x,y, ‘line_specifier’)
- >> x = 1:1:10; y = 3:3:30; plot(x,y)

- Specify line style   ‘-’ ‘--’ ‘:’   ‘-.’

  color ‘r’ ‘b’ ‘c’ ‘m’ ‘y’

  marker type ‘+’   ‘o’   ‘*’   ‘.’   ‘s’   ‘d’

- Label: xlabel (‘text as string’)

  ylabel (‘text as string’)

  title (‘text as string’)
- Example:
- year = [1988:1:1994]; sale = [8 12 20 22 18 24 27];
- plot(year, sale, ‘--r*’, ‘linewidth’, 2, ‘markersize’, 12)
- %sale2=[27 20 8 8 10 16 25]; plot(year, sale), hold, plot(year, sale2)
- xlabel(‘year’); ylabel(‘sale (million)’);   title(‘Sales Records’);

Plotting

- Contour plot

  ```
  Z = peaks(20); contour(Z); contourf(Z)
  ```

- \>>subplot(m,n,p)

  ```
  >> subplot(2,2,1); contour(Z);
  >> subplot(2,2,2); contourf(Z);
  >> subplot(2,2,3); contour(Z, 10);
  >> subplot(2,2,4); contourf(Z, 'ShowText','on');
  >> colormap autumn
  >> colorbar
  ```

- colormap: setup the color map
- colorbar: show the color bar

Script files (M-files)

- A file that contains a sequence of MATLAB commands
- It can be stored, edited later, and executed many times

- Execute script file: (1) typing its name in command window

  (2) clicking the "RUN" icon from editor window

- Input to a script file:
- (1) define and assign the value in the script file / command window
- (2) use "input" command

- Example:
- y = input ('Enter the value of y:') Value
- y = input ('Enter the value of y:' , 's') String

Output

- (1) disp: disp (name of a variable) or disp(‘text as string’)
- disp(x) ;   disp (AA) ; disp (‘AA’)
-
- (2) fprintf: display output (text and data) on the screen or to save it to a file
- fprintf(fileID, formatSpec, variable 1, …, variable N)
- fprintf(formatSpec, variable 1, …, variable N)
- >> fprintf (‘**X is %4.2f meters or %8.3f mm \n**’, 1, 1000)

- sprintf: write formatted data to an output string
- >>A = [5.00 5.00 5.00 5.00]
- >>sprintf (‘%i %d %f %e’, A)

- Special characters: for insert space, new line…
- \n: new line   \t: horizontal tab   %%: percent character

User-defined functions

- function [output argument] = function_name (input argement)

Input data

Function

Output data

- All the variables in a user-defined function are local, and is recognized only in this program and memory is not the same as the workspace
- The function file does not recognize variables with the same name outside the function
- function_name = function file name   ex: function_name.m

- Example:
- function [Value] = summation(var1, var2, var3)
- Value = var1+var2+var3;
- >> summation(1,2,3)

Anonymous functions

- A simple(one line) user-defined function without creating a separate function file (M file)
- Defined in command window, within a script file, or inside a user defined function

- name = @ (arglist) expr
- >> cube = @ (x) x^3   >>cube(3)
- >> circle = @ (x,y) x^2+y^2   >>circle(2,3)

- Mathematical expression can include built-in or user-defined function
- It is ok to use predefined variables

- Example: write an anonymous function:

$$f(x) = \frac{e^{x^2}}{\sqrt{x^2+5}}$$

Function functions

- A function to be imported into another function
  - Imports another function as an input argument

- Function handle @ : calling functions indirectly (MATLAB data type)
- ex: @cos   @FA   x = @cos

- It can be passed as an argument into another function
- Anonymous function is a function handle

- Example: >> coshandle = @cosd >>coshandle(60)

Subfunction

- A function file can contain more than one functions (primary function + subfunction)

- Example:
- function [GrSort, GrAve] = SortAve (Grades)   %primary function
- GrAve = Ave(Grades);
- GrSort = Sor(Grades);

- function a = Ave(x)   % subfunction
- a = mean(x);

- function x = Sor(x)   % subfunction
- x = sort(x);

Programming in MATLAB

- Relational operator:
- < less than   <= less than or equal to
- >greater than   >= greater than or equal to
- ~= not equal to   == equal to

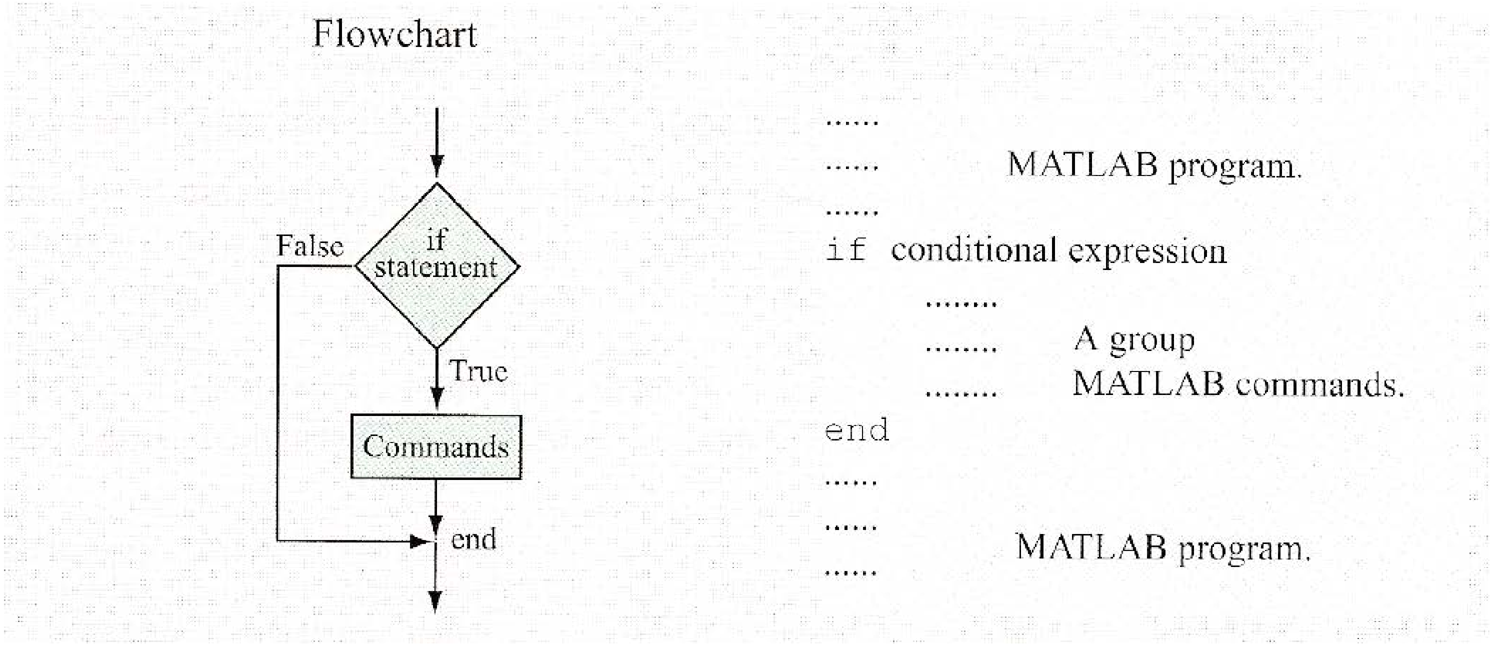
- >> 5 < 8 ➔ ans = 1 (true) ; >> 4 == 6 ➔ ans = 0 (false)

- Logical operator:
- & and   ex: A and B ➔ A&B
- | or   ex: A or B ➔ A|B
- ~ not   ex: ~= not equal to

- Note: use && in scalar operation
- Ex: if A>2 && B<3

Conditional statements

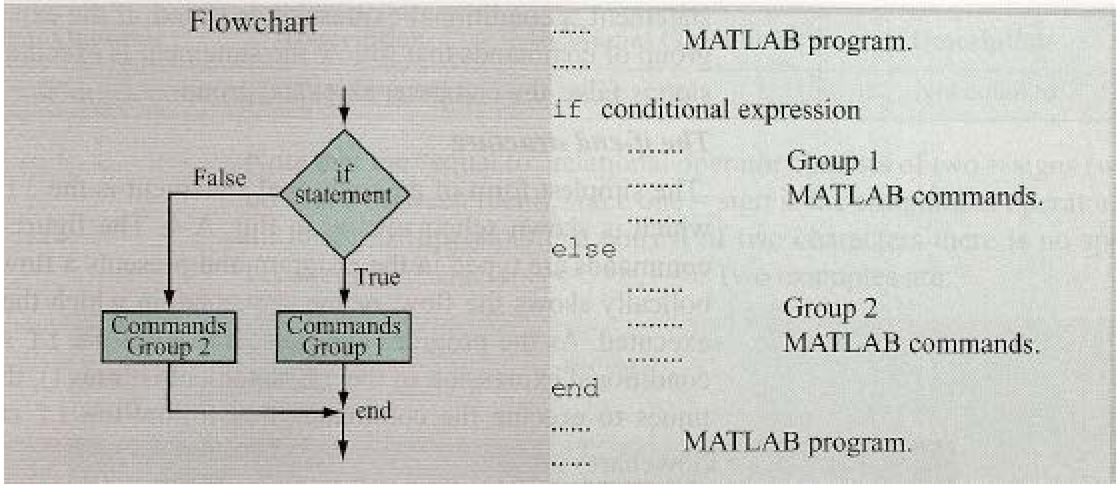
- if – else structure



```
......
......              MATLAB program.
......
if  conditional expression
        ........
        ........        A group
        ........        MATLAB commands.
end
......
......              MATLAB program.
......
```

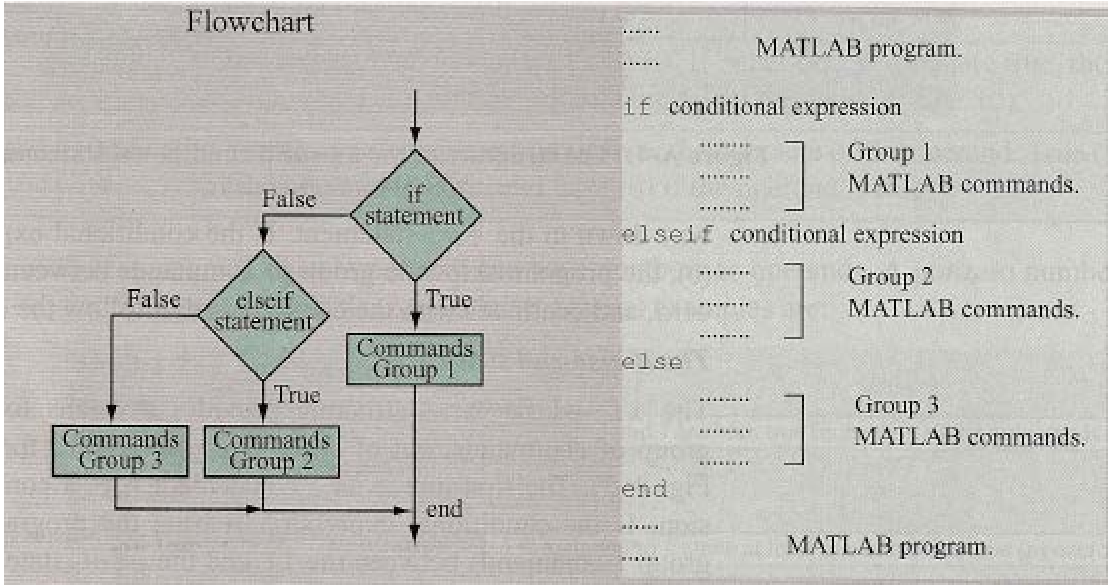
Conditional statements

- If-else-end



```
......
          MATLAB program.
......
if  conditional expression
        ........      Group 1
        ........      MATLAB commands.
        ........
else
        ........      Group 2
        ........      MATLAB commands.
        ........
end
......
          MATLAB program.
......
```

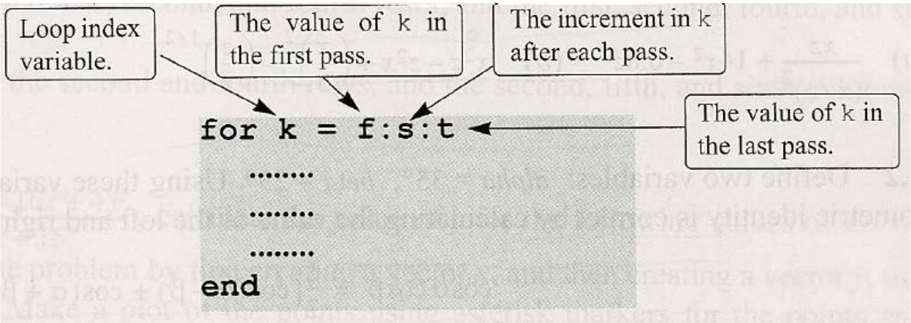
- if-elseif-end



```
......
          MATLAB program.
......
if  conditional expression
        ........   ]  Group 1
        ........   ]  MATLAB commands.
        ........   ]
elseif  conditional expression
        ........   ]  Group 2
        ........   ]  MATLAB commands.
        ........   ]
else
        ........   ]  Group 3
        ........   ]  MATLAB commands.
        ........   ]
end
......
          MATLAB program.
......
```

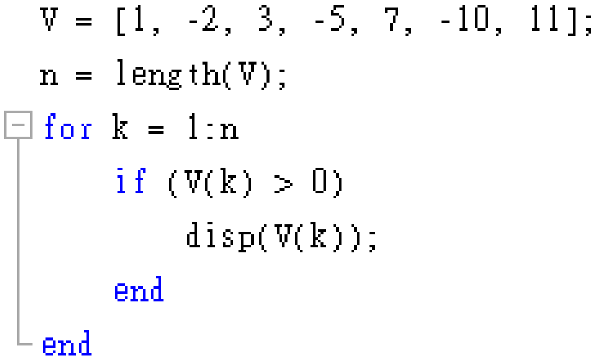
Loops

- for-end loops


Loop index variable.
The value of k in the first pass.
The increment in k after each pass.
The value of k in the last pass.

```
for k = f:s:t
    ........
    ........
    ........
end
```

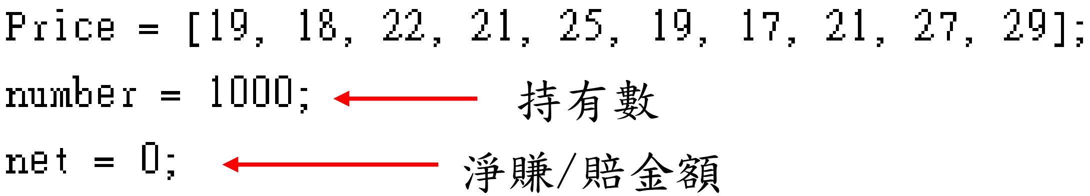
- Ex: for k = 1:2:9 ➔ k = 1, 3, 5, 7, 9
- If the increment s is omitted, its value is 1(default)
- Ex: for k = 3:7 ➔ k = 3, 4, 5, 6, 7
- Example:



```
V = [1, -2, 3, -5, 7, -10, 11];
n = length(V);
for k = 1:n
    if (V(k) > 0)
        disp(V(k));
    end
end
```

Problem

- 假設某股票10天的股價變化如下[19, 18, 22, 21, 25, 19, 17, 21, 27, 29]。剛開始持有1000張，當低於20元時買100張，高於25元時賣100張。在10天後的淨賺(賠)金額與股票數為何?



```
Price = [19, 18, 22, 21, 25, 19, 17, 21, 27, 29];
number = 1000;
net = 0;
```

股價

持有數

淨賺/賠金額

while loop

- Repeatedly execute statements while condition is true

```
while (expression)
        statements
end
```

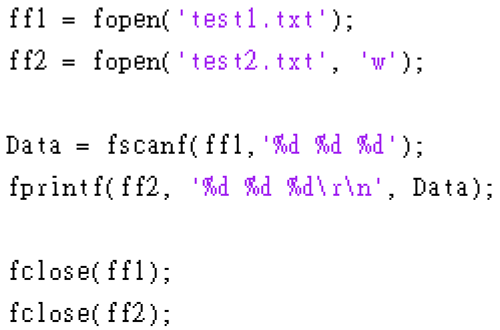
- Example1:

```
m = 0;
while (m < 10)
     m = m+1; disp(m);
end
```

- Example: If you deposit 500 dollars each year, how many years does it take to achieve 10,000 dollars (rate = 5%)?

File Input/Output

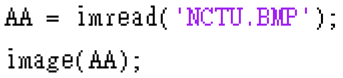
- fopen: open file or obtain information about open files
- fileID = fopen(filename);
- Example:



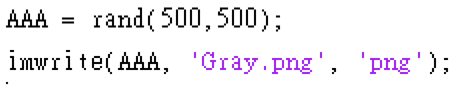
```
ff1 = fopen('test1.txt');
ff2 = fopen('test2.txt', 'w');

Data = fscanf(ff1,'%d %d %d');
fprintf(ff2, '%d %d %d\r\n', Data);

fclose(ff1);
fclose(ff2);
```

- fscanf: Read data file from text file
- fprintf: write data file to text file

- Open image files: imread
- Example:



```
AA = imread('NCTU.BMP');
image(AA);
```



```
AAA = rand(500,500);
imwrite(AAA, 'Gray.png', 'png');
```