Tuesday, April 1, 2025 5:50 PM

Notes For matlab

Variables

a variable is used to assign a value to something

- syntax to create a variable

Variable_name = value or expression

Num_one = 1

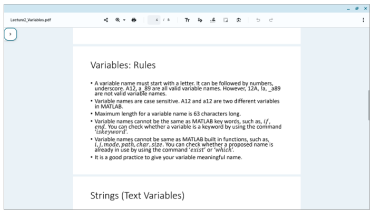
displaying a variable

use the "disp" to display the value of a variable or display stuff you typed

disp(num_one) = which would display value of variable

disp("hello world") = will display what ever is within the Parenthesis to terminal

rules for variables



- these are not meant to be memorized with practice you will understand each much better

Matlab Mathmatical Functions

- using Sin or cos will result in answers in radians to get degrees use Sind and cosd
- Ceil (amount) = this will round up (think ceiling up)
- Floor (amount) = this will round down (think floor down)
- round (amount) = this will round to the nearest integar

Vectors

- row vector

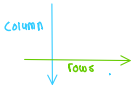
row_vector = 1 2 3 4

Size(row_vector) = 1 4 (rows, columns)

- Column Vector

column_Vector = 1; 2; 3; 4

Size(column_vector) = 4 1 (rows, column)



"Size" will give you the size of a vector including rows and columns

- ; Semicolon
  - in matlab the semicolon does many stuff including but not limited to going to the next line, suppressing the output
    - when a semicolon is placed after something in the program the program still uses it but does not display it to the terminal
    - this can also help with going to the next line

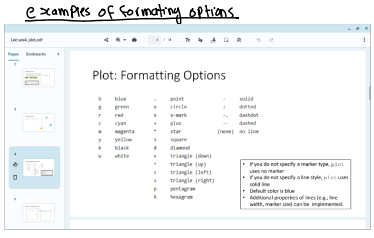
Plotting Points

- matlab command to create line graph

Plot(x-coordinate, y-coordinates, option formating)

Plot(x, y, "of")

examples of formatting options



Plot(x, y, "b--", "Linewidth", 2.5)

the "b" represents the color while the dotted line represents that the line will be dotted

Vectors Continued

- Creating a vector in matlab: manual

- row vector

row_vector = [1 2 3 4]

disp = 1 2 3 4

- Column Vector

column_Vector = [1; 2; 3; 4]

disp = 1 2 3 4

notice how semicolons were used to move the number to the next line

- Creating vectors in matlab: colon operator

a = 2 : 0.5 : 7

start, increments, end

- this will start at 2 and end at 7 it will go by 0.5 increment until it gets to 7
- if you want to go backward you will do the same exact thing you will place a start and a finish but for the increments you place a negitive number

- linspace

has a start and ending point but breaks down the increment into the equal parts

linspace(0, 10, 5)

= 0 2.5 5 7.5 10

it begans from 0 and it ends at 10 we also wanted this to be into 5 parts this is what is meant when we say broken into equal increments

squaring a vector

- when you want to square a vector you use .^ and you must include the dot this will square each term inside the vector

V1 = [2 3 4] V1.^2 = [4 9 16]

multiplying two vectors

- you really arent multiplying and actual vector but instead you are multiplying each individual term inside the vector with each individual term in the other vector but to multiply a vector you will use V1 .* V2

Finding the max and min From Vector

when you want to find max use max(V1) and whe you want to find min use min(V1)

Find the sum and Mean of a vector

when you want to find the sum of a vector use sum(V1)

when you want to find the mean of a vector use mean(V1)

Matrix

a matrix is a rectangular array of numbers arranged in rows and columns

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

Solving linear equations

- they must be in order meaning they must be x's or y's the variable doesn't really matter as long at its the same through out the column. if all of the equation have a term with x in them and one does not you can add a zero "letter" but you cannot ignore it

this is what was meon by the same letters in the colum all x next one ally

$$\begin{aligned} 3x + 2y - z &= -4 \\ -2x - 4y + 3z &= 9 \\ x + y + z &= 6 \end{aligned}$$

$$A = \begin{bmatrix} 3 & 2 & -1 \\ -2 & -4 & 3 \\ 1 & 1 & 1 \end{bmatrix} \quad B = \begin{bmatrix} -4 \\ 9 \\ 6 \end{bmatrix}$$

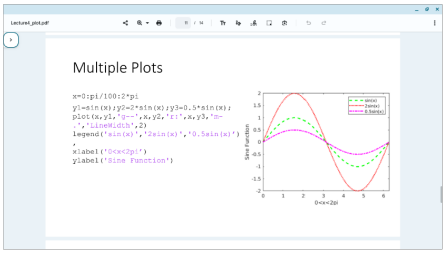
- A\B = [-1; 2]

keep in mind that B is divided by A $\frac{B}{A}$ because A has the x-values

labeling the graph

xlabel("name") will label the x-axis
ylabel("name") will label the y-axis
title("name") will give the graph a name

Graphing multiple graphs



Sub Plot

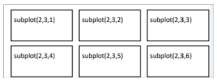
the subPlot splits the window into several sub-window

subPlot(m,n,p)

m = rows
n = columns
p = points to a particular sub window

example:

subPlot(2,3,1)



you have
if you wanted three columns you would change the two to three

Vector accessing elements

lets say you have vector a=[2 4 6 10]
you can use "end" to get the last element in the vector a(end)=10 if you want to know the second element in the vector you can use a(2)=4 or you can do this a(2:end) = will give you A(2) A(3), A(4), A(5) or you can do this A(:) = will convert a row to a column vector it does not convert a column vector to a row vector

Matrix arithmetic: addition/subtraction

- matrices must be the same sizes

$$A=\begin{bmatrix}3 & 9 & 6\\ 4 & 1 & 20\end{bmatrix} \quad B=\begin{bmatrix}5 & 8 & 10\\ 21 & 5 & 7\end{bmatrix}$$

these two are the same size to check use "size" variable

D = A+B

- the same thing goes for subtraction the size of the two matrices must be the Sa

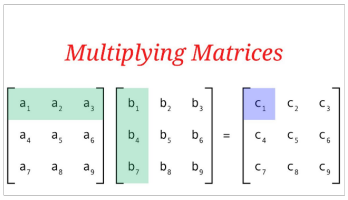
Matrix arithmetics: multiplication

- when you are multiplying matrices the number of column in on matrix has to be equal to number of rows in the other matrix

size(A) = 2 · 3 size(B) = 3 · 2

the inner dimensions match that mean you can multiply the two matrices

the reasoning on why this happens can be explained below with a picture



Creating a matrix in matlab

- matlab elements must be in brackets

matlab command:

A = [1 2 3; 4 5 6 ; 7 8 9]

this command uses ; which tells to go to the line

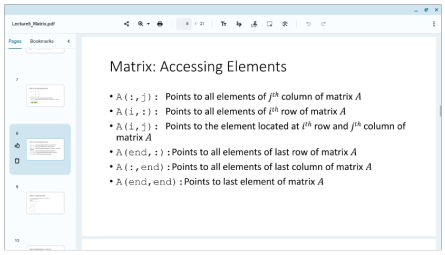
So it should look like something like this

$$A=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\end{bmatrix}$$

matrix: accessing elements

you can accesses an element by referring to its location

A(3,2) = 8
row# column#



assigning values

- a value can be assigned to any element of the matrix the command is:

A(i,j) = desired number
row# column#

deleting a row or column

- to delete a row or column use a empty or null operator

A(:,3) = [] => deletes column 3

this must be included what that does is tell the program I want that space to be empty or in other terms delete it

A(4,:) = [] => deletes row 4

Creating a sub-matrix

- a sub-matrix is a matrix created from another matrix

$$A=\begin{bmatrix}1 & 2 & 3\\ 4 & 5 & 6\\ 7 & 8 & 9\\ 10 & 11 & 12\end{bmatrix}$$

B = A([2 3], [1 2])
rows columns

result:

$$B=\begin{bmatrix}4 & 5\\ 7 & 8\end{bmatrix}$$

that is the sub-matrix

Conditional Statments

Relational Operators	Description
==	Determines equality
>=	Determines greater than or equal to
>	Determines greater than
<=	Determines less than or equal to
<	Determines less than
~=	Determines inequality

logical operator

- compare true or False

when using "and" both statments must be true if one of the statments is False the whole thing is False

when using "or" operator either one can be true they both are Not required to be true if one is False as long as the other one is true the entire Statement becomes true

Logical Operators	Description	
&&	Logical AND	Short-circuit operators
||	Logical OR	Short-circuit operators
~	Logical NOT	
&	Elementwise AND	
|	Elementwise OR	

File Management

Import/Export (Text files)

Command	Description	Example
importdata	Load data from file	A=importdata(filename)
csvread/csvwrite	Read/write comma separated value	M=csvread(filename)
dlmread/dlmwrite	Read/write ASCII delimited file of numeric data	M=dlmread(filename)
textscan	Read formatted data from text file or string	C=textscan(fileID,formatSpec)

Import/Export (Excel)

Command	Description	Example
xlsread	Read Microsoft Excel spreadsheet file	Num=xlsread(filename)
xlswrite	Write Microsoft Excel spreadsheet file	xlswrite(filename,A)
xlsinfo	Determines if file contains Microsoft Excel spreadsheet	Status=xlsinfo(filename)

Import/Export (low level files)

Command	Description	Example
fopen	Opens file or obtain information about open file	fileID=fopen(filename)
fclose	Close one or all open file	fclose(fileID)
feof	Test for end of file	Status=feof(fileID)
ferror	File I/O error information	Message=ferror (fileID)
fgetl	Read line from file, removing newline characters	Tline=fgetl(fileID)
fileread	Read contents of file as text	Text=fileread(filename)
fprintf	Write data to text file	fprintf(fileID,formatSpec,A1,...,An)
fread	Reads data from binary file	A=fread(fileID)
frewind	Move file position indicator to beginning of open file	frewind(fileID)
fscanf	Read data from text file	A=scanf(fileID,formatSpec)
ftell	Current position	Position=ftell(fileID)
fwrite	Write data to binary file	fwrite(FileID,A)

Manage File and Folders

Command	Description	Example
mkdir	Creates a new folder	mkdir('parentFolder','newdir');
movefile	Moves a file or folder; You cannot move a folder while it is on the search path.	movefile('myfile.m','d:/work');
movefile	File names must start with a letter, and contain letters, digits, and underscores.	movefile('myfile.m','oldfile.m');

Search Path Commands

Command	Description	Example
userpath	View or change default user work folder	
path	View or changes MATLAB directory search path	path(newpath)
addpath	Add folders to search path	addpath(folderName1, . . .,folderNameN)
rmpath	Remove folders from search path	Rmpath(folderName)
savepath	Save current search path	savepath
genpath	Generate path name	
pathsep	Search path separator for current platform	

equation Solver and curve Fitting

- using syms x

this should be used when you want to Solve a Function if syms x is Not Placed on top of the page Matlab will have No idea what you are talking about

must be included → syms x

```matlab
syms x
eq = x^2 - 4 == 0;   % The equation to solve
solutions = solve(eq, x);   % Solving for 'x'
disp(solutions);   % Outputs: x = -2, 2
```

= this means an assignment operator

== this means equality

PolyFit

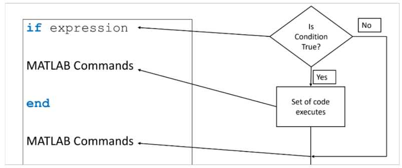
Polyfit

- `p=polyfit(x,y,n)` returns the coefficients for a polynomial $p(x)$ of degree n that is a best fit (in a least-squares sense) for the data in y.
- Example: Generate 10 points equally spaced along a sine curve in the interval $[0,4*pi]$.

```matlab
x = linspace(0,4*pi,10);
y = sin(x);
p = polyfit(x,y,7);
x1 = linspace(0,4*pi);
y1 = polyval(p,x1);
figure
plot(x,y,'o')
hold on
plot(x1,y1)
hold off
```

Conditional Statements: if

- this statement is Just a if, elseif
  - you can add as much elseif statements as you want also they must close every one of these statment with an "end"



Loops

- a loop is a repatitive block of code
- While loop and For loop
- For loop are used to calculate stuff
- while loop are used to when you want a Condition to be met
  - to terminate a while or For loop use a break statement

Function

- these can be called to action when ever you need them so instead of writing New code you can simply use a function
- the name of Function and the File of that Function must be the same if they are Not it will Not work

this is how you introduce a Function →

```matlab
function [output1, output2, ...] = functionName(input1, input2, ...)
    % Function description or comments (optional)
    % You can describe what the function does here.

    % Code to perform the intended operation
    % (The logic that your function will execute)

    % Example:
    % output1 = input1 + input2;

end
```

Closes Function → end

FunctionName will be the name of the File because when the Function gets called that is how Matlab will Find it

You do Not need to use output this is Just there to assign used input or Just input in general For later use in the Program

★ do not use "clc" or "clear" in a File where a Function is placed it will clear it and the Function will not work

roots

- roots(P) returns the roots of the Polynomials

$x^2+3x-10=0$

- the largest coefficent is 2 you will take x^2 and go all the way down to x^0 No matter what the equation is you will Find the largest coefficent and go to zero

$x^2 \; x^1 \; x^0$

$[1 \;\; 3 \;\; -10]$

x^4+3x^2+x-1

$x^4 \; x^3 \; x^2 \; x^1 \; x^0$

$[1 \;\; 0 \;\; 3 \;\; 1 \;\; -1]$

if it does Not exist like x^3 its Just a zero so this will be the equation roots will Solve

$p = [1 \; 0 \; 3 \; 1 \; -1] \Rightarrow$ roots(P)