Array multiplicⁿ A.*B

Matrix multiplicⁿ A*B

Array division < A./B a/b
A.\B b/a

Matrix Division complicated A/B
A\B

Array exponentiation 2^3 $\underset{\text{matrix}}{X}$.^$\underset{\text{matrix}}{N}$

Precedure

0 High - Parenthesis

1 - Exponentiation & Transposition '

2 +,- - Unary+, Unary-

3 - mult. & Div. (Array & matrix)

4 - Addition & subtraction

5 low - Colon operator :

6 Relational operators < <= > >= == ~=

7. Element-wise logical "and" : &

8. Element wise logical "or" : |

9. logical "and" : &&

10. logical "or" : ||

To continue a command to next line ... 3 dots

Functions

```
>> rand(3,4) ← stored function
```

creates a random (3,4) matrix with all numbers b/w 0 & 1

use help (or) doc

→ range can be changed as

```
>> 1 + rand(3,4)*9
```

so no.s are now b/w [1 & 10)

to get in other doc

To create our own function.

sin( ) sind( ) ↑ degree

→ we need to open editor

```
>>edit
```

In editor create your own fncⁿ & save it as a file (*.m)

```
>> function myRand
     a = 1 + rand(3,4)*9
   end  ← (local variable)
```

↳ Now this fncⁿ can be used

↳ The value of a variable remains only till function is running

→ so some changes to use the value in MATLAB.

x(2,end)

x(end,2)

x([2 3])

```
function a = myRand
    a = 1 + rand(3,4)*9;
end
```

Now in Matlab b = myRand

scope: set of fncⁿ that can access a variables

↳ You can also give input to function

```
function a = myRand(low, high)
    a = low + rand(3,4)*(high-low);
end
```

```
>> a = myRand(2,5);
```

↳ Can get more than 1 Outputs

```
function [a, s] = myRand(low,high)
    a = low + rand(3,4)*(high-low);
    v = a(:); ← get all element of a in a single column vector
    s = sum(v);
end
```

```
>> [a ss] = myRand[2,5);
```

To know about which ~~built in~~ functions exist

```
>> help exist
```

Sub functions

In same editor

main fncⁿ → function → first functⁿ & only fncⁿ that can be called from outside

=

subfncs — function

=

↓

can be called only by fncns within same editor file

To Make a variable in FUNCTION GLOBAL before that variable type global

| global v; ← Need to define (seperately)

Script

It is also written in editor, but does not accept values.

→ what is there in script is outputted to Matlab commandWindow

>> edit endlesson

creates & opens script.

```
fprintf ("concludes lesson");
pause(5); ← waits 5 secs b4 next command)
quit;
```

Polymorphism (output changes with input format)

sqrt(1) → 1

sqrt [1 4] → [1 2]

sum [1 2 3 4] → 10

sum $\begin{bmatrix} 1 & 3 \\ 2 & 4 \end{bmatrix}$ → [3 7]

zeros(1, 2) → [0 0]

zeros(2) → $\begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$

ones(1, 2) → $\begin{bmatrix} 1 \\ 1 \end{bmatrix}$

5* ones(2) → $\begin{bmatrix} 5 & 5 \\ 5 & 5 \end{bmatrix}$

diag(7 3 9)

$\begin{bmatrix} 7 & 0 & 0 \\ 0 & 3 & 0 \\ 0 & 0 & 9 \end{bmatrix}$

randi (10, 5, 4) ← (5, 4 matrix) if replaced by 5 it will take it as 5,5

↑ random ↑ integers ← (1 to 10)

randi ([5,10] 5, 4]

↑ integers from 5 to 10

randn (5) → [5x5 with stand devn. = 1] mean = 0

↑ normal/standard/Gaussian distribution

rng(0)

↑ sets the random numbers to initial

rng('shuffle')

↳ actually shuffles using microsec clock

Input /Output

In a function.

x= input('Gimme a number');

fprintf('This concludes lesson 3 \n')

fprintf ('%d items %.2f each\nTotal = $%5.2f\n', n, price, total);

→ %d → whole numbers (scientific notations except for) ↳ conversion character (format specifier); ampersent sign

→ %.2f → ↳ 2 digits after decimal

→ \n → escape character new line

%5.2f → fixedprint notations

↳ print using atleast 5 spaces ↳ - sign; digits b4 decimal; after decimal; decimal point

blanks b4 number ←

↳ How to write %

↳ %%

↳ \ ↳ \\

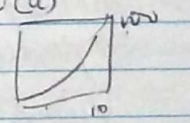
↳ ' ↳ ''

Recycling format while printing

Plotting

a=(1:10).^2

plot(a)



plot (a, b) ↑ horizontal → vertical

Two plots on same figure

↳ plot(x₁, y₁, x₂, y₂)

plot(x1, y1, 'r', x2, y2, 'k':')
↑ red ↑ black dotted
(:) → only the points

>> doc color

→ If in a figure, you don't want replace earlier plot,

>> hold on

to replace earlier plots

>> hold off

>> grid → to on

> grid → again to off

>> title(' ')

>> xlabel(' ')

>> ylabel(' ')

> legend('graph1', 'graph2')

--- bin
— wing

→ box on right side top to denote type of line & representing graph)

to turn off axis
axis off

>> axis(a b c d)
left-right range, bottom top range

Debugging.

(syntax error
└ Matlab catches this
semantic errors

⇒ Matlab has built-in debugger

In function
⇒ click - it turns to red to hold for debugging step by step

x left lim, x right lim → y
axis([0, 12, -10, 20])
to change the range of axis

imread()
to read .jpg as no.
← to see image → image()

lec-5.

Selection if / elseif / else | return to quit the function

If statement

if else statement .

function - -
└ if (condition)
block 1
else
block 2
~~else~~ elseif

⇒ end after completion

Relational operators

== is equal to

⇒ ~= is not equal to

> greater than

< lesses than

>= greaterthanequal to

<= less than equal to

(anything other than 0 is true) (2==3-1) → Answer =1 ⇒ True (else 0 for false)

» [4 -1 7 5 3] > [5 -9 6 5 -3]

① └→ [0 1 1 0 1]

» [4 -1 7 5 3] <= 4

② └→ [1 1 0 0 1]

logical operators

> logical values

Non-zero : true
zero : false

Matlab returns 1 for true.

logical operators:

&& and
|| or
~ not (one operand)

[1 -3 0] && [pi 0 0]
└→ error

→ [1 -3 0] & [pi 0 0]

array version (only single 1 or &) no need for , too
(b/w vectors
└→ [1 0 0]

Polymorphic functions.

functions that behave differently based on

1. → No. of input or output arg.

2. → Type of input or output arguments

Many built in fncs are Poly. fn
eg. (sqrt, max, size, plot, etc)

Scanned by CamScanner

How to Make our function POLYMORPHIC

Two built in fncns

nargin → # of actual input argument that fnc was called with

nargout → # ~~~ that function caller requested

ROBUSTness

→ A fncn is robust if it handles erroneous input and output arguments, and → provides a meaningful error message

comments

following %

* anything as comments in a function can be brought down to command screen using help function.

* Persistent Variables

Local Variable, BUT its value persists from one call of the fncn to the next

→ Relatively Rarely used. None of BAD side effects of global variables

FOR — While

Loops

```
loop { for n = 1:N   - control statement
     {   ~~~~        } body of loop
     { end
```

1:N → Doesn't have to be a scalar → can be any array

```
     { while total < limit     → condition / control statement
loop { body { n=n+1            → increment / decr
     {      { total = total+n;
     { end
```

Break ; Continue

*. Logical Indexing

eg: w = v(v>0) ⟶ All positive values of v will be stored in w.

(v>0) → logical values.

selects only those values of v which have logical value true.

eg :- v = [-1 0 2]

d = (v>=0) / d = v>=0

= [0 1 1]

v(d) = [illegible]

[0 2]

Logical indexing can be SIMULTANEOUSLY be done on left & right side of '='

for a vector

For an array → L.I on right changes dimension

on left assigns value without changing dimension

Logically using L.I on both sides of array can still retain indexing dimensions.

Scanned by CamScanner

Eg; toc
time for running the program

Preallocation (it's good to allocate a matrix with a size so that it saves time for reallocation of earlier row/colum before adding another element.)

$\boxed{A = zeros(N, N);}$

Datatypes

class( ) → double
  ↳ char
  ↳ logical

whos( )

double - DEFAULT type in MATLAB
- floating point representation
  64-bits (8 bytes)

single - 32-bits floating point

$2 \times 4 = 8$

Integer types - signed
  ↳ unsigned (non-negative)
  (8-, 16-, 32-, 64- bit long)

Inf → Infinite

NaN → Not-a-Number

Functions →
- class
- isa(x, 'double') - true or false.
- range check
- intmax, intmin, realmax, realmin

Strings **ASCII charcode** (0-127)

char( )
↳ gives the char at that particular number

fprintf('%s', 10)
↑ print as string

String functions

char, str2num( ), num2str( )

sprintf → assigns the value
NOT like fprintf

Structures

variables with variable.

r.name = 'mohit'

class(r) → 'struct'

class(name) → char

struct can hold arrays & arrays can hold struct.

isfield(account, 'owner')
checks if there is a field owner in ~~field~~ struct account.

rmfield(account, 'owner')
→ removes the field.

Cells

uses Pointers (i.e. stores address ~~of~~)
powerful to store heterogeneous data.
(Used more frequently than struct).

NEW SYNTAX
use: { }

⇒ Cell Arrays also exist

(In MATLAB, 2 cells pointers can ~~NOT~~ point to same location)

change something in string
strrep string replace

File Input / Output

pwd - present working directory
cd - change directory
ls - list in directory
mkdir - makes new foldes
rmdir - removes only empty folder

→ save everything in command window
using
>> save (saved as matlab.m)

or >> save mohit.m (saved as)

→ to load save file
> load
or >> load mohit.m

Excel files

xlsread
xlswrite

[num, txt, raw] = xlsread(.xls)

smallest rectangles in spreadsheet are copied

xls write → replaces only the part of existing files being written in.

text files

> Opening fid = fopen(filename, permission)
Closing fclose(fid).

fid → unique file identifier

permissions, rt, wt, at, r+t, w+t, a+t
↑
text

fgets( ) → gets one line at a time
& in loop automatically goes to next line
→ if no new text return (-1)
hence while printing lines start with ischar("line")
fwrite

In text files

parsing ~~string~~ to get numerical data is much harder.

Hence ~~[illegible]~~ Binary files are suited for numerical data

Binary file

permissions, r, w, a, r+, w+, a+.

n = fread(fid, data name type, datatype)
↓
returns ~~column~~ vectors
↳ & always in double

Scanned by CamScanner