MATLAB Basic

1. System commands

save <filename> load <filename>

who % show var whos % var details

clear /x?/ % del x or all

input(s) % show s and wait for input

format % switch short(4) / long(16) / bank(2) /

% short e / long e / rat / + (pos, neg, 0) /

% compact / loose

disp % display value without "var = "

fscanf % Read formatted data from a file

fprintf % Formatted writes to screen or file

% MUST have format char

%s % string %d % integer

%f % float %e % %f + exp

%g % whichever more compact in %f or %e

1. Notations

% % comments … % continue line

; % hide output

1. Var
   1. Declare and assign
   2. x = 3 🡪 1 X 1 matrix
   3. Multi-assign: a = 2; b = 7; c = a \* b
   4. Vector: 1-d array
      1. Row vector, space/comma delimit
      2. Col vector, ; delimit

r = [7 8 9 10 11]; t = [2, 3, 4, 5, 6]; res = r + t

% res = 9 11 13 15 17 % row vector

c = [7; 8; 9; 10; 11]

% res = 7\n 8\n 9\n 10\n 11 % col vector

* 1. Special variables

Eps % Smallest incremental number

1. Vector, Matrix and Array Commands

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

% m = 1 2 3\n 4 5 6\n 7 8 9\n

1. Data structures
   1. string consists of char.
   2. function handle: pointer

str = 'Hello!', n = 2345, d = double(n)

un = uint32(789.50),rn = 5678.92347, c = int32(rn)

logical(10) = 1 logical(0) = 0

* 1. determine ds: is…

1. Operators
   1. Assignment: NO ++ and +=
   2. Arithmatic: A, B

+, -, .\*, ./, .\ % A, B sais me size or one scalar

\* % dot product, a.col = b.row

.\* % e-wise prod, same size

/ % B\* inv(A)

~= % **not eq**

* 1. Logical

~ % not, highest precedence

& % and, equal precedence with or

| % or

1. Condition

if …; elseif…; else…; end

switch(var); case <>; …; otherwise; …; end

1. Loop: break, continue

while n > 0; …; n = n – 1; end;

for a = start:step:end;… % default step is 1

% **both end inclusive**

for a = Array; …

MATLAB Vector

1. Intro

r = [1, 2, 3, 4] % row vector

c = [1; 2; 3; 4] % col vector

v = [first:step:end] % vector

r(start:end) % access, BOTH END INCLUSIVE

1. Ops
   1. Same type, same size or one is scalar

r1 = [1, 2, 3]; % row

c1 = [1; 2; 3]; % col vector

s1 = [1]; % scalar

disp(r1 + r1) % 2, 4, 6

disp(c1 + c1) % 2\n 4\n 6

disp(r1 + s1) % 2, 3, 4

% same goes to -, .\*, ./, /\

* 1. Scalar Multiplication

r1 = [1, 2, 3]; % row

s1 = [5];

disp(r1 \* s1)

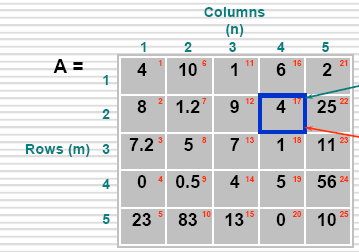
* 1. ': transpose, row <=> col
  2. Appending: horizontal: [r1, r2]; vertical: [r1; r2]
  3. Magnitude: |v| = sqrt (sigma(v\_i^2))

|v| = sqrt(sum(v.\*v)) % element wise product

* 1. Dot product: dot(vec1, vec2)

MATLAB Matrices

1. All variables are matirices ⬄ array.
   1. Vectors: matrices w/ 1 row OR 1 col
   2. Scalar: matrices w/ 1 row AND 1 col
2. Access: A(2, 4) or A(17), **START FROM 1**.



1. Extension

B = [1, 2; 3, 4] % repeate matix in each dimension

Br = repmat(B, 1, 2) % reapmat(M, d1, …, dn)

**Bc = [B, 2\*B; 3\*B, B^2] % concatenate**

1. Functions

zeros(n) % n \* n array of 0s

ones(n) % n \* n array of 1s

eye(n) % n \* n unit matrix

rand(n) % uniformly dist rand (0, 1)

magic(n) % row, col, diag same-sum array

inv(M) % inverse

MATLAB Array

1. Vecter & matrices are 1d/2d special case of arrays
2. Some special arrays
3. Multi dimensional array
   1. Create

A = [1 2 3; 4 5 6; 7 8 9] % 2-d array

A(:, :, 2) = [0 0 0; 0 0 0; 0 0 0] % add a dimension

B = cat(d, A1, …, An) % concatenate along dimension d

B = [A1, A2…]

* 1. Colon notation
     1. Create vector: start:step:end
     2. Slice array: A(start: end) similar to Python [start: end:step] notation, BUT both end included, **INDEX start from 1.**

A = 1:1:10 % 1..10

A = linspace(start, end, num)

* 1. Special multi-D arrays

zeros(n\_1, …, n\_n) % multi-D array of 0s

ones(n\_1, …, n\_n) % multi-D array of 1s

rand(n\_1, …, n\_n) % multi-D rand array

magic(n\_1, …, n\_n) % multi-D magic array

1. Array Functions

length(a) % largest length in all dimensions

ndims % number of dimensions

numel % number of elements

size % length of each dimension

sort(a, n) % sort along n-th dimcell

iscolumn % determin col vector

isrow % determin row vector

ismatrix % determin matrx

isempty % determin empty

is… % true / false

reshape % reshape to another dimension

MATLAB Functions

1. Any fuction EXCEPT anonymous one must be in file <name.m>. 1 + n sub-functions.
   1. Primary function: only 1, at first, callable from outside.
   2. Sub-functions: any number, comes after. Only visible in the file scope
   3. Private funcitons: in private folder, only visible to functions in the parent folder

function [out1, o2…] = name(var1, var2…)

% instructions, help(func\_name) to show

end

1. Anonymous functions

Name = @(args)expression

power = @(x, n) x.^n

1. Glocal variable: declare before use

Global TOTAL

1. Method as Var: feval(method, arg1, …)

MATLAB I/O

1. High level read API

% read from file

A = importdata(filename,delimiterIn,headerlinesIn)

% read from clipboard

A = importdata('-pastespecial')

for k = [1:7] % view data

disp(A.colheaders{1, k})

disp(A.data(:, k))

disp(' ')

end

MATLAB Plotting

hold on % preserve old graphic

hold off % clear old graphic



MATLAB OOP

1. Structure

classdef name<superclass

properties # instance vars

var;

end

methods # instance method

function self = set(args)

self.var = args;

end

end

methods (Static = true) # class method

function classMethd(agrs)

fprintf(“hello %s”, agrs)

end

end

end

1. An example

classdef Dog

properties % instance var

name;

age;

msg;

end

methods % instance method

function obj = Dog() % constructor

end

function obj = setInfo(obj, name, age)

obj.name = name;

obj.age = age;

end

function rst = bark(obj, times)

rst = '';

for i = 1:times;

rst = ['Hello, dog ', obj.name, '! ', rst];

end

end

end

end

1. Vecter & matrices