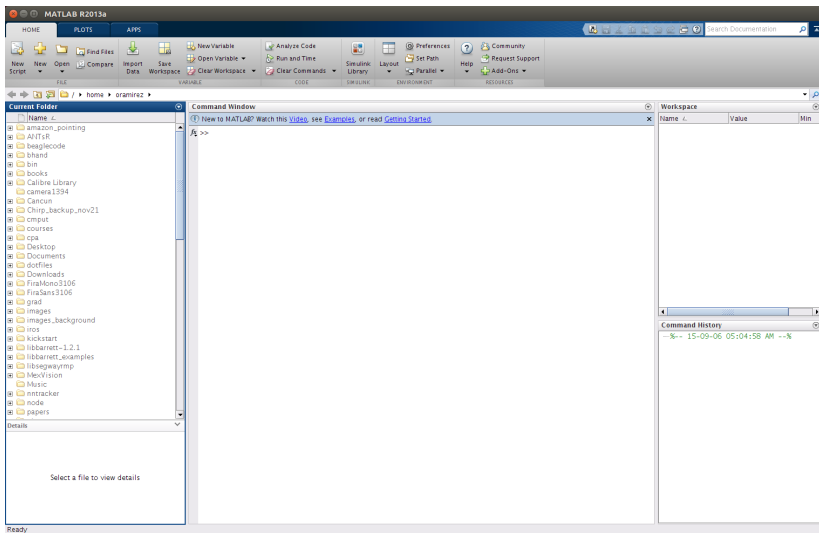


Matlab Introduction

06.09.2015

Oscar Ramirez

Matlab



Matlab

The `startup.m` file is a file you create to specify startup options. Use `startup.m` to modify the default search path, predefine variables in your workspace, or define defaults.

For example, the following statement adds the user-defined folder `/home/myname/mytools` to the search path.

```
addpath /home/myname/mytools
```

Note that it will only read the `startup.m` file for the folder where Matlab is initialized.

Matrices

$$A = \begin{bmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \end{bmatrix}$$

- A Matrix is a rectangular array of numbers.
- It's size is defined as the number of rows and columns that it contains. e.g. $A_{2 \times 3}$
- Elements are accessed by their row, column index. e.g.
 $A(2, 3) = a_{23}$

Matrices

```
>> A = rand(2, 3)
```

```
A =
```

```
    0.8147    0.1270    0.6324  
    0.9058    0.9134    0.0975
```

```
>> size(A)
```

```
ans =
```

```
     2     3
```

```
>> A(2,3)
```

```
ans =
```

```
    0.0975
```

Matrices

```
>> A(:,2)
```

```
ans =
```

```
    0.1270
```

```
    0.9134
```

```
>> A(2,:)
```

```
ans =
```

```
    0.9058
```

```
    0.9134
```

```
    0.0975
```

Matrix Addition

A and B are $m \times n$ matrices, $A + B$ is an $m \times n$ matrix defined as:

$$A + B = A_{ij} + B_{ij}$$

Properties:

- Associative property: $(A + B) + C = A + (B + C)$
- Commutative property: $A + B = B + A$

Matrix Addition

```
>> A = rand(2,3)
```

```
A =
```

0.2785	0.9575	0.1576
0.5469	0.9649	0.9706

```
>> B = rand(2,3)
```

```
B =
```

0.9572	0.8003	0.4218
0.4854	0.1419	0.9157

```
>> A + B
```

```
ans =
```

1.2357	1.7578	0.5794
1.0323	1.1068	1.8863

Scalar Multiplication

Given an $m \times n$ matrix A and a scalar α , $\alpha A = A\alpha$ is an $m \times n$ matrix defined as:

$$(\alpha A)_{ij} = \alpha \times A_{ij}$$

Properties:

□ Associative: $(\alpha\beta)A = \alpha(\beta A)$

□ Distributive:

□ $\alpha(A + B) = \alpha A + \alpha B$

□ $(\alpha + \beta)A = \alpha A + \beta A$

```
>> A * 2
```

```
>> 2 * A
```

Matrix Multiplication

Given $A_{m \times p}$ and $B_{p \times n}$, AB is an $m \times n$ matrix defined as:

$$(AB)_{ij} = \sum_{k=1}^p A_{ik} B_{kj}$$

AB is defined only when the number of columns of A equals the number of rows of B

Matrix Multiplication

$$A_{(m \times p)} B_{(p \times n)} = AB_{(m \times n)}$$

Properties:

- **Matrix multiplication is not commutative**
- Distributive:
 - $A(B + C) = AB + AC$
 - $(D + E)F = DF + EF$
- Associative: $A(BC) = (AB)C$
- $(AB)^T = B^T A^T$

Matrix Multiplication

```
>> A = rand(2,3); % Note that semicolon will suppress  
>> B = rand(3,2); % output  
>> A * B          % '%' is used for comments  
ans =  
    1.3301    0.5995  
    0.8135    0.2635
```

Transpose

Given $A_{m \times n}$, its transpose A^T is an $n \times m$ matrix defined as:

$$(A^T)_{ij} = A_{ji}$$
$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}^T = \begin{bmatrix} 1 & 4 \\ 2 & 5 \\ 3 & 6 \end{bmatrix}$$

Properties:

- $(A^T)^T = A$
- $(A + B)^T = A^T + B^T$
- $(\alpha A)^T = \alpha A^T$

>> A'

Variables

To see a list of your variables use the `who` or `whos` commands.

```
>> who
```

Your variables are:

```
A      B      ans
```

```
>> whos
```

Name	Size	Bytes	Class	Attributes
A	2x3	48	double	
B	3x2	48	double	
ans	2x2	32	double	

Matrices

Some useful ways to create matrices

```
>> B = reshape([1:1:9], 3, 3);
```

```
>> a = [1; 4; 7];
```

```
>> b = [2; 5; 8];
```

```
>> c = [3; 6; 9];
```

```
>> A = [a, b, c]
```

A =

1	2	3
4	5	6
7	8	9

Saving State

- To save the current variables:

```
save filename
```

- To load variables from a file:

```
load filename
```

Scripts

Any `*.m` file in your path can be run in Matlab by simply typing its name.

You can add and remove directories to the path using the `addpath` and `rmpath` commands. e.g. :

```
>> addpath ~/CMPUT340
```

```
>> rmpath ~/CMPUT340
```

Variables created in script files will be kept in the current workspace.

Functions

Functions are declared in `.m` files that are named after the function you are creating. Multiple functions can be placed in one `.m` file, however only the one named the same as the file will be exposed.

```
function [r1, r2] = foo(p1, p2)
    r1 = p1 * p2;
    local_var = p1 + p2;
    r2 = local_var * 2
end
```

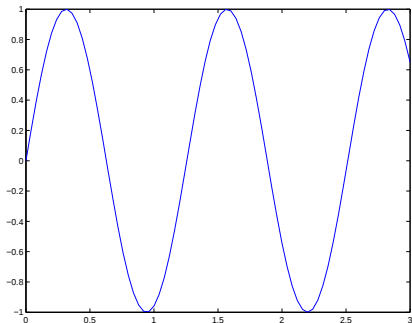
Variables created within the function do not persist after being called.

Control Statements: If

```
1 if a > 5
2     disp('hi')
3 else
4     disp('bye')
5 end;
```

Plots

```
1 n = 25;  
2 x = 0:1/n:3;  
3 y = sin(5*x);  
4 plot(x,y);  
5 print -depsc sin.eps;
```



Other useful commands

For information on any of these simply use: `help <command>`

☐ `clear`

☐ `clc`

☐ `ones`

☐ `eye`

☐ `zeros`

☐ `size`

☐ `plot3`

☐ `hold`

☐ `scatter`

P1: Exercise 1

```
1 % Vectorize the following
2 % Note the use of the tic/toc calls to time execution
3 % Compare the time once you have vectorized it
4
5 tic
6 for i = 1:1000
7     t(i) = 2*i;
8     y(i) = sin (t(i));
9 end
10 toc
11
12 clear;
```

References

- [1] PEKALSKA, E., VAN DER GLAS, M., SERLIE, I., AND VOSSEPOEL, A.
Introduction to MATLAB.
University of Twente, Department of Applied Mathematics, 2006.
- [2] SCHUURMANS, D.
CMPUT 340 Reference Material.
<https://eclass.srv.ualberta.ca/mod/page/view.php?id=1128238>, 2014.