Introduction to MATLAB

Ahmad Asadi

Amirkabir University of Technology
Department of Computer Engineering and Information Technology
ahmad.asadi@aut.ac.ir

Spring 2016

Outlines

Quick Look

What we will see

- What is MATLAB?
- Graphical User Interface of MATLAB
- Basic Syntax

What is MATLAB?

- A high level programming language being used for technical sophisticated computations
- Everything is matrix
- Stands for: **MAT**rix **LAB**oratory
- Can be assumed as a powerful super calculator
- Matrix based structure $\rightarrow$ awesome to do linear algebra

Note

Matlab is extremely broader than what we will cover in this course. We just want to understand its basics.

Look around MATLAB

Pros

- Fast and easy prototyping
- A wide variety of provided libraries including wide diversity of applications
- Great easy graphical display facilities
- Providing facilities to quickly make a little tiny application
- Quick to learn & efficient to use

Cons

- It seems slow for some sort of programs (we will see them later)
- A program that is just for personal usages (not available on web, not designed for large scale applications, not designed in a multi-user fashion, etc.)

Applications

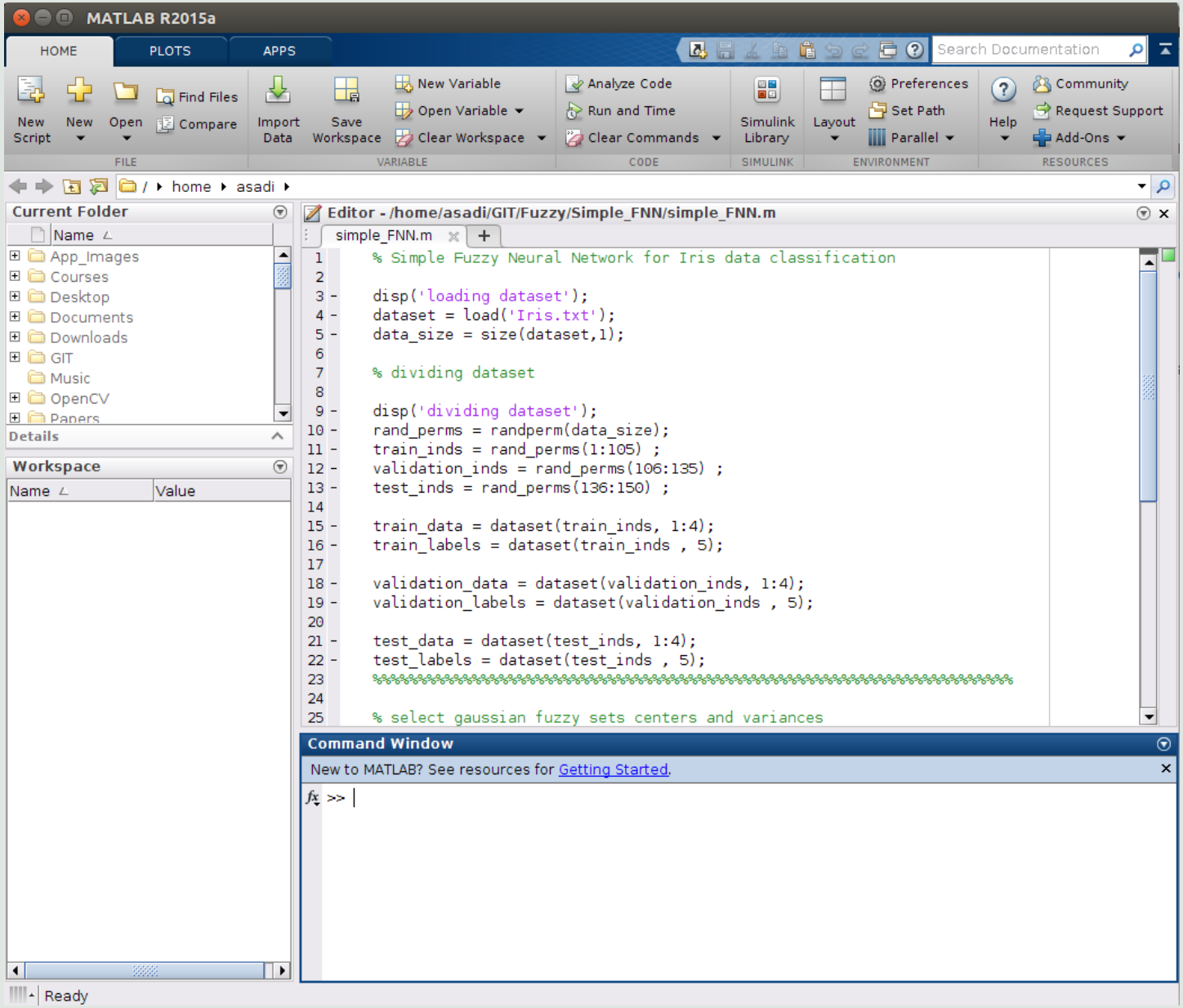
- Math and Computations
- Algorithm Development
- Modeling, Simulation and Prototyping
- Data Analysis, Exploration and Visualization
- Scientific and Engineering Graphics
- Optimized mining operations through modeling and simulation
- Automated data analysis, processing and reporting
- Forecast economical risk and profitability using financial predictive modeling
- Almost, one of the most useful handy applications for engineers and also scientists

How to work with MATLAB?

Big Picture

- Learn Rules (Syntax)
- Decompose interesting problem into simple steps
- Express each step according to MATLAB syntax
- Let MATLAB To do it!

Graphical User Interface (GUI)



Basics

Primitive data structures

Matrices & Vectors

- Almost the most primitive data structures in MATLAB $\rightarrow$ matrices
- Defined as bellow:

```
>> A = [1 2; 3 4]
A =   1 2
      3 4
```

- Separate rows by ';' and cols by ',' or ' '
- Vectors are special cases of matrices
  - **Row Vector** is an $N * 1$ matrix
  - **Column Vector** is a $1 * M$ matrix
- $size(A)$ returns dimensions of matrix A

Facilities in Creating Vectors

- Creating a vector with equally spaced intervals

```
>> A = 1:0.5:pi
A = 1.0000 1.5000 2.0000 2.5000 3.0000
```

- Creating a vector with *n* equally spaced intervals

```
>> A = linspace(0, pi, 7)
A = 0 0.5236 1.0472 1.5708 2.0944 2.6180 3.1416
```

Note

- MATLAB uses *pi* to represent π and *i* or *j* to represent imaginary unit

Matrices

There is still another useful slide!

There exist a list of useful functions being used to create matrices

- $zeros(m, n)$ creates an $m * n$ matrix of all zeros
- $ones(m, n)$ creates an $m * n$ matrix of all ones
- $eye(m, n)$ creates an $m * n$ identity matrix
- $rand(m, n)$ creates an $m * n$ uniformly distributed randoms
- $randn(m, n)$ creates an $m * n$ normally distributed randoms
- $magic(m)$ creates a square matrix with equal summation of rows, columns and diagonal
- $pascal(m)$ creates a square pascal matrix

Operations

Operations on vectors and matrices are divided into two groups

1. Matrix Operations
   Operands of these kind of operations are matrices as whole.
2. Array Operations
   Operands of these kind of operations are elements of matrices. These kind of operations are being applied to matrices, element by element.

Operations

Matrix Operations

- $+ \rightarrow$ summation
- $- \rightarrow$ subtraction
- $* \rightarrow$ multiplication
- $/ \rightarrow$ division
- $\backslash \rightarrow$ left division($A \backslash B = INV(A) * B$)
- ^$\rightarrow$ exponentiation

Array Operations

- $.' \rightarrow$ array transpose
- .^$\rightarrow$ array power
- $.* \rightarrow$ array multiplication
- $./ \rightarrow$ array division

Reading values of a particle matrix

- get value of cell on row 1, col 3 of matrix A

```
>> A(1,2)
```

- get value of cells on row 2, from col 2 to col 5 of matrix A

```
>> A(3,2:5)
```

- get value of cells from row 3 to row 6 on col 3 of matrix A

```
>> A(3:6,3)
```

- get value of cells from row 1 to row 3, from col 2 to row 4 of matrix A

```
>> A(1:3,2:4)
```

Reading values of a particle matrix

- get value of all cells on row 3 of matrix A

```
12 >> A(3,:)
```

- get value of all cells on col 2 of matrix A

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

- get value of all cells of matrix A

```
14 >> A(:,:)
```