MFE204TC ARTIFICIAL INTELLIGENCE AND DATA ANALYSIS

LECTURE 1

LONG HUANG

Xi'an Jiaotong-Liverpool University
西交利物浦大学

CONTENTS

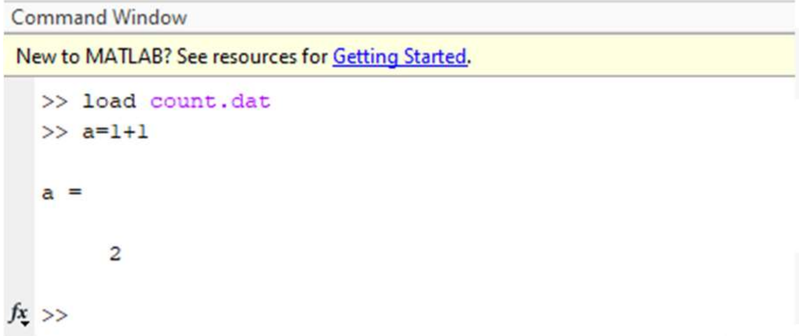
- Matlab Basics

- Data Processing
    - Importing/Exporting Data
    - Plotting Data
    - Missing Data

Certain contents of this presentation are adopted from training material at MathWorks

MATLAB BASICS – CODE EXECUTION

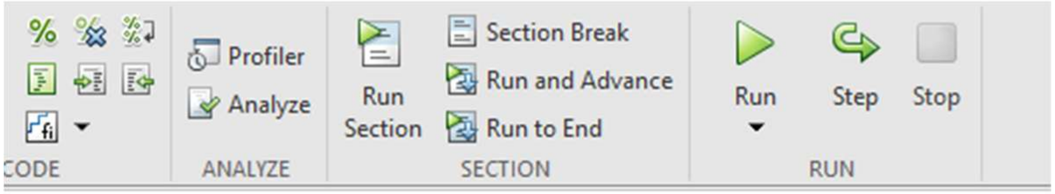
- Command line execution
  - Creating objects and modifications
    - a=1+1
  - Utility functions
    - Load count.dat

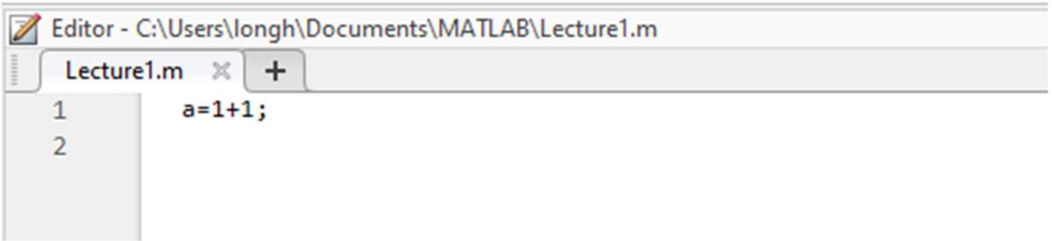


New to MATLAB? See resources for Getting Started.

```
>> load count.dat
>> a=1+1

a =

     2

>>
```

- Running program scripts (m-files)

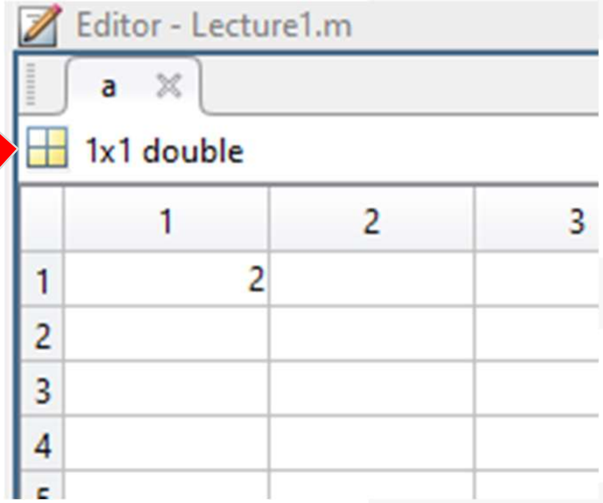




```
a=1+1;
```

BASIC RULES – MATLAB OBJECTIVES

- Objective is a matrix
  - Scalar: 1×1 matrix
  - Row vector: 1×n matrix
  - Column vector: n × 1 matrix



1x1 double

	1	2	3
1	2		
2			
3			
4			

- Matrix element
  - Has a numerical value
  - Non-digit elements: ASCII code digits
    - (i.e. A-> 65; a -> 97)
  - Logical values:
    - True ->1, false ->0

BASIC RULES – MATLAB OBJECTIVES

- Separators for column elements
  - Comma or Space
  - [1,2] -> 1 2
  - [3 4 5] -> 3 4 5

- Separators for row elements
  - Semicolon or Return

  -
    ```
    [6;7]  -> 6
              7
    ```

  -
    ```
    [8          8
     9      ->  9
     10]        10
    ```

BASIC RULES – MATLAB NOTATIONS

- Boolean variables
  - Truth Values 1 for true; 0 for false

- Primary logic symbols
  - & logical AND
  - = = logical equal
  - | logical inclusive OR
  - ~ logical NOT

- Compound logic symbols
    - <=   >=   ~=

BASIC RULES – MATLAB PROGRAMMING

- Computation
  - Combinations of elementary operations (as with most programing languages)

- Built-in constants
  - Scalars: pi (numerical π)
  - Matrices: zeros(m,n); ones(m,n); eye(m,n)

- Built-in functions
- • Numerical: sin ; cos; exp; log
- • Logical: isprime; isreal

BASIC RULES – MATLAB PROGRAMMING

- Elementary flow control in scripts

- Conditional
  - if .. elseif .. [elseif .. elseif ..] else

- Loop
  - for

- Conditional Loop
  - while

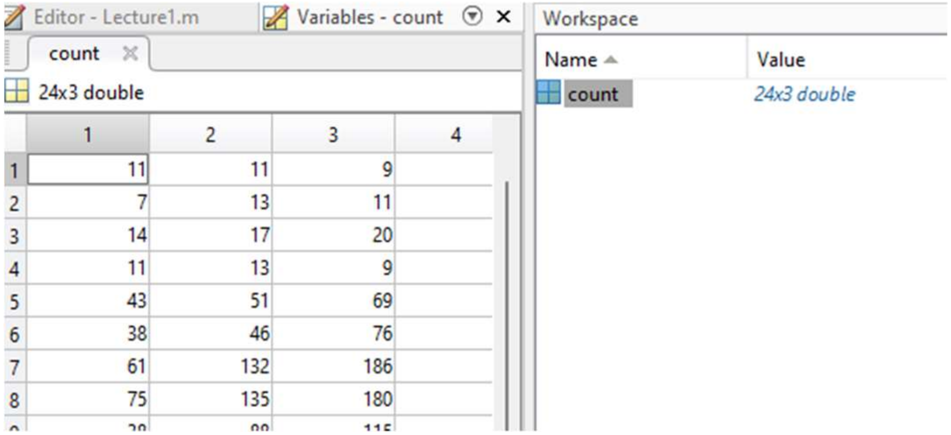
MATLAB BASICS

- If you are not familiar with the above-mentioned basics on Matlab
  - Documentation on Mathworks website
  - Introduction to Matlab, by NorthWesternU is uploaded to LMO. A good resource for students
  - Raise your request with the module teacher and TAs to seek additional support.

IMPORTING/EXPORTING DATA

- Try the following

```
load count.dat
```



	1	2	3	4
1	11	11	9	
2	7	13	11	
3	14	17	20	
4	11	13	9	
5	43	51	69	
6	38	46	76	
7	61	132	186	
8	75	135	180	

Name	Value
count	24x3 double

- Question: what are the supported formats for import and export?

  https://www.mathworks.com/help/matlab/import_export/supported-file-formats-for-import-and-export.html

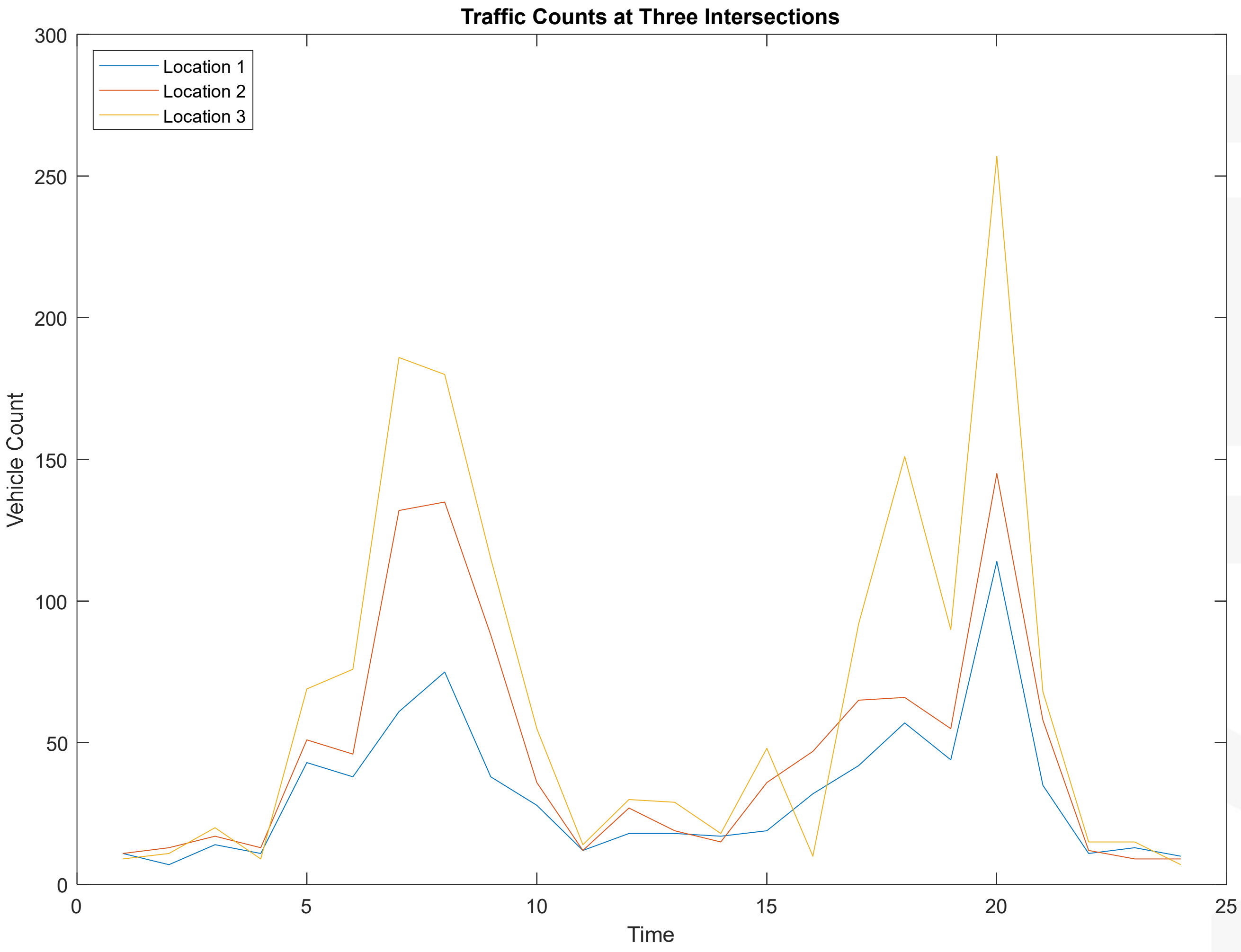
PLOTTING DATA

- Try the following

```
load count.dat
[n,p] = size(count)
t = 1:n;
plot(t,count),
legend('Location 1','Location 2','Location 3','Location','NorthWest')
xlabel('Time'), ylabel('Vehicle Count')
title('Traffic Counts at Three Intersections')
```

PLOTTING DATA

- Any other forms of plot?



REPRESENTING MISSING DATA

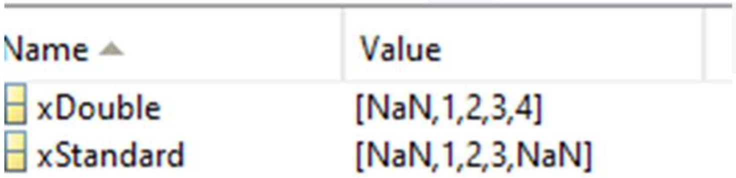
- **Missing values they often represent unreliable or unusable data points – How do we represent these?**
  - Numeric data types such as double use NaN (not a number) to represent missing values.
  - You can also use the *missing* value to represent missing numeric data or data of other typesMATLAB automatically converts the missing value to the data's native type.
  - Try the following

```
x = [NaN 1 2 3 4];
xDouble = [missing 1 2 3 4]
xDatetime = [missing datetime(2014,1:4,1)]
xString = [missing "a" "b" "c" "d"]
xCategorical = [missing categorical({'cat1' 'cat2' 'cat3' 'cat4'})]
```

TREATING MISSING DATA

- You can use the *standardizeMissing* function to convert unwanted values to the standard missing value for that data type.
  - Try the following to treat 4 as a missing double value in addition to NaN.

```
xDouble = [missing 1 2 3 4]
xStandard = standardizeMissing(xDouble,[4 NaN])
```



Name	Value
xDouble	[NaN,1,2,3,4]
xStandard	[NaN,1,2,3,NaN]

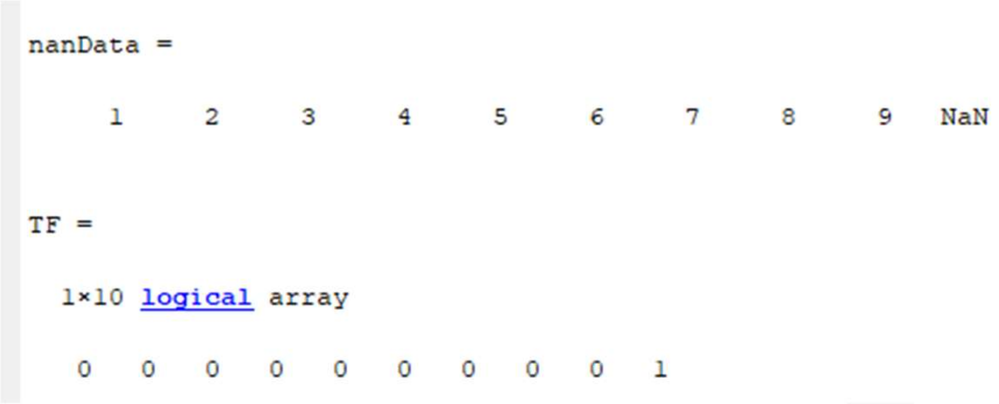
  - Use *MissingPlacement* with sort function to place NaNs to the end.

```
xSort = sort(xStandard,'MissingPlacement','last')
```

FIND MISSING DATA

- Missing data may lead to failed calculations (i.e. mean)
- *isnan* function locates any Nan value by returning a logical array

```
nanData = [1:9 NaN];
TF = isnan(nanData)
```



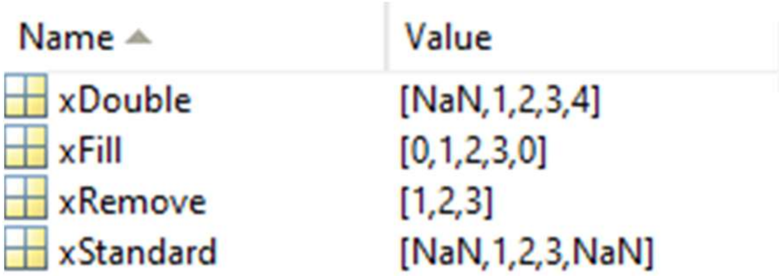
```
nanData =

     1     2     3     4     5     6     7     8     9   NaN


TF =

  1×10 logical array

   0   0   0   0   0   0   0   0   0   1
```

- *ismissing* function returns the location of missing values in data for multiple data types.

REPLACE MISSING DATA

- Use fillmissing to replace missing values with another value,
- Use rmmissing to remove missing values altogether.

```
xDouble = [missing 1 2 3 4]
xStandard = standardizeMissing(xDouble,[4 NaN])
xFill = fillmissing(xStandard,'constant',0)
xRemove = rmmissing(xStandard)
```



Name ▲	Value
xDouble	[NaN,1,2,3,4]
xFill	[0,1,2,3,0]
xRemove	[1,2,3]
xStandard	[NaN,1,2,3,NaN]