

# MATLAB FUNDAMENTALS AND PROGRAMMING TECHNIQUES

(BASIC)

*Tim Yeh*

*[support@terasoft.com.tw](mailto:support@terasoft.com.tw)*

# Course Outline

- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
- Working with Data Files
- Visualization with Array
- Automating Commands with Scripts
- Appendix: Data Type



# How to Use This Manual

- Code font is used for code, function names and URLs. It also occurs on the slides in one of the lower corners as reference to relevant files or commands for the example.

```
>> command_line_code  
>>[a,b,c] = command_in_file(d,f);
```

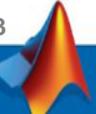
- At times, code may run off the line. The line continuation character (...) is used to show this. These are valid MATLAB statements if typed as shown, including carriage returns.

```
>> [CFlowAmounts, CFlowDates, TFactors] = ...  
cfamounts(couponRate, settle, maturity);
```

- Menu items, options and key names are highlighted in bold in the notes sections.  
Use **Ctrl+C** to break out of execution.

Click on **File, Set Path ...** to open the Path Browser.

**>> try this at the prompt**

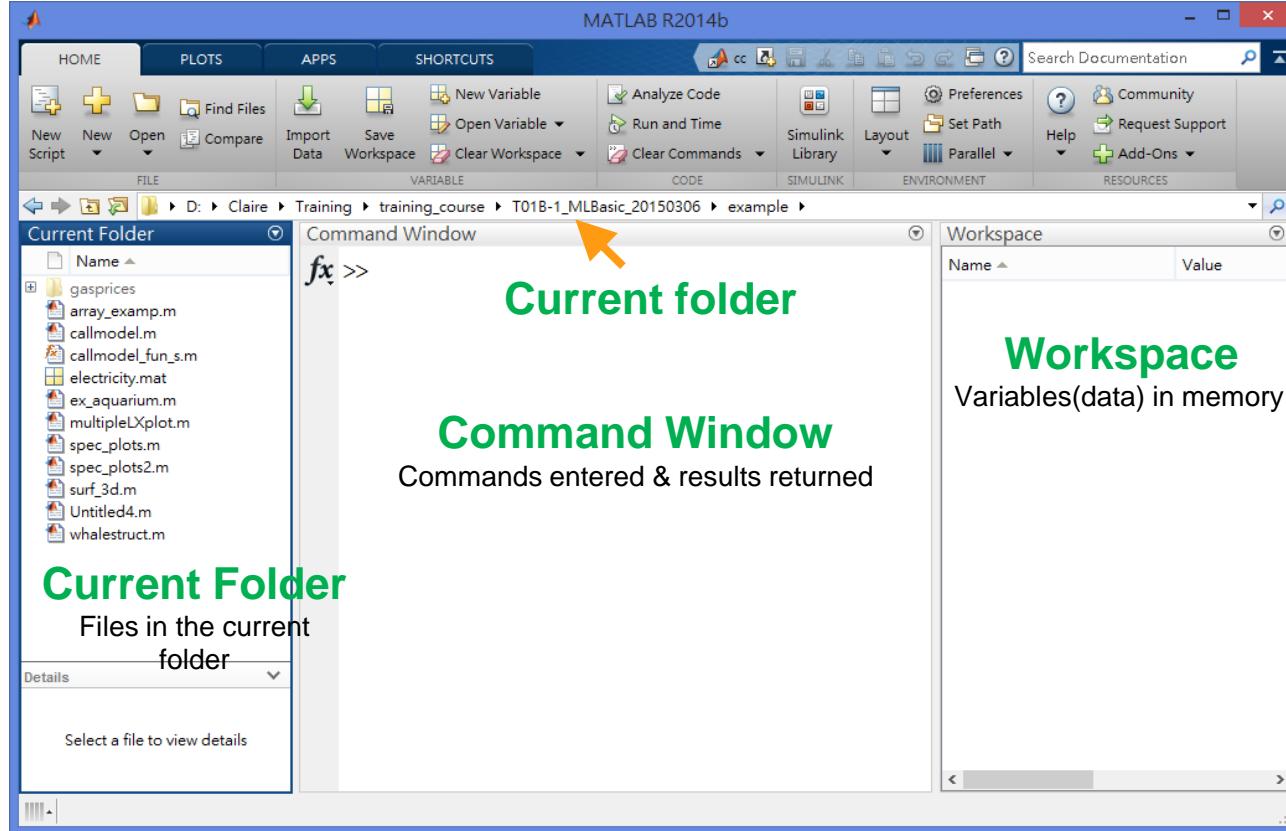


# Course Outline

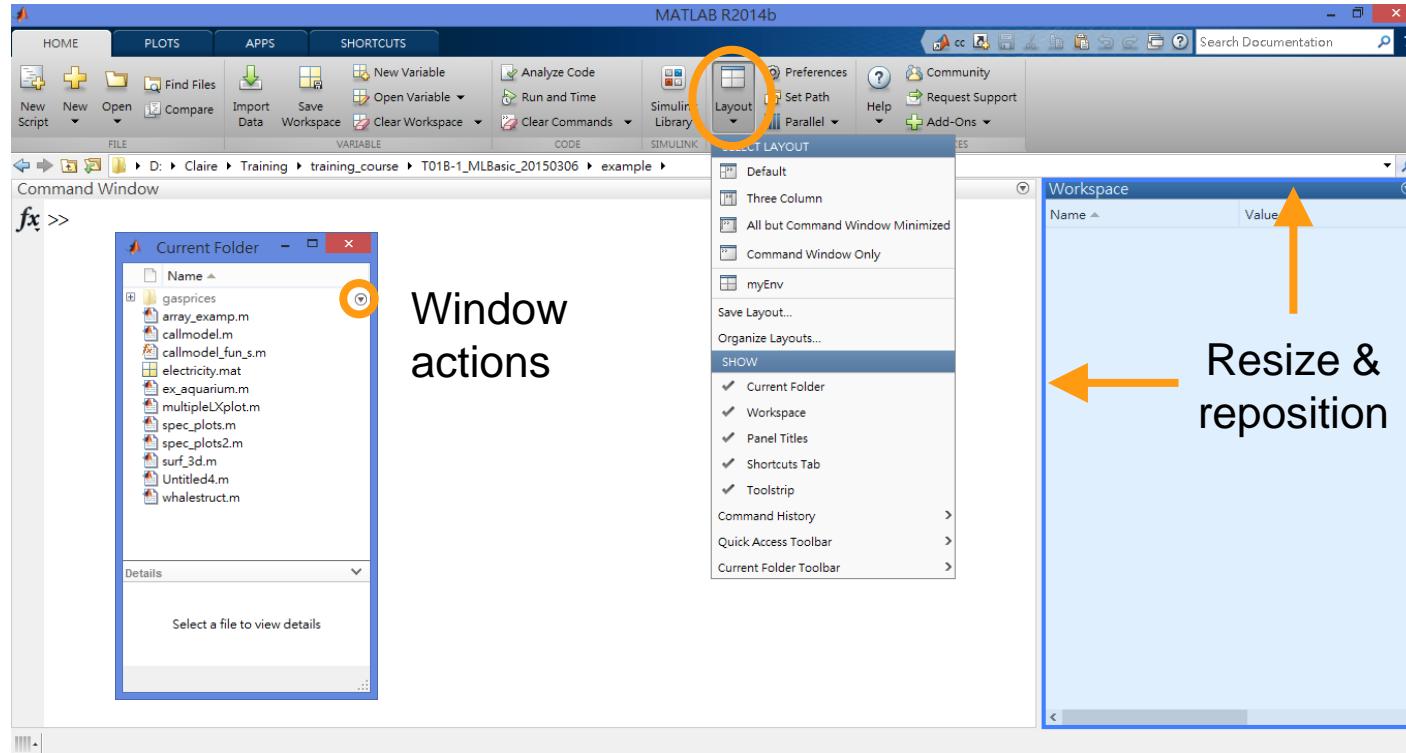
- Working with the MATLAB User Interface
  - Variables and Commands
  - Array Creation and Analysis
  - Working with Data Files
  - Visualization with Array
  - Automating Commands with Scripts
  - Appendix: Data Type



# The MATLAB® Desktop



# Customizing the Desktop



# Course Example: Gas Price Data

The image shows two windows side-by-side. On the left is the MATLAB 'Current Folder' browser. It lists several files: exchange.mat, exchange.txt, gasprices.csv (which is selected and highlighted in blue), gasprices.mat, gasprices.txt, gasprices.xls, getgas.m, gpbinaryio.m, gpfileio.m, gpfor.m, gpif.m, gpinout.m, gpinput.m, and gpwhile.m. Below the list, it says 'gasprices.csv (Microsoft Excel Comma Separated Va' and 'No details available'. An orange arrow points from the 'gasprices.csv' entry in the MATLAB browser to the corresponding file in the Microsoft Excel window on the right. The Microsoft Excel window is titled 'gasprices.csv - Microsoft Excel'. The spreadsheet contains data for 'Average Annual Gasoline (Petrol) Retail Prices in Selected Countries [\$US per gallon]'. The columns are labeled Australia, Canada, France, Germany, Italy, Japan, Mexico, South Korea, UK, and USA. The rows represent years from 1990 to 2008. The entire data range is highlighted with a red border.

Year	Australia	Canada	France	Germany	Italy	Japan	Mexico	South Korea	UK	USA
1990	1.67	3.03	2.03	4.35	3.10	1	2.03	2.82	1.10	
1991	1.96	1.92	3.45	2.9	4.5	3.46	1.3	2.49	3.01	1.14
1992	1.89	1.73	3.56	3.27	4.53	3.58	1.5	2.65	3.06	1.13
1993	1.73	1.57	3.41	3.07	3.68	4.16	1.56	2.88	2.84	1.11
1994	1.84	1.45	3.59	3.52	3.7	4.36	1.48	2.87	2.99	1.11
1995	1.95	1.53	4.26	3.96	4	4.43	1.11	2.94	3.21	1.15
1996	2.12	1.61	4.41	3.94	4.39	3.64	1.25	3.18	3.34	1.23
1997	2.05	1.62	4	3.53	4.07	3.26	1.47	3.34	3.83	1.23
1998	1.63	1.38	3.87	3.34	3.84	2.82	1.49	3.04	4.06	1.06
1999	1.72	1.52	3.85	3.42	3.87	3.27	1.79	3.8	4.29	1.17
2000	1.94	1.86	3.8	3.45	3.77	3.65	2.01	4.18	4.58	1.51
2001	1.71	1.72	3.51	3.4	3.57	3.27	2.2	3.76	4.13	1.46
2002	1.76	1.69	3.62	3.67	3.74	3.15	2.24	3.84	4.16	1.36
2003	2.19	1.99	4.35	4.59	4.53	3.47	2.04	4.11	4.7	1.59
2004	2.72	2.37	4.99	5.24	5.29	3.93	2.03	4.51	5.56	1.88
2005	3.23	2.89	5.46	5.66	5.74	4.28	2.22	5.28	5.97	2.3
2006	3.54	3.26	5.88	6.03	6.1	4.47	2.31	5.92	6.36	2.59
2007	3.85	3.59	6.6	6.88	6.73	4.49	2.4	6.21	7.13	2.8
2008	4.45	4.08	7.51	7.75	7.63	5.74	2.45	5.83	7.42	3.27

# Interactively Importing

The screenshot illustrates the MATLAB interface for importing data. On the left, the 'Current Folder' browser shows a file named 'gasprices.csv'. A context menu is open over this file, with the 'Import Data...' option highlighted by a red oval. Below the browser, two icons represent Microsoft Excel and a text editor. In the center, the 'Import' dialog box is open, showing the 'gasprices.xls' file. The 'Range' is set to 'A6:K24', 'Variable Names Row' is '5', and 'Selection' is 'Cell Array'. The 'IMPORTED DATA' table displays the first few rows of the gas prices dataset. On the right, the 'Workspace' browser shows variables for each country ('Australia', 'Canada', 'France', 'Germany', 'Italy', 'Japan', 'Mexico', 'SouthKorea', 'UK', 'USA') and the 'Year' variable, all of which are 19x1 double arrays. An orange arrow points from the 'Import Selection' button in the 'Import' dialog to the 'Workspace' browser.

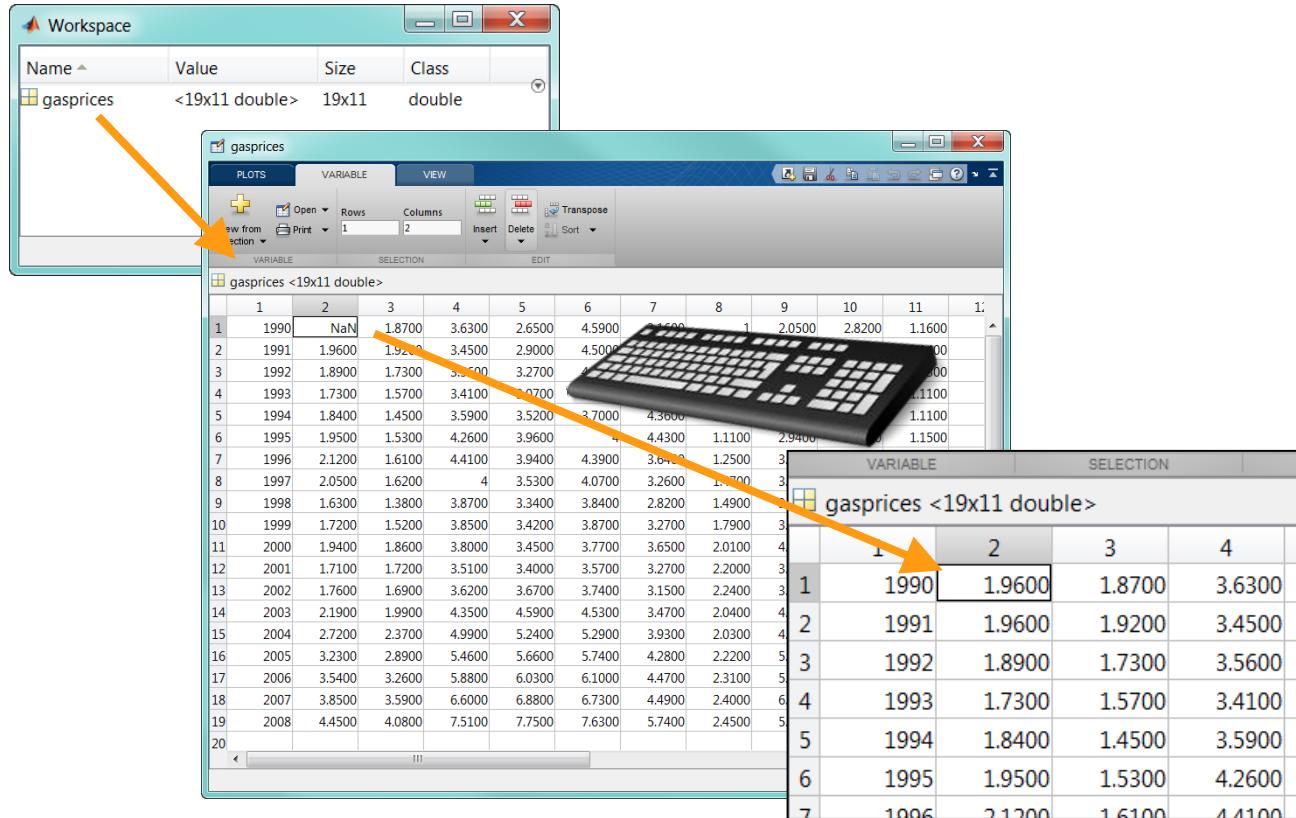
Year	Australia	Canada	France	Germany	Italy	Japan	Mexico	SouthKorea	UK	USA	
1	Average ...										
2	Source: ...										
3	http://...										
4											
5	Year	Australia	Canada	France	Germany	Italy	Japan	Mexico	SouthKorea	UK	USA
6	1990	NaN	1.8700	3.6300	2.6500	4.5900	3.1600	1	2.0500	2.8200	
7	1991	1.9600	1.9200	3.4500	2.9000	4.5000	3.4600	1.3000	2.4900	3.0100	
8	1992	1.8900	1.7300	3.5600	3.2700	4.5300	3.5800	1.5000	2.6500	3.0600	
9	1993	1.7300	1.5700	3.4100	3.0700	3.6800	4.1600	1.5600	2.8800	2.8400	
10	1994	1.8400	1.4500	3.5900	3.5200	3.7000	4.3600	1.4800	2.8700	2.9900	
11	1995	1.9500	1.5300	4.2600	3.9600	4	4.4300	1.1100	2.9400	3.2100	
12	1996	2.1200	1.6100	4.4100	3.9400	4.3900	3.6400	1.2500	3.1800	3.3400	
13	1997	2.0500	1.6200	4	3.5300	4.0700	3.2600	1.4700	3.3400	3.8300	
14	1998	1.6300	1.3800	3.8700	3.3400	3.8400	2.8200	1.4900	3.0400	4.0600	
15	1999	1.7200	1.5200	3.8500	3.4200	3.8700	3.2700	1.7900	3.8000	4.2900	
16	2000	1.9400	1.8600	3.8000	3.4500	3.7700	3.6500	2.0100	4.1800	4.5800	
17	2001	1.7100	1.7200	3.5100	3.4000	3.5700	3.2700	2.2000	3.7600	4.1300	
18	2002	1.7600	1.6900	3.6200	3.6700	3.7400	3.1500	2.2400	3.8400	4.1600	
19	2003	2.1900	1.9900	4.3500	4.5900	4.5300	3.4700	2.0400	4.1100	4.7000	
20	2004	2.7200	2.3700	4.9900	5.2400	5.2900	3.9300	2.0300	4.5100	5.5600	
21	2005	3.2300	2.8900	5.4600	5.6600	5.7400	4.2800	2.2200	5.2800	5.9700	
22	2006	3.5400	3.2600	5.8800	6.0300	6.1000	4.4700	2.3100	5.9200	6.3600	
23	2007	3.8500	3.5900	6.6000	6.8800	6.7300	4.4900	2.4000	6.2100	7.1300	
24	2008	4.4500	4.0800	7.5100	7.7500	7.6300	5.7400	2.4500	5.8300	7.4200	

# Variables in the Base Workspace

The screenshot shows the MATLAB workspace browser window. The variable 'gasprices' is listed with a value of ' $<19 \times 11$  double>', a size of '19x11', and a class of 'double'. A callout menu is open, showing options for Name, Value, Size, Bytes, Class, Min, Max, Range, Mean, Median, Mode, Var, and Std. The 'Class' option is selected. Below the browser, a table displays 19 rows of data, each consisting of 11 numerical values. An orange arrow points from the 'Size' column in the browser to the first row of the table. Another orange arrow points from the 'Class' column in the browser to the same row. The number '19' is written vertically next to the first row of the table, and the number '11' is written horizontally below it.

1990	NaN	1.87	3.63	2.65	4.59	3.16	1	2.05	2.82	1.16
1991	1.96	1.92	3.45	2.9	4.5	3.46	1.3	2.49	3.01	1.14
1992	1.89	1.73	3.56	3.27	4.53	3.58	1.5	2.65	3.06	1.13
1993	1.73	1.57	3.41	3.07	3.68	4.16	1.56	2.88	2.84	1.11
1994	1.84	1.45	3.59	3.52	3.7	4.36	1.48	2.87	2.99	1.11
1995	1.95	1.53	4.26	3.96	4	4.43	1.11	2.94	3.21	1.15
1996	2.12	1.61	4.41	3.94	4.39	3.64	1.25	3.18	3.34	1.23
1997	2.05	1.62	4	3.53	4.07	3.26	1.47	3.34	3.83	1.23
1998	1.63	1.38	3.87	3.34	3.84	2.82	1.49	3.04	4.06	1.06
1999	1.72	1.52	3.85	3.42	3.87	3.27	1.79	3.8	4.29	1.17
2000	1.94	1.86	3.8	3.45	3.77	3.65	2.01	4.18	4.58	1.07
2001	1.71	1.72	3.51	3.4	3.57	3.27	2.2	3.76	4.13	1.07
2002	1.76	1.69	3.62	3.67	3.74	3.15	2.24	3.84	4.16	1.07
2003	2.19	1.99	4.35	4.59	4.53	3.47	2.04	4.11	4.7	1.59
2004	2.72	2.37	4.99	5.24	5.29	3.93	2.03	4.51	5.56	1.68
2005	3.23	2.89	5.46	5.66	5.74	4.28	2.22	5.28	5.97	2.3
2006	3.54	3.26	5.88	6.03	6.1	4.47	2.31	5.92	6.36	2.59
2007	3.85	3.59	6.6	6.88	6.73	4.49	2.4	6.21	7.13	2.8
2008	4.45	4.08	7.51	7.75	7.63	5.74	2.45	5.83	7.42	3.27

# The Variable Editor



# New Variables

The screenshot shows the MATLAB environment with three windows:

- Variable Editor (Left):** Shows a table titled "gasprices" with 20 rows (Year) and 11 columns (Price). The first few rows of data are:

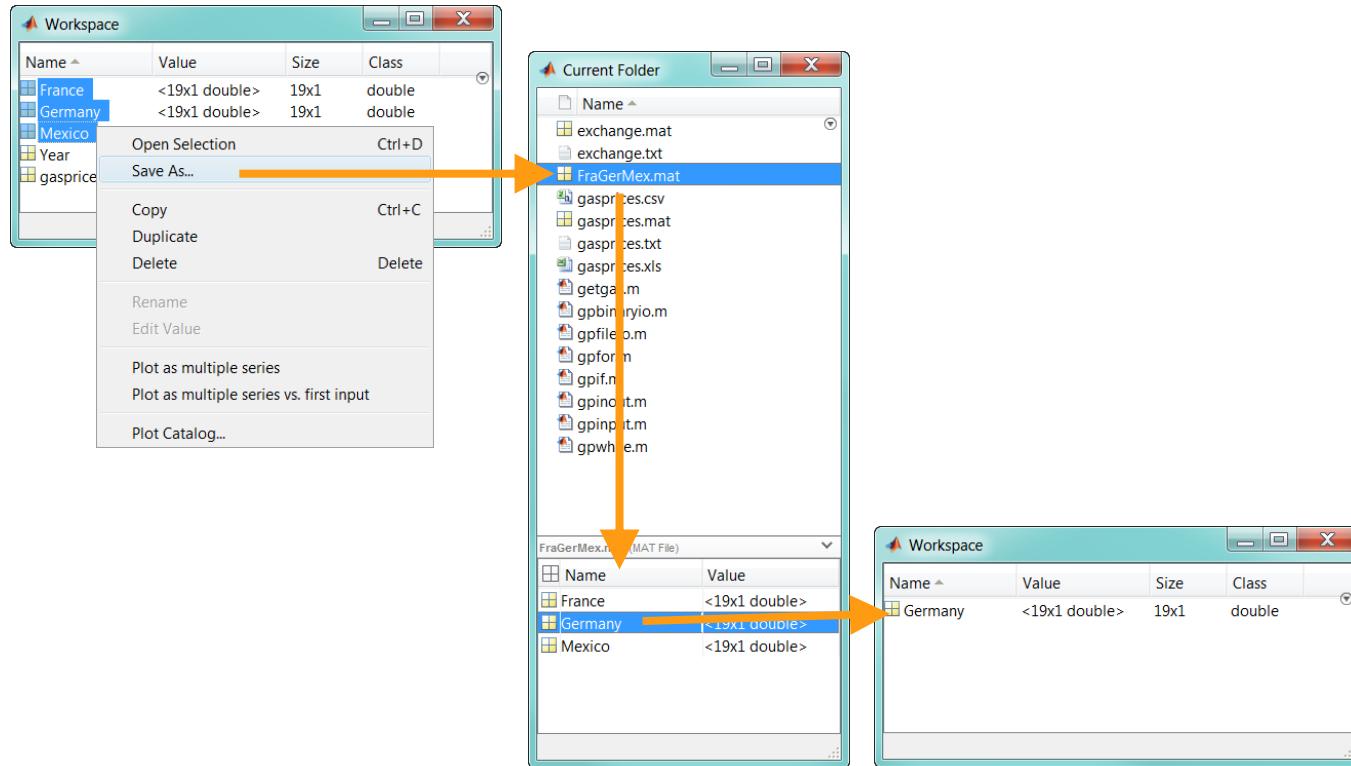
	1	2	3	4	5	6	7	8	9	10
1	1990	1.9600	1.8700	3.6300	2.6500	4.5900	3.1600	1	2.0500	2.8400
2	1991	1.9600	1.9200	3.4500	2.9000	4.5000	3.4600	1.3000	2.4900	2.9000
3	1992	1.8900	1.7300	3.5600	3.2700	4.5300	3.5800	1.5000	2.6500	2.9000
4	1993	1.7300	1.5700	3.4100	3.0700	3.6800	4.1600	1.5600	2.8800	2.8400

- Workspace Browser (Top Right):** Shows the current workspace with two variables:

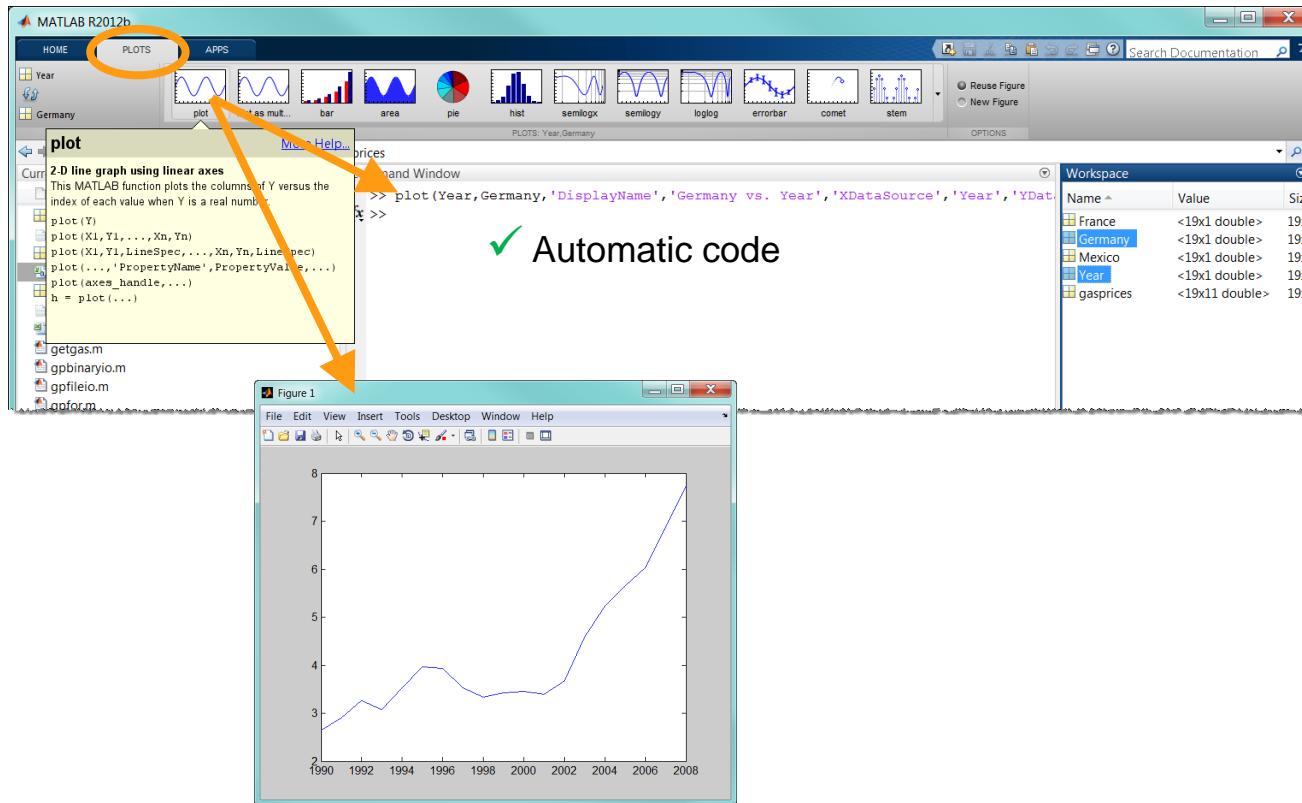
Name	Value	Size	Class
gasprices	<19x11 double>	19x11	double
gasprices1	<19x1 double>	19x1	double

A context menu is open over the "gasprices1" entry, with "Rename" highlighted.
- Workspace Catalog (Bottom Right):** Shows a list of available plots:
  - Open Selection
  - Save As...
  - Copy
  - Duplicate
  - Delete
  - Rename
  - Edit Value
  - plot(gasprices1)
  - bar(gasprices1)
  - area(gasprices1)
  - pie(gasprices1)
  - hist(gasprices1)
  - Plot Catalog...

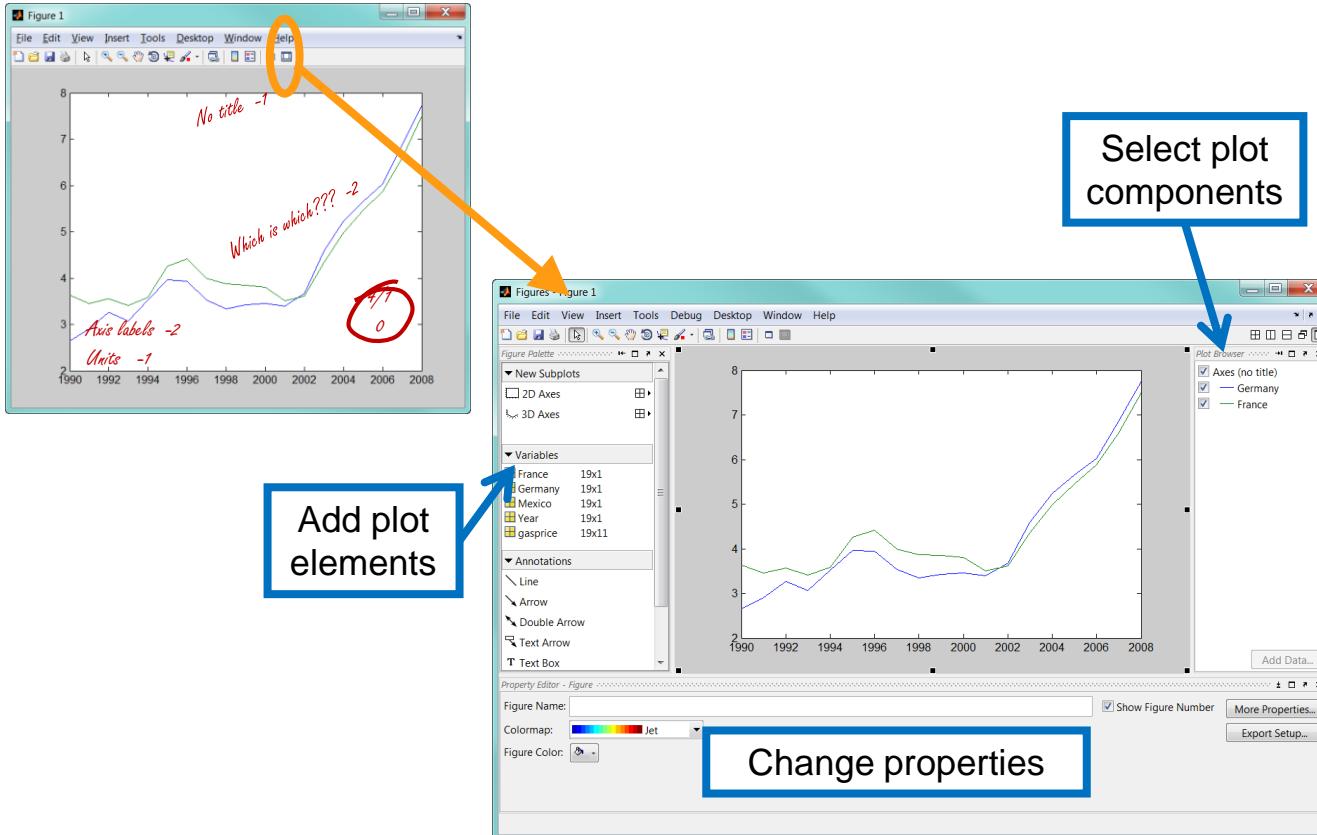
# Saving and Loading Variables



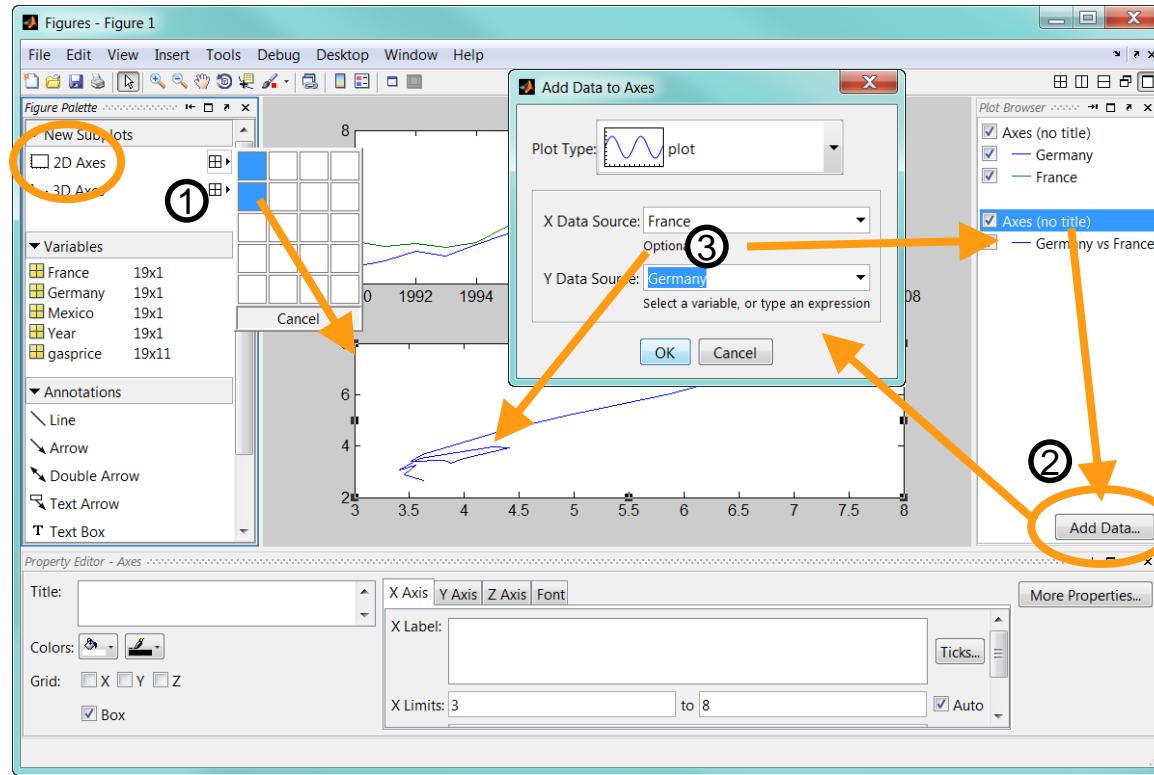
# Plotting the Data



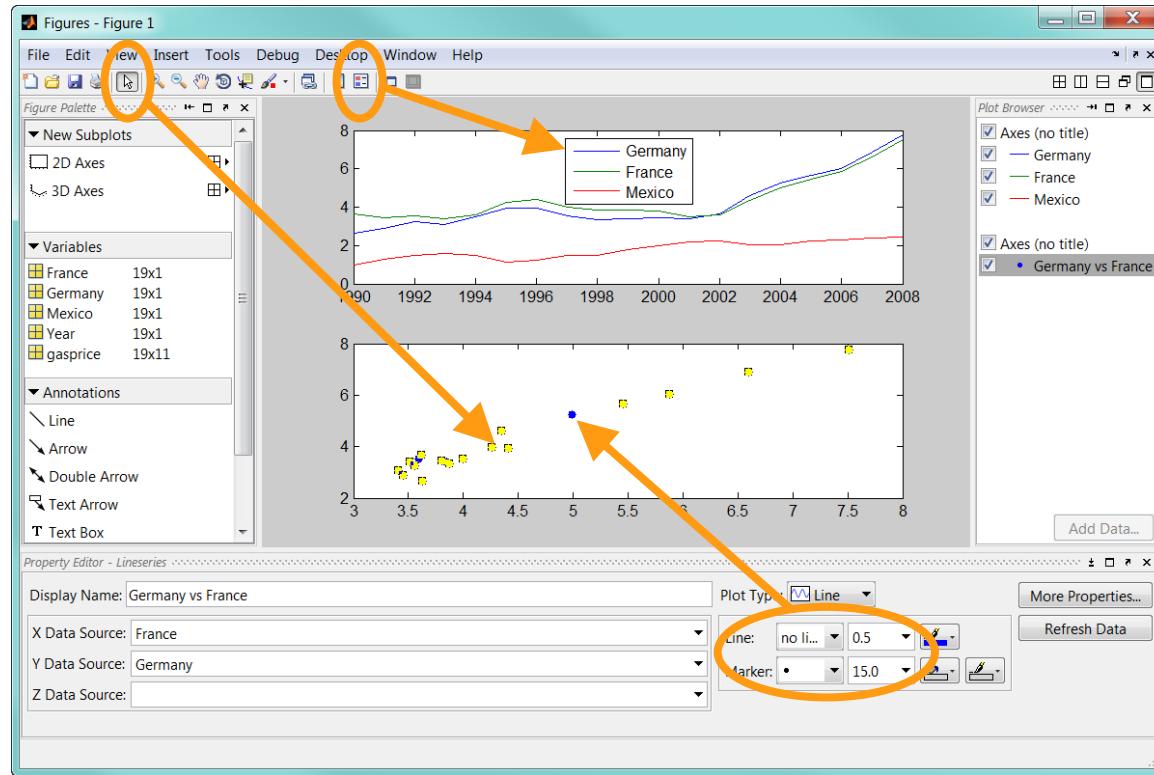
# Plot Tools



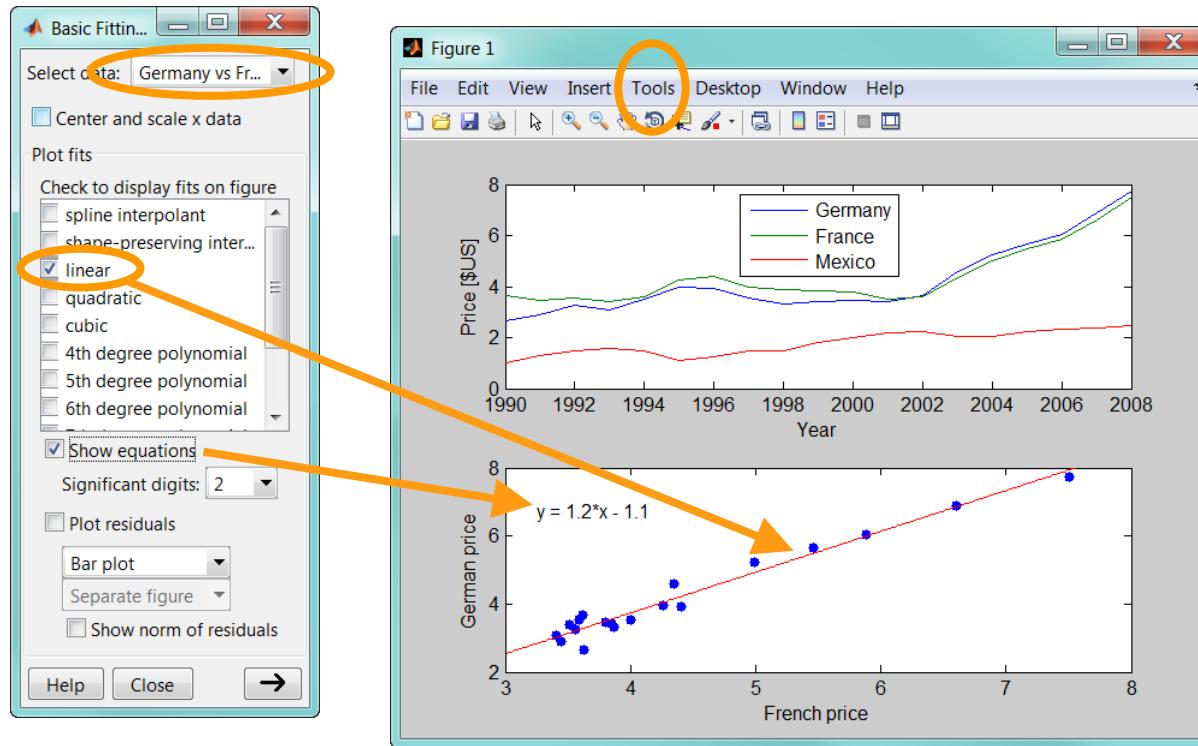
# Multiple Plots



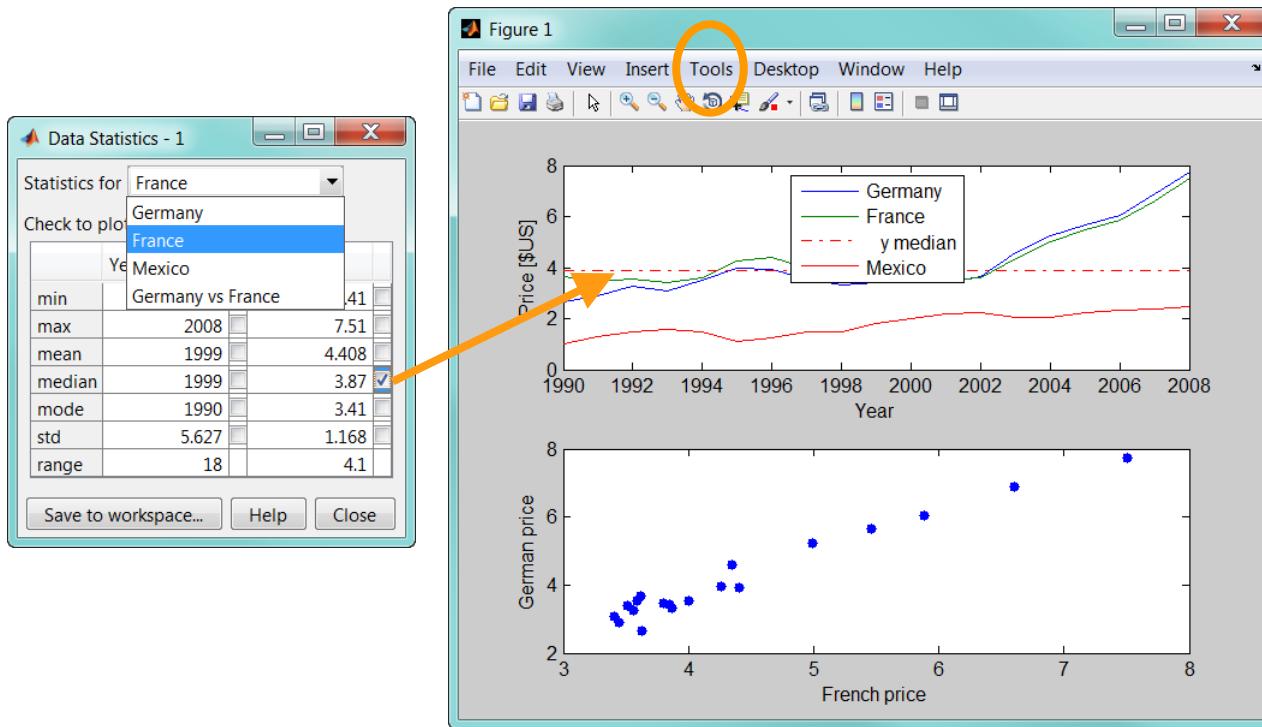
# Formatting the Plot



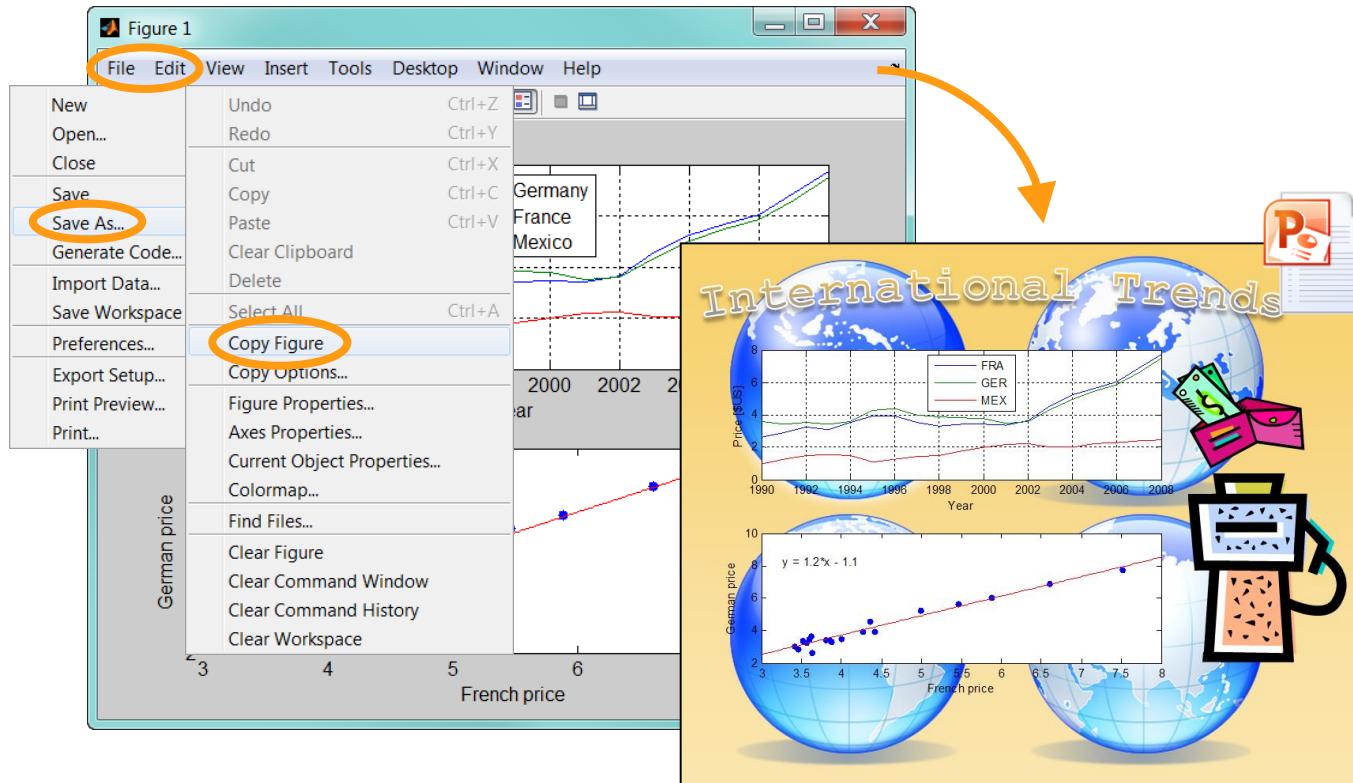
# Basic Fitting Tool



# Data Statistics Tool

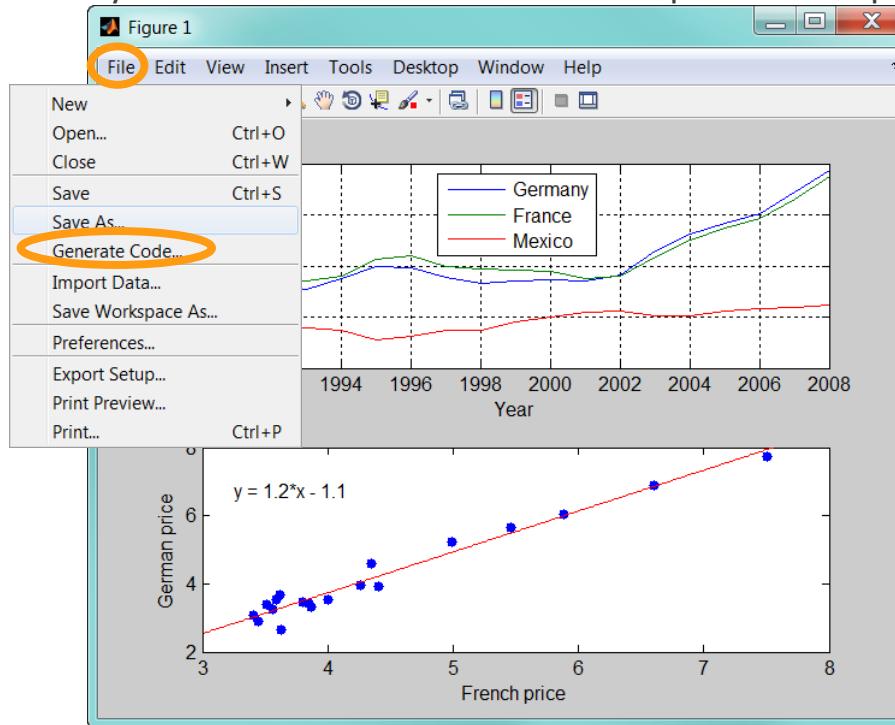


# Exporting to Another Application

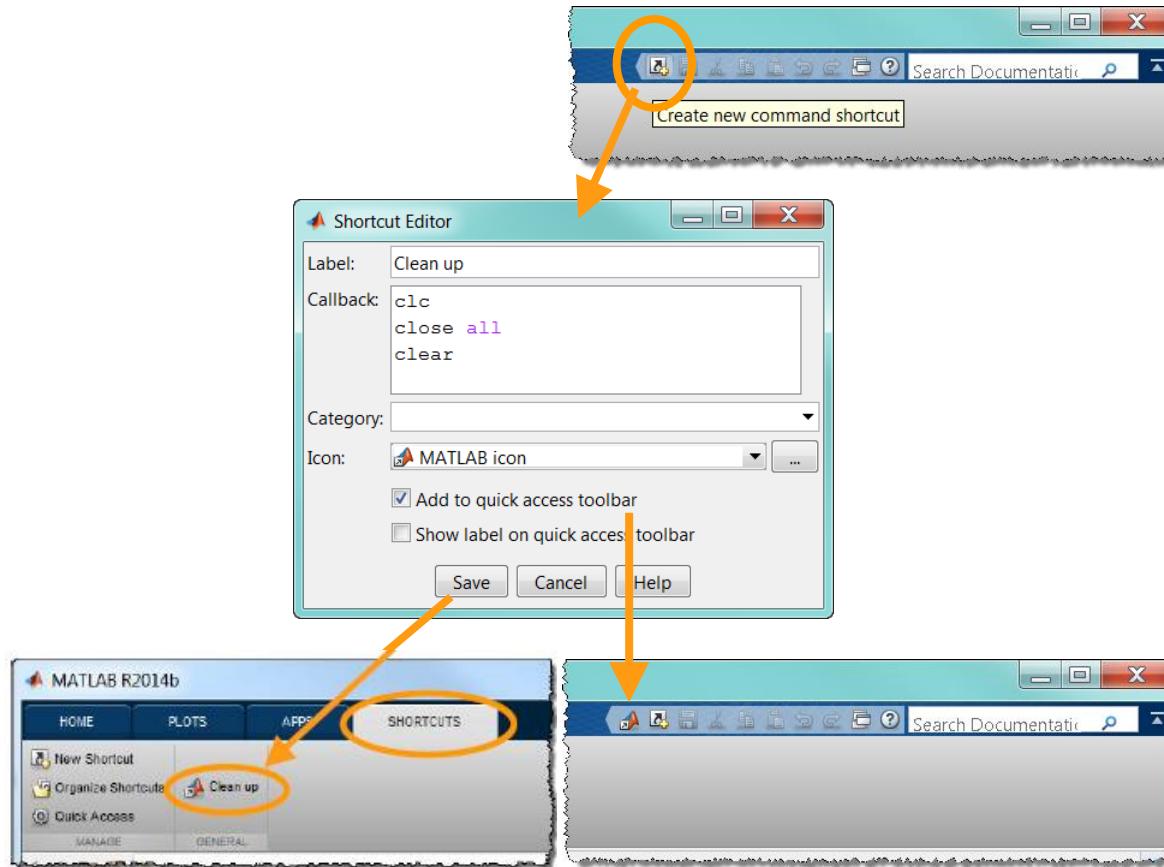


# Generate M-file

- Automatically Create MATLAB Commands to Reproduce Graphics



# Shortcuts



# Help and Documentation

The diagram illustrates the MATLAB Help and Documentation interface. On the left, there are two icons: a magnifying glass labeled "search" and an open book labeled "browse". Orange arrows point from these icons to the corresponding sections in the Help window. The Help window shows the documentation for the `ones` function. It includes:

- Syntax:**

```
Y = ones(n)
Y = ones(m,n)
Y = ones([m n])
Y = ones(n,n,p,...)
Y = ones([n n p ...])
Y = ones(size(A))
Y = ones(...,classname)
ones([m,n,...],classname)
```
- Description:**

`Y = ones(n)` returns an  $n$ -by- $n$  matrix of 1s. An error message appears if  $n$  is not a scalar.

`Y = ones(m,n)` or `Y = ones([m n])` returns an  $m$ -by- $n$  matrix of ones.

`Y = ones(m,n,p,...)` or `Y = ones([m n p ...])` returns an  $m$ -by- $n$ -by- $p$ -by-... array of 1s.
- Note:** The size inputs  $m$ ,  $n$ ,  $p$ , ... should be nonnegative integers. Negative integers are treated as 0.
- Examples:**

```
x = ones(2,3,'int8');
```
- See Also:**

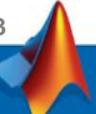
[complex](#) | [eye](#) | [true](#) | [zeros](#)

help  
doc  
docsearch



# Chapter 1 Test Your Knowledge

1. Where does MATLAB display a listing of the variables currently in memory and their associated attributes?
  - A. Command Window
  - B. Workspace browser
  - C. Current Directory browser
  - D. Command History
  
2. The default MATLAB variable type is:
  - A. Single
  - B. Double
  - C. Cell



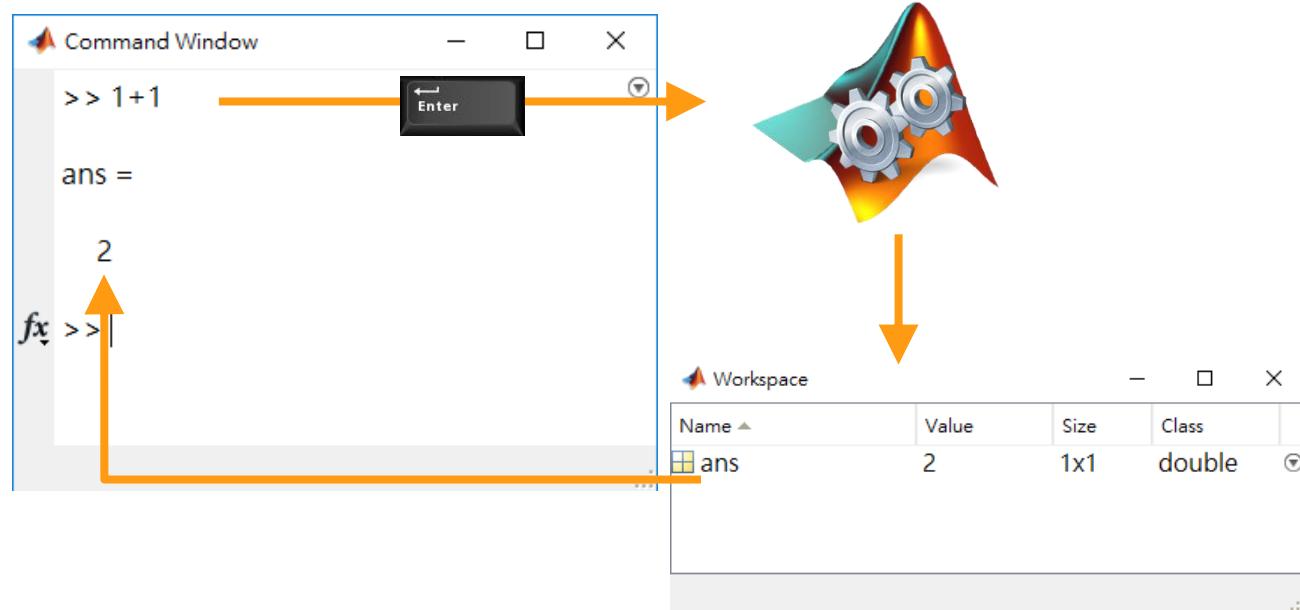
# Course Outline

- Working with the MATLAB User Interface
- Variables and Commands
  - Array Creation and Analysis
  - Working with Data Files
  - Visualization with Array
  - Automating Commands with Scripts
  - Appendix: Data Type



# High-level Calculator

Operators: + - \* / ^



# MATLAB® Commands

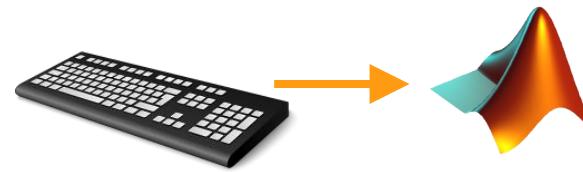


Diagram illustrating the MATLAB environment showing the Command Window and Workspace.

**Command Window:**

```
>> x = 42;
>> y = x*sin(x);
>> z = x - y
z =
    80.4939
fx >> |
```

The Command Window shows the execution of three commands: assignment of x to 42, calculation of y as x times the sine of x, and calculation of z as x minus y. The result z is displayed as 80.4939. An orange arrow points from the Enter key icon to the workspace table.

**Workspace:**

Name	Value	Size	Class
x	42	1x1	double
y	-38.4939	1x1	double
z	80.4939	1x1	double

The Workspace table lists the variables x, y, and z with their corresponding values, sizes, and classes. A calculator icon is shown next to the workspace table.



# Storing Data in Variables

```
>> x = 6*7;
```

Assignment

1) Evaluate right-hand side

2) Assign to variable  
on left-hand side



42



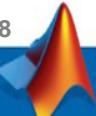
# Chapter 2 Test Your Knowledge

1. The default MATLAB variable type is:

- A. Single
- B. Double
- C. Cell

2. Calculate question below :

- $0.1 \times \frac{2}{5} + 8^7$
- $\sin(0.8\pi)$
- $10^{\ln(13) \times 6}$
- Find  $e^x$ ,  $x = \frac{\sqrt{(5+8)}}{49}$

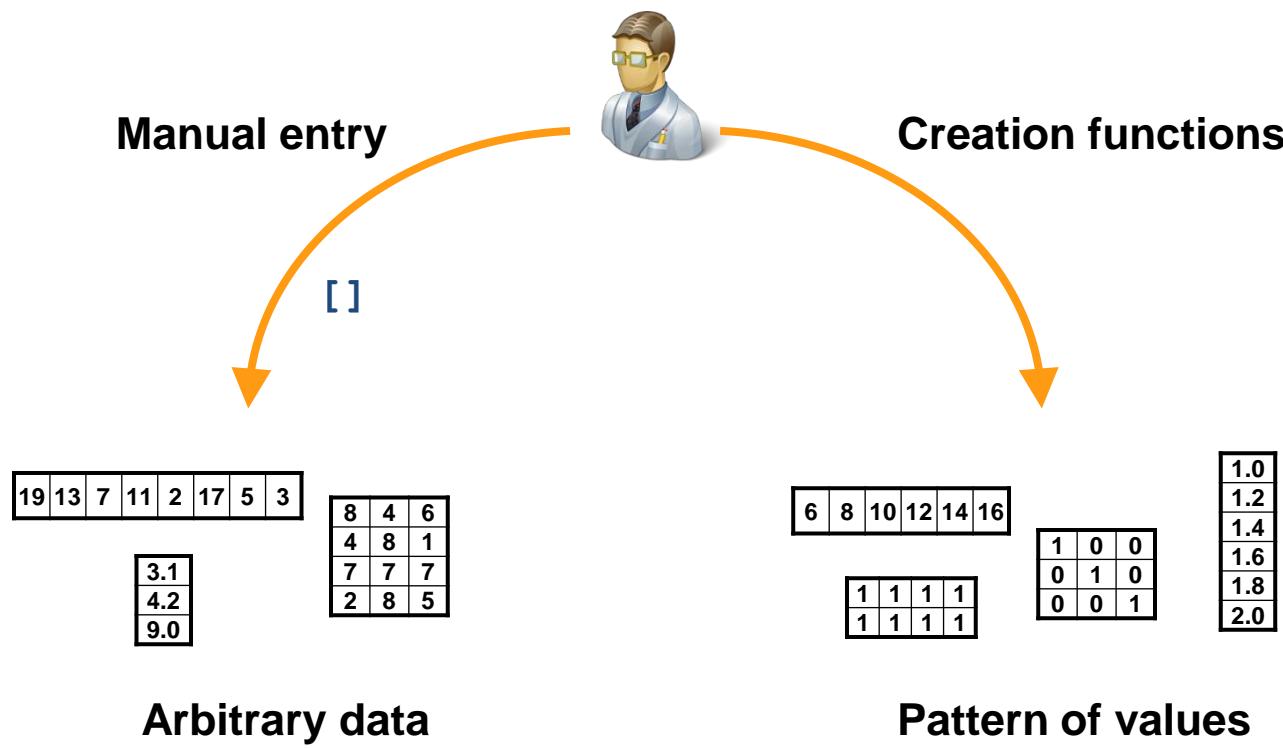


# Course Outline

- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
  - Working with Data Files
  - Visualization with Array
  - Automating Commands with Scripts
  - Appendix: Data Type



# Creating Array

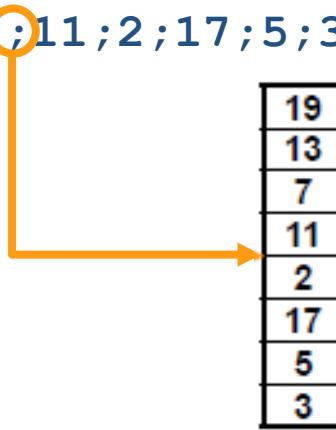


# Entering Vectors Manually

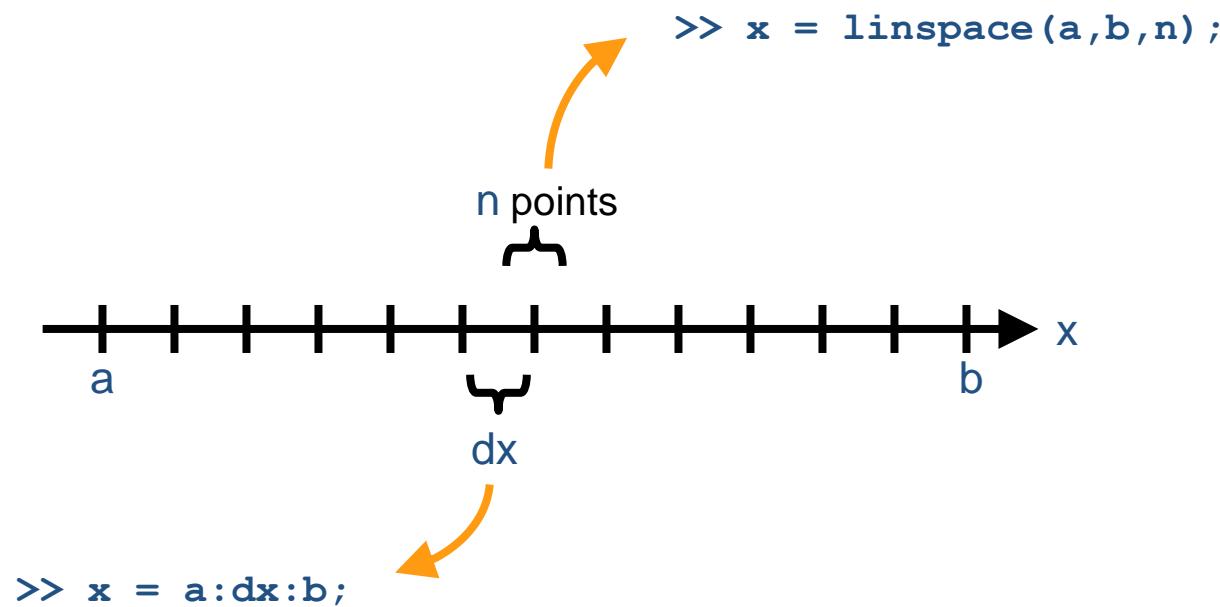
```
>> x = [19 13 7 11 2 17 5 3]
```



```
>> x = [19;13;7;11;2;17;5;3]
```



## Creating Vectors of Equally Spaced Values



# Entering Matrices Manually

```
>> A = [1,2,3; 4,5,6; 7,8,9];
```

or

```
>> A = [1 2 3; 4 5 6; 7 8 9];
```

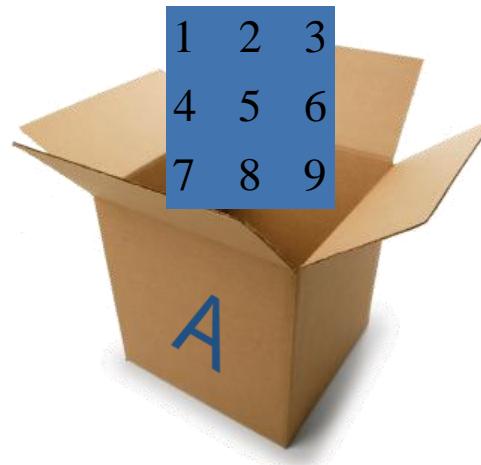
or

```
>> A = [1 2 3
```

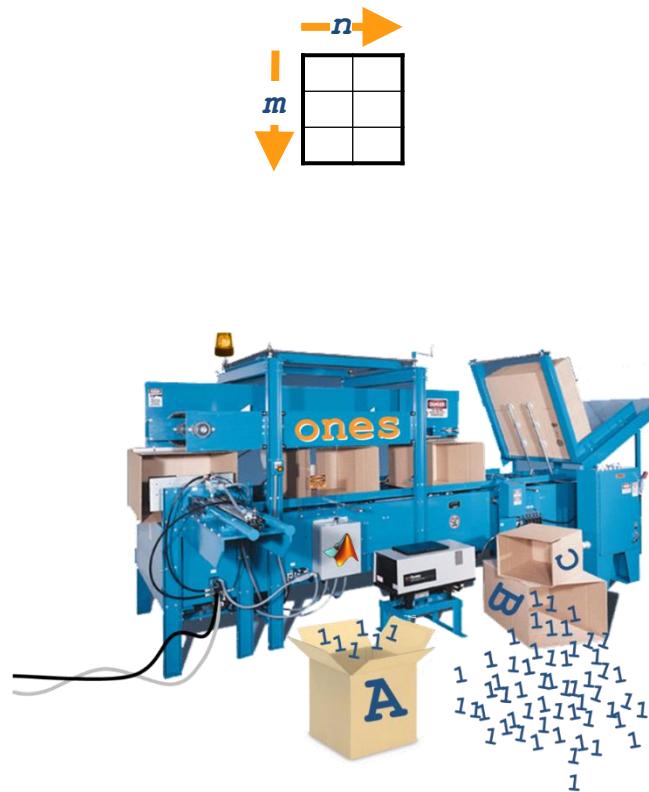
```
4 5 6
```

```
7 8 9]
```

} data entry mode



# Creating Matrices



`>> A = fun(m, n);`

companion  
eye  
gallery  
hadamard  
hankel  
hilb  
invhilb  
magic  
ones

pascal  
rand  
randi  
randn  
rosser  
toeplitz  
vander  
wilkinson  
zeros



## Concatenating Arrays

A	<table border="1"><tr><td>2</td><td>3</td></tr><tr><td>5</td><td>7</td></tr><tr><td>11</td><td>13</td></tr></table>	2	3	5	7	11	13
2	3						
5	7						
11	13						

B	<table border="1"><tr><td>-1</td><td>1</td></tr><tr><td>1</td><td>-1</td></tr></table>	-1	1	1	-1
-1	1				
1	-1				

C	<table border="1"><tr><td>0</td></tr><tr><td>8</td></tr><tr><td>0</td></tr></table>	0	8	0
0				
8				
0				

```
>> X = [A;B]
```

2	3
5	7
11	13
-1	1
1	-1

```
>> Y = [A,C]
```

2	3	0
5	7	8
11	13	0

!!!Please notice the dimension



## Row, Column Indexing

`>> gpdata(1, 2)`

`>> gpdata(3, 6)`

`>> gpdata(2, end)`

	1	2	3	4	5	6	7	8	9	10	11	12
1	1990	NaN	1.8700	3.6300	2.6500	4.5900	3.1600	1	2.0500	2.8200	1.1600	
2	1991	1.9600	1.9200	3.4500	2.9000	4.5000	3.4600	1.3000	2.4900	3.0100	1.1400	
3	1992	1.8900	1.7300	3.5600	3.2700	4.5300	3.5800	1.5000	2.6500	3.0600	1.1300	
4	1993	1.7300	1.5700	3.4100	3.0700	3.6800	4.1600	1.5600	2.8800	2.8400	1.1100	
5	1994	1.8400	1.4500	3.5900	3.5200	3.7000	4.3600	1.4800	2.8700	2.9900	1.1100	
6	1995	1.9500	1.5300	4.2600	3.9600	4	4.4300	1.1100	2.9400	3.2100	1.1500	
7	1996	2.1200	1.6100	4.4100	3.9400	4.3900	3.6400	1.2500	3.1800	3.3400	1.2300	
8	1997	2.0500	1.6200	4	3.5300	4.0700	3.2600	1.4700	3.3400	3.8300	1.2300	
9	1998	1.6300	1.3800	3.8700	3.3400	3.8400	2.8200	1.4900	3.0400	4.0600	1.0600	
10	1999	1.7200	1.5200	3.8500	3.4200	3.8700	3.2700	1.7900	3.8000	4.2900	1.1700	
11	2000	1.9400	1.8600	3.8000	3.4500	3.7700	3.6500	2.0100	4.1800	4.5800	1.5100	
12	2001	1.7100	1.7200	3.5100	3.4000	3.5700	3.2700	2.2000	3.7600	4.1300	1.4600	
13	2002	1.7600	1.6900	3.6200	3.6700	3.7400	3.1500	2.2400	3.8400	4.1600	1.3600	
14	2003	2.1900	1.9900	4.3500	4.5900	4.5300	3.4700	2.0400	4.1100	4.7000	1.5900	
15	2004	2.7200	2.3700	4.9900	5.2400	5.2900	3.9300	2.0300	4.5100	5.5600	1.8800	
16	2005	3.2300	2.8900	5.4600	5.6600	5.7400	4.2800	2.2200	5.2800	5.9700	2.3000	
17	2006	3.5400	3.2600	5.8800	6.0300	6.1000	4.4700	2.3100	5.9200	6.3600	2.5900	
18	2007	3.8500	3.5900	6.6000	6.8800	6.7300	4.4900	2.4000	6.2100	7.1300	2.8000	
19	2008	4.4500	4.0800	7.5100	7.7500	7.6300	5.7400	2.4500	5.8300	7.4200	3.2700	
20												

`>> gpdata(end, 2)`

`>> gpdata(end, end)`

# Multiple Row, Column Indices

```
>> Year = gpdata(:,1)
```

```
>> gpdata([3,4],6:9)
```

	1	2	3	4	5	6	7	8	9	10	11	12
1	1990	Nan	1.8700	3.6300	2.6500	4.5900	3.1600	1	2.0500	2.8200	1.1600	
2	1991	1.9600	1.9200	3.4500	2.9000	4.5000	3.4600	1.3000	2.4900	3.0100	1.1400	
3	1992	1.8900	1.7300	3.5600	3.2700	4.5300	3.5800	1.5000	2.6500	3.0600	1.1300	
4	1993	1.7300	1.5700	3.4100	3.0700	3.6800	4.1600	1.5600	2.8800	2.8400	1.1100	
5	1994	1.8400	1.4500	3.5900	3.5200	3.7000	4.3600	1.4800	2.8700	2.9900	1.1100	
6	1995	1.9500	1.5300	4.2600	3.9600	4	4.4300	1.1100	2.9400	3.2100	1.1500	
7	1996	2.1200	1.6100	4.4100	3.9400	4.3900	3.6400	1.2500	3.1800	3.3400	1.2300	
8	1997	2.0500	1.6200	4	3.5300	4.0700	3.2600	1.4700	3.3400	3.8300	1.2300	
9	1998	1.6300	1.3800	3.8700	3.3400	3.8400	2.8200	1.4900	3.0400	4.0600	1.0600	
10	1999	1.7200	1.5200	3.8500	3.4200	3.8700	3.2700	1.7900	3.8000	4.2900	1.1700	
11	2000	1.9400	1.8600	3.8000	3.4500	3.7700	3.6500	2.0100	4.1800	4.5800	1.5100	
12	2001	1.7100	1.7200	3.5100	3.4000	3.5700	3.2700	2.2000	3.7600	4.1300	1.4600	
13	2002	1.7600	1.6900	3.6200	3.6700	3.7400	3.1500	2.2400	3.8400	4.1600	1.3600	
14	2003	2.1900	1.9900	4.3500	4.5900	4.5300	3.4700	2.0400	4.1100	4.7000	1.5900	
15	2004	2.7200	2.3700	4.9900	5.2400	5.2900	3.9300	2.0300	4.5100	5.5600	1.8800	
16	2005	3.2300	2.8900	5.4600	5.6600	5.7400	4.2800	2.2200	5.2800	5.9700	2.3000	
17	2006	3.5400	3.2600	5.8800	6.0300	6.1000	4.4700	2.3100	5.9200	6.3600	2.5900	
18	2007	3.8500	3.5900	6.6000	6.8800	6.7300	4.4900	2.4000	6.2100	7.1300	2.8000	
19	2008	4.4500	4.0800	7.5100	7.7500	7.6300	5.7400	2.4500	5.8300	7.4200	3.2700	
20												

```
>> gp08 = gpdata(end,2:end)
```

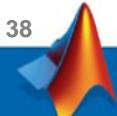
# Linear Indexing

```
>> A = magic(5)  
A =  
    1 17 6 24 11 1 16 8 21 15  
    2 23 7 5 12 7 17 14 22 16  
    3 4 8 6 13 13 18 20 23 22  
    4 10 9 12 14 19 19 21 24 3  
    5 11 10 18 15 25 20 2 25 9
```

Indices Data



end



## Exercise

- ◆ `A = magic(5)`
- ◆ `a=A(?)`
- ◆ `b=A(?)`
- ◆ `c=A(?)`

`A =`

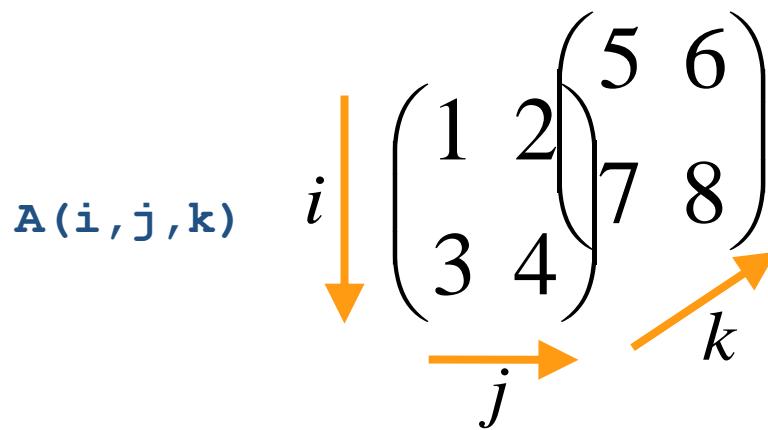
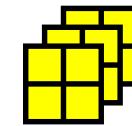
17	24	1	8	15
23	5	7	14	16
4	6	13	20	22
10	12	19	21	3
11	18	25	2	9

Diagram illustrating the output of `A = magic(5)`. The matrix A is a 5x5 magic square. Arrows point to specific elements:

- Arrow labeled `a` points to the element `1` at position (3, 3).
- Arrow labeled `c` points to the element `4` at position (3, 1).
- Arrow labeled `b` points to the element `11` at position (5, 1).



# Multidimensional Arrays

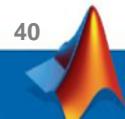


**Construct**

```
>> A(:,:,1) = [1, 2; 3, 4];
>> A(:,:,2) = [5, 6; 7, 8];
>> B = rand(2,2,3);
```

**Access**

```
>> A(:,2,1)
ans =
2
4
```



# Chapter 3-1 Test Your Knowledge

1. (Select all that apply) Which of the following will create a matrix with three rows?
  - A. `A = [zeros(2,4);ones(1,4)];`
  - B. `A = [1;2;3,4;5;6];`
  - C. `A = [1,2;3,4;5,6]';`
  - D. `A = rand(3);`
2. Given a 5-by-5 matrix A, `A(4:end,3:4)` will produce a matrix of what size?
  - A. 1-by-2
  - B. 2-by-2
  - C. 2-by-3
  - D. 3-by-2
3. (Select all that apply) Which commands are equivalent to the command  
`>> x = 1.4:2:6.8;?`
  - A. `X = [1.4 2 6.8];`
  - B. `X = [1.4 6.8];`
  - C. `X = [1.4 3.4 5.4];`
  - D. `X = [1.4 3.4 5.4 6.8];`
  - E. `X = [3.4 5.4];`



# Array Operations

```
>> GMSum = Germany + Mexico
```

The diagram illustrates the element-wise addition of two 19x1 arrays, *Germany* and *Mexico*, to produce the array *GMSum*. The arrays are represented as vertical columns of numbers, separated by a plus sign. The resulting array *GMSum* is also shown as a column of numbers.

Germany	Mexico	GMSum
3.65	2.65	1.00
4.20	2.90	1.30
4.77	3.27	1.50
4.63	3.07	1.56
5.00	3.52	1.48
5.07	3.96	1.11
5.19	3.94	1.25
5.00	3.53	1.47
4.83	3.34	1.49
5.21	3.42	1.79
5.46	3.45	2.01
5.60	3.40	2.20
5.91	3.67	2.24
6.63	4.59	2.04
7.27	5.24	2.03
7.88	5.66	2.22
8.34	6.03	2.31
9.28	6.88	2.40
10.20	7.75	2.45

```
>> load gasprices
```

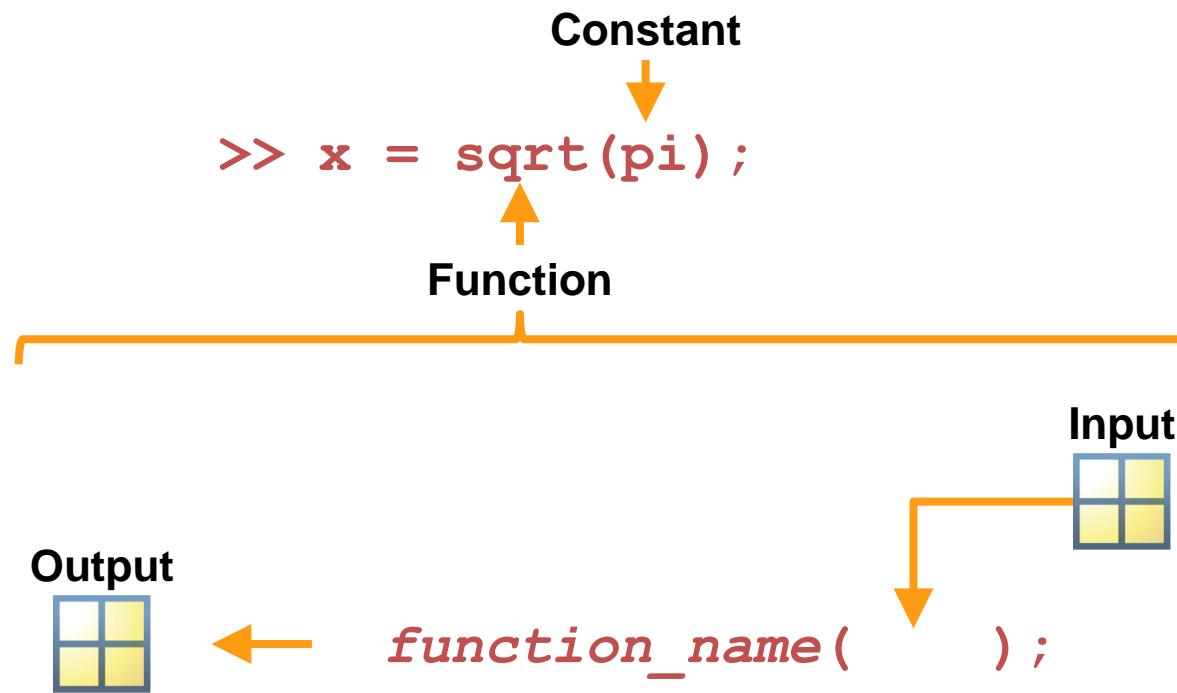
$$\begin{matrix} \text{19-by-1} \\ \text{array} \end{matrix} = \begin{matrix} \text{19-by-1} \\ \text{array} \end{matrix} \otimes \begin{matrix} \text{1-by-1} \\ \text{array} \end{matrix}$$

$$\begin{matrix} \text{n-by-1} \\ \text{array} \end{matrix} + \begin{matrix} \text{1-by-1} \\ \text{array} \end{matrix} = \begin{matrix} \text{n-by-1} \\ \text{array} \end{matrix}$$

$$\begin{matrix} \text{n-by-1} \\ \text{array} \end{matrix} + \begin{matrix} \text{n-by-1} \\ \text{array} \end{matrix} = \begin{matrix} \text{n-by-1} \\ \text{array} \end{matrix}$$



# Using Built-In Functions and Constants



# Mathematical Functions

```
>> y = sin(x);
```

“Vectorized”

x

0	0.7854	1.5708	2.3562	3.1416
---	--------	--------	--------	--------



y

0	0.7071	1.0000	0.7071	0.0000
---	--------	--------	--------	--------

sin  
sind  
sinh  
asin  
exp  
log  
log2  
log10  
sqrt  
nthroot  
abs  
angle  
floor  
ceil  
round  
mod



# Matrix Operations

**Inner dimensions must be equal!!**

**Prices**

1.96	1.87	3.63	2.65	4.59	3.16	1.00	2.05	2.82	1.16
1.96	1.92	3.45	2.90	4.50	3.46	1.30	2.49	3.01	1.14
1.89	1.73	3.56	3.27	4.53	3.58	1.50	2.65	3.06	1.13
1.73	1.57	3.41	3.07	3.68	4.16	1.56	2.88	2.84	1.11
1.84	1.45	3.59	3.52	3.70	4.36	1.48	2.87	2.99	1.11
1.95	1.53	4.26	3.96	4.00	4.43	1.11	2.94	3.21	1.15
2.12	1.61	4.41	3.94	4.39	3.64	1.25	3.18	3.34	1.23
2.05	1.62	4.00	3.53	4.07	3.26	1.47	3.34	3.83	1.23
1.63	1.38	3.87	3.34	3.84	2.82	1.49	3.04	4.06	1.06
1.72	1.52	3.85	3.42	3.87	3.27	1.79	3.80	4.29	1.17
1.94	1.86	3.80	3.45	3.77	3.65	2.01	4.18	4.58	1.51
1.71	1.72	3.51	3.40	3.57	3.27	2.20	3.76	4.13	1.46
1.76	1.69	3.62	3.67	3.74	3.15	2.24	3.84	4.16	1.36
2.19	1.99	4.35	4.59	4.53	3.47	2.04	4.11	4.70	1.59
2.72	2.37	4.99	5.24	5.29	3.93	2.03	4.51	5.56	1.8
3.23	2.89	5.46	5.66	5.74	4.28	2.22	5.28	5.97	2.30
3.54	3.26	5.88	6.03	6.10	4.47	2.31	5.92	6.36	2.59
3.85	3.59	6.60	6.88	6.73	4.49	2.40	6.21	7.13	2.80
4.45	4.08	7.51	7.75	7.63	5.74	2.45	5.83	7.42	3.2

**rpop/100**



0.0210
0.0370
0.0701
0.0900
0.0660
0.1390
0.1180
0.0530
0.0670
0.3360

**WAvgPrices**

2.1547
2.2662
2.3495
2.3339
2.4143
2.5245
2.5206
2.4468
2.2712
2.4766
2.7190
2.5711
2.5750
2.9206
3.3485
3.602
4.618
4.006
4.705

**19-by-1**

19-by-10

10-by-1

$$2.72 * 0.0210 + 2.37 * 0.0370 + 4.99 * 0.0701 + 5.24 * 0.0900 + \dots + 5.56 * 0.0670 + 1.88 * 0.3360 = 3.3489$$

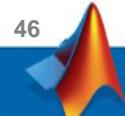
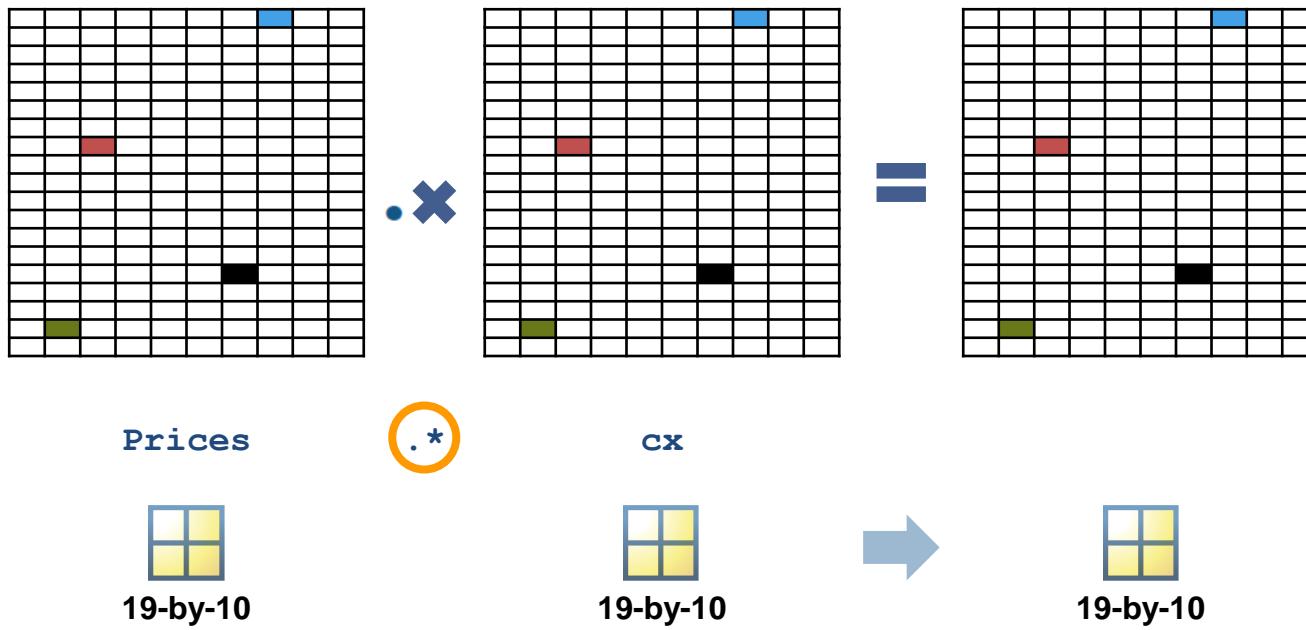


$$= \begin{matrix} 12\text{-by-}24 \end{matrix} \times \begin{matrix} 1\text{-by-}1 \end{matrix}$$

$$\begin{matrix} m\text{-by-}n \end{matrix} + \begin{matrix} m\text{-by-}n \end{matrix} = \begin{matrix} m\text{-by-}n \end{matrix}$$

$$\begin{matrix} m\text{-by-}n \end{matrix} + \begin{matrix} 1\text{-by-1} \end{matrix} = \begin{matrix} m\text{-by-}n \end{matrix}$$

# Array Operations



## Array Operations (Cont.)

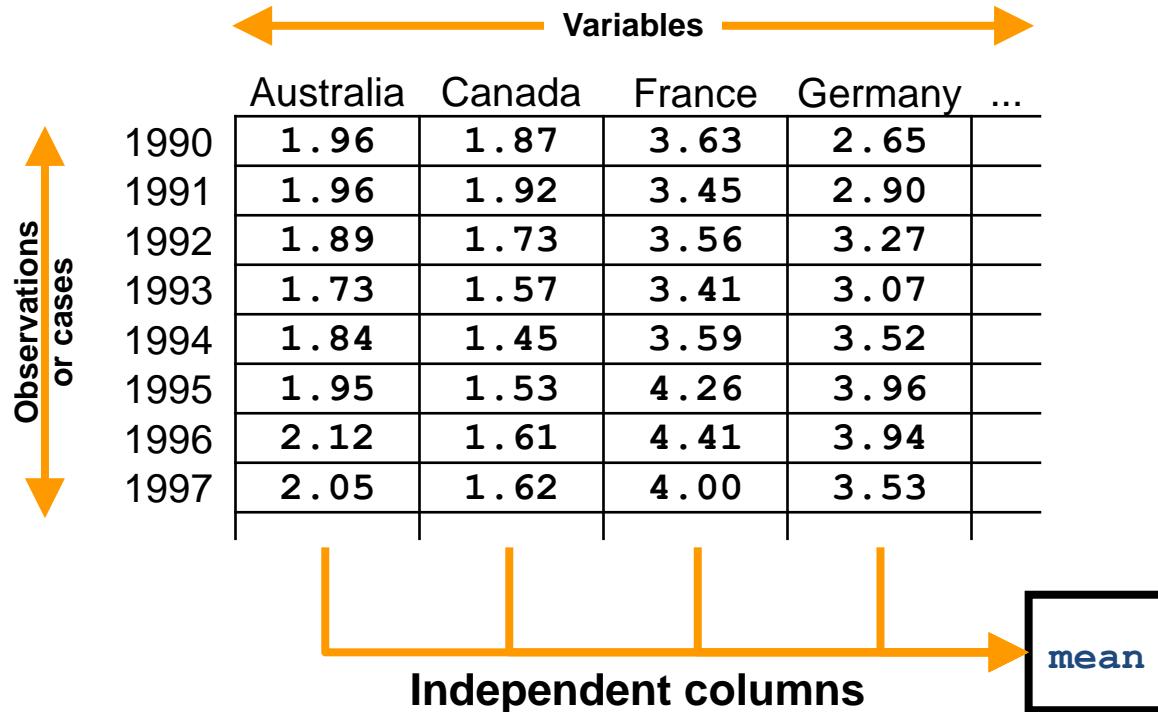
```
>> GMRatio = Germany ./ Mexico
```

Matrices  
must have  
the same  
dimensions!!

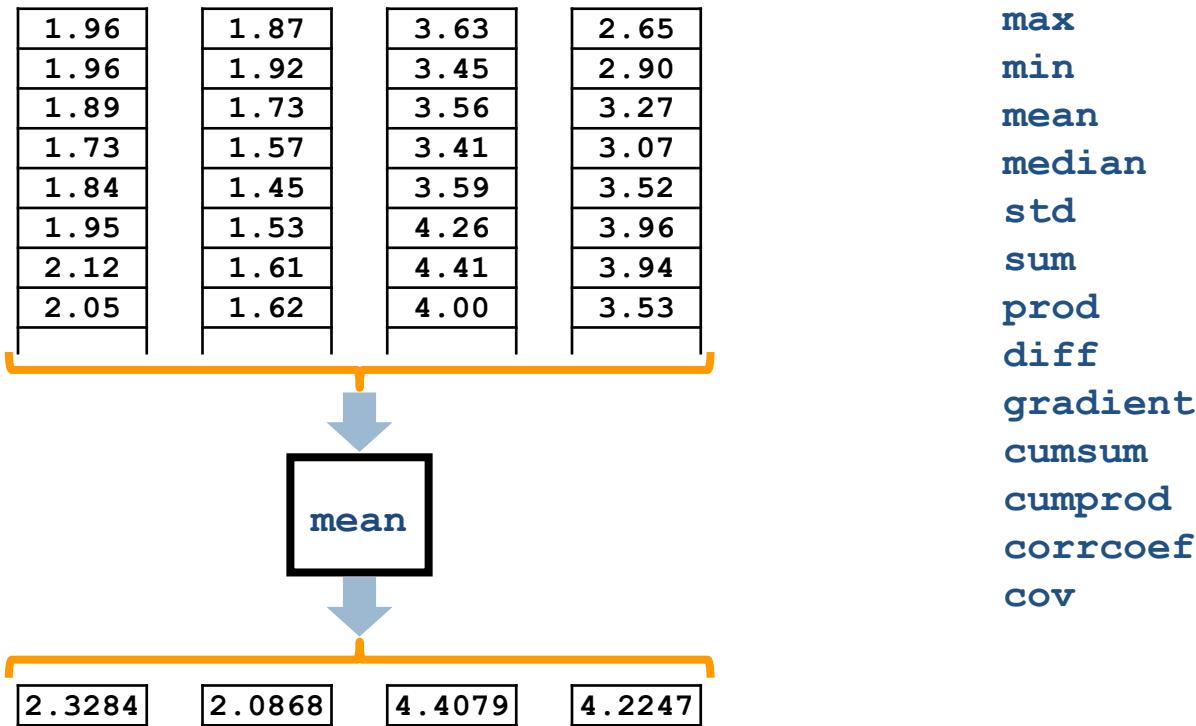
2.65	2.65	1.00
2.23	2.90	1.30
2.18	3.27	1.50
1.97	3.07	1.56
2.38	3.52	1.48
3.57	3.96	1.11
3.15	3.94	1.25
2.40	3.53	1.47
2.24	3.34	1.49
1.91	3.42	1.79
1.72	3.45	2.01
1.55	3.40	2.20
1.64	3.67	2.24
2.25	4.59	2.04
2.58	5.24	2.03
2.55	5.66	2.22
2.61	6.03	2.31
2.87	6.88	2.40
3.16	7.75	2.45



# Data in the MATLAB® Environment



# Statistical Operations



# Example: Array Operations

- In most languages - use loops:

```
>> tic; for I = 1:10000  
Density(I) = Mass(I) / (Length(I)*Width(I)*Height(I));  
end; toc  
elapsed_time =  
4.7260
```

Use TIC and TOC to measure elapsed time

- In MATLAB - use array operations:

```
>> tic; Density = Mass ./ (Length.*Width.*Height); toc  
elapsed_time =  
0
```

Vectorized code is much faster than loops

```
>> array_examp
```



# Chapter 3-2 Test Your Knowledge

1. (Select all that apply) Given two matrices, **A** and **B**, where **A** is a 2-by-3 matrix and **B** is a 3-by-2 matrix, which of the following operations are valid?
  - A. **A+B**
  - B. **A .+B**
  - C. **A\*B**
  - D. **A .\*B**
2. Given a vector **x**, what is the command to add 3 to each element, double that value, then sum all the resulting values?
  - A. **sum (2\*x+3)**
  - B. **sum (2\* [x (k)+3] )**
  - C. **sum [2\*x+3]**
  - D. **sum (2\* (x+3) )**

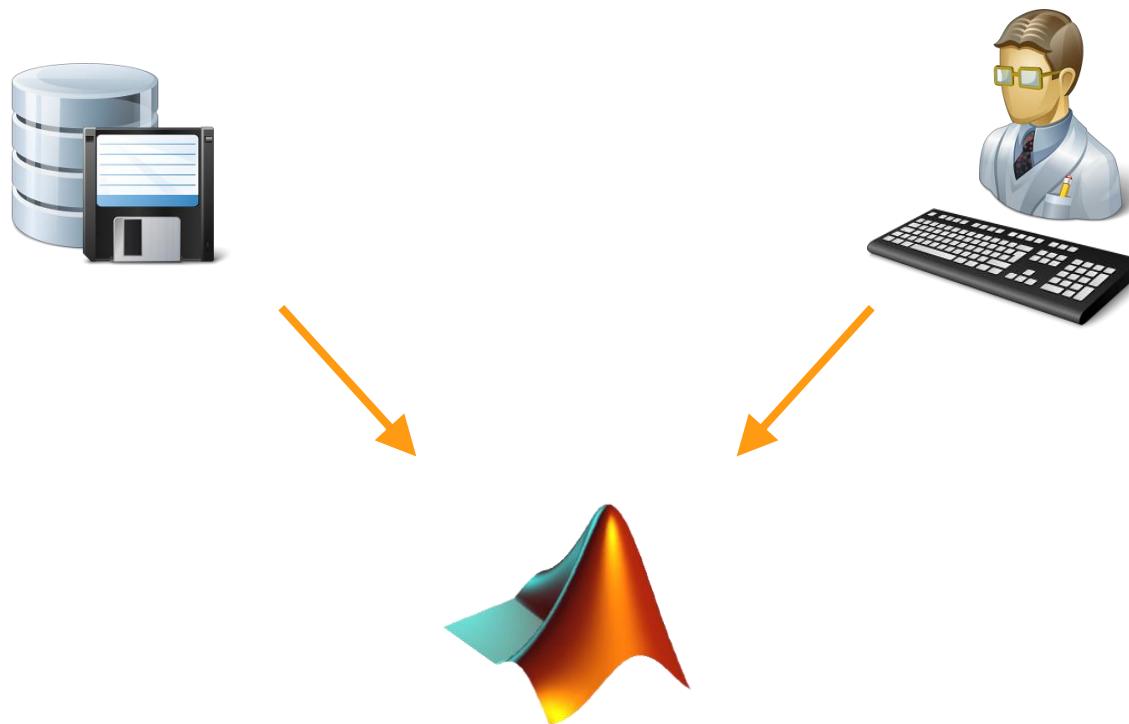


# Course Outline

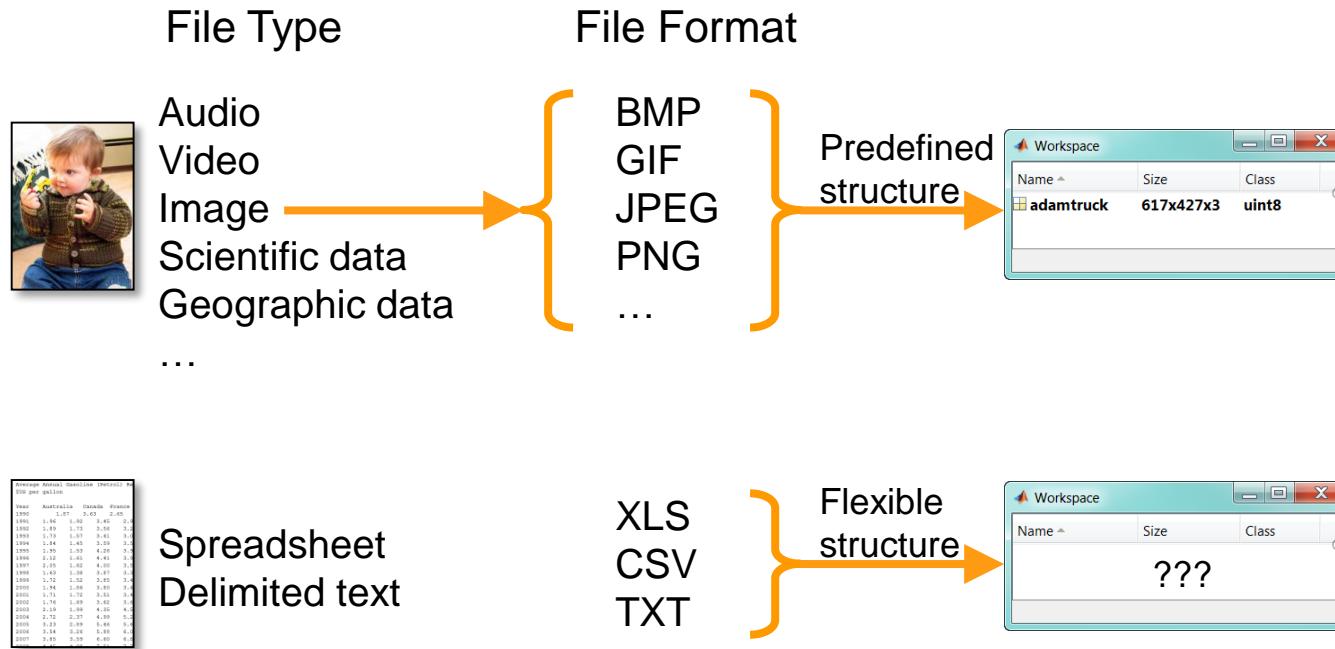
- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
- Working with Data Files
- Visualization with Array
- Automating Commands with Scripts
- Appendix: Data Type



# Getting Data into MATLAB®



# File Types and File Formats



# Reading and Writing Fixed-Structure Files



`audioread`

```
1.20E+01,9.38E-07,4.  
1.30E+01,6.11E-08,9.  
1.40E+01,4.58E-07,6.  
1.50E+01,6.89E-07,3.  
1.60E+01,1.66E-07,2.  
1.70E+01,2.13E-07,2.  
1.80E+01,2.38E-07,3.  
1.90E+01,4.13E-07,9.  
2.00E+01,8.23E-07,7.  
2.10E+01,6.60E-07,4.  
2.20E+01,9.11E-07,8.  
2.30E+01,7.77E-07,1.
```

`cdfread`

`csvread`

`dlmread`

`h5read`



`imread`



`ncread`

`urlread`

`xmlread`

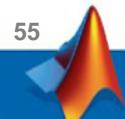
...



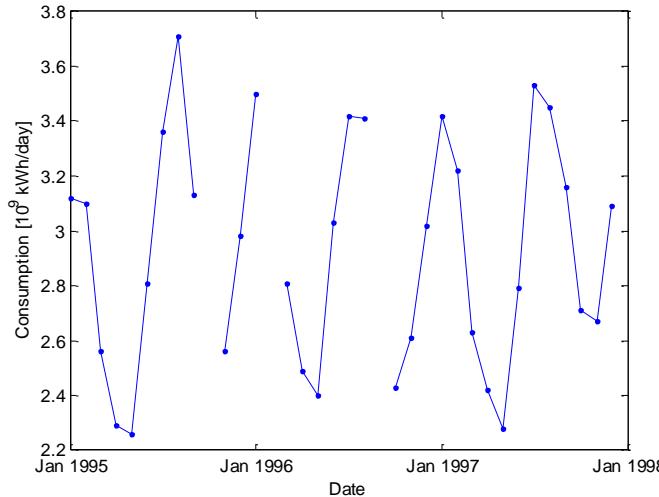
`data = xyzread('file.xyz')`



`xyzwrite(data,'file.xyz')`



## Dealing with Missing Data



**mean**

3.190
0
<del>4.030</del>
4.380
0
<b>NaN</b>
>> <b>nanmean(x)</b>
3.430
0
3.050
0

**mean**

3.190
0
<del>4.030</del>
0
4.380
0
<b>NaN</b>
>> <b>x(isnan(x)) =</b>
[ ] ; 3.050
>> <b>mean(x)</b>

```
>> load electricityNaNs  
>> x = Residential(215:220);  
>> mean(x)  
>> nnz(isnan(x))
```

**mean**

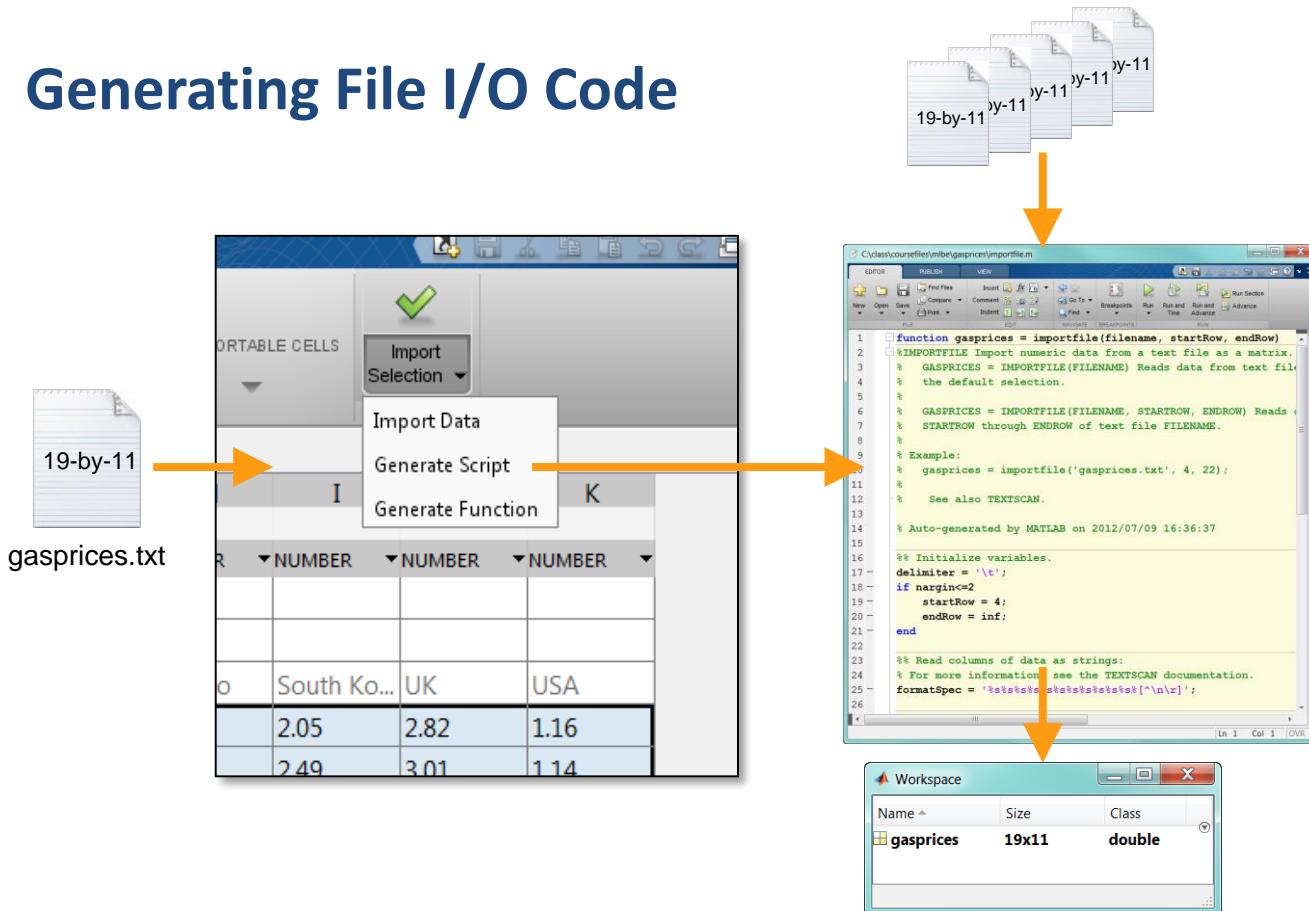
3.190
0
4.030
0
4.380
0
<b>NaN</b>
3.430
0
3.190
3.050
0
4.030
0
4.380
0

ans =  
NaN

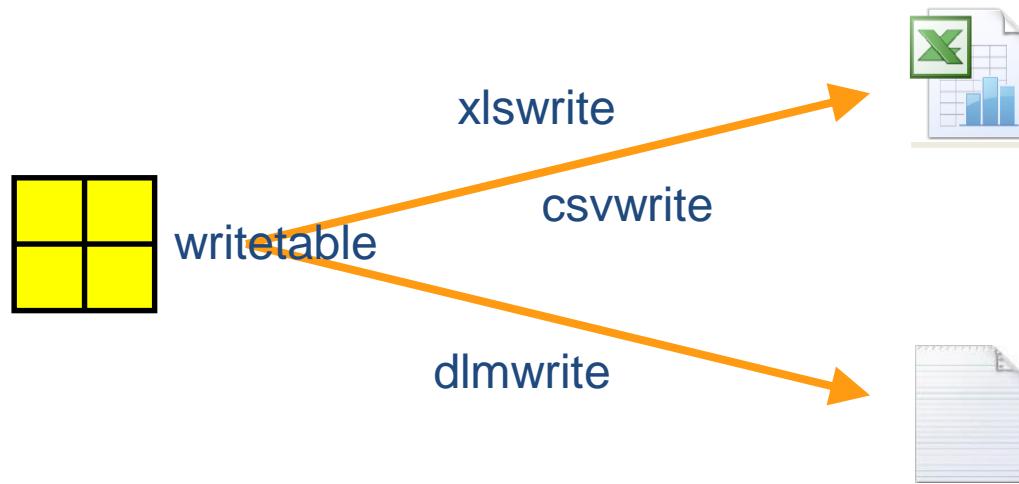
**mean**

3.190
0
4.030
0
4.380
0
<b>3.880</b>
8
3.430
3.050
0
3.8808
>> <b>x(isnan(x)) =</b>
[ ] ; 3.050
>> <b>mean(x)</b>
0

# Generating File I/O Code



# Exporting Spreadsheets



# Chapter 4 Test Your Knowledge

1. (Select all that apply) Which of the following can you change when importing a delimited text file with the Import Tool?
  - A. Which columns to import.
  - B. The format (text, number, date, etc.) of each column.
  - C. The delimiter character.
  - D. How to import missing values.
  
2. Suppose “ABC” is a supported format in MATLAB. Which command would import data from an ABC file mydata.abc into a matrix A?
  - A. `mydata = abcread(A);`
  - B. `abcread(A,mydata);`
  - C. `abcread('mydata.abc',A);`
  - D. `A = abcread('mydata.abc');`
  - E. `A = abcread(mydata);`



# Course Outline

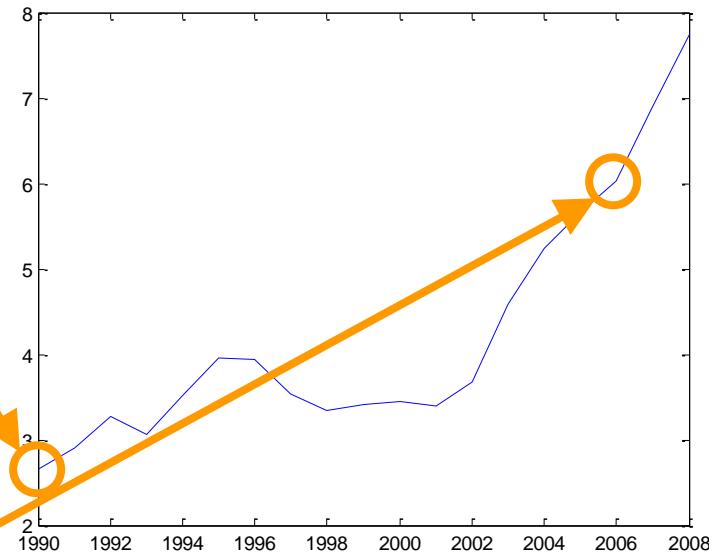
- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
- Working with Data Files
- Visualization with Array
- Automating Commands with Scripts
- Appendix: Data Type



# Plotting Vectors

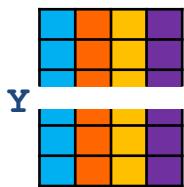
```
>> plot(Year,Germany)
```

1990	2.65
1991	2.90
1992	3.27
1993	3.07
1994	3.52
1995	3.96
1996	3.94
1997	3.53
1998	3.34
1999	3.42
2000	3.45
2001	3.40
2002	3.67
2003	4.59
2004	5.24
2005	5.66
2006	6.03
2007	6.88
2008	7.75

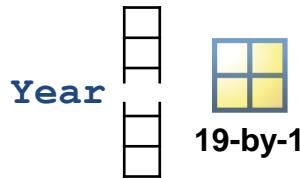
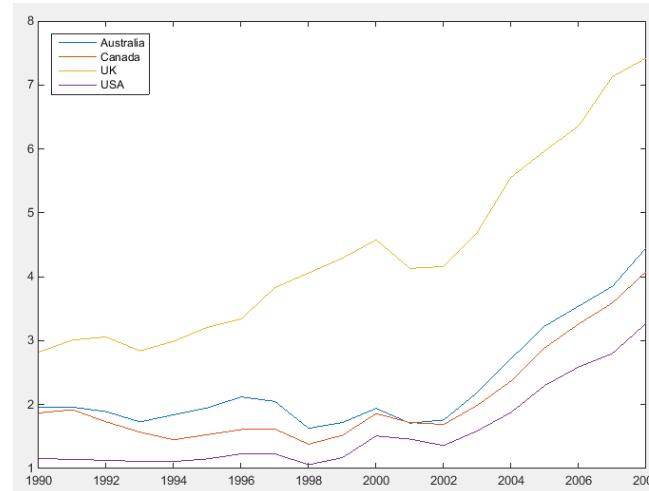


## Plotting Multiple Columns

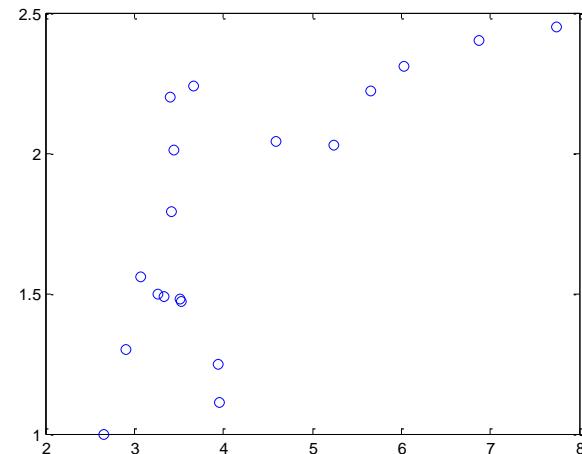
```
>> EndIdx = [1 2 9 10];  
>> Y = Prices(:, EndIdx);  
>> plot(Year,Y)  
>> legend(country(EndIdx), 'Location', 'NW')
```



4 plots



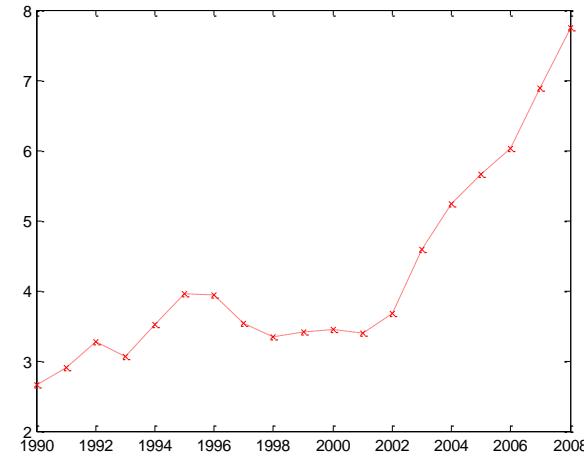
# Plot Options



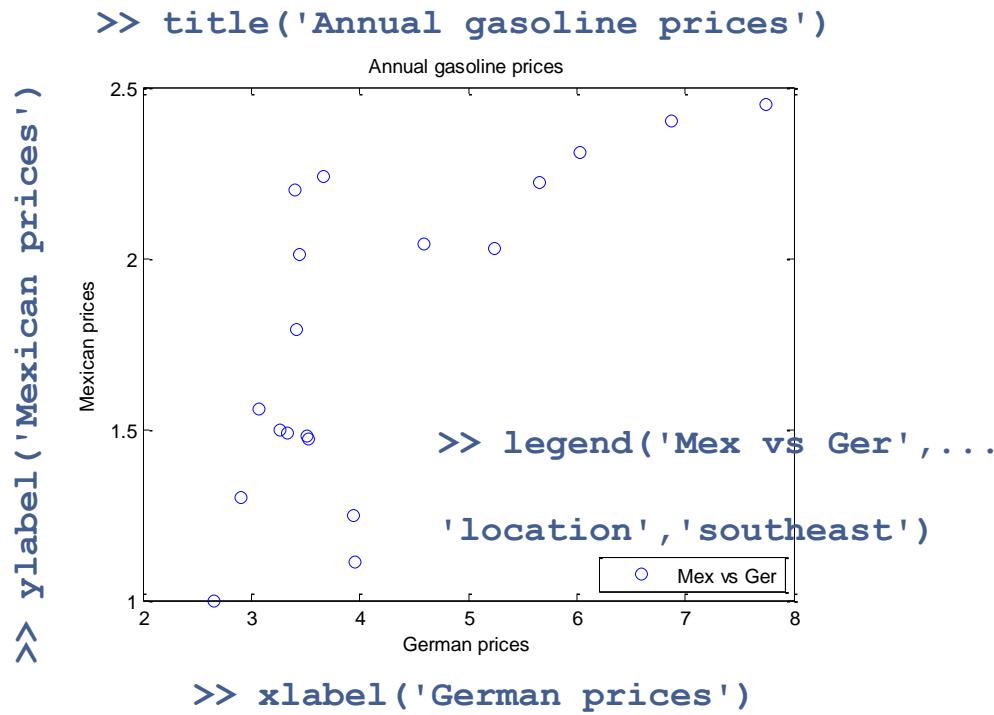
```
>> plot(Germany,Mexico,'o')
```

Data Type

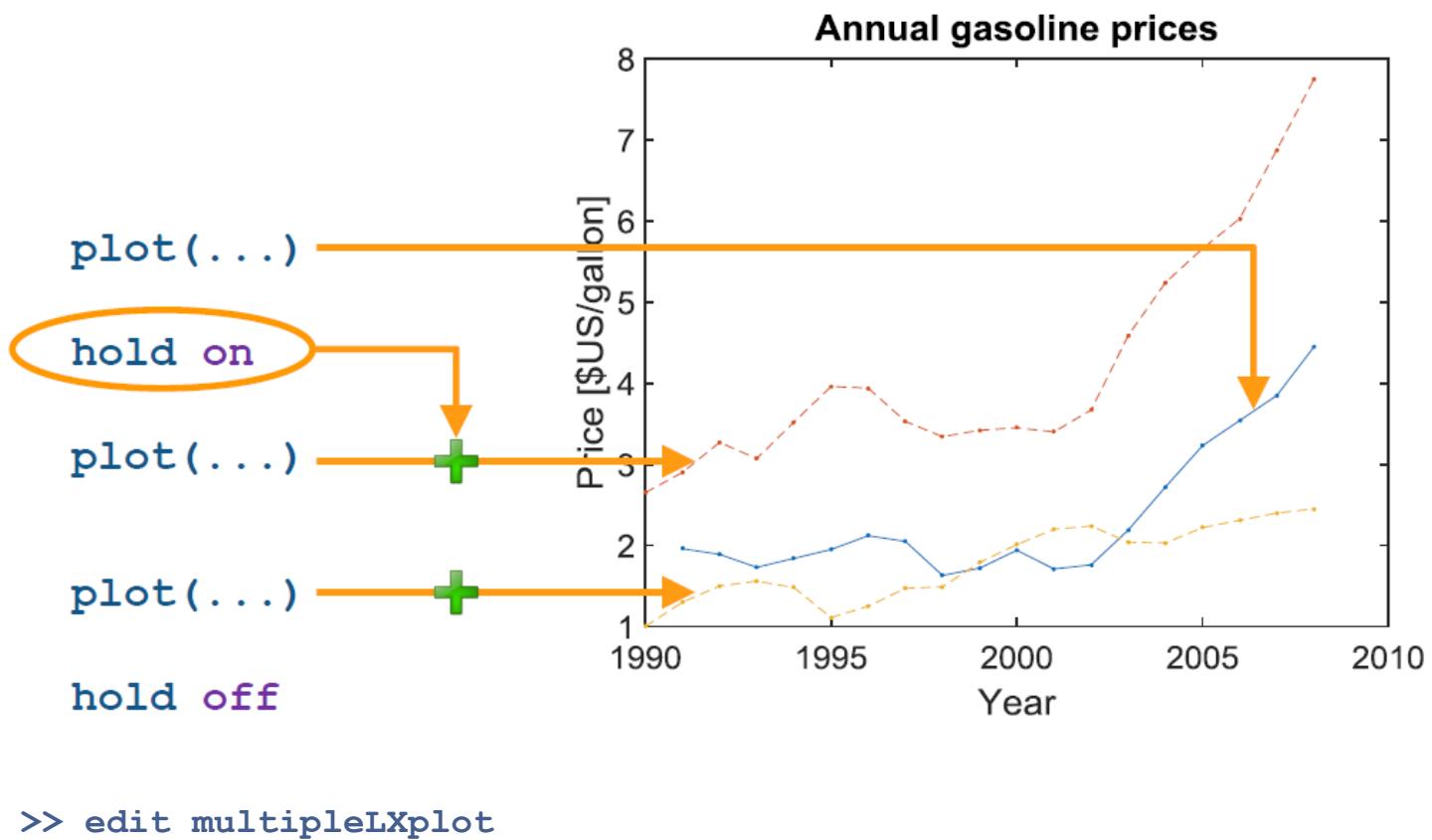
```
>> plot(Year,Germany,'rx:')
```



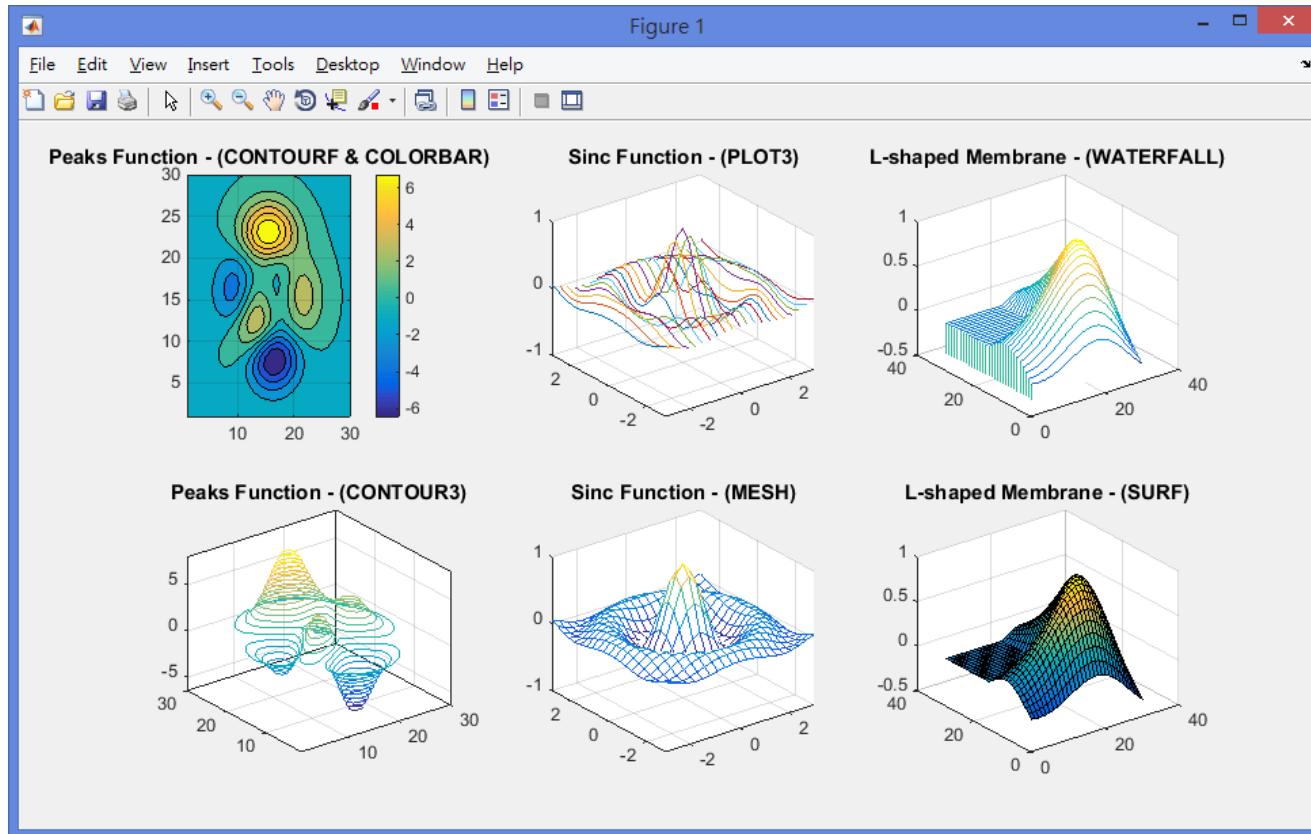
# Annotating Plots



# Adding Plots



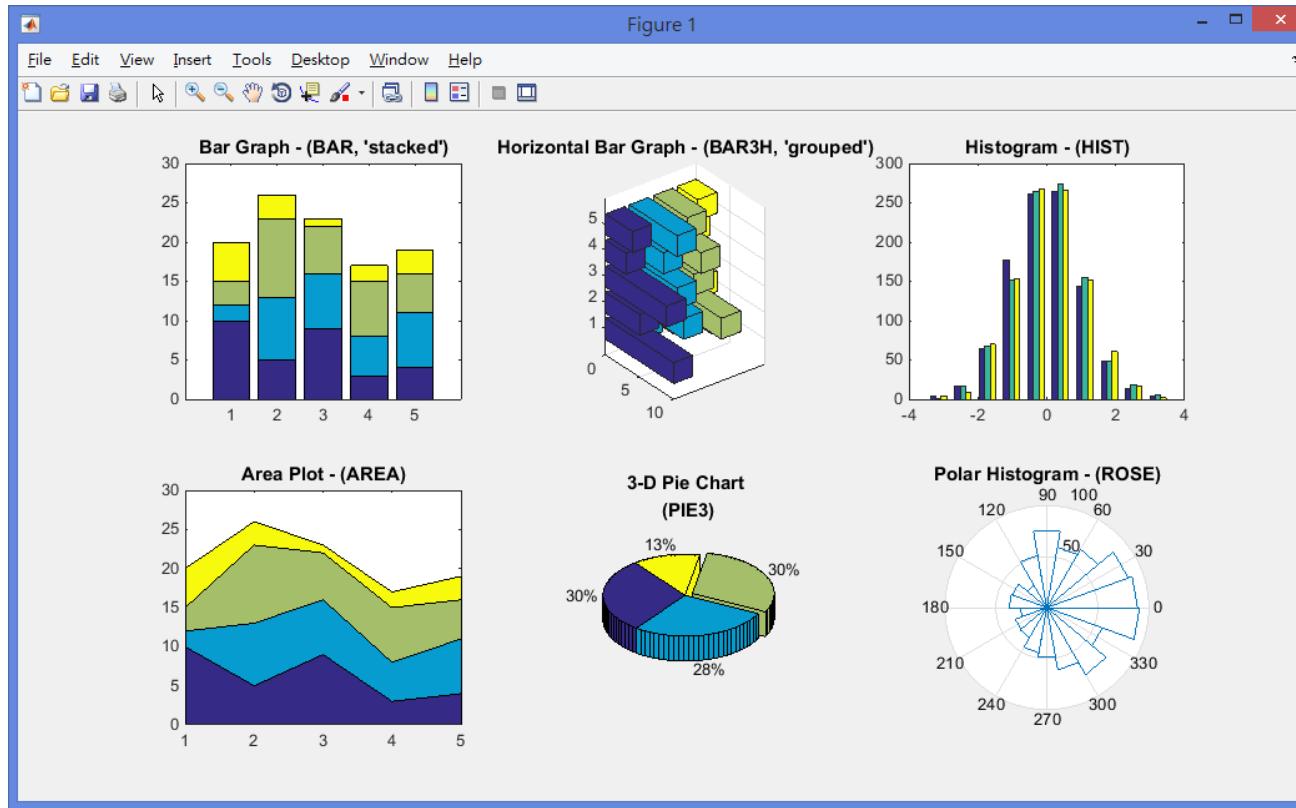
# 3-D Surface Plotting



`>> surf_3d`



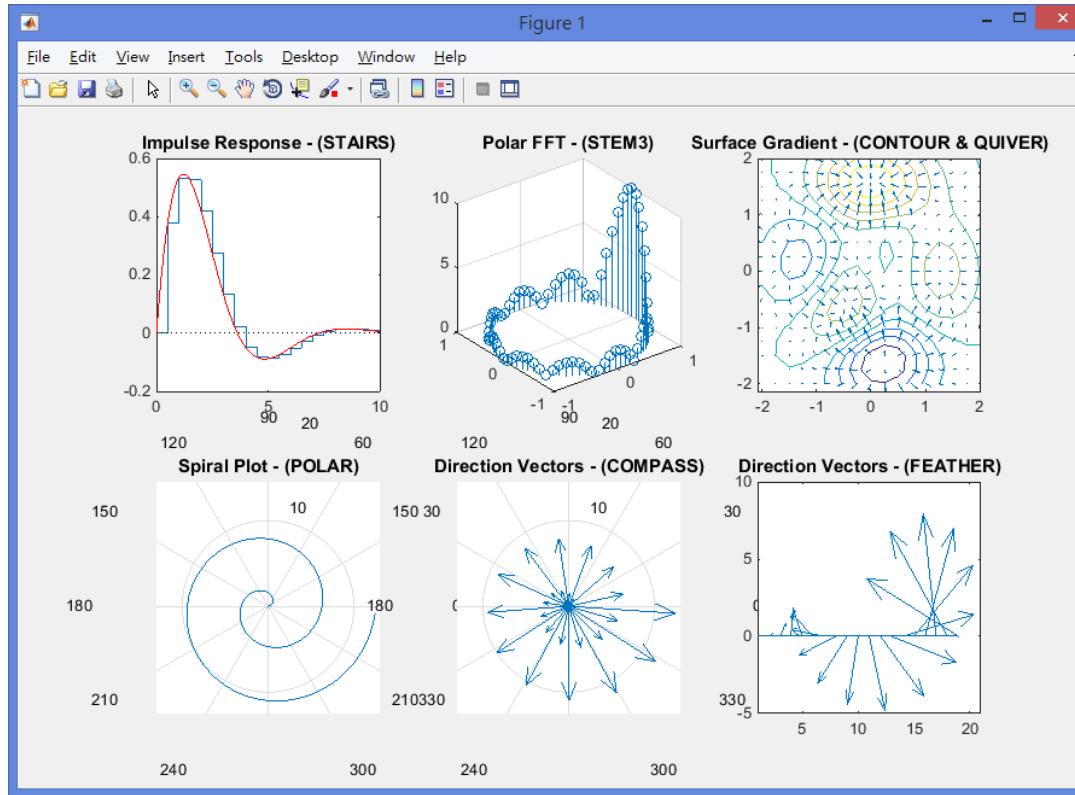
# Specialized Plotting Routines



`>> spec_plots`



# Specialized Plotting Routines (Continued)

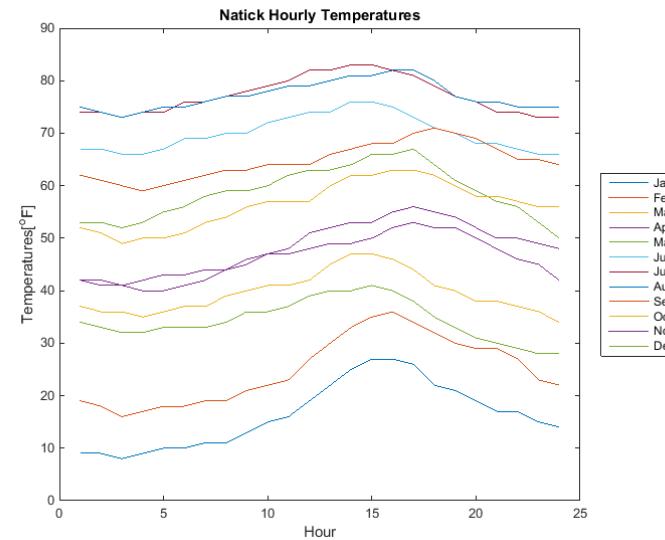


```
>> spec_plots2
```



## Exercise: Natick Hourly Temperatures

1. Load `natickData.mat`.
2. Calculate the average temperature for each month.
3. Determine which month had the largest standard deviation.
4. Determine which month had the largest range ( $\text{max} - \text{min}$ ).
5. Plot temperature vs. time for each month.



```
>> edit  
natickTemps
```



# Chapter 5 Test Your Knowledge

1. (Select all that apply) Which of the following will create a scatter plot of `frogs` on the horizontal axis and `GDP` on the vertical axis, with red markers at the data points?
  - A. `plot(GDP,frogs,'ro')`
  - B. `plot(GDP,frogs,'o','r')`
  - C. `plot(frogs,GDP,'ro')`
  - D. `plot(frogs,GDP,'red')`
2. If `A` is a 15-by-7 matrix, which of the following commands will result in five line plots on the same axes?
  - A. `plot(A(:))`
  - B. `plot(A(:,3:end))`
  - C. `plot(A(11:end,:))`
  - D. `plot(A(2:6))`



# Course Outline

- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
- Working with Data Files
- Visualization with Array
- Automating Commands with Scripts
- Appendix: Data Type



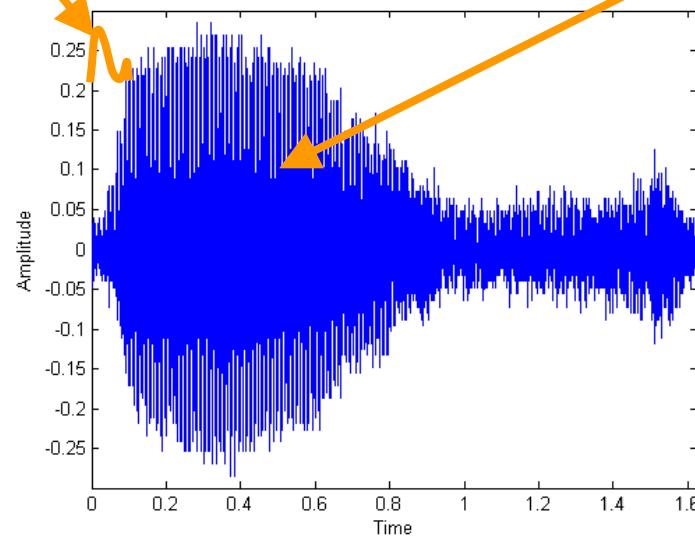
# Example: Modeling a Whal Call



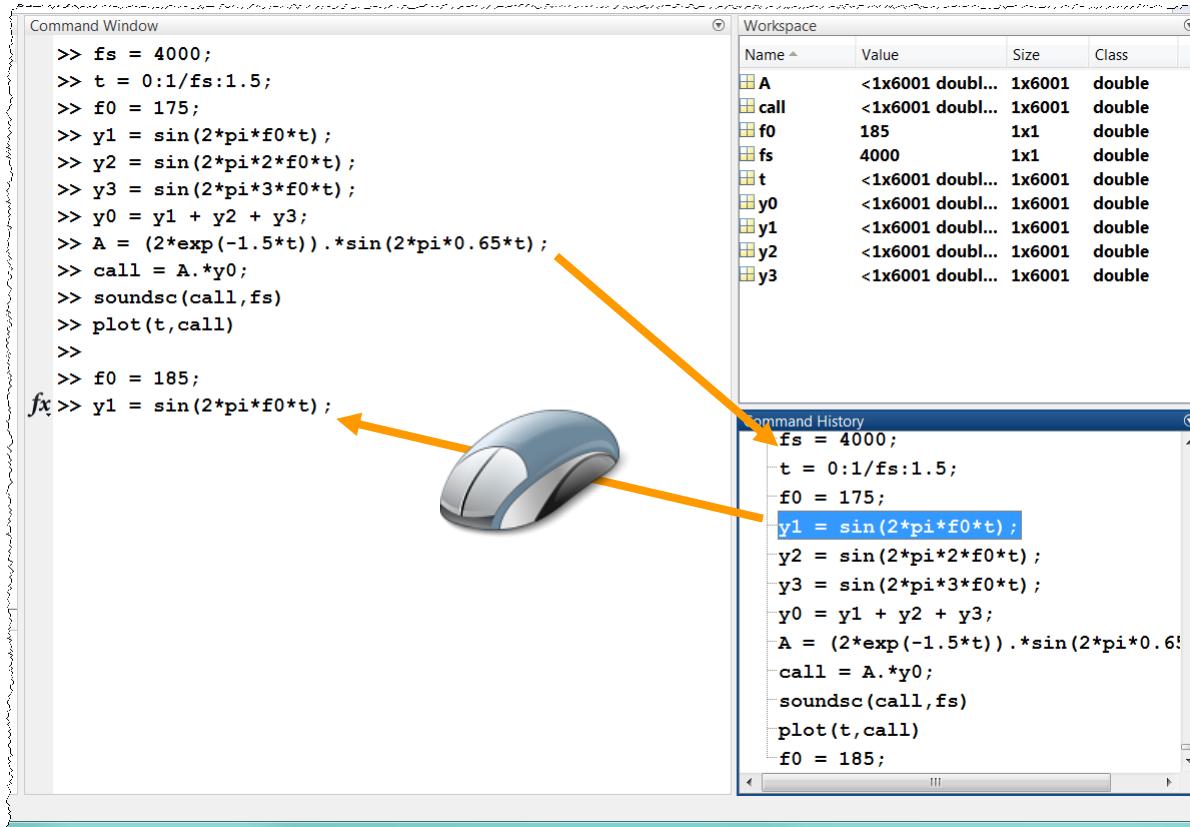
Amplitude modulated:  $y(t) = A(t)y_0(t)$

Decaying oscillation  
 $A(t) = A_0 e^{-Bt} \sin(2\pi f_m t)$

Sum of harmonics of a fundamental frequency  
 $y_0(t) = \sum_n \sin(2\pi n f_0 t)$

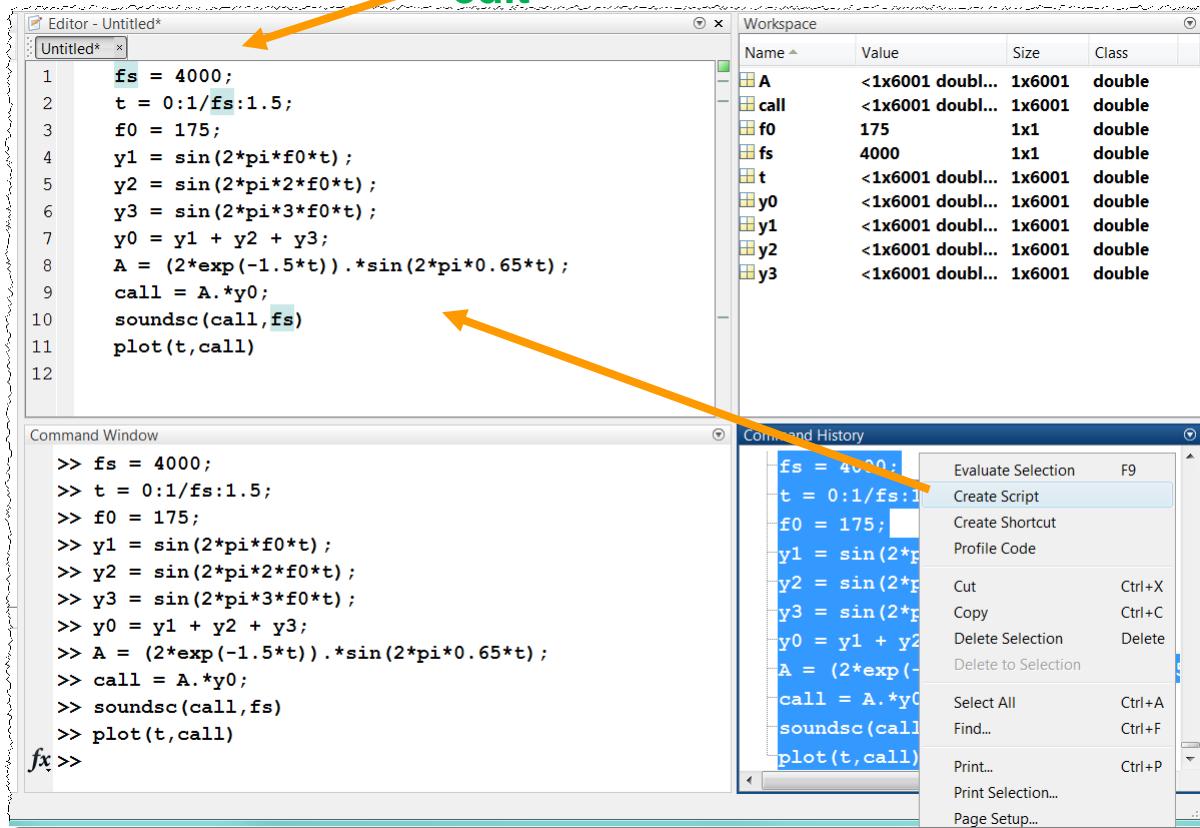


# The Command History



# The MATLAB® Editor

edit



# Script Files

```
H1 line % CALLMODEL Models a blue whale B call.  
%  
% Uses a model of the form y = A.*y0  
% where A = A0*exp(-B*t).*sin(2*pi*fm*t)  
% and y0 is a sum of harmonics  
% yn = sin(2*pi*n*f0*t)  
  
% Create the time base for the signal.  
fs = 4000;  
t = 0:(1/fs):1.5;  
  
% Set the fundamental frequency of the call.  
f0 = 175;  
  
% Create the harmonics.  
y0 = sin(2*pi*f0*t) + sin(2*pi*2*f0*t) +  
sin(2*pi*3*f0*t);
```

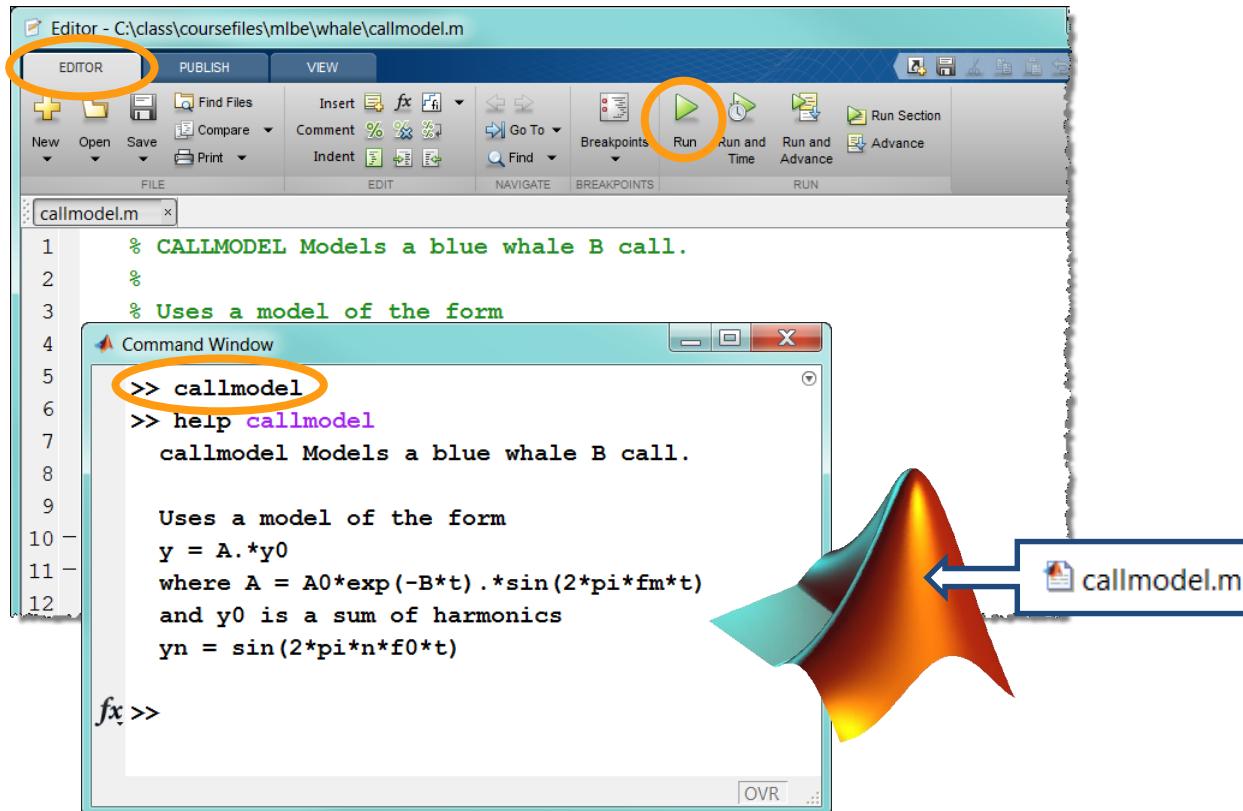
Code

Help

Comments

```
>> edit callmodel.m
```

# Running a Script



# Code Sections

- Create the time base for the signal
- Set the fundamental frequency of the call
- Create the harmonics
- Create the envelope
- Create the call
- Plot the model call and listen to it



%% Set the fundamental frequency of the call.  
f0 = 175;

%% Create the harmonics.  
y0 = sin(2\*pi\*f0\*t) + sin(2\*pi\*2\*f0\*t) + sin(2\*pi\*3\*f0\*t);

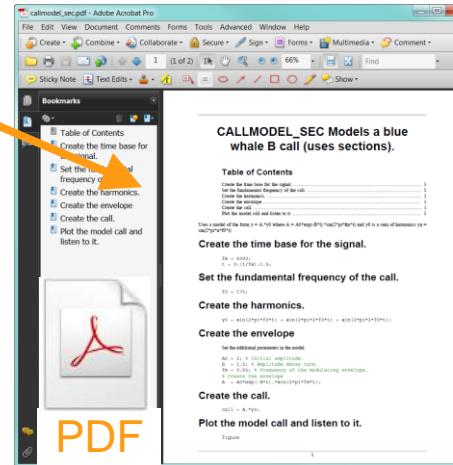
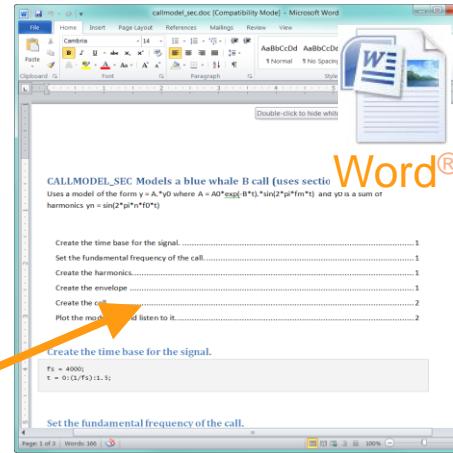
%% Create the envelope  
% Set the additional parameters in the model.  
A0 = 2; % Initial amplitude.  
B = 1.5; % Amplitude decay rate.  
fm = 0.65; % Frequency of the modulating envelope.  
% Create the envelope  
A = A0\*exp(-B\*t).\*sin(2\*pi\*fm\*t);

%% Create the call.  
call = A.\*y0;



# Publishing Code

```
% Set the fundamental frequency of the call.  
f0 = 175;  
  
%% Create the harmonics.  
y0 = sin(2*pi*f0*t) + sin(2*pi*2*f0*t) + sin(2*pi*3*f0*t);  
  
%% Create the envelope  
% Set the additional parameters in the model.  
A0 = 2; % Initial amplitude.  
B = 1.5; % Amplitude decay rate.  
fm = 0.65; % Frequency of the modulating envelope.  
% Create the envelope  
A = A0*exp(-B*t).*sin(2*pi*fm*t);  
  
%% Create the call.  
call = A.*y0;
```



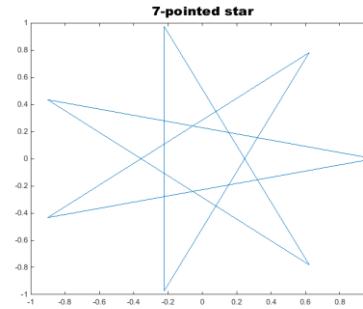
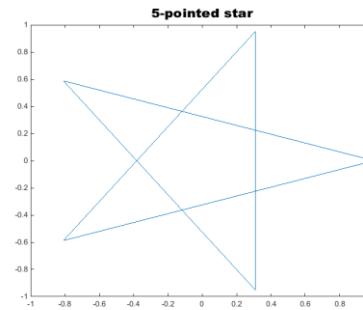
# Exercise: N-Pointed Star

- Create a script to draw a 5-pointed and a 7-pointed star using the equations

$$x = \cos\left(\frac{(n-1)t\pi}{n}\right)$$

$$y = \sin\left(\frac{(n-1)t\pi}{n}\right)$$

with  $t = 0, 1, 2, \dots, 100$



```
>> edit nPointStar
```



# Chapter 6 Test Your Knowledge

- T/F: Anything following a % sign is ignored by MATLAB as a comment.
- T/F: In section mode, you can modify your code and rerun it without having to save the file.
- T/F: Script files can access and modify any variables already in the base MATLAB workspace.



## Q & A

Thank you very much

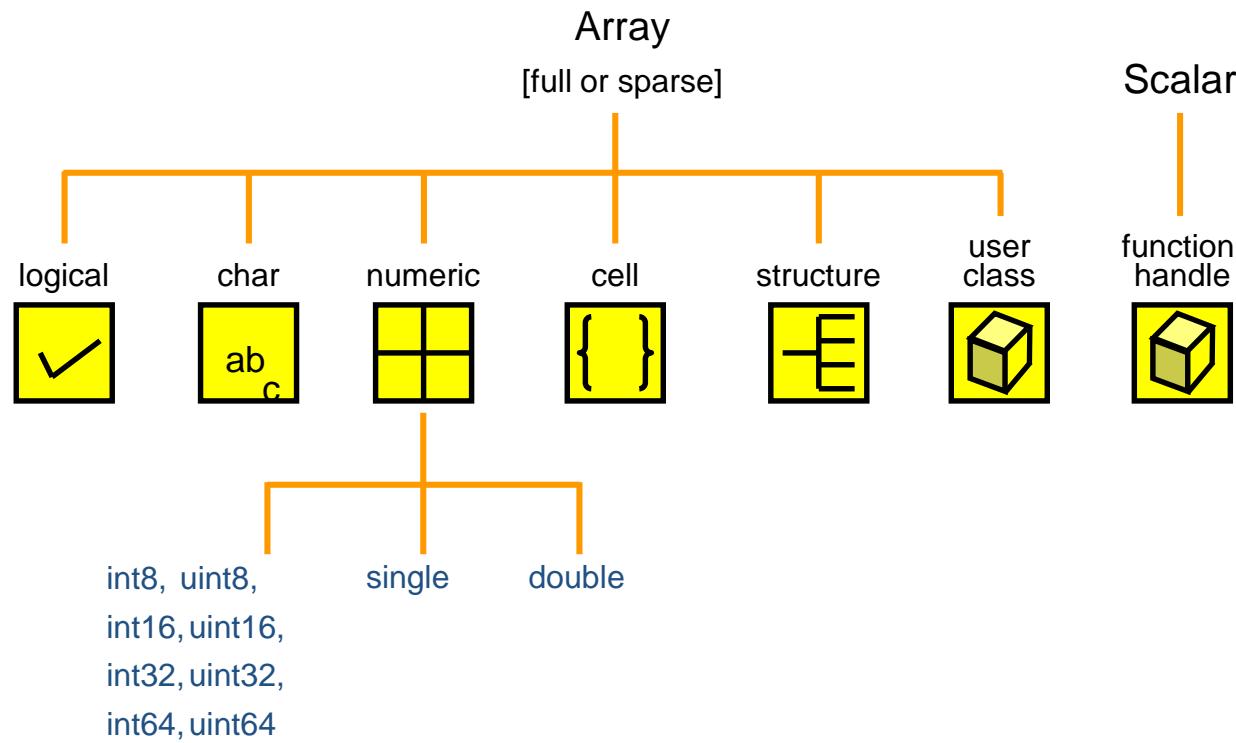
Have a good time

# Course Outline

- Working with the MATLAB User Interface
- Variables and Commands
- Array Creation and Analysis
- Working with Data Files
- Visualization with Array
- Automating Commands with Scripts
- Appendix: Data Type



# MATLAB® Data Types



# Integer Arrays



$$\begin{pmatrix} 1 & 2 \\ 3 & 4 \end{pmatrix}$$

**Construct**    `>> A = uint8([1, 2; 3, 4]);`

**Access**    `>> A(:,2)`  
`ans =`  
`2`  
`4`



## Nondouble Arithmetic

back

$$\begin{array}{|c|c|} \hline \text{single} & + & \text{single} \\ \hline \end{array} = \begin{array}{|c|c|} \hline \text{single} \\ \hline \end{array}$$

$$\begin{array}{c} \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \\ \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \end{array} + \begin{array}{c} \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \\ \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \end{array} = \begin{array}{c} \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \\ \boxed{\textcolor{blue}{\square}} \quad \boxed{\textcolor{blue}{\square}} \end{array}$$

double              single              single

$$\begin{array}{|c|c|} \hline \text{int8} & + & \text{double} & = & \begin{array}{|c|c|} \hline \text{int8} & \\ \hline \end{array} \end{array}$$

$$\begin{array}{|c|c|} \hline \text{int8} & + & \begin{array}{|c|c|} \hline & \\ \hline & \\ \hline \end{array} & = & \begin{array}{|c|c|} \hline & \\ \hline & \\ \hline \end{array} \\ \hline \end{array}$$

$$\begin{array}{|c|c|} \hline \text{int8} & + & \text{int8} & = & \text{int8} \\ \hline \end{array}$$

$$\begin{array}{|c|c|} \hline \text{int8} & \text{int16} \end{array} + \begin{array}{|c|c|} \hline \text{int8} & \text{int16} \end{array} = \begin{array}{|c|c|} \hline \text{int8} & \text{int16} \end{array}$$

# Characters and Strings

back

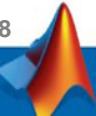
```
>> y = x      ——————> variable  
>> y = 'x'    ——————> character
```

```
>> MarkA = 'Friends, Romans, countrymen, lend me your ears';
```

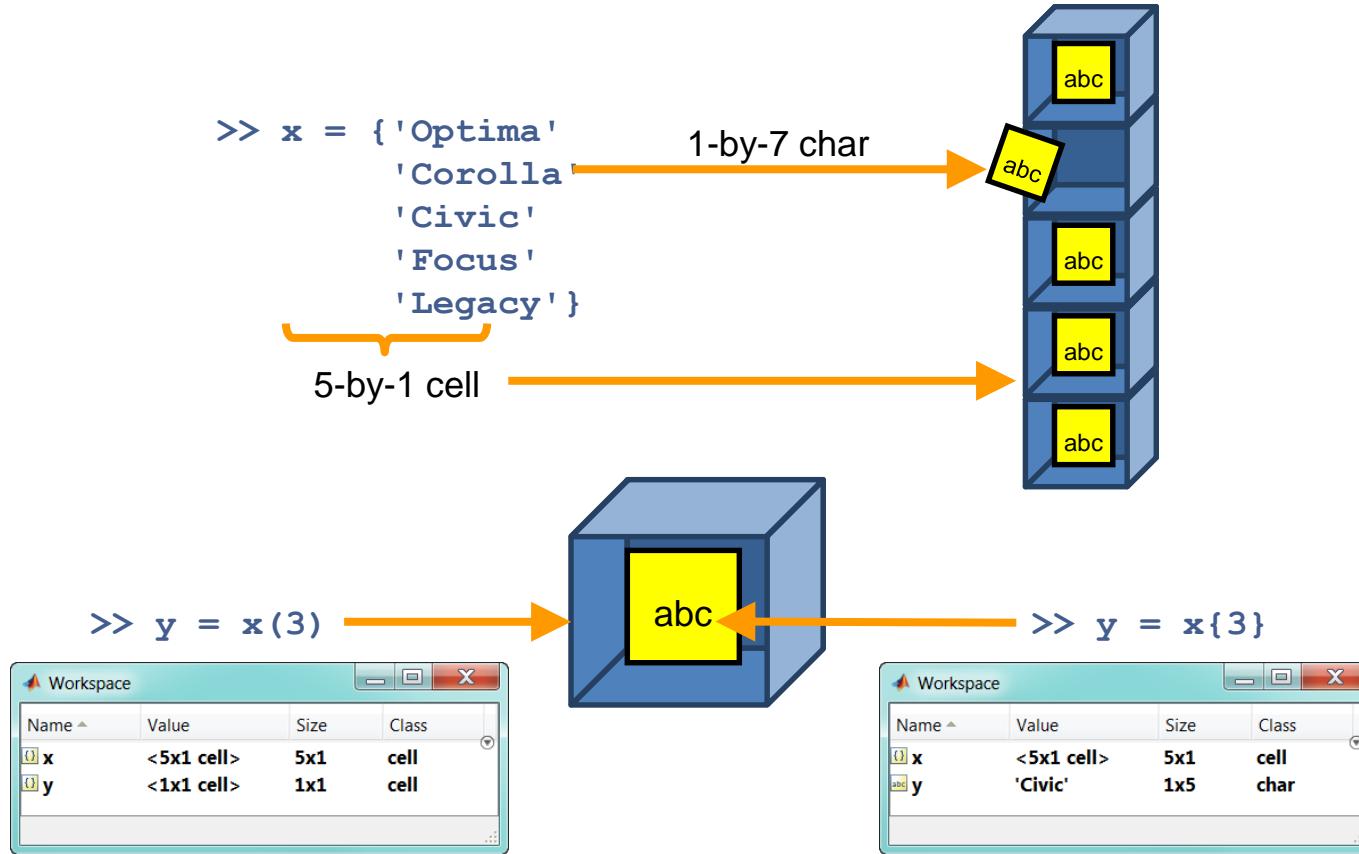
1-by-46 char array

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
'F'	'r'	'i'	'e'	'n'	'd'	's'	', '	' '	'R'	'o'	'm'	'a'	'n'	's'	', '	' '	'c'

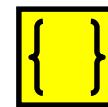
```
>> FriendNationality = MarkA(10:15)
```



# Cell Arrays (1)



## Cell Arrays (2)



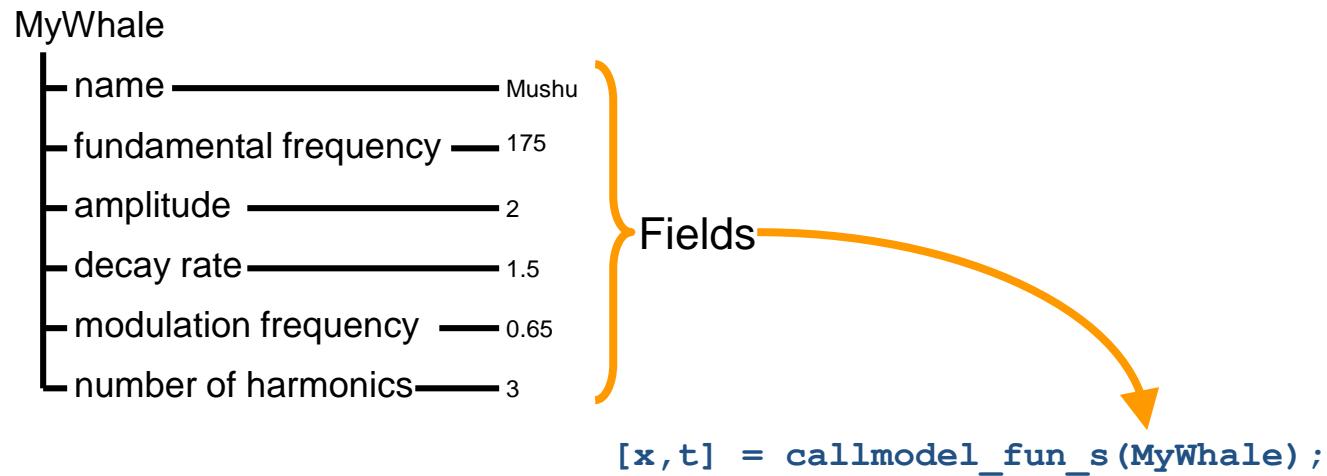
3.14		foo
bar	$\begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$	2
	B	

**Construct**   `>> A = {pi,[],'foo';...  
'bar',eye(2),2;...  
[],B,[]};`

**Access**   `>> A{2,2}(:,2)  
ans =  
0  
1`



# Structures

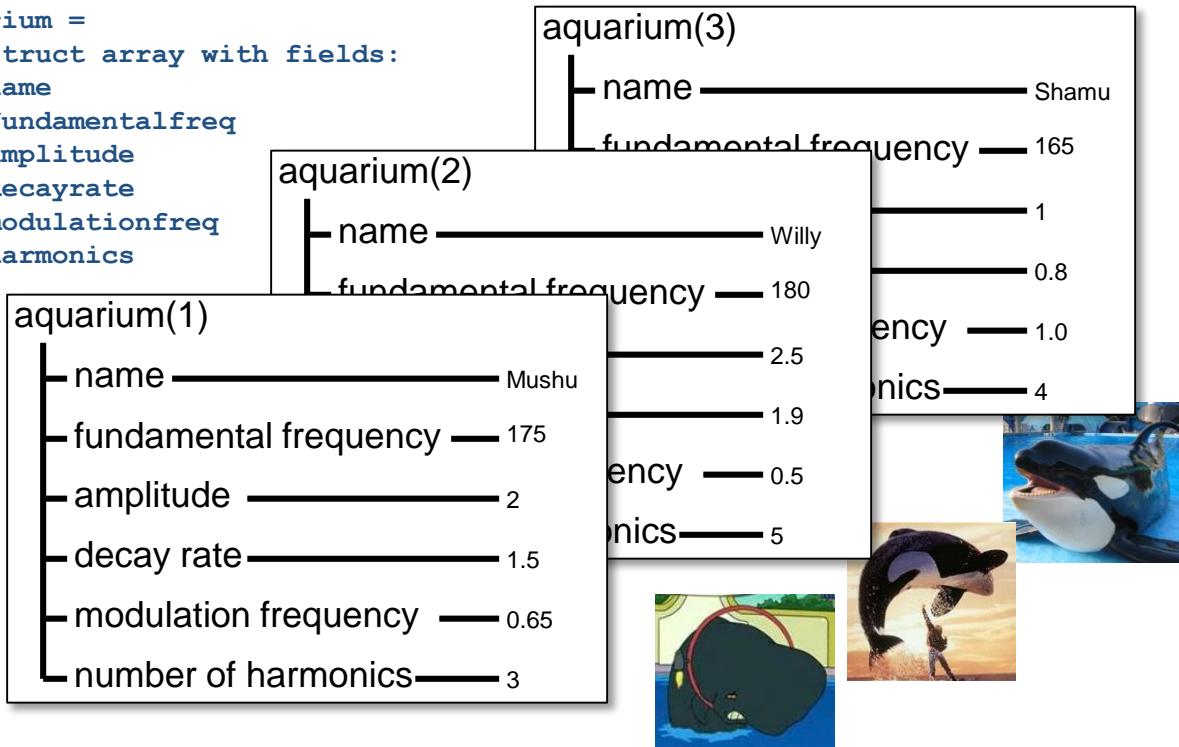


```
>> edit  
whalestruct
```

# Structure Arrays

```
>> aquarium
```

```
aquarium =  
1x3 struct array with fields:  
  name  
  fundamentalfreq  
  amplitude  
  decayrate  
  modulationfreq  
  harmonics
```



```
>> ex_aquarium
```



# Indexing into Structure Arrays

