MathWorks MATLAB&SIMULINK

Introduction to MATLAB® Tutorial

MATLAB® is a high-level language and interactive environment for numerical computation, visualization, and programming. Using MATLAB®, you can analyze data, develop algorithms, and create models and applications. The language, tools, and built-in math functions enable you to explore multiple approaches and reach a solution faster than with spreadsheets or traditional programming languages, such as C/C++ or Java™.

Installing and Activating MATLAB®

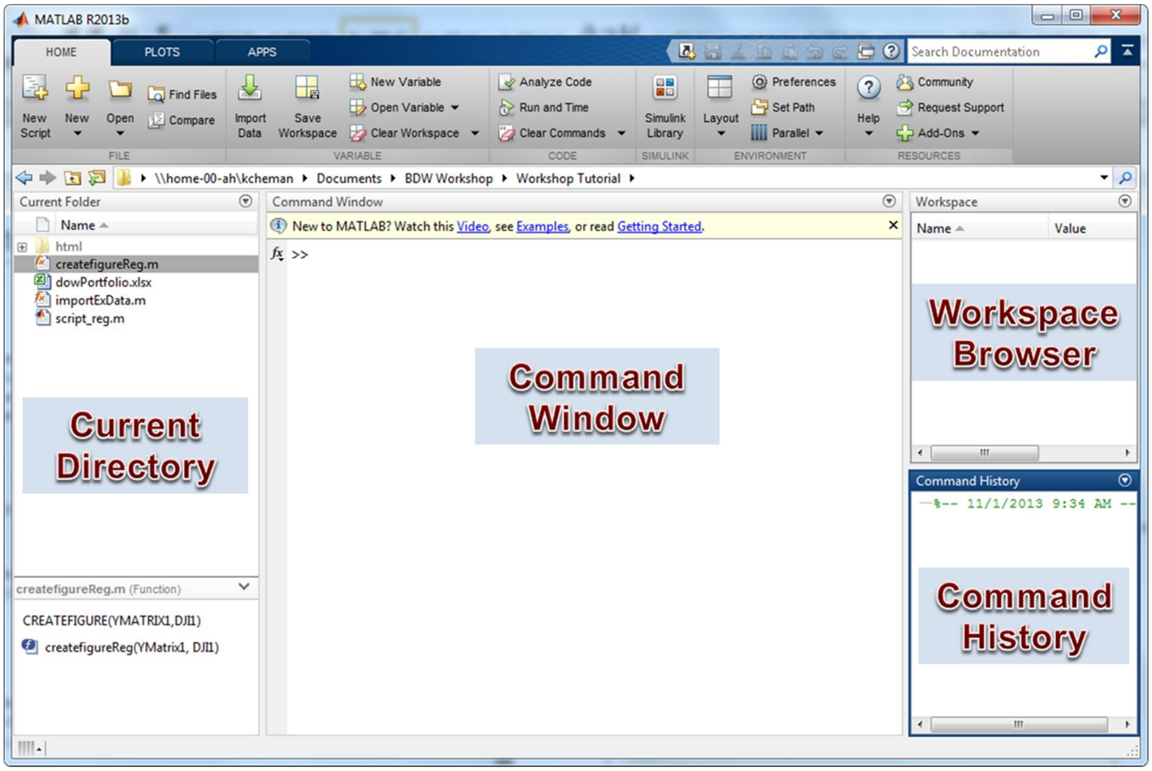
Follow the instructions that you received by email or during the Workshop to install and activate MATLAB®. You will now be able to launch MATLAB® through November 15, 2013. You have access to the following products:

- MATLAB® R2013b
- Statistics Toolbox
- Neural Network Toolbox
- Curve Fitting Toolbox
- Optimization Toolbox

User Interface

When you start MATLAB®, the desktop appears in its default layout. The desktop includes these panels:

- Current Directory – Access your files.
- Command Window – Enter commands at the command line, indicated by the prompt: >>.
- Workspace Browser – Explore data that you create or import from files.
- Command History – View or rerun commands that you entered at the command line.



New to MATLAB? Watch this Video, see Examples, or read Getting Started.

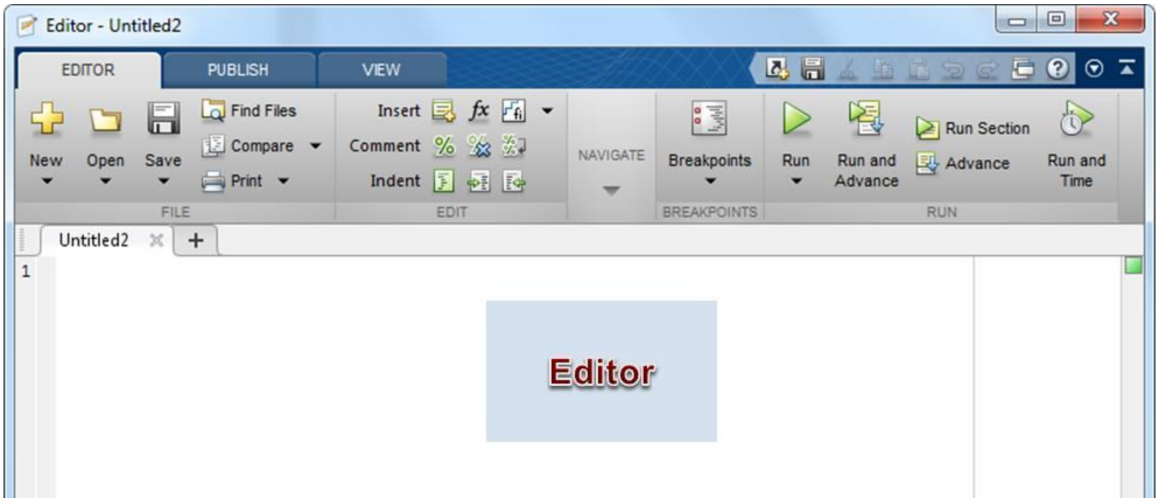
```
fx >>
```

```
%-- 11/1/2013 9:34 AM --
```

Other features of the desktop include:

- Toolstrip containing the Home, Plots and Apps tabs; each tab groups functionality associated with common tasks. Additional tabs, such as Editor, Publish and Variable, appear in the Toolstrip as needed to support your workflow.
- Quick access toolbar displays frequently used options such as cut, copy, paste and help. You can customize the options available in the quick access toolbar to suit your typical workflow.
- Search Documentation box allows you to search the documentation.

The MATLAB® Editor allows you to create a new file (e.g., script or function) in MATLAB®. You can open a new, untitled file by (1) typing `edit` at the command line or (2) using the New button from the Toolstrip.

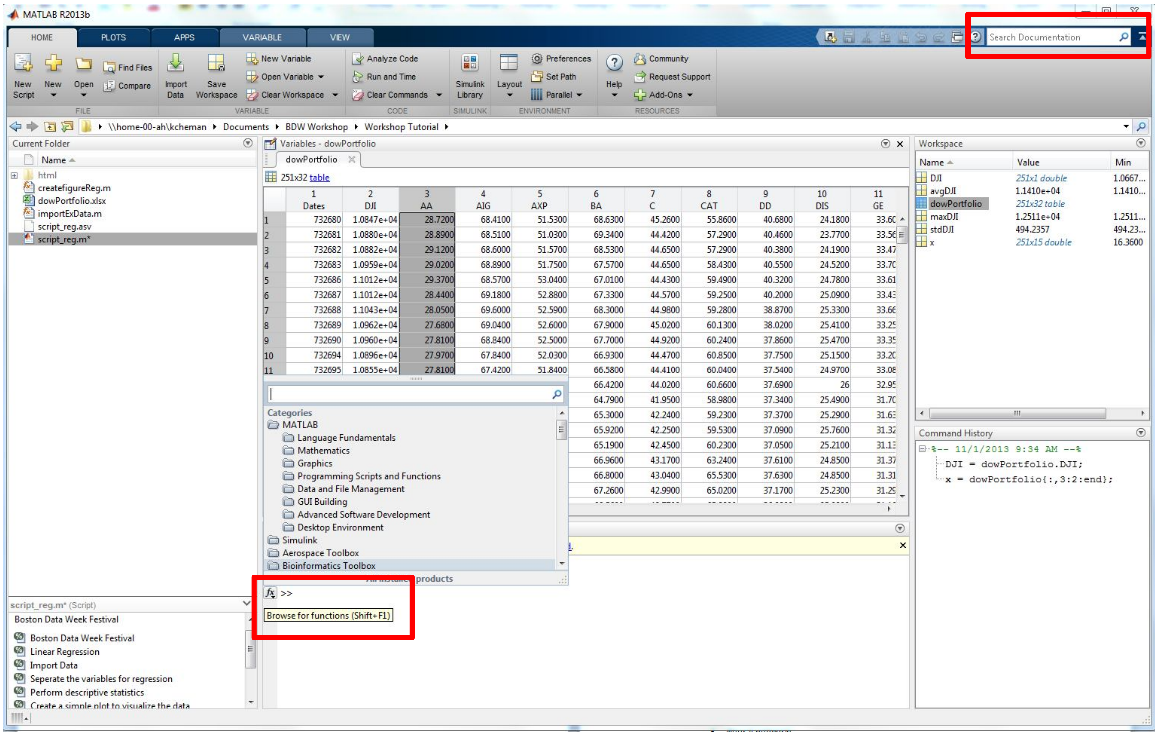


Documentation: http://www.mathworks.com/help/matlab/learn_matlab/desktop.html

Getting Help

There are several ways to get MATLAB® help:

- Click the Function Browser in the Command Window. (See graphic below.)
- Search Documentation in the top right of the user interface.

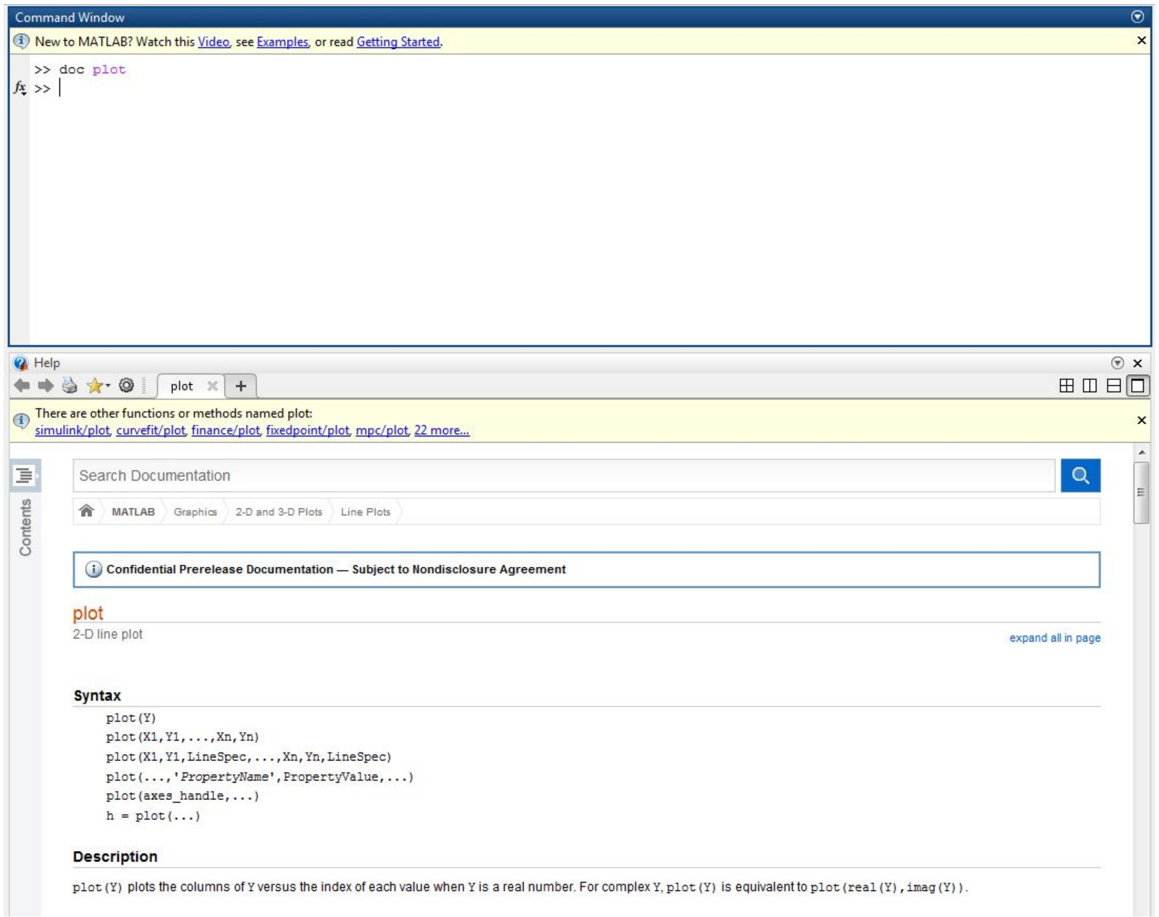


	1	2	3	4	5	6	7	8	9	10	11
	Dates	DJI	AA	AIG	AXP	BA	C	CAT	DD	DIS	GE
1	732680	1.0847e+04	28.7200	68.4100	51.5300	68.6300	45.2600	55.8600	40.6800	24.1800	33.60
2	732681	1.0880e+04	28.8900	68.5100	51.0300	69.3400	44.4200	57.2900	40.4600	23.7700	33.56
3	732682	1.0882e+04	29.1200	68.6000	51.5700	68.5300	44.6500	57.2900	40.3800	24.1900	33.47
4	732683	1.0959e+04	29.0200	68.8900	51.7500	67.5700	44.6500	58.4300	40.5500	24.5200	33.70
5	732686	1.1012e+04	29.3700	68.5700	53.0400	67.0100	44.4300	59.4900	40.3200	24.7800	33.61
6	732687	1.1012e+04	28.4400	69.1800	52.8800	67.3300	44.5700	59.2500	40.2000	25.0900	33.43
7	732688	1.1043e+04	28.0500	69.6000	52.5900	68.3000	44.9800	59.2800	38.8700	25.3300	33.66
8	732689	1.0962e+04	27.6800	69.0400	52.6000	67.9000	45.0200	60.1300	38.0200	25.4100	33.25
9	732690	1.0960e+04	27.8100	68.8400	52.5000	67.7000	44.9200	60.2400	37.8600	25.4700	33.35
10	732694	1.0896e+04	27.9700	67.8400	52.0300	66.9300	44.4700	60.8500	37.7500	25.1500	33.20
11	732695	1.0855e+04	27.8100	67.4200	51.8400	66.5800	44.4100	60.0400	37.5400	24.9700	33.08
						66.4200	44.0200	60.6600	37.6900	26	32.95
						64.7900	41.9500	58.9800	37.3400	25.4900	31.70
						65.3000	42.2400	59.2300	37.3700	25.2900	31.63
						65.9200	42.2500	59.5300	37.0900	25.7600	31.32
						65.1900	42.4500	60.2300	37.0500	25.2100	31.13
						66.9600	43.1700	63.2400	37.6100	24.8500	31.37
						66.8000	43.0400	65.5300	37.6300	24.8500	31.31
						67.2600	42.9900	65.0200	37.1700	25.2300	31.29

Name	Value	Min
DJI	251x1 double	1.0667...
avgDJI	1.1410e+04	1.1410...
dowPortfolio	251x32 table	
maxDJI	1.2511e+04	1.2511...
stdDJI	494.2357	494.23...
x	251x15 double	16.3600

```
%-- 11/1/2013 9:34 AM --%
  DJI = dowPortfolio.DJI;
  x = dowPortfolio{:,3:2:end};
```

- Search the Documentation from the command line using `doc`.



New to MATLAB? Watch this Video, see Examples, or read Getting Started.

```
>> doc plot
>> |
```

There are other functions or methods named plot:
simulink/plot, curvefit/plot, finance/plot, fixedpoint/plot, mpc/plot, 22 more...

Search Documentation

MATLAB > Graphics > 2-D and 3-D Plots > Line Plots

Confidential Prerelease Documentation — Subject to Nondisclosure Agreement

plot
2-D line plot

expand all in page

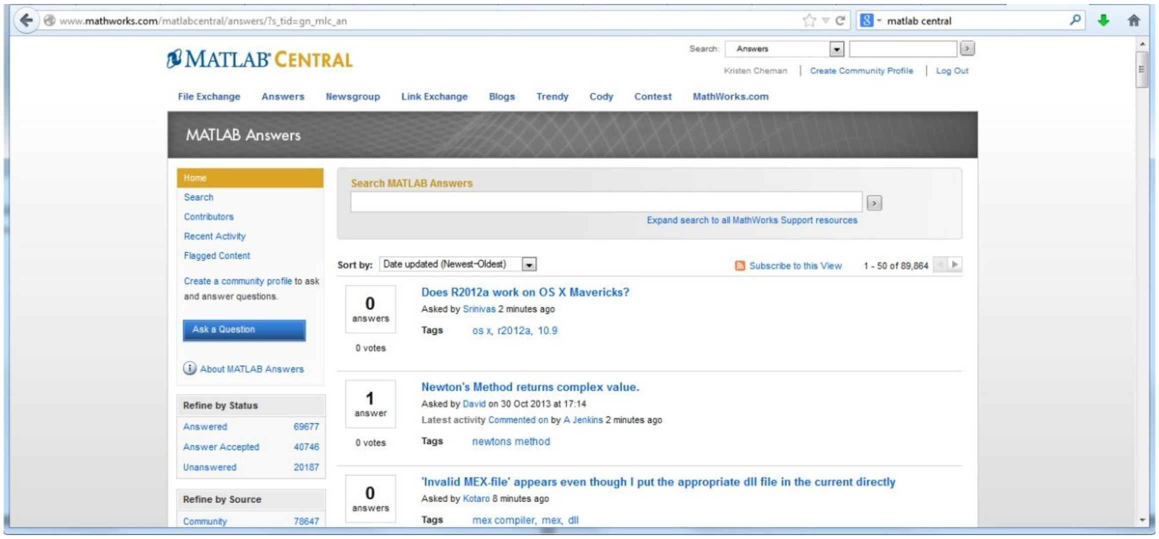
Syntax

```
plot(Y)
plot(X1,Y1,...,Xn,Yn)
plot(X1,Y1,LineSpec,...,Xn,Yn,LineSpec)
plot(...,'PropertyName',PropertyValue,...)
plot(axes_handle,...)
h = plot(...)
```

Description

`plot(Y)` plots the columns of `Y` versus the index of each value when `Y` is a real number. For complex `Y`, `plot(Y)` is equivalent to `plot(real(Y),imag(Y))`.

- Visit www.mathworks.com -> Support and search Documentation.
- Post a question to MATLAB® Central.



Create a community profile to ask and answer questions.

Refine by Status

Answered	69677
Answer Accepted	40746
Unanswered	20187

Refine by Source

Community	78647

Search MATLAB Answers

Expand search to all MathWorks Support resources

Sort by: Date updated (Newest-Oldest) Subscribe to this View 1 - 50 of 89,864

0 answers, 0 votes
Does R2012a work on OS X Mavericks?
Asked by Srinivas 2 minutes ago
Tags os x, r2012a, 10.9

1 answer, 0 votes
Newton's Method returns complex value.
Asked by David on 30 Oct 2013 at 17:14
Latest activity Commented on by A Jenkins 2 minutes ago
Tags newtons method

0 answers
'Invalid MEX-file' appears even though I put the appropriate dll file in the current directly
Asked by Kotaro 8 minutes ago
Tags mex compiler, mex, dll

MATLAB Warm-up

Manipulating Data

You can define variables at the command line. Try:

```
a = 3;

b = 4
```

Notice the difference when you remove the semicolon at the end of a line.

MATLAB® has several built in functions to easily create data: `rand`, `zeros`, `ones`, `magic`, and `diag` are examples.

Create a 4-by-3 matrix of random numbers:

```
X = rand(4,3)
```

You notice that `( )` are used for indexing, order of operations, and passing arguments to a function. `[ ]` are used for concatenation and defining multiple outputs.

For example, you can define the following vectors:

```
vec1 = [1:5]

vec2 = [0:5:25]

vec3 = [25:-3:2]
```

Create a vector of every other value from 3 to 10:

```
vec = [3:2:10]
```

There are three types of indexing that you can use to extract data from an array:

1. Extract a single element.

   Try extracting the third element in vector `vec1`:

   ```
   third = vec1(3)
   ```

   Try extracting the elements at locations (1,2) and (2,3) from matrix X:

   ```
   loc1 = X(1,2)
   loc2 = X(2,3)
   ```

2. Extract multiple elements.

Try extracting the first and fourth elements in vector `v` and call it `v1`. Then extract the third through last elements vector `v` and call it `v2`:

```
v1 = vec1([1 4])
v2 = vec1(3:end)
```

Try extracting the third row from matrix `X` and call it `m1`. Then extract the first two rows, but only the first and second column in those rows; call it `m2`.

```
m1 = X(3,:)
m2 = X(1:2,1:2)
```

3. Logical indexing. This allows you to use a single, logical array for the matrix subscript.

   Try the following:

   ```
   newX = X(X>.5)
   ```

MATLAB® allows you to easily perform matrix and element-wise operations and scalar expansion. For example, if we define two matrices:

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

B = [5 6; 7 8]
```

Then:

```
C = A*B

D = A.*B

E = A.^3 + B^2
```

Other useful operations are: `^` (power), `\` (left divide), and `'` (transpose).

Try:

```
A+1
```

Try:

```
A+[1 2]
```

Data Analysis

You may have interest in descriptive statistics about your dataset. Both MATLAB® and the Statistics Toolbox offer functions to assist you.

Try the following:

1. Generate a 10-by-10 array of random numbers between 1 and 10.

   ```
   randarray = rand(10,10)*10
   ```

2. Find the maximum value of each column.

   ```
   maxval = max(randarray)
   ```

3. Calculate the mean of each column.

   ```
   meanval = mean(randarray)
   ```

4. Calculate the standard deviation of each column.

   ```
   sigma = std(randarray)
   ```

Documentation: http://www.mathworks.com/help/matlab/descriptive-statistics.html

Scripts and Functions

Program files can be *scripts* that simply execute a series of MATLAB® statements, or they can be *functions* that also accept input arguments and produce output. Both scripts and functions contain MATLAB® code, and both are stored in text files with a `.m` extension.

Try the following:

1. Create a new script that computes the area of a triangle. In the script type:

   ```
   b = 5;
   h = 3;
   a = 0.5*(b.* h)
   ```

2. Save the file as `triarea.m`.
3. Call the script from the command line by typing: `triarea`.

To calculate the area of a triangle with different dimensions, you would need to update the values of b and h in the script and rerun it.

Instead you could convert the script into a function.

The function calling syntax looks like:

```
[out1, out2, ..., outN] = functionname(in1, in2, ..., inN)
```

1. Revise the file to look like a function:

   ```
   function a = triarea(b,h)
   a = 0.5*(b.* h);
   ```

2. Save the file and then call the function with different base and height values from the command line without modifying the script. Try:

   ```
   a1 = triarea(1,5)
   ```

```
        a2 = triarea(2,10)
        a3 = triarea(3,6)
```

Logic

Once data is loaded, you may want to perform operations on it:

Conditional Control Statements: `if`, `elseif`, and `else`

The `if`, `elseif` , and `else` commands evaluate an expression and execute a group of statements when the expression is true. The syntax is:

```
if expression
   statements
          :
elseif expression
   statements
          :
else
   statements
          :
end
```

Try creating a conditional statement to check if a number is even or odd:

```
x = 10;
if mod(x,2) == 0
     disp('x is even')
else
     disp('x is odd')
end
```

Documentation: http://www.mathworks.com/help/matlab/ref/if.html

For Loops

For loops allow you to execute statements a specified number of times. The syntax is:

```
for index = values
   statements
          :
end
```

Try creating a for loop to perform 10! (factorial):

```
f = 1;
for ii = 1:10
   f = f*ii;
end
```

Type `f` on the command line to see the result. Then, use MATLAB's built in `factorial` to check your work.

```
    factorial(10)
```

Documentation: http://www.mathworks.com/help/matlab/ref/for.html

Problem #1: Linear Regression Models

Introduction

Regression analysis is a statistical technique for estimating the relationships among variables. Independent variables (often denoted by X) are also called explanatory or predictor variables and the dependent variable (often called y) is the response variable.

Problem Definition

The Dow Jones Index (DJI) is used as a benchmark of market performance. The Dow Jones Index (response variable) is a weighted average of 30 big named stocks or components (predictor variables). The problem is to perform multiple linear regression on a subset of predictors (15 out of 30) to see how well you can predict the response.

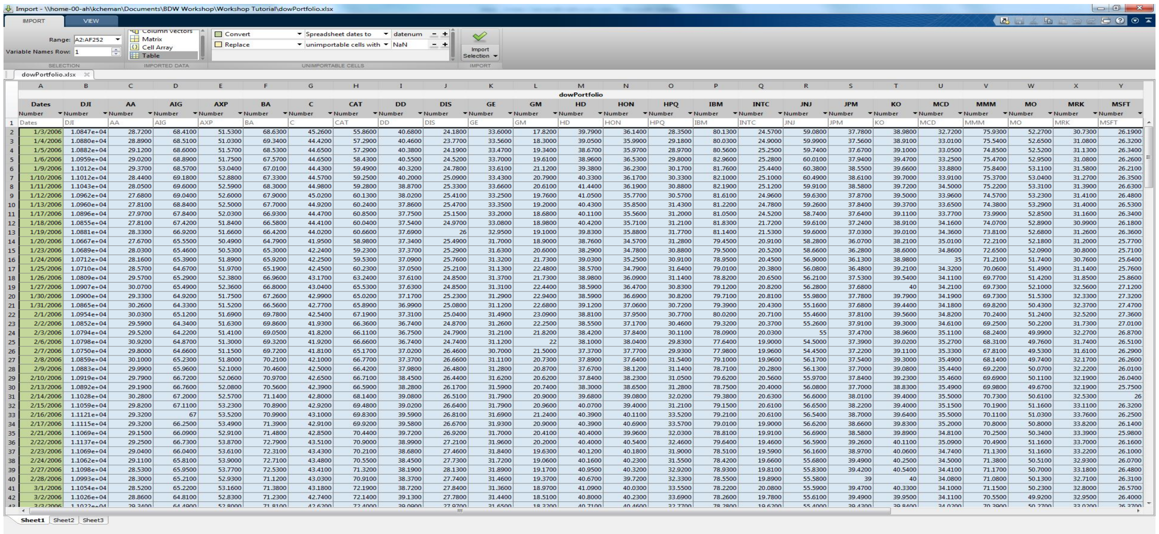
Dataset: `dowPortfolio.csv`

This dataset includes (1) trading days between January 3 and December 29, 2006 in the first column, (2) the DJI for each date in column 2, and (3) each component of the DJI in columns 3 through 32. (The Dow Jones Index is a weighted average of columns 3 through 32.)

Importing Data

With the User Interface

1. Right- or double-click on the filename in the Current Folder window to launch the Import wizard. The wizard will display the file in a spreadsheet-like format.
2. Select the range to import, data type, and import rules. You can use the defaults, but change the data type to Table.



Import - \\home-00-ah\kcheman\Documents\BDW Workshop\Workshop Tutorial\dowPortfolio.xlsx

Range: A2:AF252
Variable Names Row: 1

Convert Spreadsheet dates to datenum
Replace unimportable cells with NaN

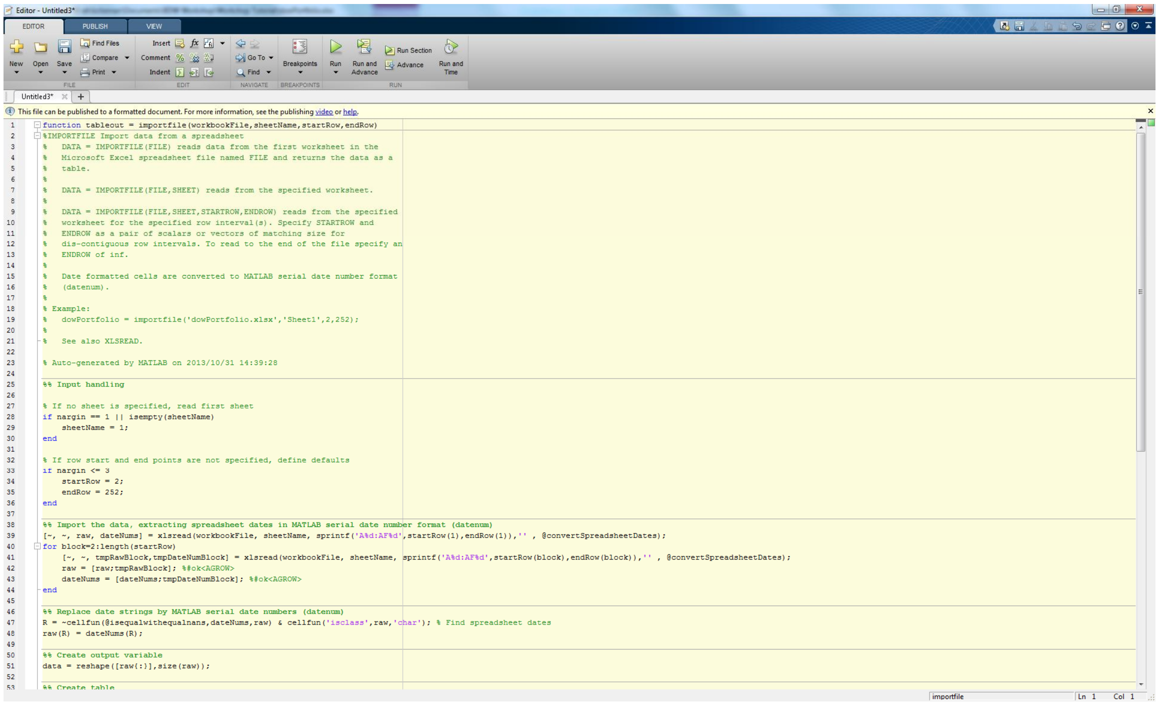
dowPortfolio

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y
	Dates	DJI	AA	AIG	AXP	BA	C	CAT	DD	DIS	GE	GM	HD	HON	HPQ	IBM	INTC	JNJ	JPM	KO	MCD	MMM	MO	MRK	MSFT
	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number	Number
1	Dates	DJI	AA	AIG	AXP	BA	C	CAT	DD	DIS	GE	GM	HD	HON	HPQ	IBM	INTC	JNJ	JPM	KO	MCD	MMM	MO	MRK	MSFT
2	1/3/2006	1.0847e+04	28.7200	68.4100	51.5300	68.6300	45.2600	55.8600	40.6800	24.1800	33.6000	17.8200	39.7900	36.1400	28.3500	80.1300	24.5700	59.0800	37.7800	38.9800	32.7200	75.9300	52.2700	30.7300	26.1900
3	1/4/2006	1.0880e+04	28.8900	68.5100	51.0300	69.3400	44.4200	57.2900	40.4600	23.7700	33.5600	18.3000	39.0500	35.9900	29.1800	80.0300	24.9000	59.9900	37.5600	38.9100	33.0100	75.5400	52.6500	31.0800	26.3200
4	1/5/2006	1.0882e+04	29.1200	68.6000	51.5700	68.5300	44.6500	57.2900	40.3800	24.1900	33.4700	19.3400	38.6700	35.9700	28.9700	80.5600	25.2500	59.7400	37.6700	39.1000	33.0500	74.8500	52.5200	31.1300	26.3400
5	1/6/2006	1.0959e+04	29.0200	68.8900	51.7500	67.5700	44.6500	58.4300	40.5500	24.5200	33.7000	19.6100	38.9600	36.5300	29.8000	82.9600	25.2800	60.0100	37.9400	39.4700	33.2500	75.4700	52.9500	31.0800	26.2600
6	1/9/2006	1.1012e+04	29.3700	68.5700	53.0400	67.0100	44.4300	59.4900	40.3200	24.7800	33.6100	21.1200	39.3800	36.2300	30.1700	81.7600	25.4400	60.3800	38.5500	39.6600	33.8800	75.8400	53.1100	31.5800	26.2100
7	1/10/2006	1.1012e+04	28.4400	69.1800	52.8800	67.3300	44.5700	59.2500	40.2000	25.0900	33.4300	20.7900	40.3300	36.1700	30.3300	82.1000	25.1000	60.4900	38.6100	39.7000	33.9100	75.3700	53.0400	31.2700	26.3500
8	1/11/2006	1.1043e+04	28.0500	69.6000	52.5900	68.3000	44.9800	59.2800	38.8700	25.3300	33.6600	20.6100	41.4400	36.1900	30.8800	82.1900	25.1200	59.9100	38.5800	39.7200	34.5000	75.2200	53.3100	31.3900	26.6300
9	1/12/2006	1.0962e+04	27.6800	69.0400	52.6000	67.9000	45.0200	60.1300	38.0200	25.4100	33.2500	19.7600	41.0500	35.7700	30.5700	81.6100	24.9600	59.6300	37.8700	39.5000	33.9600	74.5700	53.2300	31.4100	26.4800
10	1/13/2006	1.0960e+04	27.8100	68.8400	52.5000	67.7000	44.9200	60.2400	37.8600	25.4700	33.3500	19.2000	40.4300	35.8500	31.4300	81.2200	24.7800	59.2600	37.8400	39.3700	33.6500	74.3800	53.2900	31.4000	26.5300
11	1/17/2006	1.0896e+04	27.9700	67.8400	52.0300	66.9300	44.4700	60.8500	37.7500	25.1500	33.2000	18.6800	40.1100	35.5600	31.2000	81.0500	24.5200	58.7400	37.6400	39.1100	33.7700	73.9900	52.8500	31.1600	26.3400
12	1/18/2006	1.0855e+04	27.8100	67.4200	51.8400	66.5800	44.4100	60.0400	37.5400	24.9700	33.0800	18.9800	40.4200	35.7100	31.2100	81.8300	21.7200	59.6100	37.2400	38.9100	34.1600	74.0700	52.8900	30.9900	26.1800
13	1/19/2006	1.0881e+04	28.3300	66.9200	51.6600	66.4200	44.0200	60.6600	37.6900	26	32.9500	19.1000	39.8300	35.8800	31.7700	81.1400	21.5300	59.6000	37.0300	39.0100	34.3600	73.8100	52.6800	31.2600	26.3600
14	1/20/2006	1.0667e+04	27.6700	65.5500	50.4900	64.7900	41.9500	58.9800	37.3400	25.4900	31.7000	18.9000	38.7600	34.5700	31.2800	79.4500	20.9100	58.2800	36.0700	38.2100	35.0100	72.2100	52.1800	31.2000	25.7700
15	1/23/2006	1.0689e+04	28.0300	65.4600	50.5300	65.3000	42.2400	59.2300	37.3700	25.2900	31.6300	20.6000	38.2900	34.7800	30.8800	79.5000	20.5200	58.6600	36.2800	38.6000	34.8600	72.6500	52.0900	30.8000	25.7100
16	1/24/2006	1.0712e+04	28.1600	65.3900	51.8900	65.9200	42.2500	59.5300	37.0900	25.7600	31.3200	21.7300	39.0300	35.2500	30.9100	78.9500	20.4500	56.9000	36.1300	38.9800	35	71.2100	51.7400	30.7600	25.6400
17	1/25/2006	1.0710e+04	28.5700	64.6700	51.9700	65.1900	42.4500	60.2300	37.0500	25.2100	31.1300	22.4800	38.5700	34.7900	31.6400	79.0100	20.3800	56.0800	36.4800	39.2100	34.3200	70.0600	51.4900	31.1400	25.7600
18	1/26/2006	1.0809e+04	29.5700	65.2900	52.3800	66.9600	43.1700	63.2400	37.6100	24.8500	31.3700	21.7300	38.9800	36.0900	31.1400	78.8200	20.6500	56.2100	37.5300	39.5400	34.1100	69.7700	51.4200	31.8500	25.8600
19	1/27/2006	1.0907e+04	30.0700	65.4900	52.3600	66.8000	43.0400	65.5300	37.6300	24.8500	31.3100	22.4400	38.5900	36.4700	30.8300	79.1200	20.8200	56.2800	37.6800	40	34.2100	69.7300	52.1000	32.5600	27.1200
20	1/30/2006	1.0900e+04	29.3300	64.9200	51.7500	67.2600	42.9900	65.0200	37.1700	25.2300	31.2900	22.9400	38.5900	36.6900	30.8200	79.7100	20.8100	55.9800	37.7800	39.7900	34.1900	69.7300	51.5300	32.3300	27.3200
21	1/31/2006	1.0865e+04	30.2600	64.3300	51.5200	66.5600	42.7700	65.8900	36.9900	25.0800	31.1200	22.6800	39.1200	37.0600	30.7200	79.3900	20.4300	55.1600	37.6800	39.4400	34.1800	69.8200	50.4300	32.3700	27.4700
22	2/1/2006	1.0954e+04	30.0300	65.1200	51.6900	69.7800	42.5400	67.1900	37.3100	25.0400	31.4900	23.0900	38.8100	37.9500	30.7700	80.0200	20.7100	55.4600	37.8100	39.5600	34.8200	70.2400	51.2400	32.5200	27.3600
23	2/2/2006	1.0852e+04	29.5900	64.3400	51.6300	69.8600	41.9300	66.3600	36.7400	24.8700	31.2600	22.2500	38.5500	37.1700	30.4600	79.3200	20.3700	55.2600	37.9100	39.3000	34.6100	69.2500	50.2200	31.7300	27.0100
24	2/3/2006	1.0794e+04	29.5200	64.2200	51.4100	69.0500	41.8200	66.1100	36.7500	24.7900	31.2100	21.8200	38.4200	37.8400	30.1100	78.0900	20.0300	55	37.4700	38.9600	35.1100	68.2400	49.9900	32.2700	26.8700
25	2/6/2006	1.0798e+04	30.9200	64.8700	51.3000	69.3200	41.9200	66.6600	36.7400	24.7400	31.1200	22	38.1000	38.0400	29.8300	77.6400	19.9000	54.5000	37.3900	39.0200	35.2700	68.3100	49.7600	31.7400	26.5100
26	2/7/2006	1.0750e+04	29.8000	64.6600	51.1500	69.7200	41.8100	65.1700	37.0200	26.4600	30.7000	21.5000	37.3700	37.7700	29.9300	77.9800	19.9600	54.4500	37.2200	39.1100	35.3300	67.8100	49.5300	31.6100	26.2900
27	2/8/2006	1.0859e+04	30.1000	65.2300	51.8000	70.2100	42.1000	66.7700	37.3700	26.6600	31.1100	20.7300	37.8900	37.6400	31.5400	79.1000	19.9600	56.1700	37.5400	39.3000	35.4900	68.1400	49.7400	32.1700	26.2600
28	2/9/2006	1.0883e+04	29.9900	65.9600	52.1000	70.4600	42.5000	66.4200	37.9800	26.4800	31.2800	20.8700	37.6700	38.1200	31.1400	78.7100	20.2800	56.1300	37.7000	39.0800	35.4400	69.2200	50.0700	32.2200	26.0100
29	2/10/2006	1.0919e+04	29.7900	66.7200	52.0600	70.9700	42.6500	66.7100	38.4500	26.4400	31.6200	20.6200	37.8400	38.2300	31.0500	79.6200	20.5600	55.9700	37.8400	39.2300	35.4600	69.6900	50.1100	32.1900	26.0400
30	2/13/2006	1.0892e+04	29.1900	66.7600	52.0800	70.5600	42.3900	66.5900	38.2800	26.1700	31.5900	20.7400	38.3000	38.6500	31.2800	78.7500	20.4000	56.0800	37.7000	38.8300	35.4900	69.9800	49.6700	32.1900	25.7500
31	2/14/2006	1.1028e+04	30.2800	67.2000	52.5700	71.1400	42.8000	68.1400	39.0800	26.5100	31.7900	20.9000	39.6800	39.0800	32.0200	79.3800	20.6300	56.6000	38.0100	39.4000	35.5000	70.7300	50.6100	32.5300	26
32	2/15/2006	1.1059e+04	29.8200	67.1100	53.2300	70.8900	42.9200	69.4800	39.0200	26.6400	31.7900	20.9600	40.0700	39.4000	31.2100	79.1500	20.6100	56.6500	38.2200	39.4000	35.1500	70.1900	51.1600	33.1100	26.3200
33	2/16/2006	1.1121e+04	29.3200	67	53.5200	70.9900	43.1000	69.8300	39.5900	26.8100	31.6900	21.2400	40.3900	40.1100	33.5200	79.2100	20.6100	56.5400	38.7000	39.6400	35.5000	70.1100	51.0300	33.7600	26.2500
34	2/17/2006	1.1115e+04	29.3200	66.2500	53.4900	71.3900	42.9100	69.9200	39.5800	26.6700	31.9300	20.9000	40.3900	40.6900	33.5700	79.0100	19.9000	56.6200	38.6600	39.8300	35.2000	70.8000	50.8000	33.8200	26.1400
35	2/21/2006	1.1069e+04	29.1500	66.0900	52.9100	71.4800	42.8500	70.4400	39.7200	26.9200	31.7000	20.4100	40.4000	39.9600	32.0300	78.8100	19.9100	56.6900	38.5800	39.8900	34.8100	70.2500	50.3400	33.3900	25.9800
36	2/22/2006	1.1137e+04	29.2500	66.7300	53.8700	72.7900	43.5100	70.9000	38.9900	27.2100	31.9600	20.2000	40.4000	40.5400	32.4600	79.6400	19.4600	56.5900	39.2600	40.1100	35.0900	70.4900	51.1600	33.7000	26.1600
37	2/23/2006	1.1069e+04	29.0400	66.0400	53.6100	72.3100	43.4300	70.2100	38.6800	27.4600	31.8400	19.6300	40.1200	40.1800	31.9000	78.5100	19.5900	56.1600	38.9700	40.0600	34.7400	71.1300	51.1600	33.2200	26.1000
38	2/24/2006	1.1062e+04	29.1100	65.8100	53.9000	72.7100	43.4800	70.5500	38.4500	27.7300	31.7200	19.0600	40.1600	40.2300	31.5500	78.4200	19.6600	55.6800	39.4900	40.2500	34.5000	71.3800	50.5100	32.9300	26.0700
39	2/27/2006	1.1098e+04	28.5300	65.9500	53.7700	72.5300	43.4100	71.3200	38.1900	28.1300	31.8900	19.1700	40.9500	40.3200	32.9200	78.9300	19.8100	55.8300	39.4200	40.5400	34.4100	71.1700	50.7000	33.1800	26.4800
40	2/28/2006	1.0993e+04	28.3000	65.2100	52.9300	71.1200	43.0300	70.9100	38.3700	27.7400	31.4600	19.3700	40.6700	39.7200	32.3300	78.5500	19.8900	55.5800	39	40	34.0800	71.0800	50.1300	32.7100	26.3100
41	3/1/2006	1.1054e+04	28.5200	65.2200	53.1600	71.3800	43.1800	72.1900	38.7200	27.8400	31.3600	18.9700	41.0900	40.0300	33.5500	78.2200	20.0800	55.5900	39.4700	40.3300	34.1000	71.1500	50.2300	32.8000	26.5700
42	3/2/2006	1.1026e+04	28.8600	64.8100	52.8300	71.2300	42.7400	72.1400	39.1300	27.7800	31.4400	18.5100	40.8000	40.2300	33.6900	78.2600	19.7800	55.6100	39.4900	39.9500	34.1100	70.5500	49.9200	32.9500	26.4000
43	3/3/2006	1.1022e+04	29.3400	64.4900	52.8000	71.8100	42.6200	72.4000	39.0900	27.9700	31.6500	18.3200	40.7100	40.4600	32.7700	78.2800	19.6200	55.4000	39.4300	39.8400	34.0200	70.3900	50.2700	33.0200	26.3700

3. Choose Generate Function. Save the function as `importExData.m`.

   The Import button drop-down menu allows you to auto-generate code for a script or function to perform the exact steps from the Import wizard. Everything done manually can be translated to the programmatic equivalent.

4. Press Import Selection to save the table as a variable called `dowPortfolio`.



Editor - Untitled3*

This file can be published to a formatted document. For more information, see the publishing video or help.

```matlab
function tableout = importfile(workbookFile,sheetName,startRow,endRow)
%IMPORTFILE Import data from a spreadsheet
%   DATA = IMPORTFILE(FILE) reads data from the first worksheet in the
%   Microsoft Excel spreadsheet file named FILE and returns the data as a
%   table.
%
%   DATA = IMPORTFILE(FILE,SHEET) reads from the specified worksheet.
%
%   DATA = IMPORTFILE(FILE,SHEET,STARTROW,ENDROW) reads from the specified
%   worksheet for the specified row interval(s). Specify STARTROW and
%   ENDROW as a pair of scalars or vectors of matching size for
%   dis-contiguous row intervals. To read to the end of the file specify an
%   ENDROW of inf.
%
%   Date formatted cells are converted to MATLAB serial date number format
%   (datenum).
%
% Example:
%   dowPortfolio = importfile('dowPortfolio.xlsx','Sheet1',2,252);
%
%   See also XLSREAD.

% Auto-generated by MATLAB on 2013/10/31 14:39:28

%% Input handling

% If no sheet is specified, read first sheet
if nargin == 1 || isempty(sheetName)
    sheetName = 1;
end

% If row start and end points are not specified, define defaults
if nargin <= 3
    startRow = 2;
    endRow = 252;
end

%% Import the data, extracting spreadsheet dates in MATLAB serial date number format (datenum)
[~, ~, raw, dateNums] = xlsread(workbookFile, sheetName, sprintf('A%d:AF%d',startRow(1),endRow(1)),'' , @convertSpreadsheetDates);
for block=2:length(startRow)
    [~, ~, tmpRawBlock,tmpDateNumBlock] = xlsread(workbookFile, sheetName, sprintf('A%d:AF%d',startRow(block),endRow(block)),'' , @convertSpreadsheetDates);
    raw = [raw;tmpRawBlock]; %#ok<AGROW>
    dateNums = [dateNums;tmpDateNumBlock]; %#ok<AGROW>
end

%% Replace date strings by MATLAB serial date numbers (datenum)
R = ~cellfun(@isequalwithequalnans,dateNums,raw) & cellfun('isclass',raw,'char'); % Find spreadsheet dates
raw(R) = dateNums(R);

%% Create output variable
data = reshape([raw{:}],size(raw));

%% Create table
```

From the Command Line

You can also import from the command line, using the `readtable` command:

```
readtable('dowPortfolio.csv');
```

Supported File Formats

You can import a long list of file formats using MATLAB®, including text, spreadsheet, and XML.

Documentation: http://www.mathworks.com/help/matlab/import_export/supported-file-formats.html

Useful Data Types

Tables

In this example, you imported your data as a `table` data type. Tables serve as an excellent container for tabular data of dissimilar types and allow you to import data of different sizes and data types. In this case, all fields are numeric and the date is converted to a `datenum` type.

Documentation: http://www.mathworks.com/help/matlab/tables.html

Categorical Arrays

Also useful are categorical arrays, which allow you to store arrays of qualitative data with values from a finite set of discrete, nonnumeric data.

Documentation: http://www.mathworks.com/help/matlab/categorical-arrays.html

Create New Variables from `table`

To create new variables from the table:

1. Double-click on the `dowPortfolio` variable in the Workspace to open and view the table.
2. Right-click on column 2 containing the DJI data.
3. In the drop down menu, select New Workspace Variable from Selection -> Separate Workspace Variable(s).

   A new variable `DJI` appears in the Workspace and MATLAB® echoes the commands in the Command History:

   ```
   DJI = dowPortfolio.DJI;
   ```

   This variable `DJI` represents your response variable.

4. From the command line, create a new variable `x` from the variable `dowPortfolio`, which combines only 15 components of the DJI—say every other column starting with column 3.

   ```
   x = dowPortfolio{:,3:2:end};
   ```

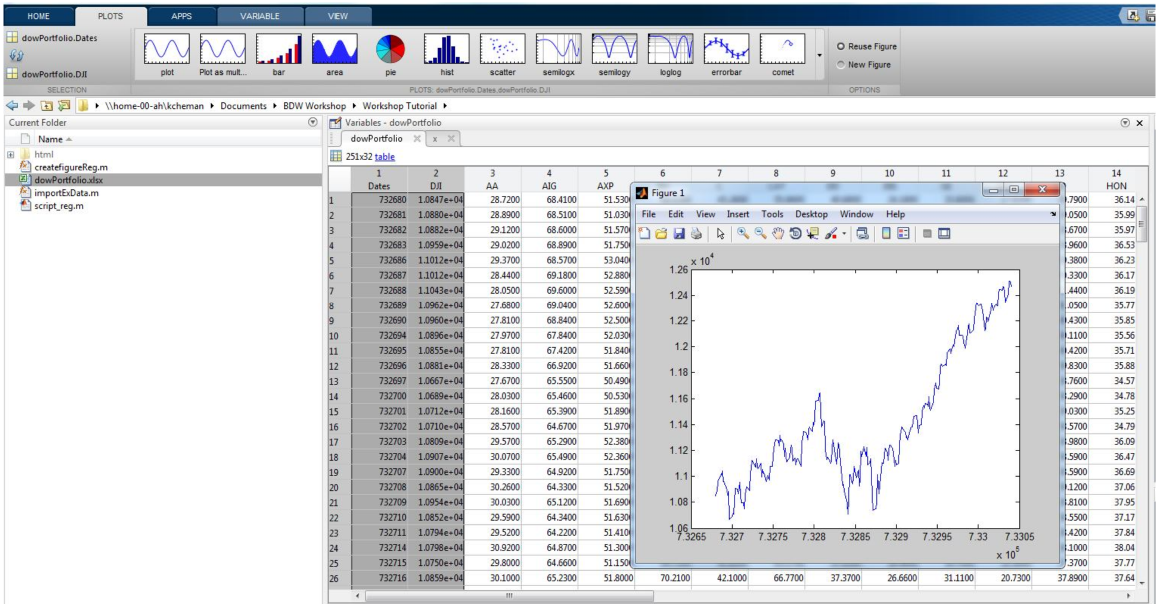
   This variable `x` contains your predictors and you will use it later to try to model `DJI` using just these 15 components. (Note: These 15 components were selected arbitrarily; you could create a model with all of the components or other subsets of the components too.)

Documentation: http://www.mathworks.com/help/matlab/matrices-and-arrays.html

Data Analysis and Visualization

The Plots tab allows you to create various plots from single line graphs to bar and pie charts to histograms to scatter plots. Some plot options may be greyed out, if they don't make sense based on your data selection.

1. Open the table and select the `Dates` and `DJI` columns. Choose the plot icon to see the DJI over time.



PLOTS: dowPortfolio.Dates,dowPortfolio.DJI

Variables - dowPortfolio

dowPortfolio | x

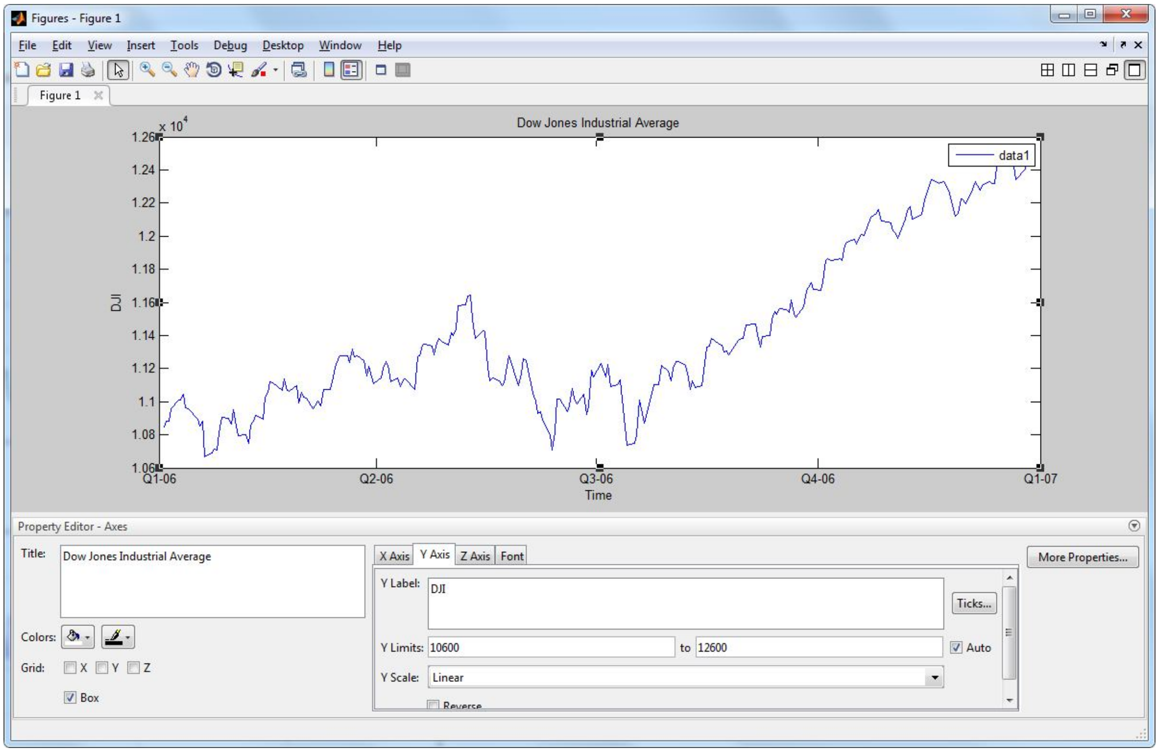
251x32 table

	1 Dates	2 DJI	3 AA	4 AIG	5 AXP	6	7	8	9	10	11	12	13	14 HON
1	732680	1.0847e+04	28.7200	68.4100	51.530								.7900	36.14
2	732681	1.0880e+04	28.8900	68.5100	51.030								.0500	35.99
3	732682	1.0882e+04	29.1200	68.6000	51.570								.6700	35.97
4	732683	1.0959e+04	29.0200	68.8900	51.750								.9600	36.53
5	732686	1.1012e+04	29.3700	68.5700	53.040								.3800	36.23
6	732687	1.1012e+04	28.4400	69.1800	52.880								.3300	36.17
7	732688	1.1043e+04	28.0500	69.6000	52.590								.4400	36.19
8	732689	1.0962e+04	27.6800	69.0400	52.600								.0500	35.77
9	732690	1.0960e+04	27.8100	68.8400	52.500								.4300	35.85
10	732694	1.0896e+04	27.9700	67.8400	52.030								.1100	35.56
11	732695	1.0855e+04	27.8100	67.4200	51.840								.4200	35.71
12	732696	1.0881e+04	28.3300	66.9200	51.660								.8300	35.88
13	732697	1.0667e+04	27.6700	65.5500	50.490								.7600	34.57
14	732700	1.0689e+04	28.0300	65.4600	50.530								.2900	34.78
15	732701	1.0712e+04	28.1600	65.3900	51.890								.0300	35.25
16	732702	1.0710e+04	28.5700	64.6700	51.970								.5700	34.79
17	732703	1.0809e+04	29.5700	65.2900	52.380								.9800	36.09
18	732704	1.0907e+04	30.0700	65.4900	52.360								.5900	36.47
19	732707	1.0900e+04	29.3300	64.9200	51.750								.5900	36.69
20	732708	1.0865e+04	30.2600	64.3300	51.520								.1200	37.06
21	732709	1.0954e+04	30.0300	65.1200	51.690								.8100	37.95
22	732710	1.0852e+04	29.5900	64.3400	51.630								.5500	37.17
23	732711	1.0794e+04	29.5200	64.2200	51.410								.4200	37.84
24	732714	1.0798e+04	30.9200	64.8700	51.300								.1000	38.04
25	732715	1.0750e+04	29.8000	64.6600	51.150								.3700	37.77
26	732716	1.0859e+04	30.1000	65.2300	51.8000	70.2100	42.1000	66.7700	37.3700	26.6600	31.1100	20.7300	37.8900	37.64

You'll notice that MATLAB® echoes the command in the Command Window, so that you can recreate the steps easily.

```
plot(dowPortfolio.Dates,dowPortfolio.DJI)
```

2. Correct the dates in the plot by typing `datetick` in the Command Window.
3. Add chart labels from the Figure window.



Property Editor - Axes

Title: Dow Jones Industrial Average

Colors:

Grid: □ X □ Y □ Z

☑ Box

X Axis | Y Axis | Z Axis | Font

Y Label: DJI

Y Limits: 10600 to 12600 ☑ Auto

Y Scale: Linear

□ Reverse

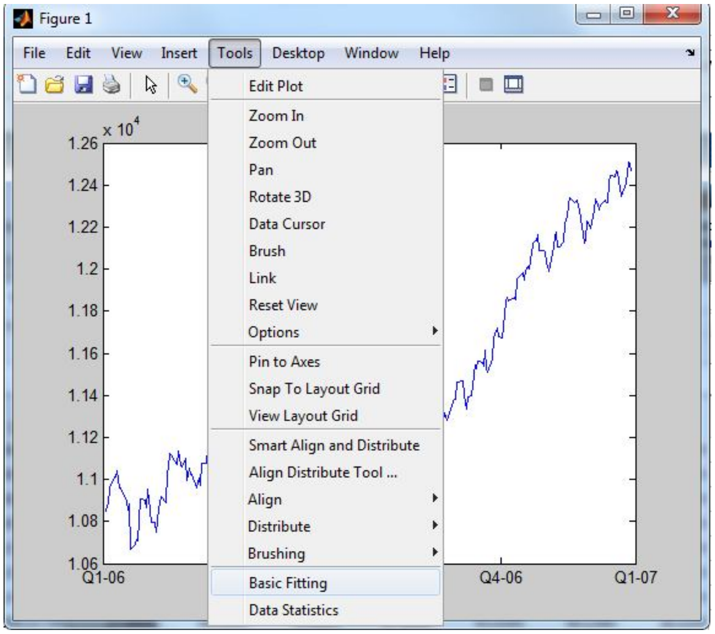
4. Auto-generate the code to produce this figure by selecting File -> Generate Code in the Figure window. Save the code as `createfigureReg.m`.

Basic Fitting

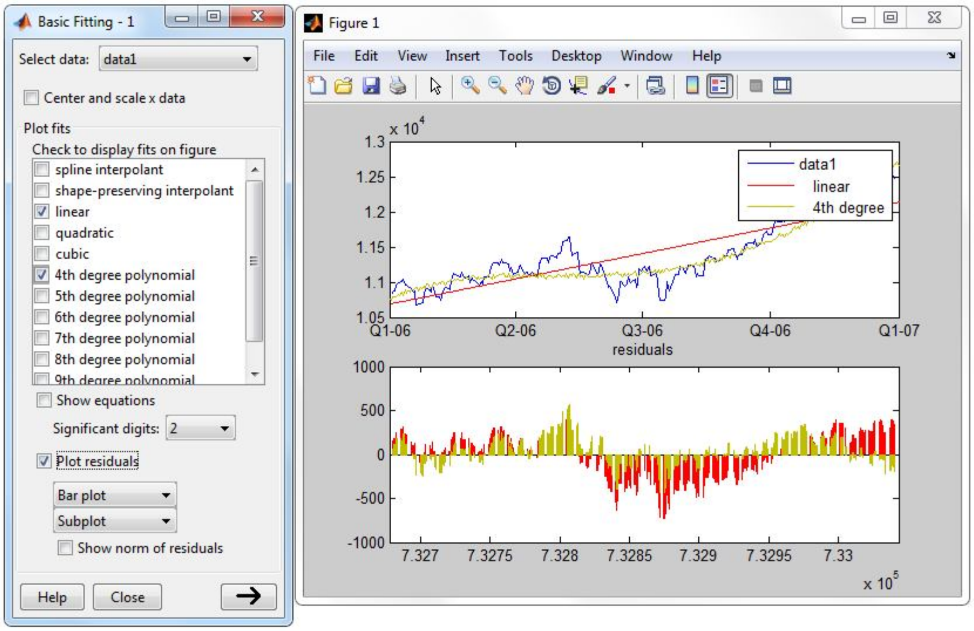
The Basic Fitting Tool allows you to:

- Select a model and plotting options
- Examine and export model coefficients and norms of residuals
- Examine and export interpolated and extrapolated values

1. From the Figure window, you can access the Basic Fitting Tool by selecting Tools -> Basic Fitting:



2. Check several plot fits to compare them against `DJI`.
3. Plot residuals.
4. Use the arrow button to view the model equations.


Basic Fitting - 1

Select data: data1

□ Center and scale x data

Plot fits

Check to display fits on figure

□ spline interpolant
□ shape-preserving interpolant
☑ linear
□ quadratic
□ cubic
☑ 4th degree polynomial
□ 5th degree polynomial
□ 6th degree polynomial
□ 7th degree polynomial
□ 8th degree polynomial
□ 9th degree polynomial

□ Show equations

Significant digits: 2

☑ Plot residuals

Bar plot

Subplot

□ Show norm of residuals

Documentation: http://www.mathworks.com/help/matlab/data_analysis/interactive-fitting.html?searchHighlight=basic+fitting#f1-15488

Linear Regression

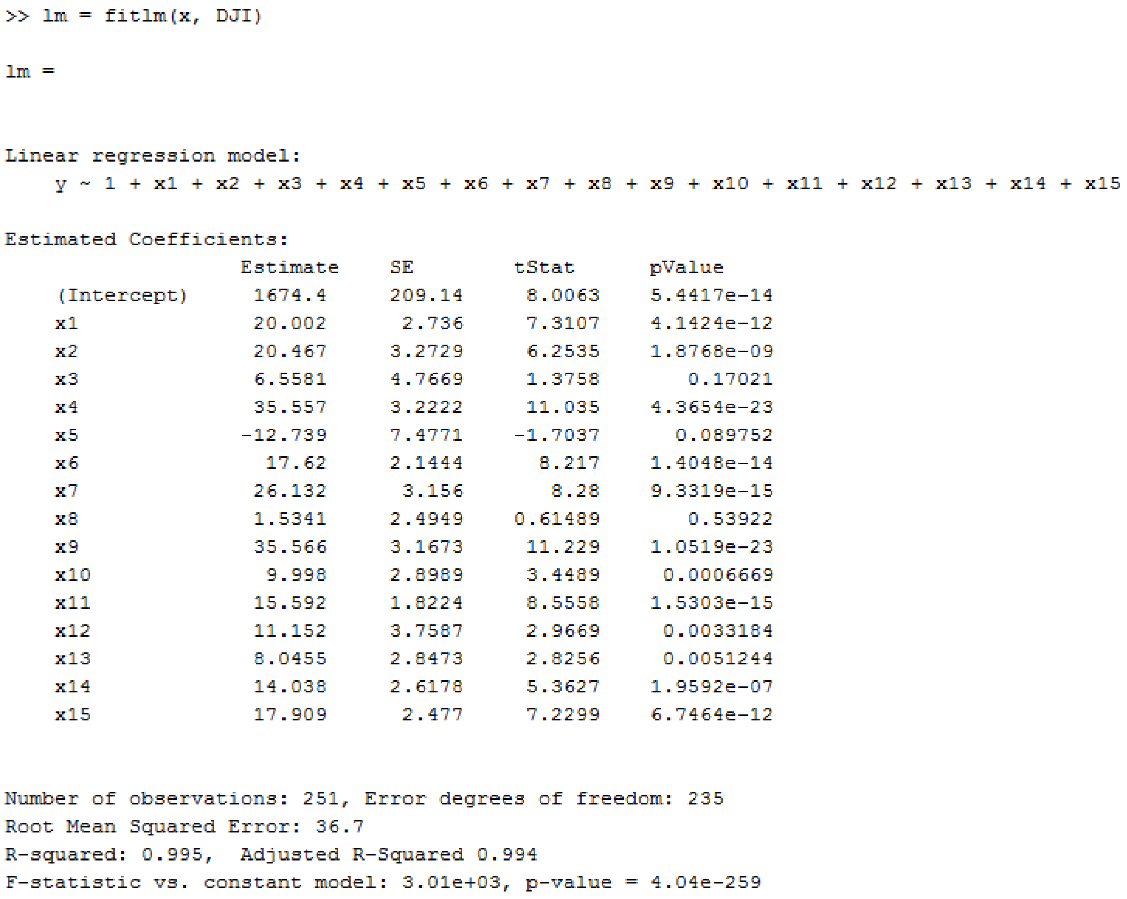
You can also use built-in functions and write your own code to apply regression techniques, such as linear regression. Next, perform linear regression between the 15 predictors in `x` against the response `DJI`.

1. Pull up the documentation for `fitlm`. Look at your options for function syntax.
2. Create a linear regression model of the responses (`DJI`), fit to the data matrix (`x`) by using the `fitlm` function:

   ```
   lm = fitlm(x, DJI)
   ```

   You get the following output:

   - Linear regression model (In this case, it is a linear formula with 16 terms, including 15 predictors.)
   - Estimated coefficients for each variable in the regression model (`Estimate`)
   - Standard error for each intercept in the regression model (`SE`)
   - t-statistic (`tStat`) – The t-statistic for your variable gets compared with values in the *Student's t distribution* to determine the p-value (which is the number that you really need to be looking at).
   - p-values (`pValue`) - The p-value for each term tests the null hypothesis that the coefficient is equal to zero (no effect). A predictor that has a low p-value (< 0.05) is likely to be a meaningful addition to your model because changes in the predictor's value are related to changes in the response variable.
3. Look at the R-squared value to get insight into how well the model fits the data. Recall that an R-squared value of 1 indicates that the regression line perfectly fits the data.



```
>> lm = fitlm(x, DJI)

lm = 


Linear regression model:
    y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15

Estimated Coefficients:
                   Estimate    SE        tStat      pValue    
    (Intercept)     1674.4    209.14      8.0063    5.4417e-14
    x1              20.002     2.736      7.3107    4.1424e-12
    x2              20.467    3.2729      6.2535    1.8768e-09
    x3              6.5581    4.7669      1.3758       0.17021
    x4              35.557    3.2222      11.035    4.3654e-23
    x5             -12.739    7.4771     -1.7037      0.089752
    x6               17.62    2.1444       8.217    1.4048e-14
    x7              26.132     3.156        8.28    9.3319e-15
    x8              1.5341    2.4949     0.61489       0.53922
    x9              35.566    3.1673      11.229    1.0519e-23
    x10              9.998    2.8989      3.4489     0.0006669
    x11             15.592    1.8224      8.5558    1.5303e-15
    x12             11.152    3.7587      2.9669     0.0033184
    x13             8.0455    2.8473      2.8256     0.0051244
    x14             14.038    2.6178      5.3627    1.9592e-07
    x15             17.909     2.477      7.2299    6.7464e-12


Number of observations: 251, Error degrees of freedom: 235
Root Mean Squared Error: 36.7
R-squared: 0.995,  Adjusted R-Squared 0.994
F-statistic vs. constant model: 3.01e+03, p-value = 4.04e-259
```

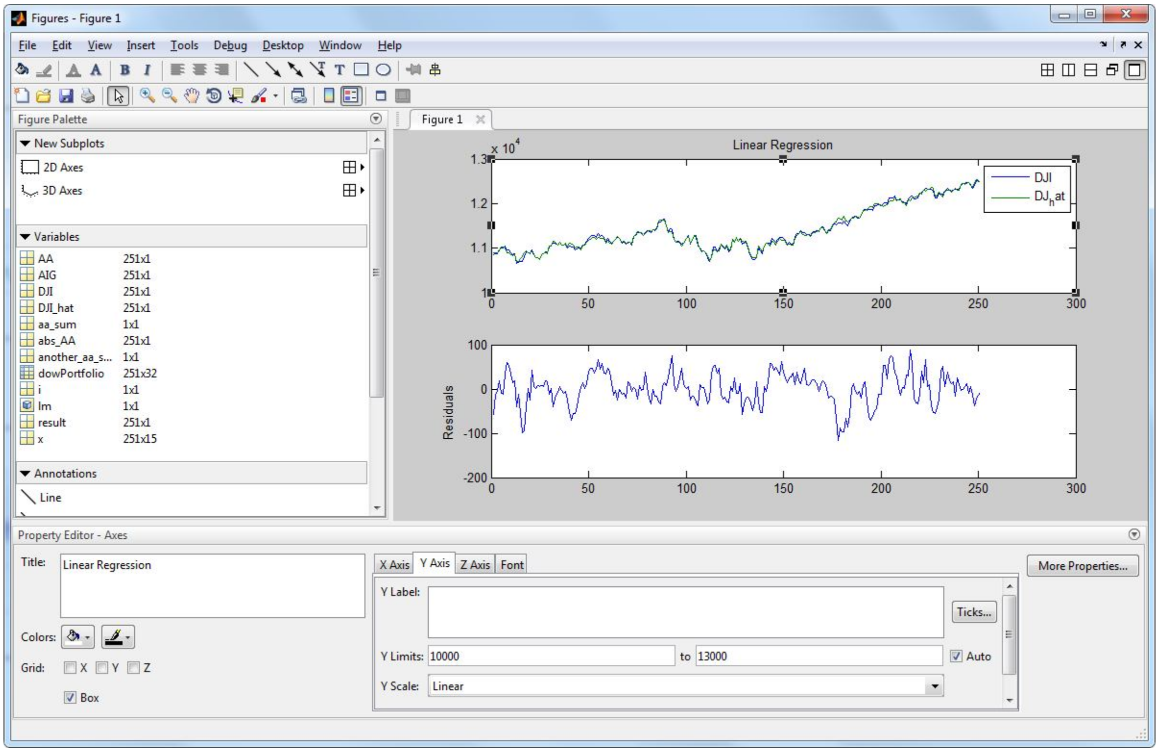
4. Test the performance of your model by generating a predicted `DJI` using the `predict` function:

```
DJI_hat = predict(lm,x)
```

Visualize Results

You can create graphics to visualize the model results, as well as the residuals, which indicate how well the model fits the dataset.

1. Select `DJI` and `DJI_hat` in the Workspace. In the Plots tab, select Plot as multiple series. A figure is displayed that shows a plot of `DJI` (the response variable) with `DJI_hat`(the prediction for `DJI`). You can see that the prediction for `DJI` very closely resembles the original `DJI`.
2. Use the Figure Palette to add a subplot to the Figure that shows the residuals (i.e., `DJI – DJI_hat`.) You'll notice that the residuals are small compared to the size of the `DJI` variable, indicating that our prediction was a good one.



Variables	
AA	251x1
AIG	251x1
DJI	251x1
DJI_hat	251x1
aa_sum	1x1
abs_AA	251x1
another_aa_s...	1x1
dowPortfolio	251x32
i	1x1
lm	1x1
result	251x1
x	251x15

Title: Linear Regression

Colors:

Grid: □ X □ Y □ Z

☑ Box

Y Label:

Y Limits: 10000 to 13000 ☑ Auto

Y Scale: Linear

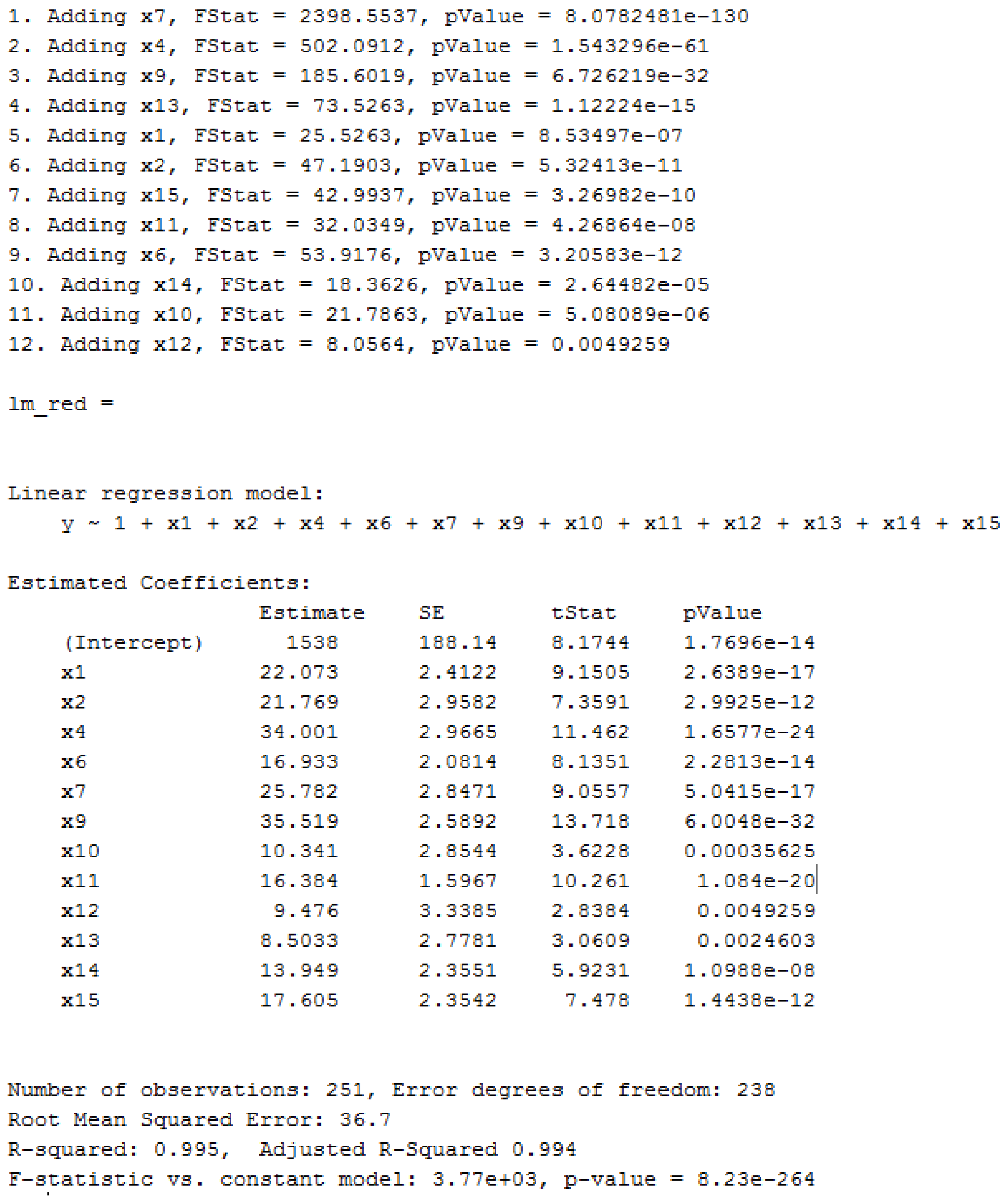
Stepwise Regression

Stepwise regression is a way of calibrating or adjusting your model to make it simpler by using fewer predictors by preforming multiple linear regressions. Now, you will use stepwise regression to reduce the 15 predictors in x to a smaller subset of predictors. The reason for doing this is to determine a fewer number of stocks that you can buy, which perform as well as the DJI itself.

1. The `stepwiselm` function is used to implement stepwise regression in MATLAB®. Open the documentation to see your argument options.
2. Implement stepwise regression. Restrict the upper bound to be a linear model, so that you don't have interactions or higher order terms.

```
lm_red = stepwiselm(x,DJI,'constant','Upper','linear')
```

3. Look at the output. How many predictors does the new model have? (Answer: 12)



```
1. Adding x7, FStat = 2398.5537, pValue = 8.0782481e-130
2. Adding x4, FStat = 502.0912, pValue = 1.543296e-61
3. Adding x9, FStat = 185.6019, pValue = 6.726219e-32
4. Adding x13, FStat = 73.5263, pValue = 1.12224e-15
5. Adding x1, FStat = 25.5263, pValue = 8.53497e-07
6. Adding x2, FStat = 47.1903, pValue = 5.32413e-11
7. Adding x15, FStat = 42.9937, pValue = 3.26982e-10
8. Adding x11, FStat = 32.0349, pValue = 4.26864e-08
9. Adding x6, FStat = 53.9176, pValue = 3.20583e-12
10. Adding x14, FStat = 18.3626, pValue = 2.64482e-05
11. Adding x10, FStat = 21.7863, pValue = 5.08089e-06
12. Adding x12, FStat = 8.0564, pValue = 0.0049259

lm_red = 


Linear regression model:
    y ~ 1 + x1 + x2 + x4 + x6 + x7 + x9 + x10 + x11 + x12 + x13 + x14 + x15

Estimated Coefficients:
                   Estimate    SE        tStat     pValue    
    (Intercept)      1538      188.14    8.1744    1.7696e-14
    x1             22.073      2.4122    9.1505    2.6389e-17
    x2             21.769      2.9582    7.3591    2.9925e-12
    x4             34.001      2.9665    11.462    1.6577e-24
    x6             16.933      2.0814    8.1351    2.2813e-14
    x7             25.782      2.8471    9.0557    5.0415e-17
    x9             35.519      2.5892    13.718    6.0048e-32
    x10            10.341      2.8544    3.6228    0.00035625
    x11            16.384      1.5967    10.261     1.084e-20
    x12             9.476      3.3385    2.8384     0.0049259
    x13            8.5033      2.7781    3.0609     0.0024603
    x14            13.949      2.3551    5.9231    1.0988e-08
    x15            17.605      2.3542     7.478    1.4438e-12


Number of observations: 251, Error degrees of freedom: 238
Root Mean Squared Error: 36.7
R-squared: 0.995,  Adjusted R-Squared 0.994
F-statistic vs. constant model: 3.77e+03, p-value = 8.23e-264
```

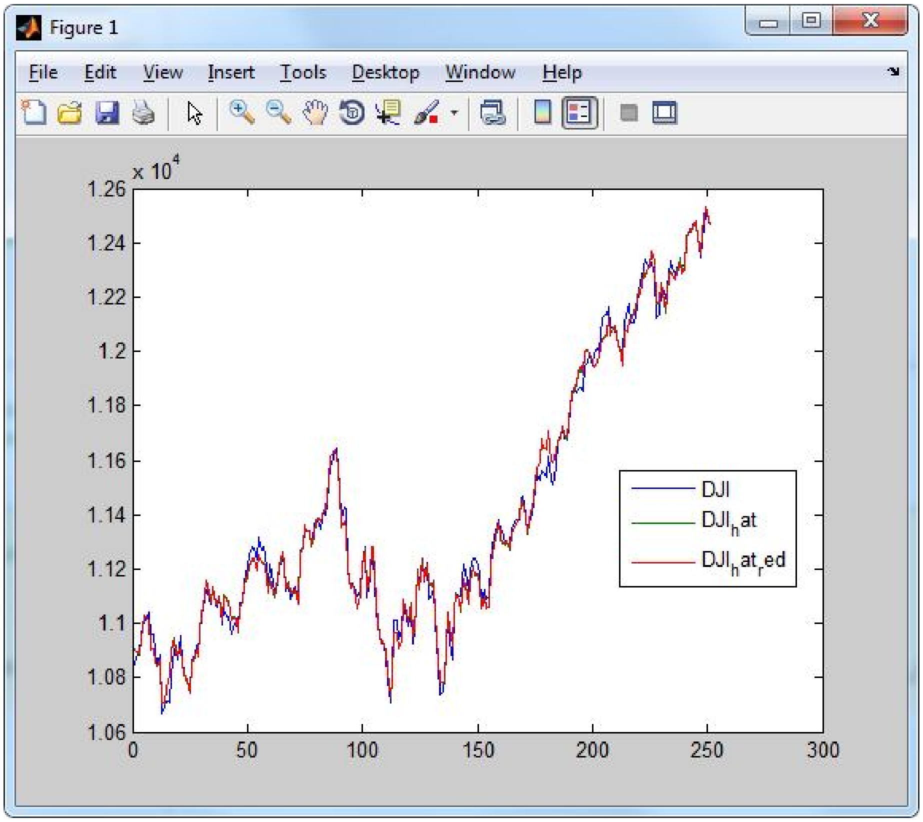
4. Look at the R-squared value of the `lm_red` model (with reduced predictors). How does it compare to the previous R-squared value?
5. Predict the `DJI` with the new stepwise model using the `predict` function:

```
DJI_hat_red = predict(lm_red,x);
```

6. Plot the original response variable `DJI` with the two predictions `DJI_hat` (from the `fitlm` model) and `DJI_hat_red` (from the `stepwiselm` model) either by (1) using the interface to select the three variables in the Workspace and using the Plots tab or (2) at the command line:

```
figure
```

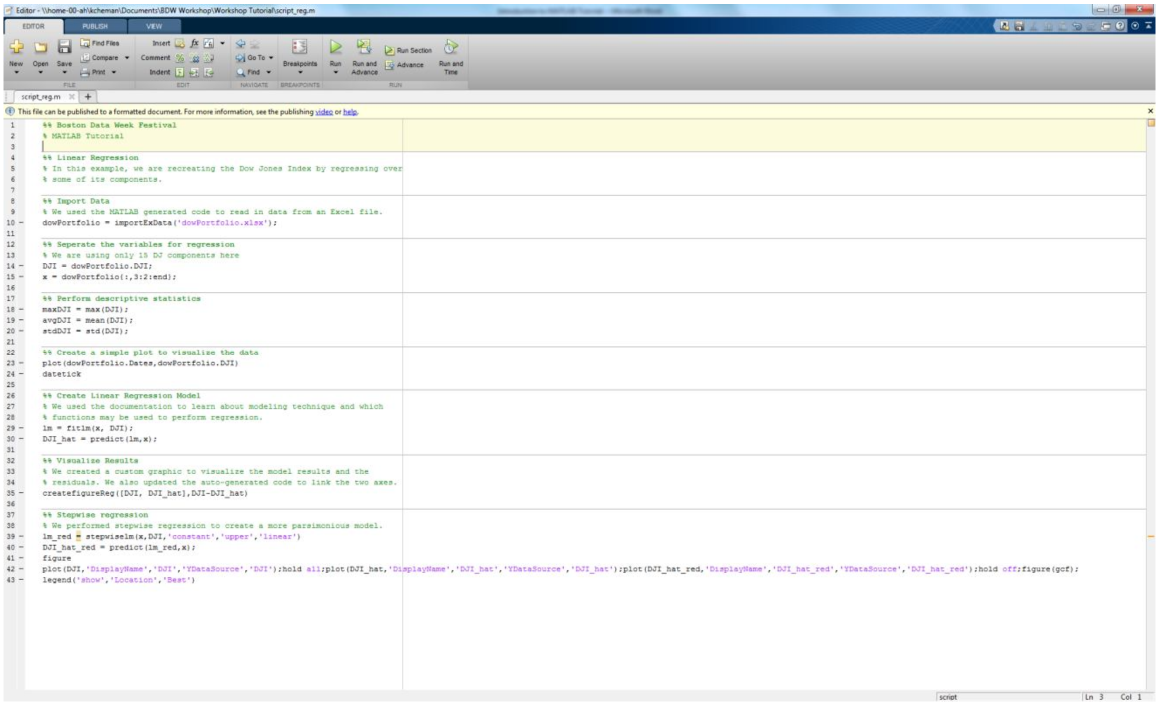
```
plot(DJI,'DisplayName','DJI','YDataSource','DJI');hold
all;plot(DJI_hat,'DisplayName','DJI_hat','YDataSource','DJI_hat');plot
(DJI_hat_red,'DisplayName','DJI_hat_red','YDataSource','DJI_hat_red');
hold off;figure(gcf);

legend('show','Location','Best')
```



7. Notice that both models resulted in predictions that were close to the original `DJI`.

Scripts

The script for this example problem is called `script_reg.m`, which utilizes the auto-generated functions `importExData.m` and `createfigureReg.m`.



This file can be published to a formatted document. For more information, see the publishing video or help.

```
%% Boston Data Week Festival
% MATLAB Tutorial

%% Linear Regression
% In this example, we are recreating the Dow Jones Index by regressing over
% some of its components.

%% Import Data
% We used the MATLAB generated code to read in data from an Excel file.
dowPortfolio = importExData('dowPortfolio.xlsx');

%% Seperate the variables for regression
% We are using only 15 DJ components here
DJI = dowPortfolio.DJI;
x = dowPortfolio{:,3:2:end};

%% Perform descriptive statistics
maxDJI = max(DJI);
avgDJI = mean(DJI);
stdDJI = std(DJI);

%% Create a simple plot to visualize the data
plot(dowPortfolio.Dates,dowPortfolio.DJI)
datetick

%% Create Linear Regression Model
% We used the documentation to learn about modeling technique and which
% functions may be used to perform regression.
lm = fitlm(x, DJI);
DJI_hat = predict(lm,x);

%% Visualize Results
% We created a custom graphic to visualize the model results and the
% residuals. We also updated the auto-generated code to link the two axes.
createfigureReg([DJI, DJI_hat],DJI-DJI_hat)

%% Stepwise regression
% We performed stepwise regression to create a more parsimonious model.
lm_red = stepwiselm(x,DJI,'constant','upper','linear')
DJI_hat_red = predict(lm_red,x);
figure
plot(DJI,'DisplayName','DJI','YDataSource','DJI');hold all;plot(DJI_hat,'DisplayName','DJI_hat','YDataSource','DJI_hat');plot(DJI_hat_red,'DisplayName','DJI_hat_red','YDataSource','DJI_hat_red');hold off;figure(gcf);
legend('show','Location','Best')
```

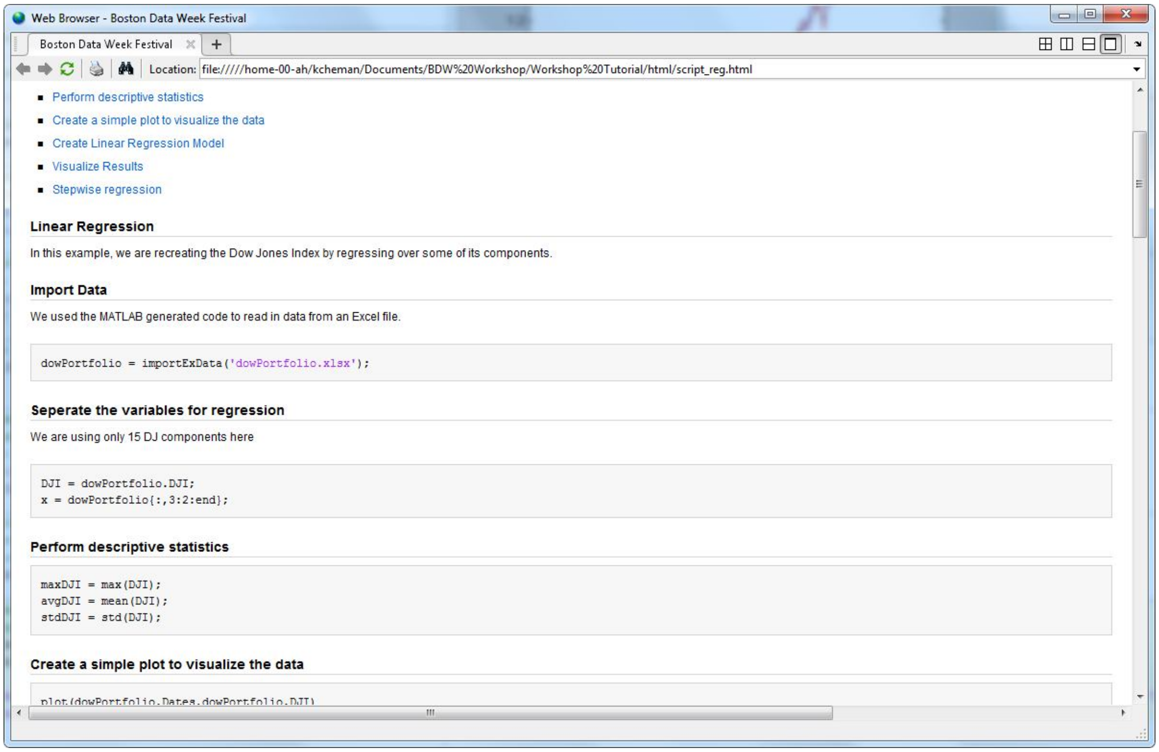
MATLAB® allows you to run sections of code at a time, working in chunks. A code section contains contiguous lines of code that you want to evaluate as a group in a MATLAB® script, beginning with two comment characters (%%).

Documentation: http://www.mathworks.com/help/matlab/scripts.html

Publishing Reports

By now, you should have a main linear regression script and two supporting functions.

The Publish tab allows you to execute code, including creating plots and evaluating results. You can then share your results (as HTML) with anyone, even if they don’t have MATLAB®.


Web Browser - Boston Data Week Festival
Boston Data Week Festival
+
Location: file://///home-00-ah/kcheman/Documents/BDW%20Workshop/Workshop%20Tutorial/html/script_reg.html

- Perform descriptive statistics
- Create a simple plot to visualize the data
- Create Linear Regression Model
- Visualize Results
- Stepwise regression

Linear Regression

In this example, we are recreating the Dow Jones Index by regressing over some of its components.

Import Data

We used the MATLAB generated code to read in data from an Excel file.

```
dowPortfolio = importExData('dowPortfolio.xlsx');
```

Seperate the variables for regression

We are using only 15 DJ components here

```
DJI = dowPortfolio.DJI;
x = dowPortfolio{:,3:2:end};
```

Perform descriptive statistics

```
maxDJI = max(DJI);
avgDJI = mean(DJI);
stdDJI = std(DJI);
```

Create a simple plot to visualize the data

```
plot(dowPortfolio.Dates,dowPortfolio.DJI)
```

Documentation: http://www.mathworks.com/help/matlab/matlab_prog/publishing-matlab-code.html?searchHighlight=publish

Saving Your Work

The `save` function allows you to save all variables from the current Workspace into a MATLAB® formatted binary file (MAT-file).

Documentation: http://www.mathworks.com/help/matlab/ref/save.html

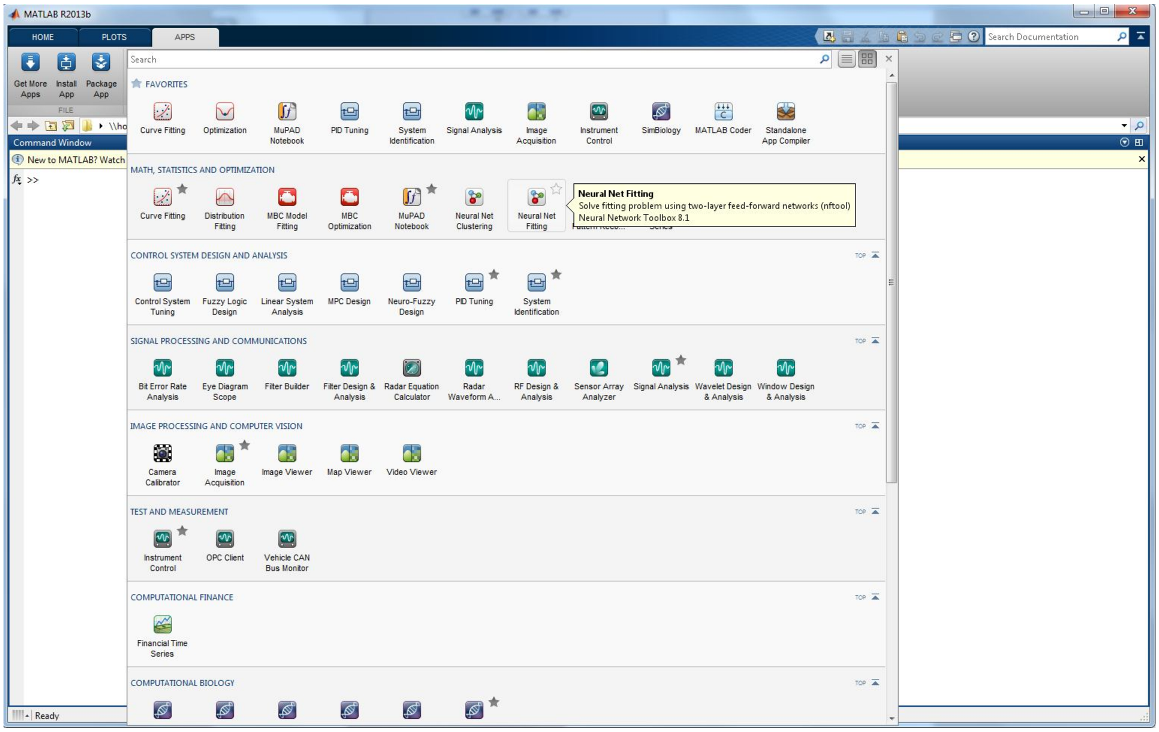
Problem #2: Neural Networks

This time, start with all 30 components as predictors. Utilize a neural network to model the response `DJI` using the predictors `x30`. You can do this in MATLAB® in two ways:

1. Use an App (`nftool`), as described in Using the Neural Network Fitting Tool.
2. Use command-line functions, as described in Using Command-Line Functions.

It is generally best to start with the App, and then to use the App to automatically generate command-line scripts.

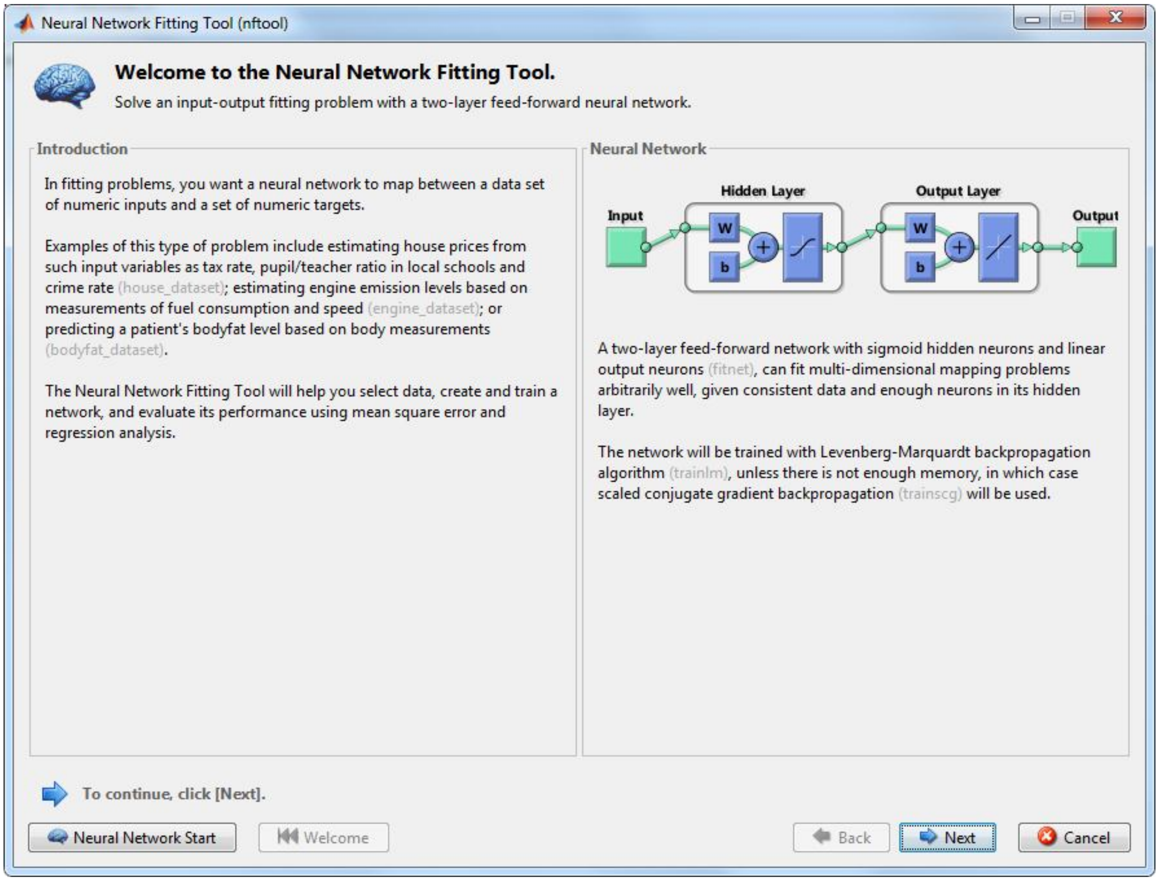
1. Fom the command line, create a new variable `x30` with all 30 predictors.
   ```
   x30 = dowPortfolio{:,3:32};
   ```
2. Locate the Neural Network Fitting Tool in the Apps tab.



Neural Net Fitting
Solve fitting problem using two-layer feed-forward networks (nftool)
Neural Network Toolbox 8.1

The App provides a step-by-step walk through for curve fitting (or other neural network applications), including from data selection through training, validation, and testing.

3. Launch the Neural Network Fitting Tool. The first panel describes the kind of problems that a fitting network is suitable for.



Welcome to the Neural Network Fitting Tool.

Solve an input-output fitting problem with a two-layer feed-forward neural network.

Introduction

In fitting problems, you want a neural network to map between a data set of numeric inputs and a set of numeric targets.

Examples of this type of problem include estimating house prices from such input variables as tax rate, pupil/teacher ratio in local schools and crime rate (house_dataset); estimating engine emission levels based on measurements of fuel consumption and speed (engine_dataset); or predicting a patient's bodyfat level based on body measurements (bodyfat_dataset).

The Neural Network Fitting Tool will help you select data, create and train a network, and evaluate its performance using mean square error and regression analysis.

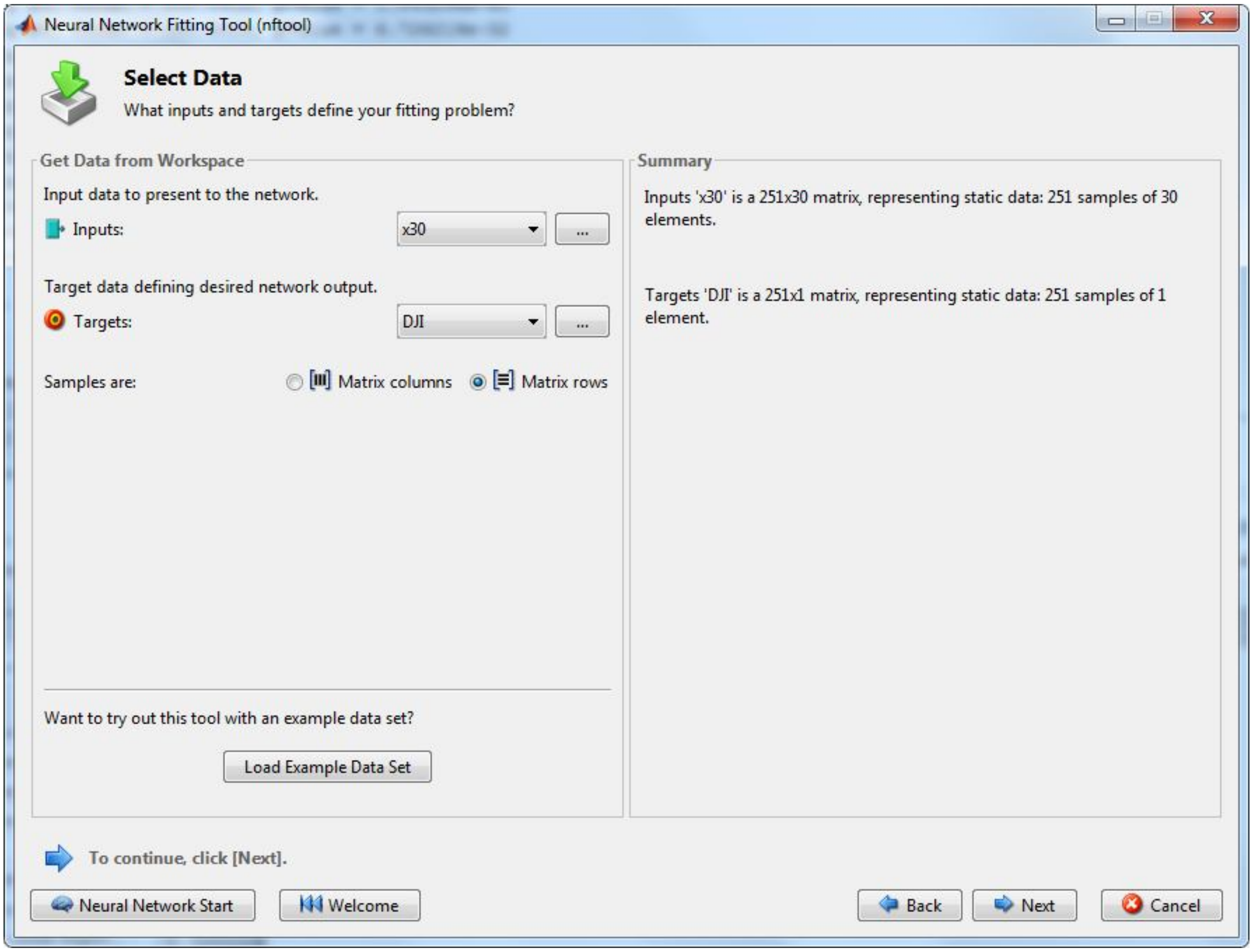
Neural Network

A two-layer feed-forward network with sigmoid hidden neurons and linear output neurons (fitnet), can fit multi-dimensional mapping problems arbitrarily well, given consistent data and enough neurons in its hidden layer.

The network will be trained with Levenberg-Marquardt backpropagation algorithm (trainlm), unless there is not enough memory, in which case scaled conjugate gradient backpropagation (trainscg) will be used.

To continue, click [Next].

4. Select Data. Set the variable `x30` (predictors) as inputs and `DJI` (response) as the target. Recall that your samples are represented as rows in `x30`, so use the radio button to select Matrix rows.



Neural Network Fitting Tool (nftool)

Select Data

What inputs and targets define your fitting problem?

Get Data from Workspace

Input data to present to the network.

Inputs: x30

Target data defining desired network output.

Targets: DJI

Samples are: ○ Matrix columns ● Matrix rows

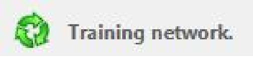
Want to try out this tool with an example data set?

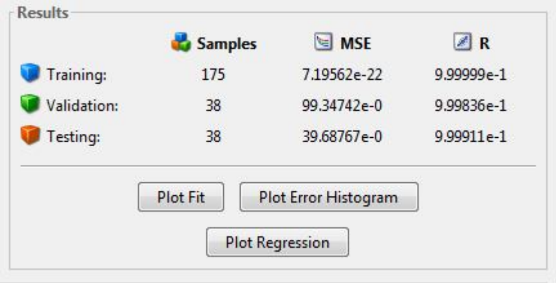
Summary

Inputs 'x30' is a 251x30 matrix, representing static data: 251 samples of 30 elements.

Targets 'DJI' is a 251x1 matrix, representing static data: 251 samples of 1 element.

To continue, click [Next].

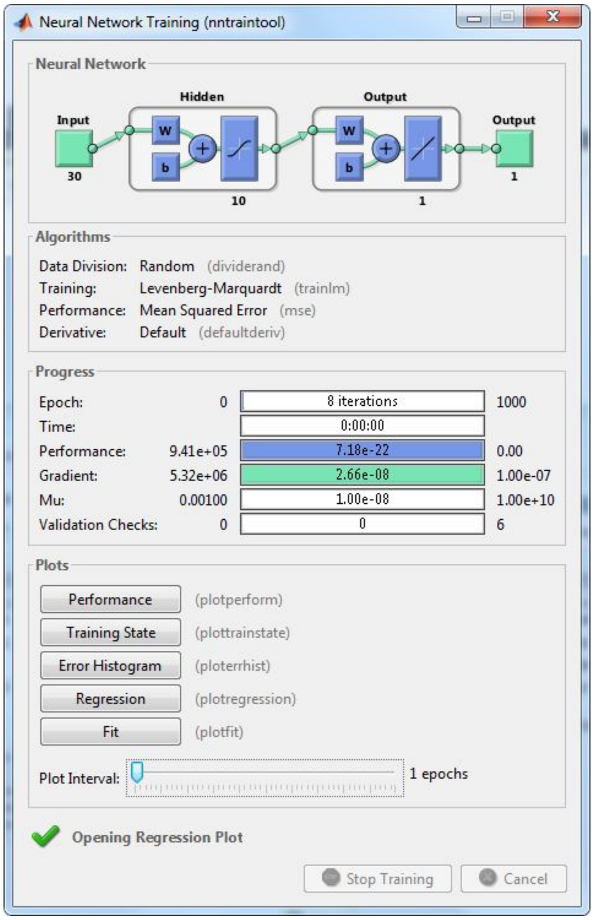
5. Divide the dataset into Training, Validation, and Testing data. Use the defaults for this example.
6. Define the architecture of the neural network. On this panel, you can choose the details of the network used to solve the problem. With more neurons, you can solve more difficult problems; with fewer neurons, the more compact your network will be. Use the defaults for this example. If you don't get a good solution, you can later return here to make changes.
7. Train the network to fit the inputs and targets. Training runs until generalization stops improving (i.e., mean square error increases). You will see the  Training network. icon as the application is running and MSE and R statistics when it finishes.



Results

	Samples	MSE	R
Training:	175	7.19562e-22	9.99999e-1
Validation:	38	99.34742e-0	9.99836e-1
Testing:	38	39.68767e-0	9.99911e-1

8. Look at the output to see details of the training algorithms and status, including network accuracy, as well as useful plots.



Algorithms

Data Division: Random (dividerand)
Training: Levenberg-Marquardt (trainlm)
Performance: Mean Squared Error (mse)
Derivative: Default (defaultderiv)

Progress

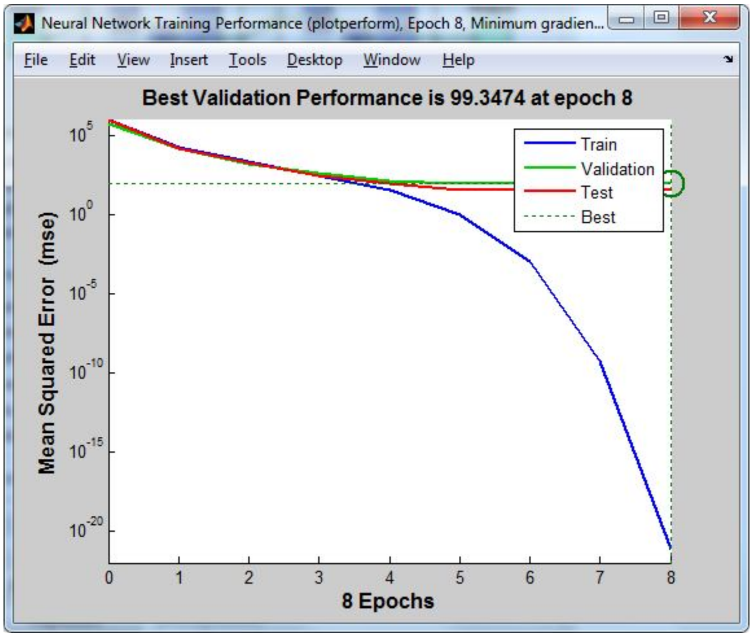
Epoch:	0	8 iterations	1000
Time:		0:00:00	
Performance:	9.41e+05	7.18e-22	0.00
Gradient:	5.32e+06	2.66e-08	1.00e-07
Mu:	0.00100	1.00e-08	1.00e+10
Validation Checks:	0	0	6

(plotperform)
(plottrainstate)
(ploterrhist)
(plotregression)
(plotfit)

Plot Interval: 1 epochs

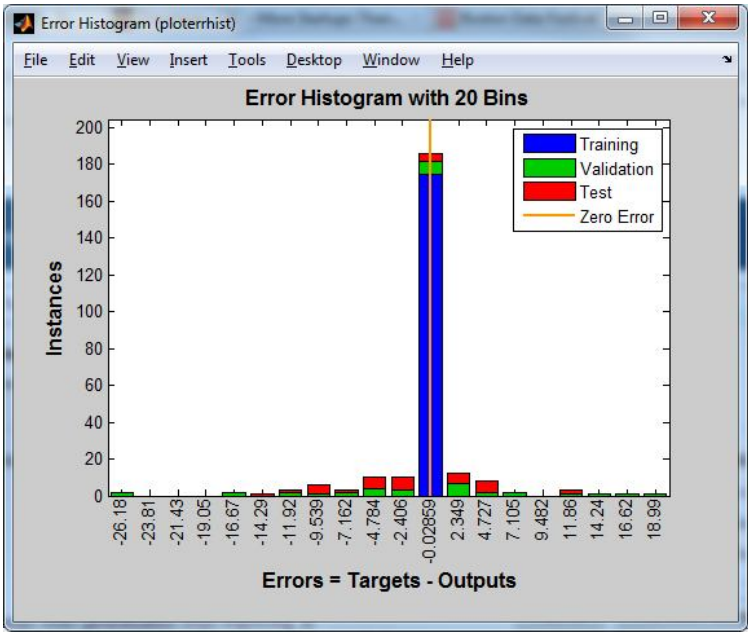
✓ Opening Regression Plot

9. Generate a Performance plot to see how the mean squared error decreases as training (the blue plot) continues. Training stops when validation (the green plot) stops decreasing. The red line shows how well the network did on the test data, showing how well your network will generalize to new data.


Neural Network Training Performance (plotperform), Epoch 8, Minimum gradien...
File Edit View Insert Tools Desktop Window Help
Best Validation Performance is 99.3474 at epoch 8
Mean Squared Error (mse)
10^{5}
10^{0}
10^{-5}
10^{-10}
10^{-15}
10^{-20}
0
1
2
3
4
5
6
7
8
8 Epochs
Train
Validation
Test
Best

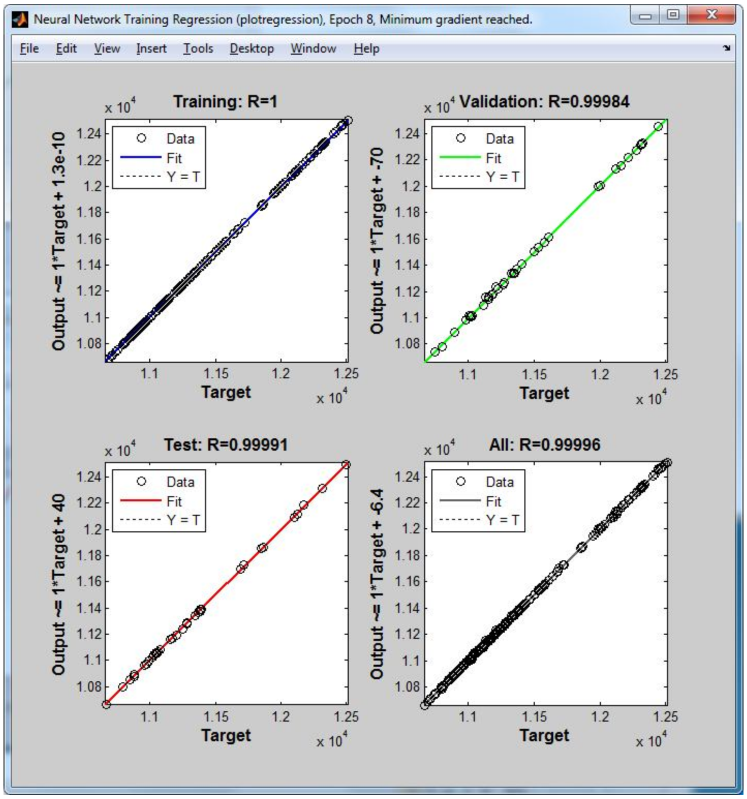
Note that an epoch is a single pass through the entire training set. At the beginning of each epoch, MATLAB® readjusts the network weights, based on where it left off in the previous epoch.

10. The Error Histogram shows you the distribution of errors. Most of the errors are grouped around 0—as indicated by the vertical yellow line. If they're not, you may need more neurons for better results.



11. View the Regression plot to see network outputs versus targets with respect to targets for training, validation, and test sets.

The regression line (in color) shows how well the network outputs are centered around the targets. For a perfect fit, the data should fall along a 45 degree line, where the network outputs are equal to the targets. (The perfect fit is represented by the dotted line.) The overall distance between the regression line and dotted line is summarized by the R-value, which should be near 1 for a good fit. The following regression plots display the network outputs For this problem, the fit is good for all data sets, since with R values are nearly 1 in each case.

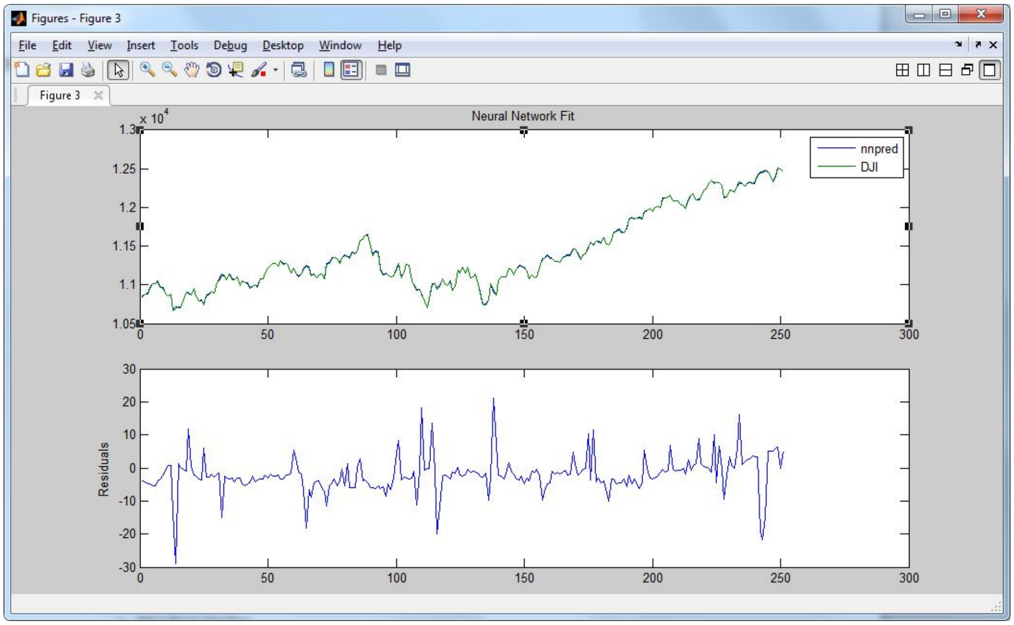


12. Evaluate network. You can use this panel to get tips on improving the network, such as adjusting the network size or using a larger dataset. You can also test the training network on additional data. Skip this step for this example.
13. Deploy solution. Click Save Results to save your variables to the Workspace. (You can also use the this panel to generate a script that captures all of the neural network fitting steps.)
14. You should now see an object called `net` in the Workspace; this is your neural network. Predict the DJI using the network and your input data:

```
nnpred = net(x30')
```

Note that you must use the transpose of x30, since you defined the network to use Matrix rows as input samples.

15. Plot the prediction `nnpred'` (Note: again use `'`) with the original `DJI`, as well as the residuals:



Additional information: http://www.mathworks.com/help/nnet/gs/fit-data-with-a-neural-network.html
Video: http://www.mathworks.com/videos/getting-started-with-neural-network-toolbox-68794.html

Extra Credit Problem (Try on your own): Regression using Machine Learning

Linear (and non-linear) regression techniques can be used to fit models when you know the structure of the model apriori. For example, with a linear model, you can assume the linear dependence of the predictors. Linear regression is a parametric regression technique. The Statistics Toolbox also offers nonparametric regression techniques, such as Regression Trees.

1. Use the data in Activity 1 (i.e., `DJI` and `x`) and perform the same regression fit using the RegressionTree class.
2. Use the class's `predict` method to predict the responses.

Additional Machine Learning Resources

Machine Learning

Discovery Page: http://www.mathworks.com/discovery/machine-learning.html

Relevant Products

- Statistics Toolbox (http://www.mathworks.com/products/statistics/)
- Neural Network Toolbox (http://www.mathworks.com/products/neural-network/)
- Curve Fitting Toolbox (http://www.mathworks.com/help/curvefit/)

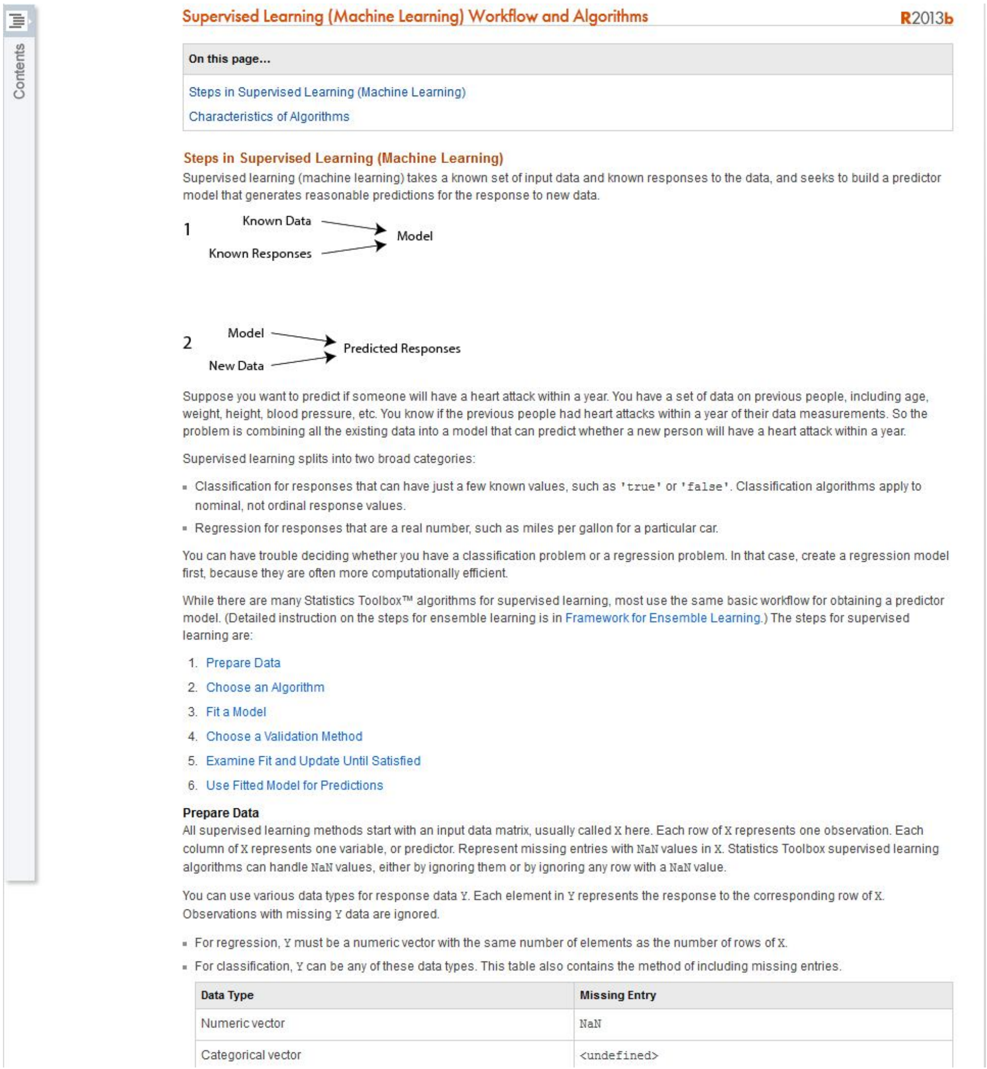
Supervised Learning

MATLAB® and its toolboxes include two categories of supervised learning algorithms:

- Classification: for categorical response values, where the data can be separated into specific “classes”
    - Support vector machines (SVM) for binary classification
    - Neural networks
    - Naïve Bayes classifiers
    - Decision trees for regression and classification
    - Linear and Quadratic Discriminant Analysis Classification
    - Nearest neighbors (kNN) for classification
- Regression: for continuous-response values
    - Multiple Linear regression with multiple predictor variables
    - Stepwise regression for variable selection
    - Multivariate regression
    - Regularization
    - Mixed effects models
    - Model assessment for plotting and diagnostic statistics
    - Nonlinear regression
    - Nonlinear mixed-effects models
    - Nonlinear Model Assessment
    - Generalized linear regression
    - Decision trees for regression and classification

There is a useful Getting Started guide in the documentation:
http://www.mathworks.com/help/stats/supervised-learning-machine-learning-workflow-and-algorithms.html#bswluhd

The guide walks you through the steps in Supervised Learning: Prepare Data, Choose an Algorithm, Fit a Model, etc.



Supervised Learning (Machine Learning) Workflow and Algorithms

R2013b

On this page…

Steps in Supervised Learning (Machine Learning)

Characteristics of Algorithms

Steps in Supervised Learning (Machine Learning)

Supervised learning (machine learning) takes a known set of input data and known responses to the data, and seeks to build a predictor model that generates reasonable predictions for the response to new data.

Suppose you want to predict if someone will have a heart attack within a year. You have a set of data on previous people, including age, weight, height, blood pressure, etc. You know if the previous people had heart attacks within a year of their data measurements. So the problem is combining all the existing data into a model that can predict whether a new person will have a heart attack within a year.

Supervised learning splits into two broad categories:

- Classification for responses that can have just a few known values, such as `'true'` or `'false'`. Classification algorithms apply to nominal, not ordinal response values.
- Regression for responses that are a real number, such as miles per gallon for a particular car.

You can have trouble deciding whether you have a classification problem or a regression problem. In that case, create a regression model first, because they are often more computationally efficient.

While there are many Statistics Toolbox™ algorithms for supervised learning, most use the same basic workflow for obtaining a predictor model. (Detailed instruction on the steps for ensemble learning is in Framework for Ensemble Learning.) The steps for supervised learning are:

1. Prepare Data
2. Choose an Algorithm
3. Fit a Model
4. Choose a Validation Method
5. Examine Fit and Update Until Satisfied
6. Use Fitted Model for Predictions

Prepare Data

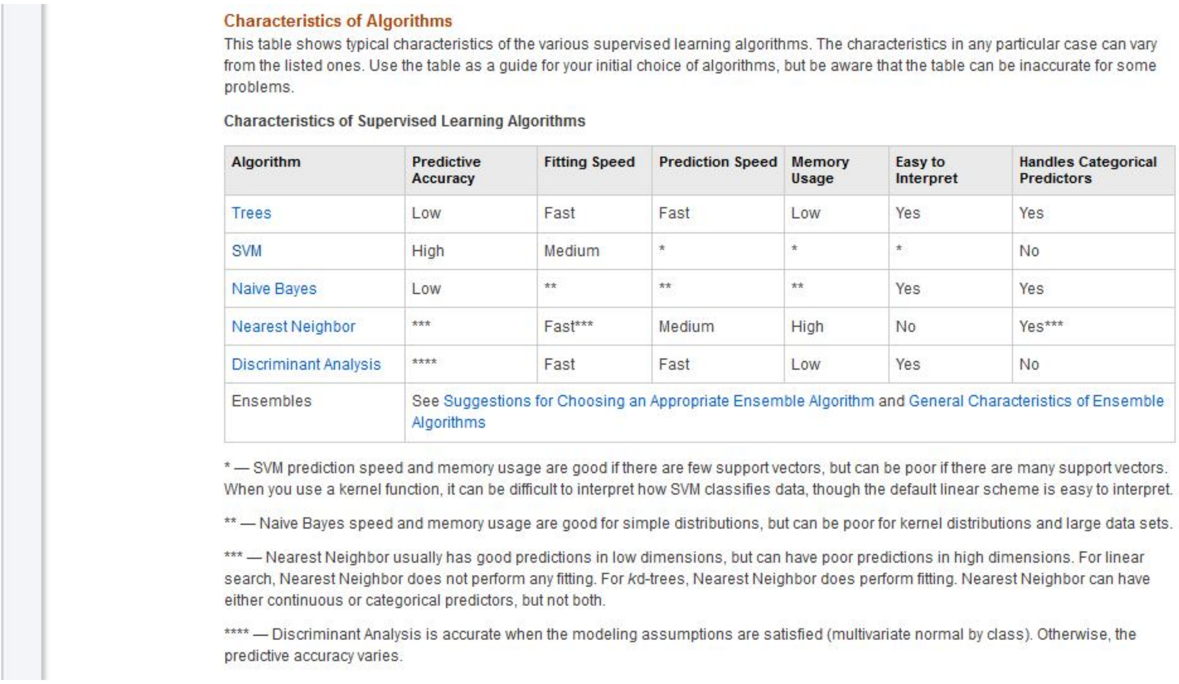
All supervised learning methods start with an input data matrix, usually called `X` here. Each row of `X` represents one observation. Each column of `X` represents one variable, or predictor. Represent missing entries with `NaN` values in `X`. Statistics Toolbox supervised learning algorithms can handle `NaN` values, either by ignoring them or by ignoring any row with a `NaN` value.

You can use various data types for response data `Y`. Each element in `Y` represents the response to the corresponding row of `X`. Observations with missing `Y` data are ignored.

- For regression, `Y` must be a numeric vector with the same number of elements as the number of rows of `X`.
- For classification, `Y` can be any of these data types. This table also contains the method of including missing entries.

Data Type	Missing Entry
Numeric vector	`NaN`
Categorical vector	`<undefined>`

The guide also lists the characteristics of various algorithms to help you choose which one to try first.



Characteristics of Algorithms

This table shows typical characteristics of the various supervised learning algorithms. The characteristics in any particular case can vary from the listed ones. Use the table as a guide for your initial choice of algorithms, but be aware that the table can be inaccurate for some problems.

Characteristics of Supervised Learning Algorithms

Algorithm	Predictive Accuracy	Fitting Speed	Prediction Speed	Memory Usage	Easy to Interpret	Handles Categorical Predictors
Trees	Low	Fast	Fast	Low	Yes	Yes
SVM	High	Medium	*	*	*	No
Naive Bayes	Low	**	**	**	Yes	Yes
Nearest Neighbor	***	Fast***	Medium	High	No	Yes***
Discriminant Analysis	****	Fast	Fast	Low	Yes	No
Ensembles	See Suggestions for Choosing an Appropriate Ensemble Algorithm and General Characteristics of Ensemble Algorithms					

* — SVM prediction speed and memory usage are good if there are few support vectors, but can be poor if there are many support vectors. When you use a kernel function, it can be difficult to interpret how SVM classifies data, though the default linear scheme is easy to interpret.

** — Naive Bayes speed and memory usage are good for simple distributions, but can be poor for kernel distributions and large data sets.

*** — Nearest Neighbor usually has good predictions in low dimensions, but can have poor predictions in high dimensions. For linear search, Nearest Neighbor does not perform any fitting. For *kd*-trees, Nearest Neighbor does perform fitting. Nearest Neighbor can have either continuous or categorical predictors, but not both.

**** — Discriminant Analysis is accurate when the modeling assumptions are satisfied (multivariate normal by class). Otherwise, the predictive accuracy varies.

Additional information: http://www.mathworks.com/discovery/supervised-learning.html

Unsupervised Learning

MATLAB® and its toolboxes include the most common unsupervised learning method - cluster analysis for exploratory data analysis to find hidden patterns or grouping in data:

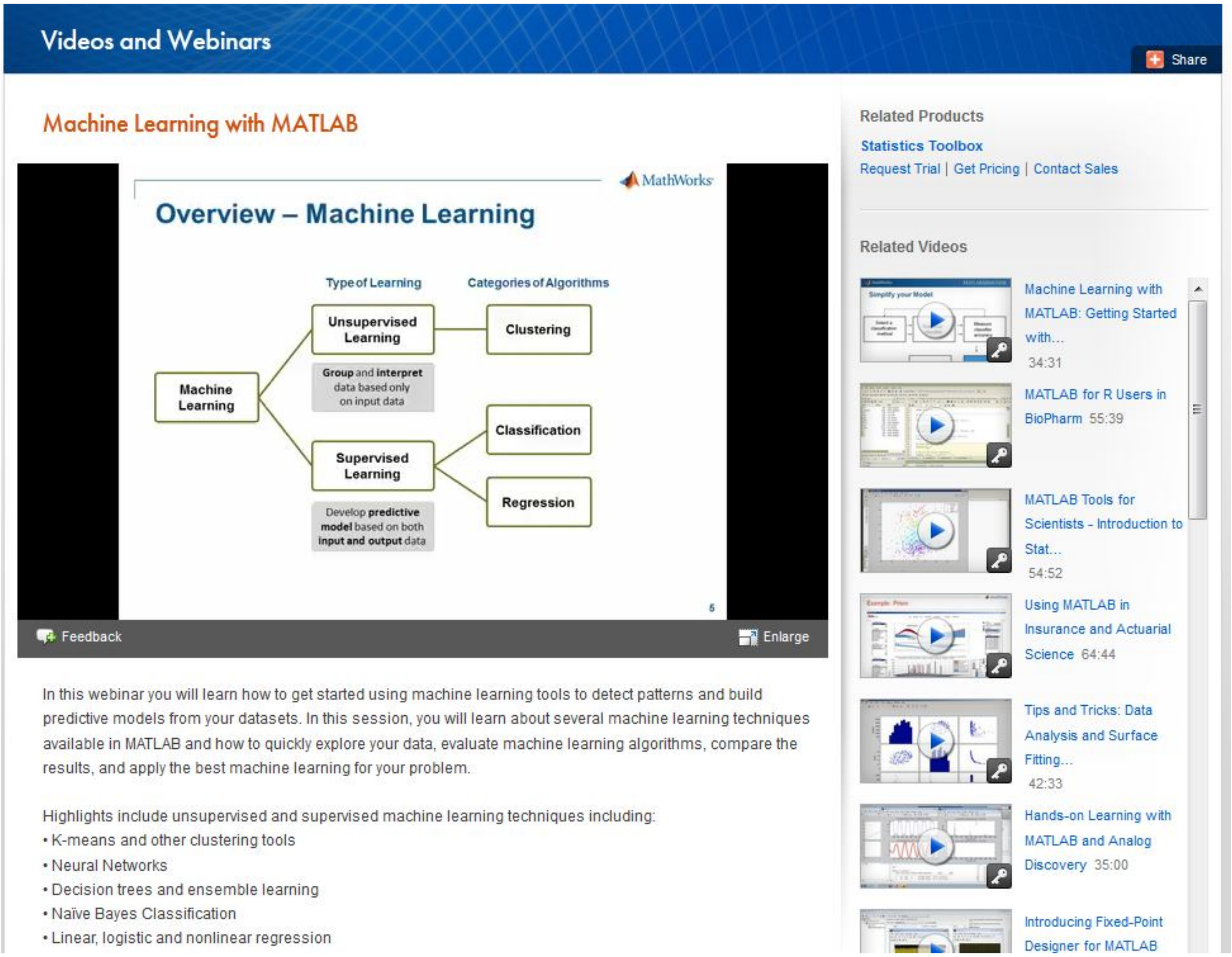
- Hierarchical clustering to produce nested sets of clusters
- k-Means clustering by minimizing mean distance
- Gaussian mixture models
- Self-organizing maps to identify prototype vectors for clusters of examples, example distributions, and similarity relationships between clusters
- Hidden Markov models for data generation

Additional information:

http://www.mathworks.com/discovery/unsupervised-learning.html
http://www.mathworks.com/discovery/cluster-analysis.html

Machine Learning Webinar

To learn more about how to get started using machine learning tools to detect patterns and build predictive models from datasets, visit the following Webinar link: http://www.mathworks.com/videos/machine-learning-with-matlab-81984.html.



Machine Learning with MATLAB

In this webinar you will learn how to get started using machine learning tools to detect patterns and build predictive models from your datasets. In this session, you will learn about several machine learning techniques available in MATLAB and how to quickly explore your data, evaluate machine learning algorithms, compare the results, and apply the best machine learning for your problem.

Highlights include unsupervised and supervised machine learning techniques including:

- K-means and other clustering tools
- Neural Networks
- Decision trees and ensemble learning
- Naïve Bayes Classification
- Linear, logistic and nonlinear regression

Related Products

Statistics Toolbox

Request Trial | Get Pricing | Contact Sales

Related Videos

Machine Learning with MATLAB: Getting Started with... 34:31

MATLAB for R Users in BioPharm 55:39

MATLAB Tools for Scientists - Introduction to Stat... 54:52

Using MATLAB in Insurance and Actuarial Science 64:44

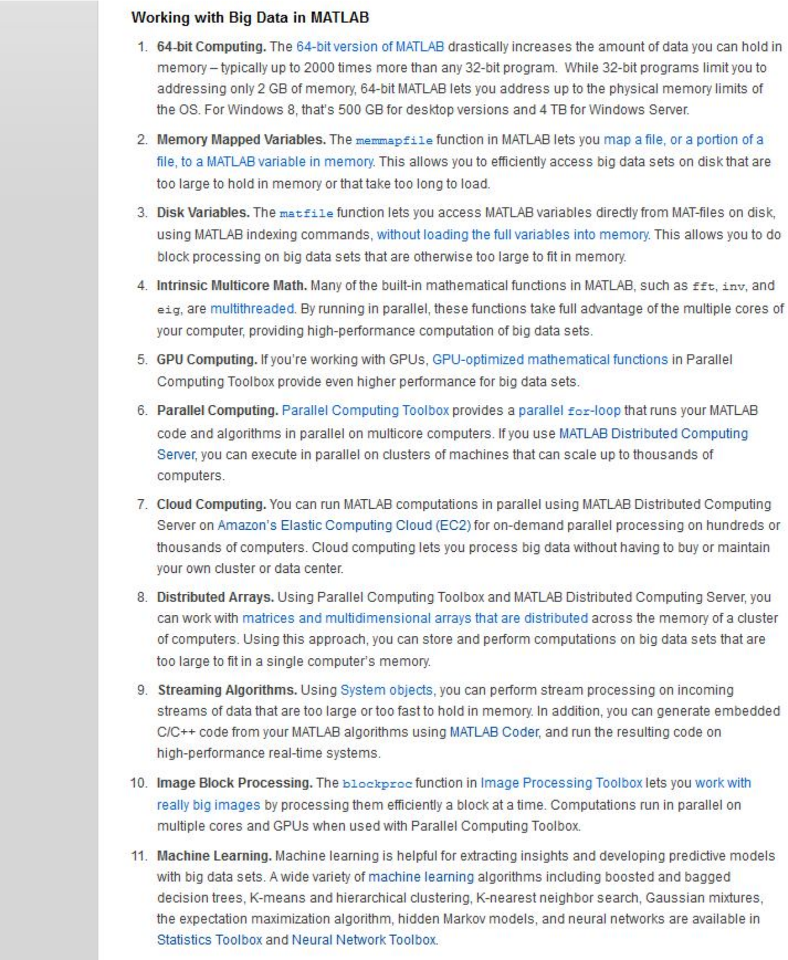
Tips and Tricks: Data Analysis and Surface Fitting... 42:33

Hands-on Learning with MATLAB and Analog Discovery 35:00

Introducing Fixed-Point Designer for MATLAB

Working with Large Data Sets

Big data sets may not fit into available memory, may take too long to process, or may stream too quickly to store. Standard algorithms are usually not designed to process big data sets in reasonable amounts of time or memory. MATLAB® provides a number of tools to tackle these challenges.



Working with Big Data in MATLAB

1. **64-bit Computing.** The 64-bit version of MATLAB drastically increases the amount of data you can hold in memory – typically up to 2000 times more than any 32-bit program. While 32-bit programs limit you to addressing only 2 GB of memory, 64-bit MATLAB lets you address up to the physical memory limits of the OS. For Windows 8, that's 500 GB for desktop versions and 4 TB for Windows Server.
2. **Memory Mapped Variables.** The `memmapfile` function in MATLAB lets you map a file, or a portion of a file, to a MATLAB variable in memory. This allows you to efficiently access big data sets on disk that are too large to hold in memory or that take too long to load.
3. **Disk Variables.** The `matfile` function lets you access MATLAB variables directly from MAT-files on disk, using MATLAB indexing commands, without loading the full variables into memory. This allows you to do block processing on big data sets that are otherwise too large to fit in memory.
4. **Intrinsic Multicore Math.** Many of the built-in mathematical functions in MATLAB, such as `fft`, `inv`, and `eig`, are multithreaded. By running in parallel, these functions take full advantage of the multiple cores of your computer, providing high-performance computation of big data sets.
5. **GPU Computing.** If you're working with GPUs, GPU-optimized mathematical functions in Parallel Computing Toolbox provide even higher performance for big data sets.
6. **Parallel Computing.** Parallel Computing Toolbox provides a parallel `for`-loop that runs your MATLAB code and algorithms in parallel on multicore computers. If you use MATLAB Distributed Computing Server, you can execute in parallel on clusters of machines that can scale up to thousands of computers.
7. **Cloud Computing.** You can run MATLAB computations in parallel using MATLAB Distributed Computing Server on Amazon's Elastic Computing Cloud (EC2) for on-demand parallel processing on hundreds or thousands of computers. Cloud computing lets you process big data without having to buy or maintain your own cluster or data center.
8. **Distributed Arrays.** Using Parallel Computing Toolbox and MATLAB Distributed Computing Server, you can work with matrices and multidimensional arrays that are distributed across the memory of a cluster of computers. Using this approach, you can store and perform computations on big data sets that are too large to fit in a single computer's memory.
9. **Streaming Algorithms.** Using System objects, you can perform stream processing on incoming streams of data that are too large or too fast to hold in memory. In addition, you can generate embedded C/C++ code from your MATLAB algorithms using MATLAB Coder, and run the resulting code on high-performance real-time systems.
10. **Image Block Processing.** The `blockproc` function in Image Processing Toolbox lets you work with really big images by processing them efficiently a block at a time. Computations run in parallel on multiple cores and GPUs when used with Parallel Computing Toolbox.
11. **Machine Learning.** Machine learning is helpful for extracting insights and developing predictive models with big data sets. A wide variety of machine learning algorithms including boosted and bagged decision trees, K-means and hierarchical clustering, K-nearest neighbor search, Gaussian mixtures, the expectation maximization algorithm, hidden Markov models, and neural networks are available in Statistics Toolbox and Neural Network Toolbox.

Additional information: http://www.mathworks.com/discovery/big-data-matlab.html