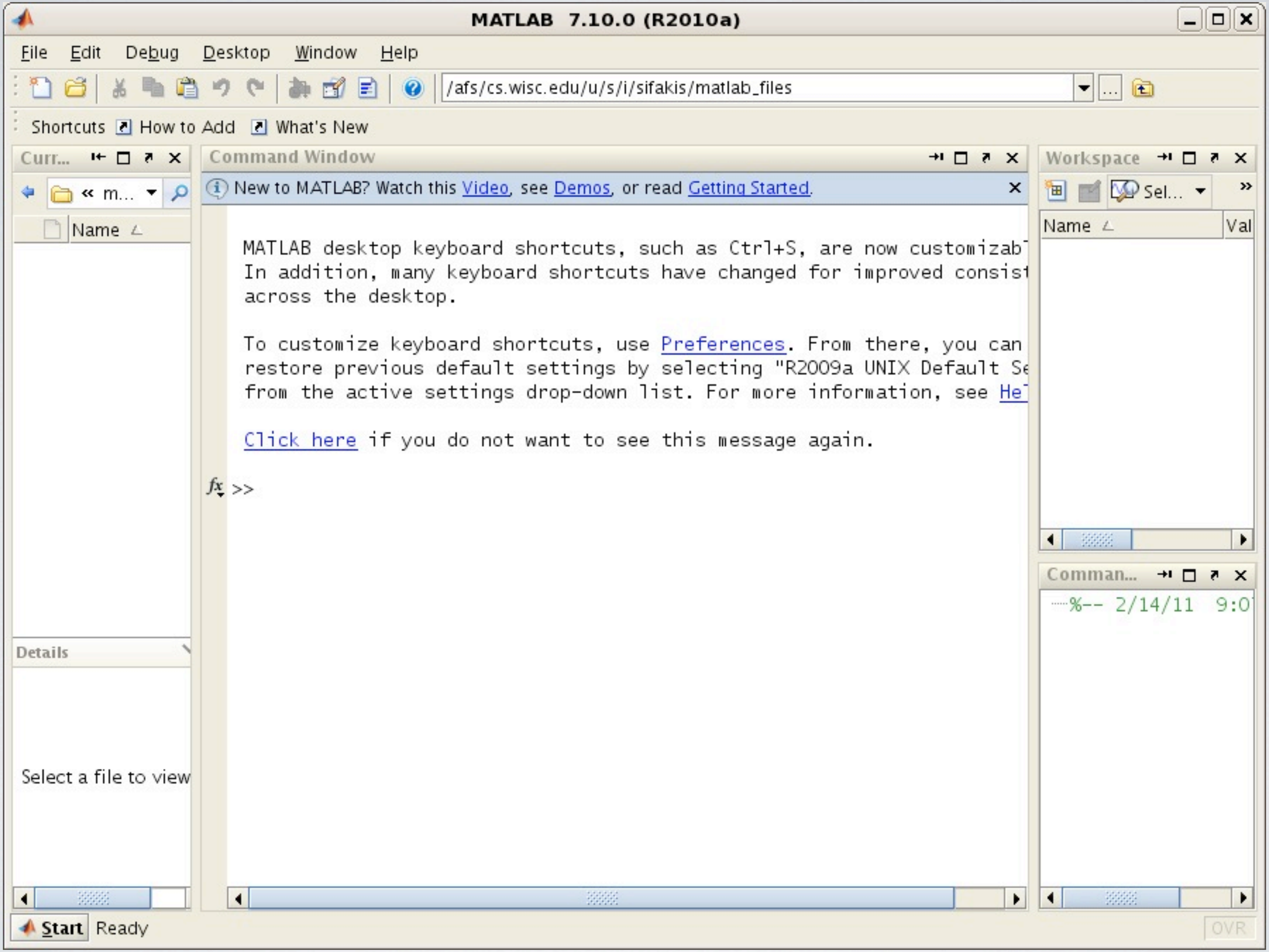
MATLAB BASICS

CS412 Spring 2011

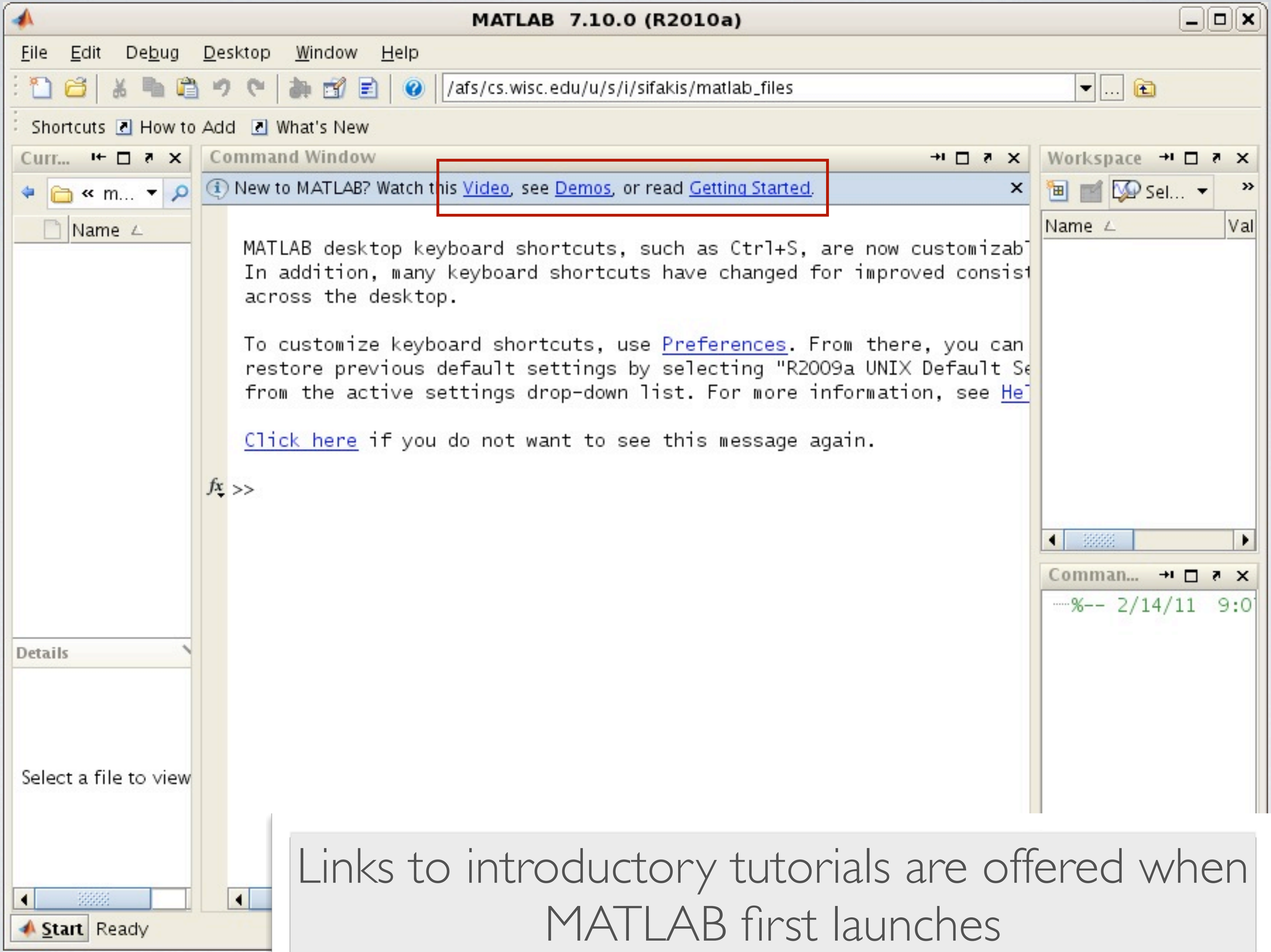
Instructor : Eftychios Sifakis

To launch MATLAB from an instructional lab machine:
Enter *matlab* at the shell prompt



```
MATLAB desktop keyboard shortcuts, such as Ctrl+S, are now customizab
In addition, many keyboard shortcuts have changed for improved consis
across the desktop.

To customize keyboard shortcuts, use Preferences. From there, you can
restore previous default settings by selecting "R2009a UNIX Default Se
from the active settings drop-down list. For more information, see He

Click here if you do not want to see this message again.

>>
```

```
%-- 2/14/11  9:0
```

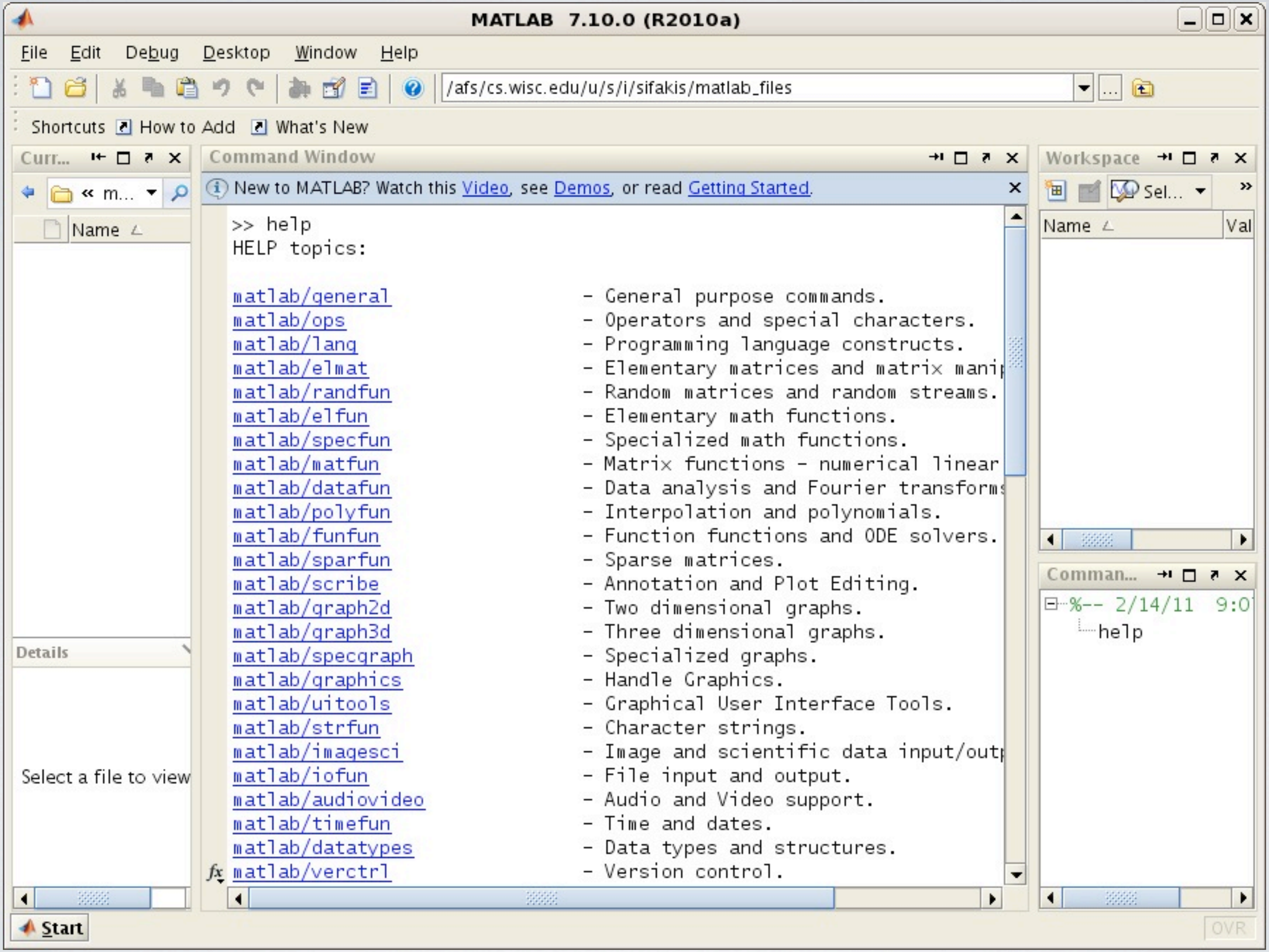


New to MATLAB? Watch this Video, see Demos, or read Getting Started.

```
MATLAB desktop keyboard shortcuts, such as Ctrl+S, are now customizab
In addition, many keyboard shortcuts have changed for improved consist
across the desktop.

To customize keyboard shortcuts, use Preferences. From there, you can
restore previous default settings by selecting "R2009a UNIX Default Se
from the active settings drop-down list. For more information, see He

Click here if you do not want to see this message again.

>>
```

```
%-- 2/14/11  9:0
```

Links to introductory tutorials are offered when MATLAB first launches

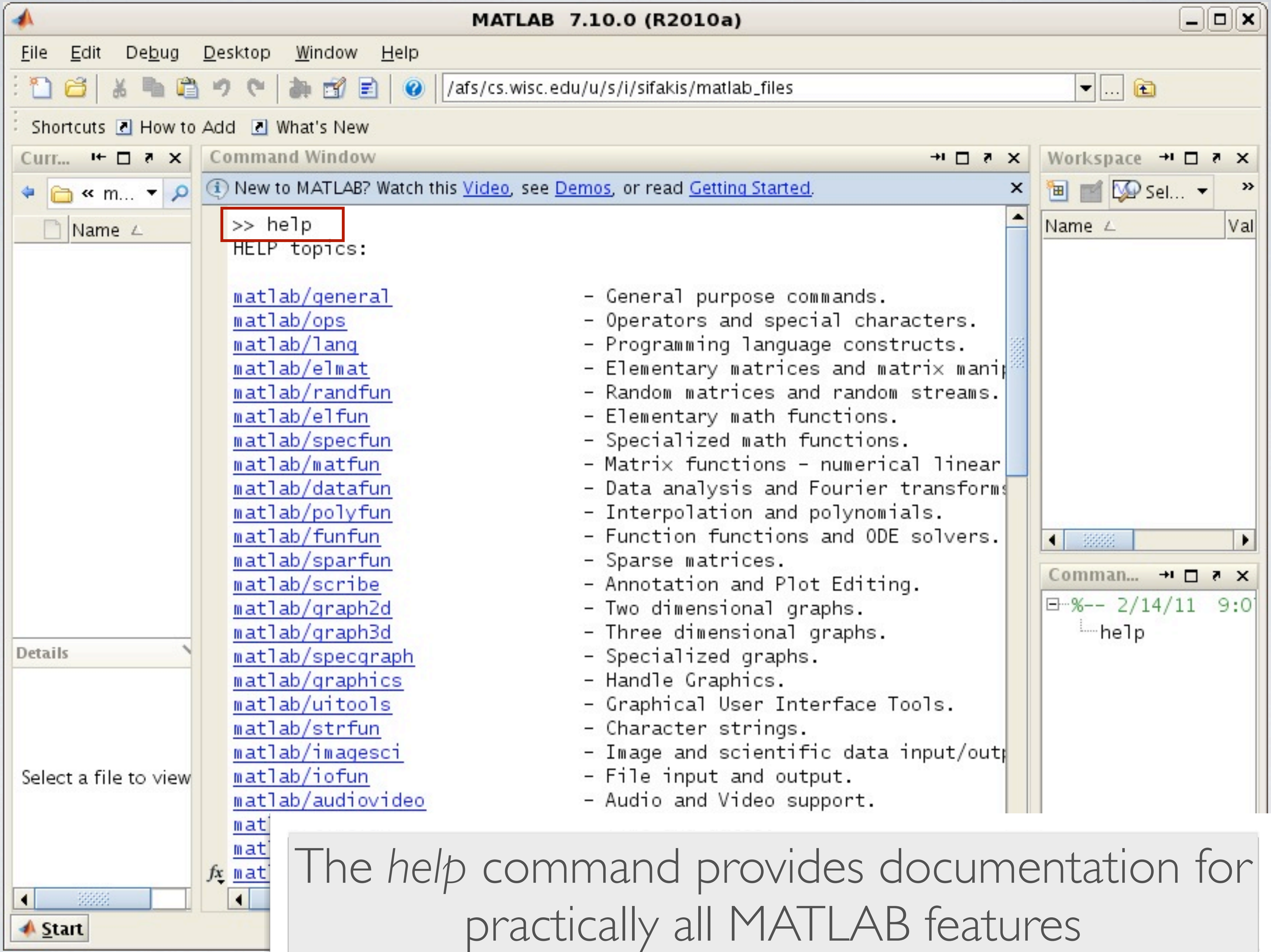


/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files

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

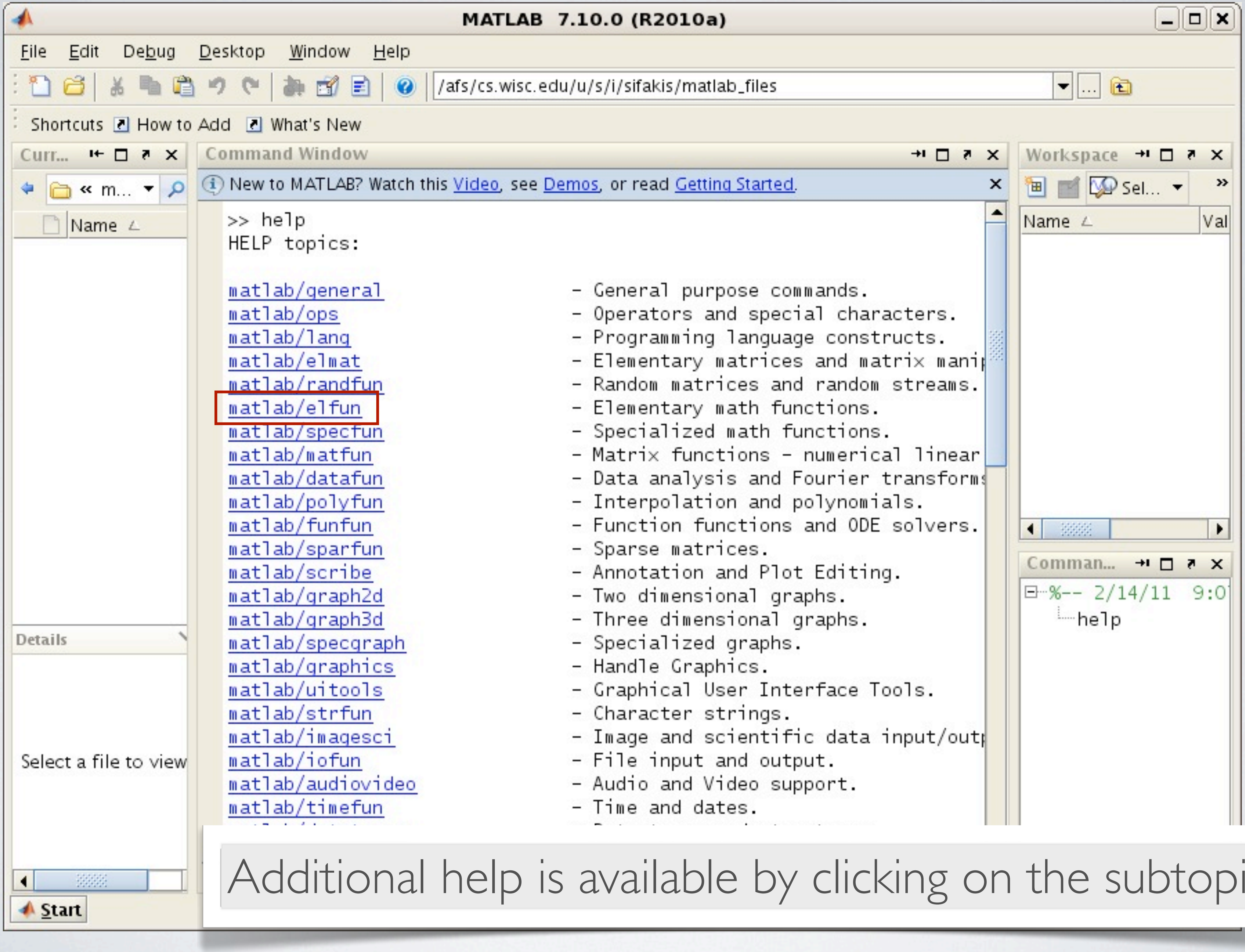
```
>> help
HELP topics:

matlab/general                 - General purpose commands.
matlab/ops                     - Operators and special characters.
matlab/lang                    - Programming language constructs.
matlab/elmat                   - Elementary matrices and matrix mani
matlab/randfun                 - Random matrices and random streams.
matlab/elfun                   - Elementary math functions.
matlab/specfun                 - Specialized math functions.
matlab/matfun                  - Matrix functions - numerical linear
matlab/datafun                 - Data analysis and Fourier transform
matlab/polyfun                 - Interpolation and polynomials.
matlab/funfun                  - Function functions and ODE solvers.
matlab/sparfun                 - Sparse matrices.
matlab/scribe                  - Annotation and Plot Editing.
matlab/graph2d                 - Two dimensional graphs.
matlab/graph3d                 - Three dimensional graphs.
matlab/specgraph               - Specialized graphs.
matlab/graphics                - Handle Graphics.
matlab/uitools                 - Graphical User Interface Tools.
matlab/strfun                  - Character strings.
matlab/imagesci                - Image and scientific data input/out
matlab/iofun                   - File input and output.
matlab/audiovideo              - Audio and Video support.
matlab/timefun                 - Time and dates.
matlab/datatypes               - Data types and structures.
matlab/verctrl                 - Version control.
```

```
%-- 2/14/11  9:0
  help
```


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Start
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Name
Val
Comman...
%-- 2/14/11 9:0
help

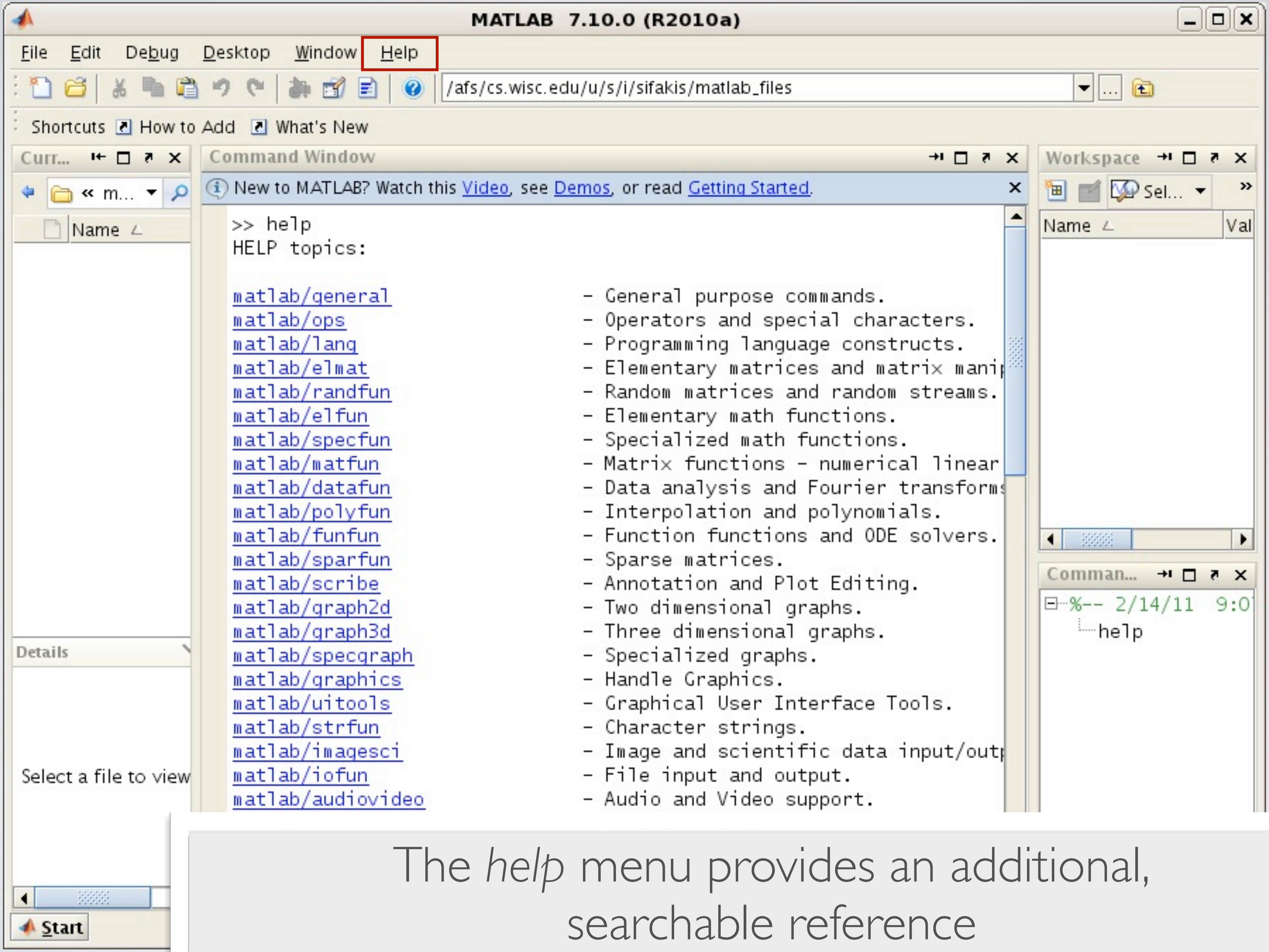
```
>> help
HELP topics:

matlab/general                - General purpose commands.
matlab/ops                    - Operators and special characters.
matlab/lang                   - Programming language constructs.
matlab/elmat                  - Elementary matrices and matrix mani
matlab/randfun                - Random matrices and random streams.
matlab/elfun                  - Elementary math functions.
matlab/specfun                - Specialized math functions.
matlab/matfun                 - Matrix functions - numerical linear
matlab/datafun                - Data analysis and Fourier transform
matlab/polyfun                - Interpolation and polynomials.
matlab/funfun                 - Function functions and ODE solvers.
matlab/sparfun                - Sparse matrices.
matlab/scribe                 - Annotation and Plot Editing.
matlab/graph2d                - Two dimensional graphs.
matlab/graph3d                - Three dimensional graphs.
matlab/specgraph              - Specialized graphs.
matlab/graphics               - Handle Graphics.
matlab/uitools                - Graphical User Interface Tools.
matlab/strfun                 - Character strings.
matlab/imagesci               - Image and scientific data input/outp
matlab/iofun                  - File input and output.
matlab/audiovideo             - Audio and Video support.
mat
mat
mat
```

The *help* command provides documentation for practically all MATLAB features


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Start
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Name
Val
Comman...
%-- 2/14/11 9:0
help

```
>> help
HELP topics:

matlab/general                 - General purpose commands.
matlab/ops                     - Operators and special characters.
matlab/lang                    - Programming language constructs.
matlab/elmat                   - Elementary matrices and matrix mani
matlab/randfun                 - Random matrices and random streams.
matlab/elfun                   - Elementary math functions.
matlab/specfun                 - Specialized math functions.
matlab/matfun                  - Matrix functions - numerical linear
matlab/datafun                 - Data analysis and Fourier transform
matlab/polyfun                 - Interpolation and polynomials.
matlab/funfun                  - Function functions and ODE solvers.
matlab/sparfun                 - Sparse matrices.
matlab/scribe                  - Annotation and Plot Editing.
matlab/graph2d                 - Two dimensional graphs.
matlab/graph3d                 - Three dimensional graphs.
matlab/specgraph               - Specialized graphs.
matlab/graphics                - Handle Graphics.
matlab/uitools                 - Graphical User Interface Tools.
matlab/strfun                  - Character strings.
matlab/imagesci                - Image and scientific data input/out
matlab/iofun                   - File input and output.
matlab/audiovideo              - Audio and Video support.
matlab/timefun                 - Time and dates.
```

Additional help is available by clicking on the subtopics

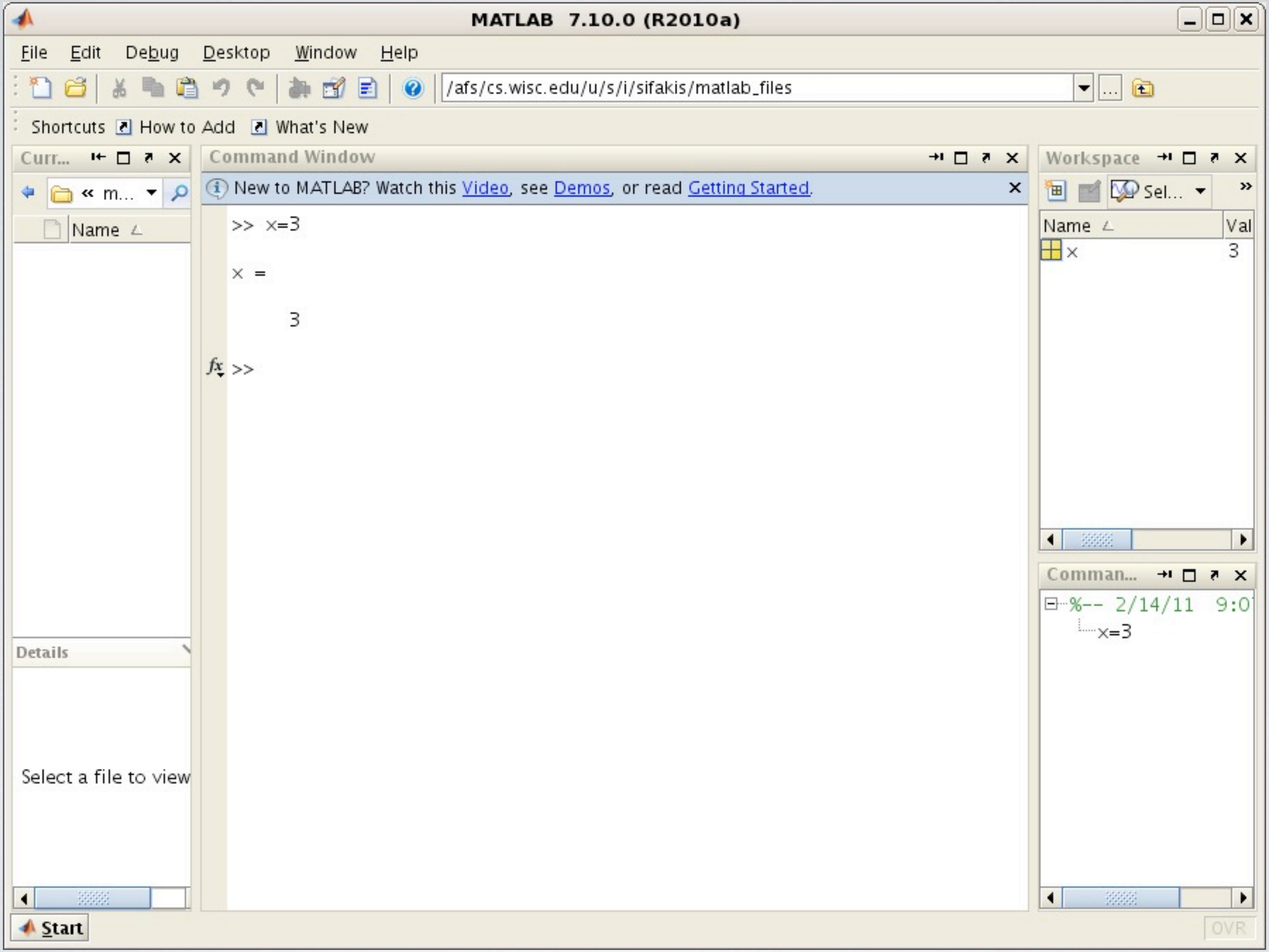


/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files

```
>> help
HELP topics:

matlab/general                - General purpose commands.
matlab/ops                    - Operators and special characters.
matlab/lang                   - Programming language constructs.
matlab/elmat                  - Elementary matrices and matrix mani
matlab/randfun                - Random matrices and random streams.
matlab/elfun                  - Elementary math functions.
matlab/specfun                - Specialized math functions.
matlab/matfun                 - Matrix functions - numerical linear
matlab/datafun                - Data analysis and Fourier transform
matlab/polyfun                - Interpolation and polynomials.
matlab/funfun                 - Function functions and ODE solvers.
matlab/sparfun                - Sparse matrices.
matlab/scribe                 - Annotation and Plot Editing.
matlab/graph2d                - Two dimensional graphs.
matlab/graph3d                - Three dimensional graphs.
matlab/specgraph              - Specialized graphs.
matlab/graphics               - Handle Graphics.
matlab/uitools                - Graphical User Interface Tools.
matlab/strfun                 - Character strings.
matlab/imagesci               - Image and scientific data input/out
matlab/iofun                  - File input and output.
matlab/audiovideo             - Audio and Video support.
```

```
%-- 2/14/11   9:0
  help
```

The *help* menu provides an additional, searchable reference



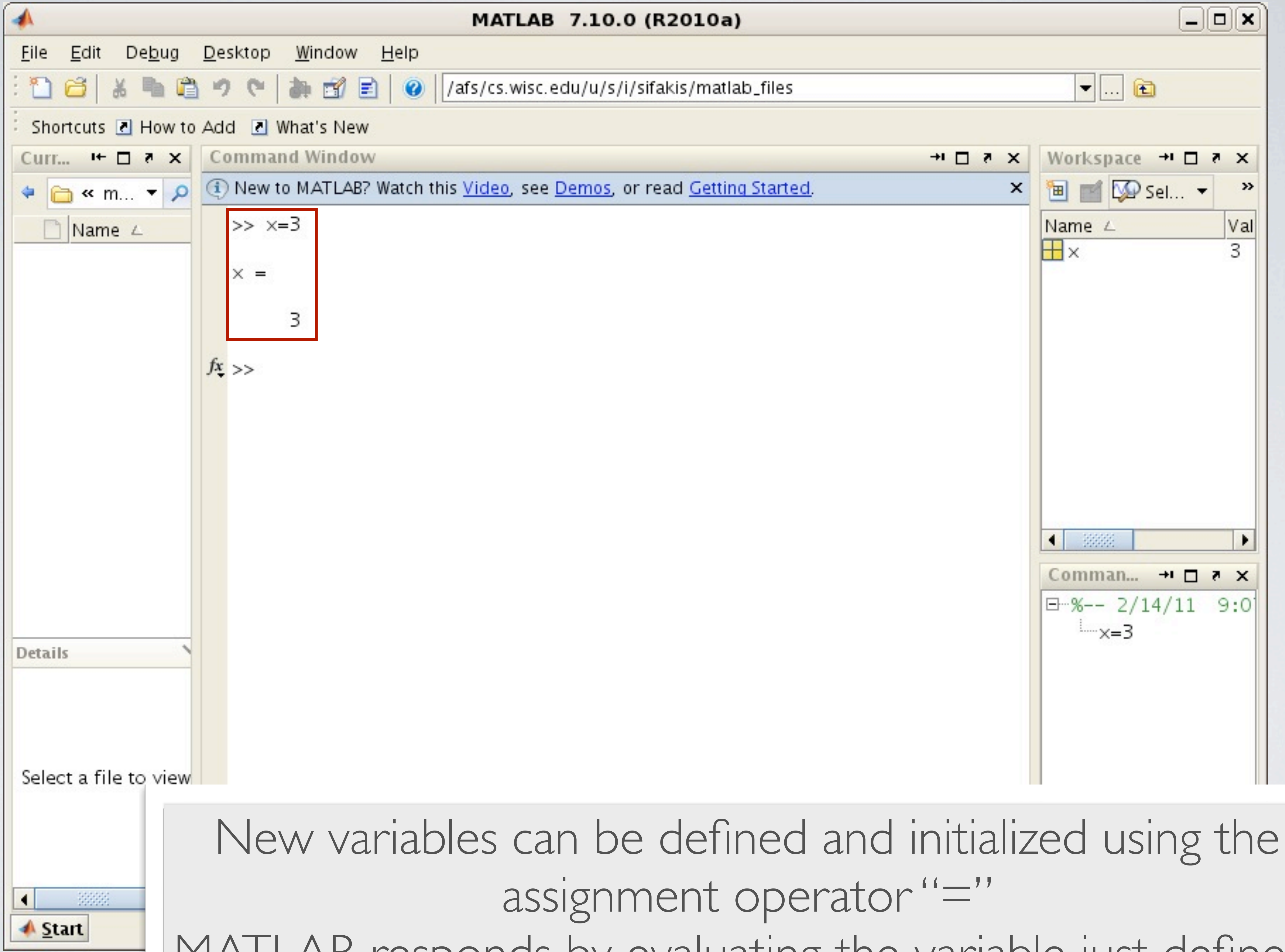
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files

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

```
>> x=3

x =

     3

>>
```

Name	Val
x	3

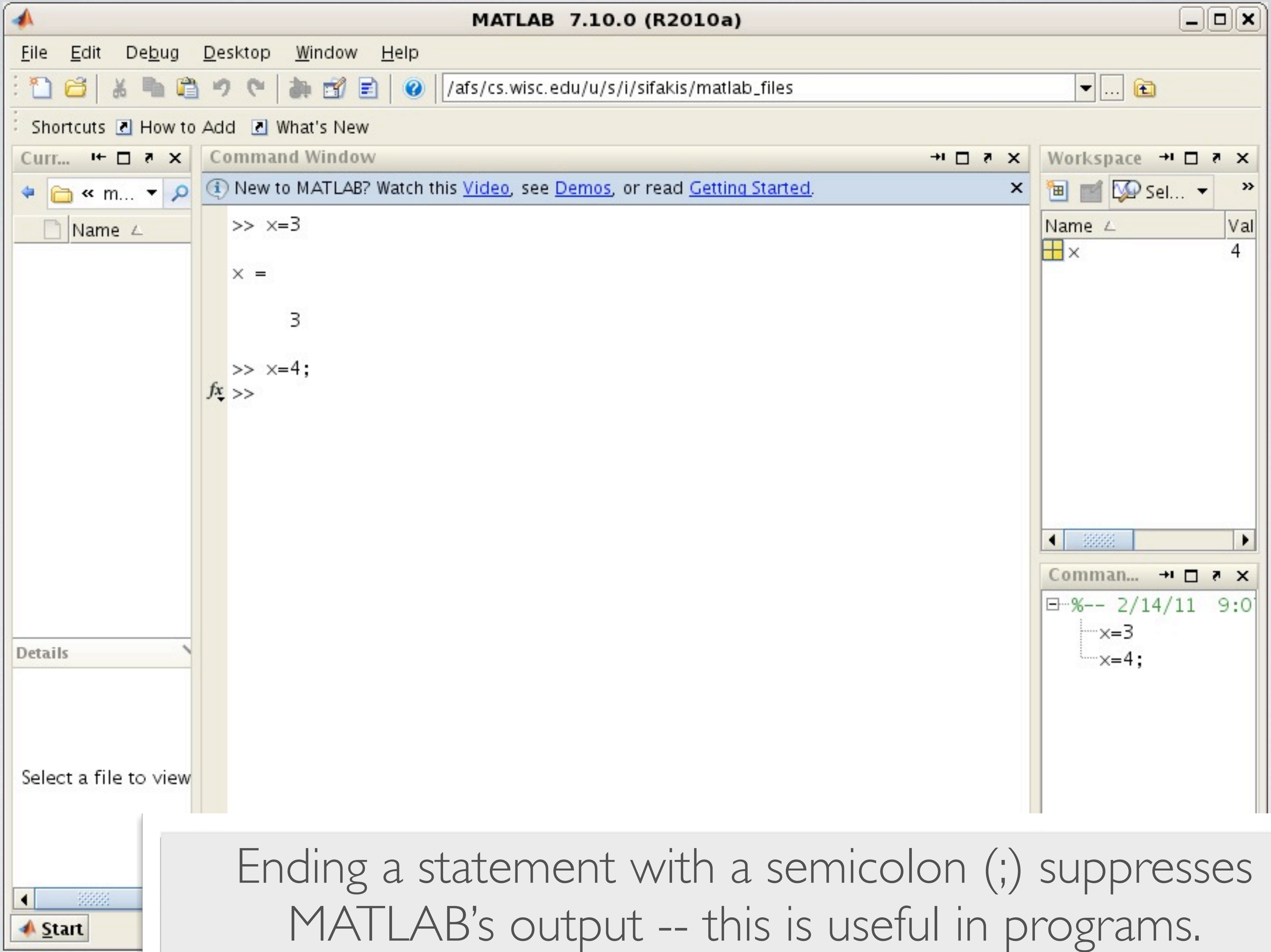
```
%-- 2/14/11  9:0
   x=3
```


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
Shortcuts How to Add What's New
Curr...
Name
Details
Select a file to view
Start
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Name
Val
x
3
Comman...
%-- 2/14/11 9:0
x=3

```
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
```

```
>> x=3

x =

     3

>>
```

New variables can be defined and initialized using the assignment operator "="
MATLAB responds by evaluating the variable just defined.


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Start
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Comman...

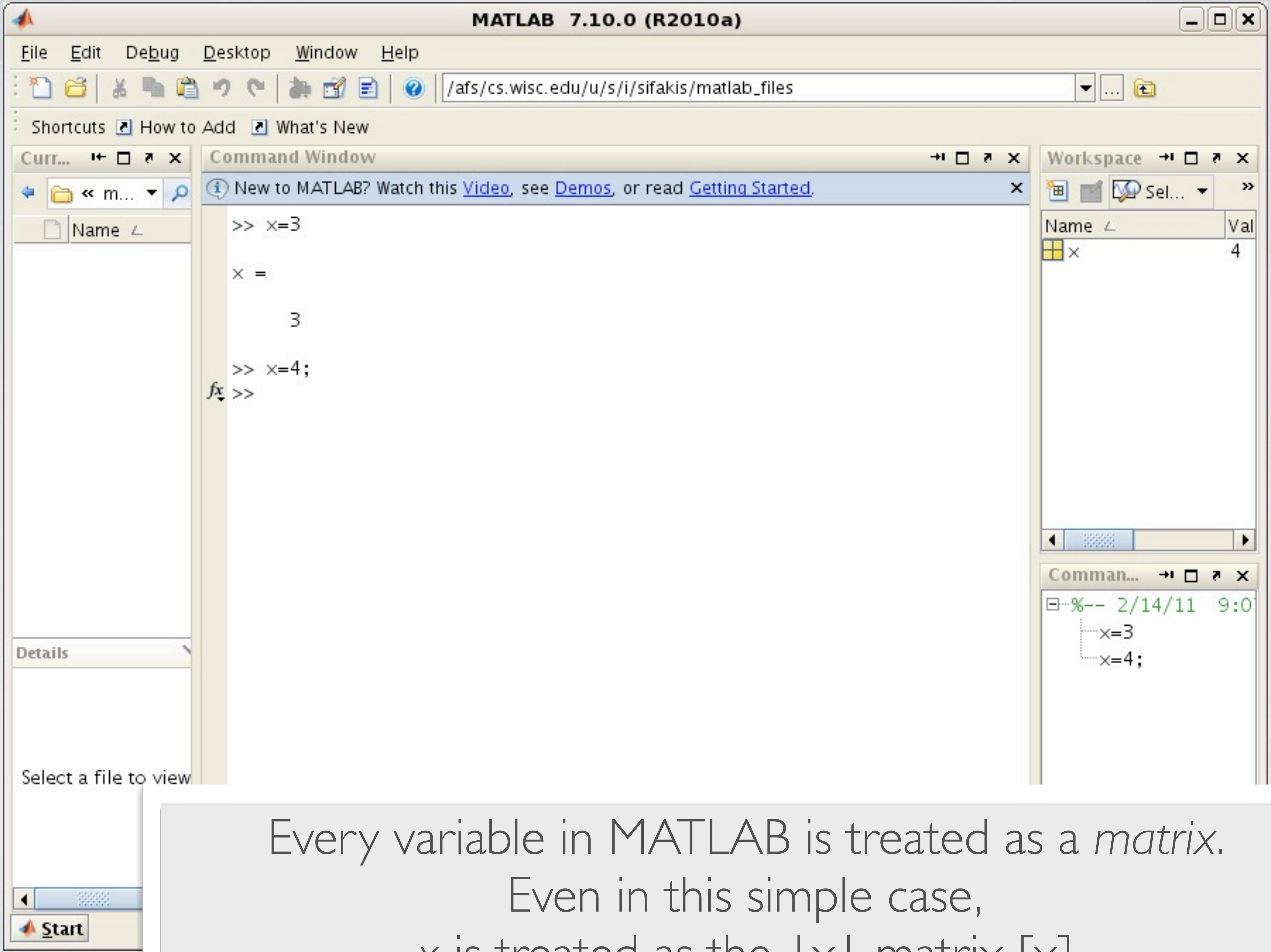
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files

```
>> x=3

x =

     3

>> x=4;
>>
```

Name	Val
x	4

```
%-- 2/14/11  9:0
x=3
x=4;
```

Ending a statement with a semicolon (;) suppresses MATLAB's output -- this is useful in programs.



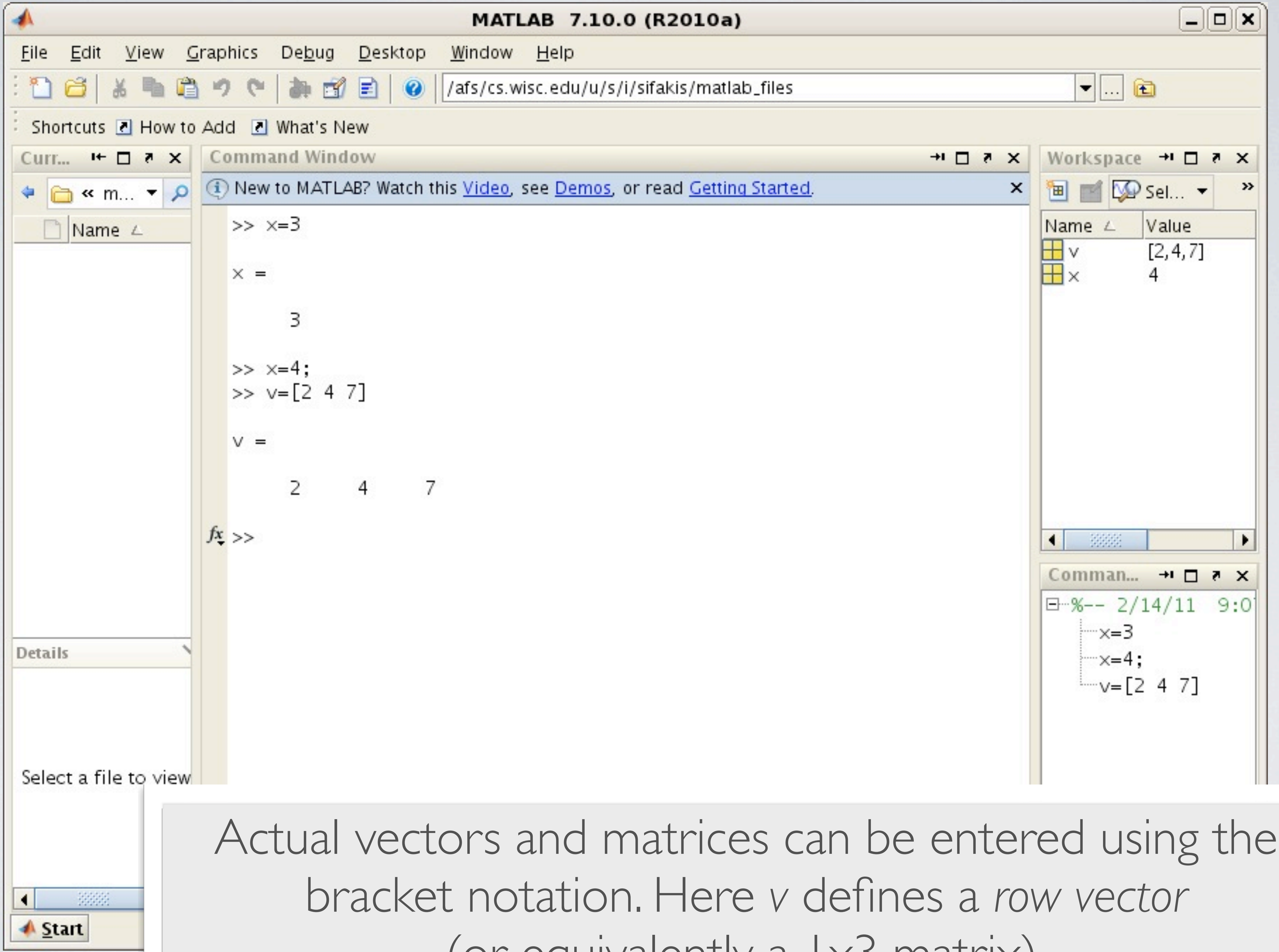
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files

```
>> x=3

x =

     3

>> x=4;
>>
```

Name	Val
x	4

```
%-- 2/14/11  9:0
  x=3
  x=4;
```

Every variable in MATLAB is treated as a *matrix*.
Even in this simple case,
x is treated as the 1x1 matrix [x]


MATLAB 7.10.0 (R2010a)
File Edit View Graphics Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Start
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Comman...

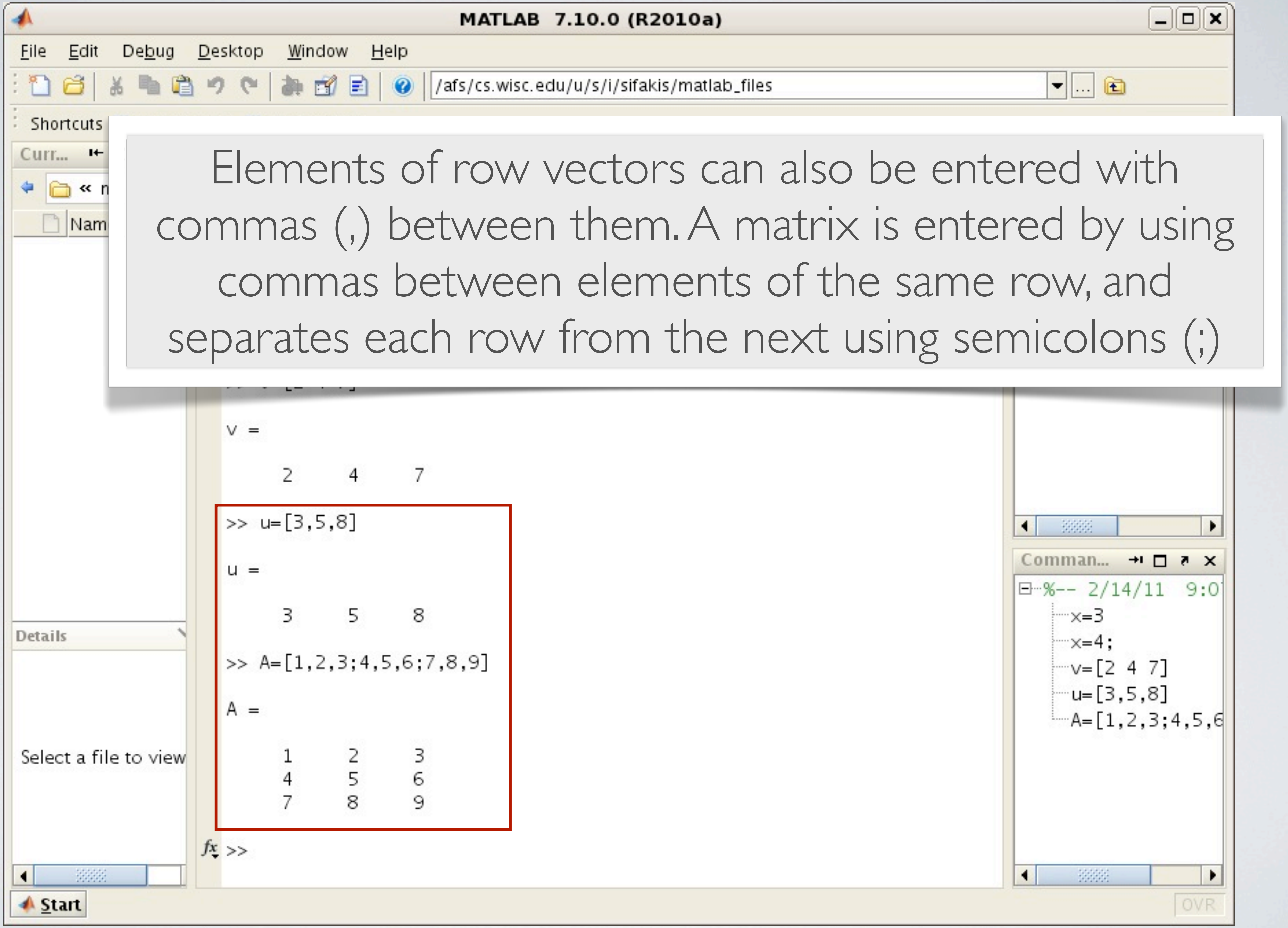
```
>> x=3

x =

     3

>> x=4;
>> v=[2 4 7]

v =

     2     4     7

>>
```

Name	Value
v	[2,4,7]
x	4

```
%-- 2/14/11   9:0
   x=3
   x=4;
   v=[2 4 7]
```

Actual vectors and matrices can be entered using the bracket notation. Here *v* defines a *row vector* (or, equivalently a 1x3 matrix).

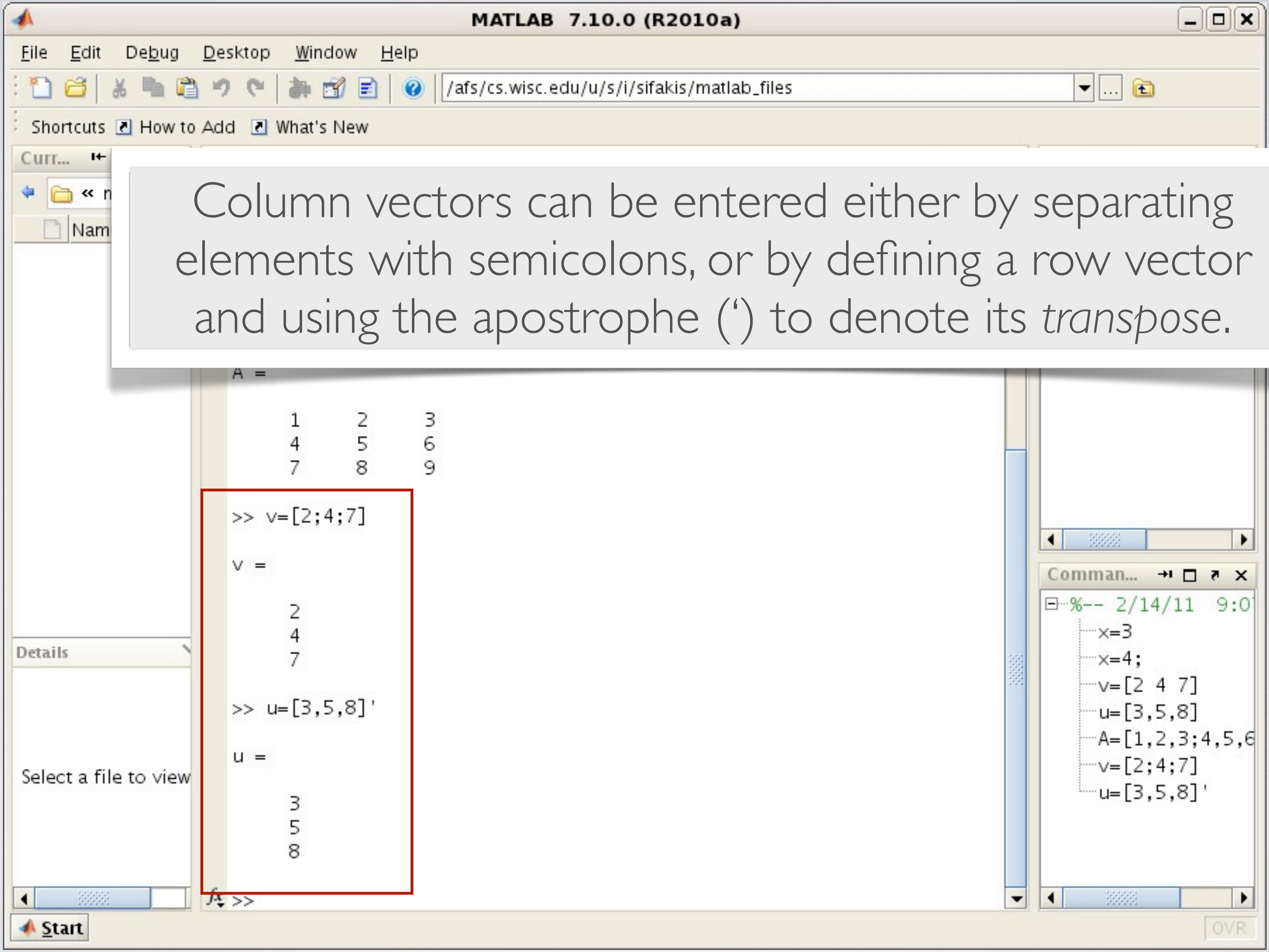
Elements of row vectors can also be entered with commas (,) between them. A matrix is entered by using commas between elements of the same row, and separates each row from the next using semicolons (;)



```
v =

     2     4     7

>> u=[3,5,8]

u =

     3     5     8

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

A =

     1     2     3
     4     5     6
     7     8     9

>>
```

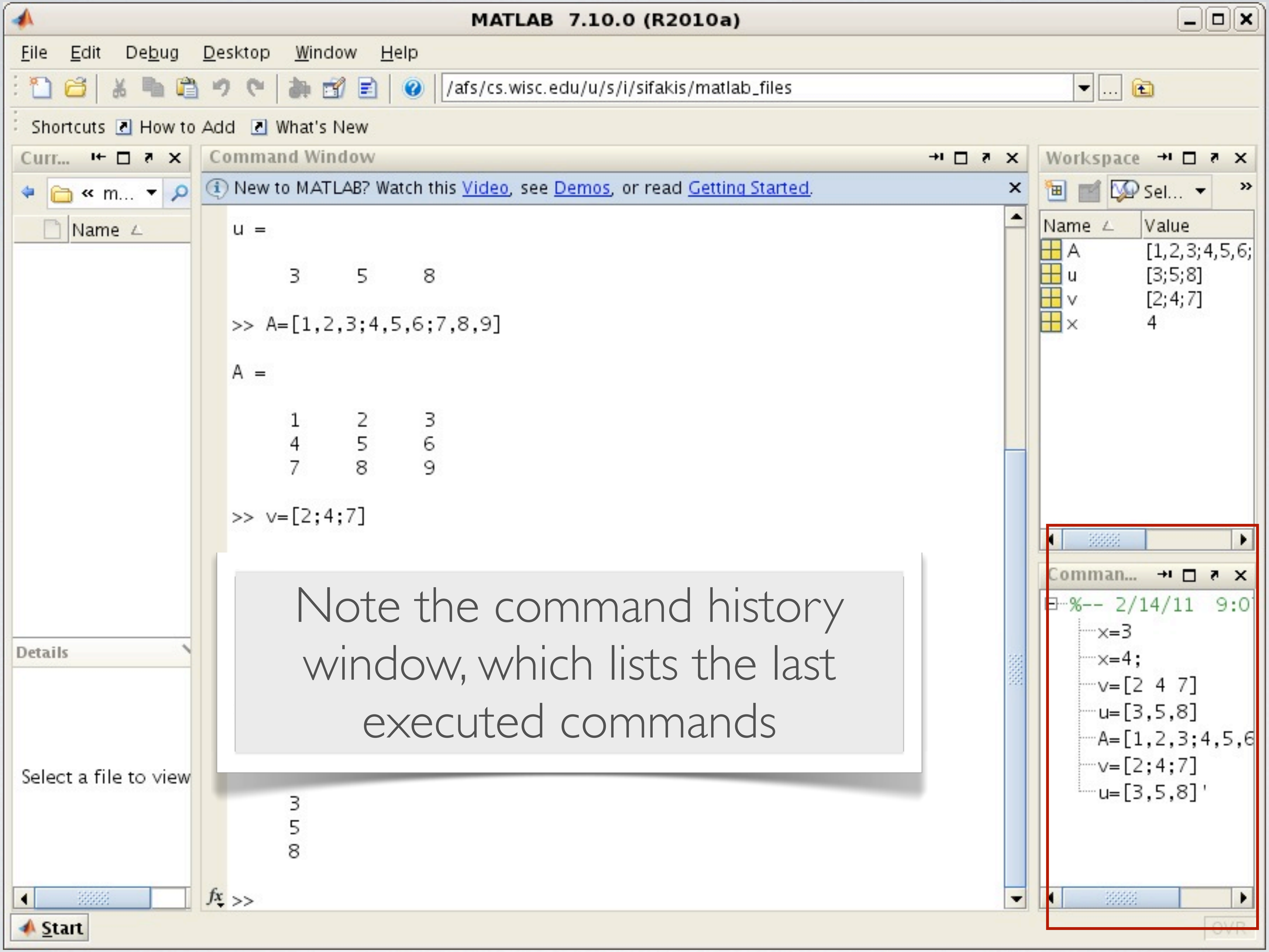
```
%-- 2/14/11   9:0
x=3
x=4;
v=[2 4 7]
u=[3,5,8]
A=[1,2,3;4,5,6
```

Column vectors can be entered either by separating elements with semicolons, or by defining a row vector and using the apostrophe (‘) to denote its *transpose*.


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
Nam
Details
Select a file to view
Comman...
Start
OVR

```
A =

     1     2     3
     4     5     6
     7     8     9

>> v=[2;4;7]

v =

     2
     4
     7

>> u=[3,5,8]'

u =

     3
     5
     8

>>
```

```
%-- 2/14/11  9:0
x=3
x=4;
v=[2 4 7]
u=[3,5,8]
A=[1,2,3;4,5,6
v=[2;4;7]
u=[3,5,8]'
```



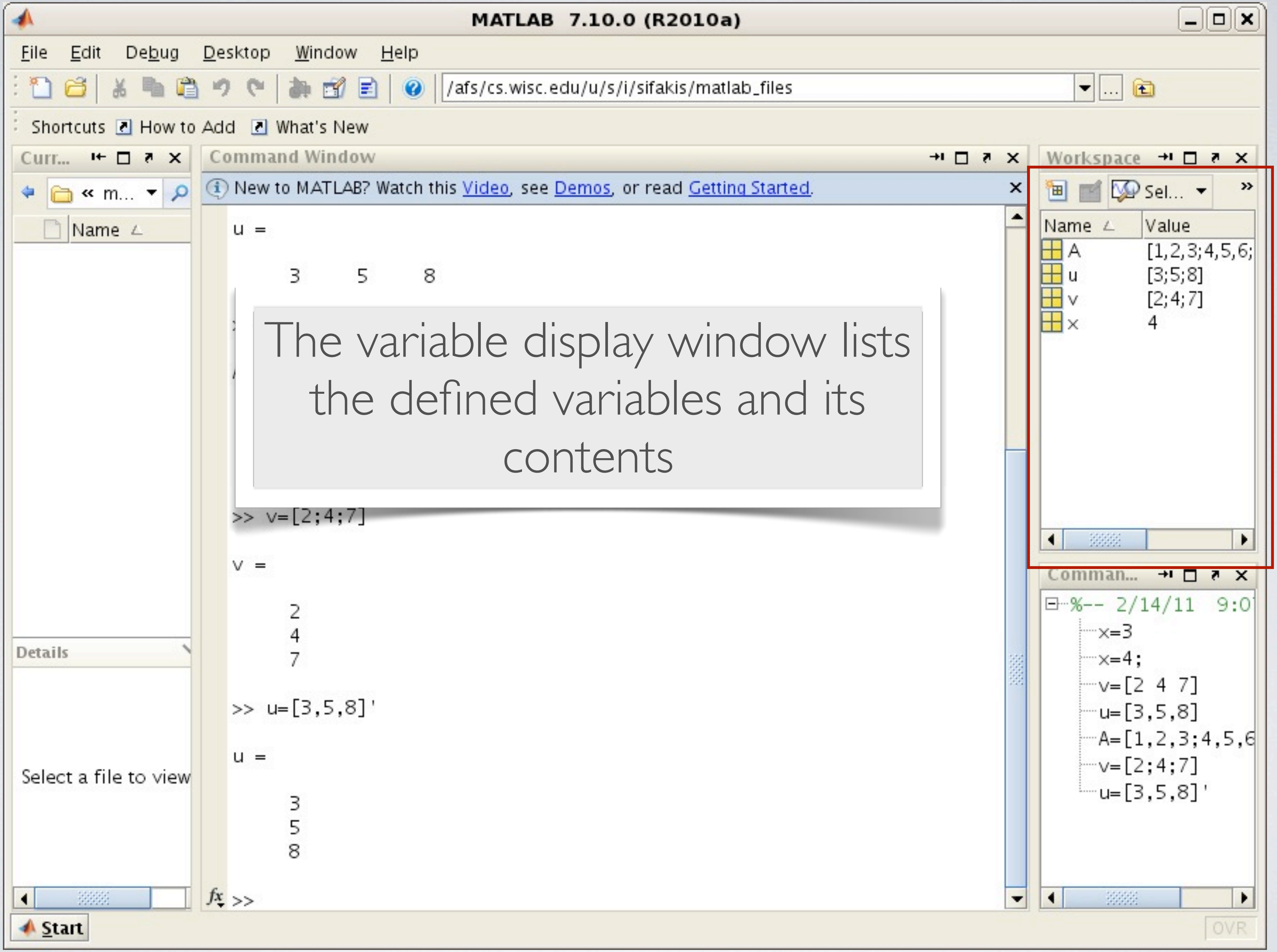
```
u =

     3     5     8

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

A =

     1     2     3
     4     5     6
     7     8     9

>> v=[2;4;7]

     3
     5
     8

>>
```

Name	Value
A	[1,2,3;4,5,6;
u	[3;5;8]
v	[2;4;7]
x	4

```
%-- 2/14/11  9:0
  x=3
  x=4;
  v=[2 4 7]
  u=[3,5,8]
  A=[1,2,3;4,5,6
  v=[2;4;7]
  u=[3,5,8]'
```

Note the command history window, which lists the last executed commands

The variable display window lists the defined variables and its contents



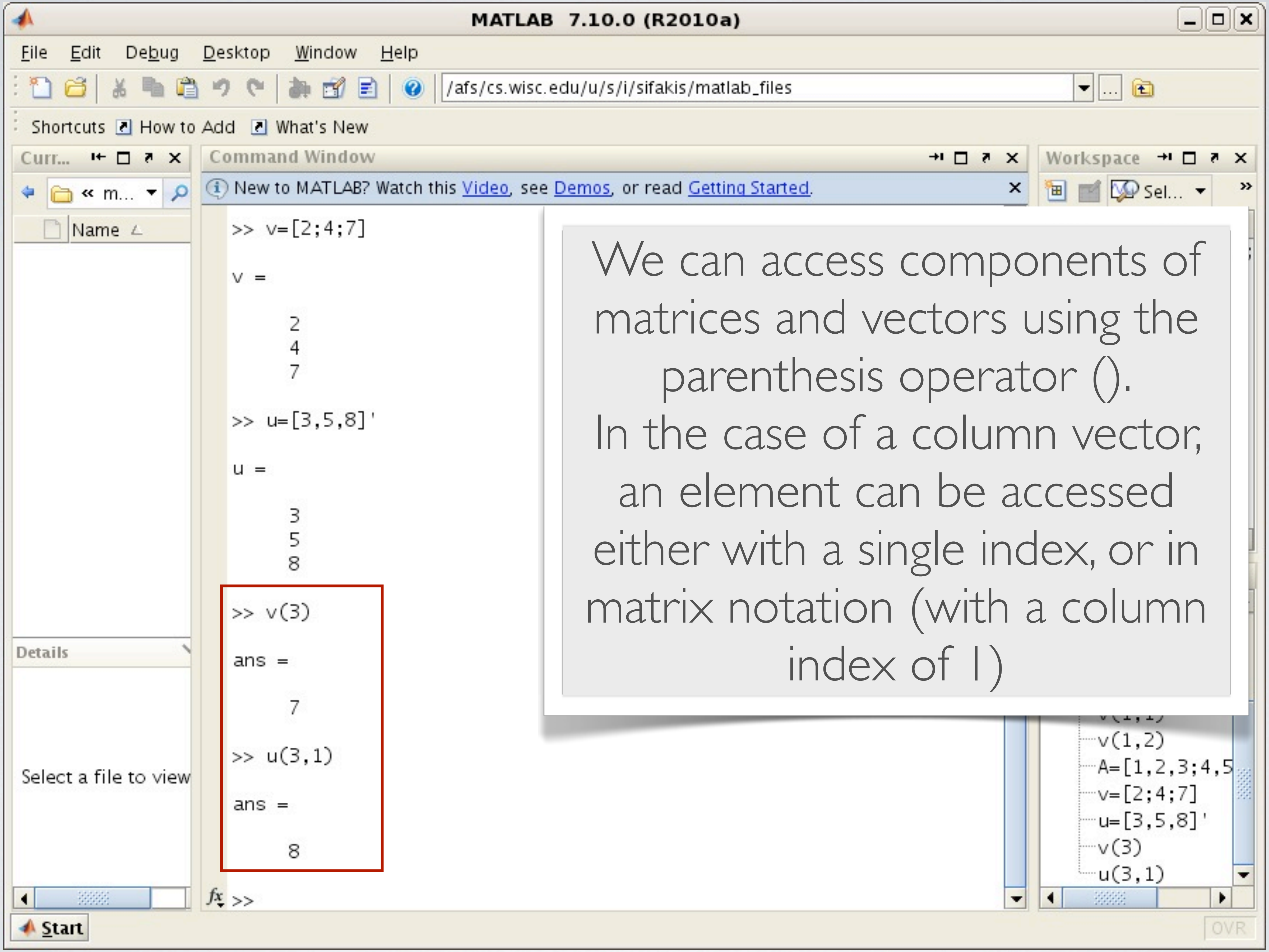
Select a file to view

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

```
u =

     3     5     8

>> v=[2;4;7]

v =

     2
     4
     7

>> u=[3,5,8]'

u =

     3
     5
     8

>>
```

Name	Value
A	[1,2,3;4,5,6;
u	[3;5;8]
v	[2;4;7]
x	4

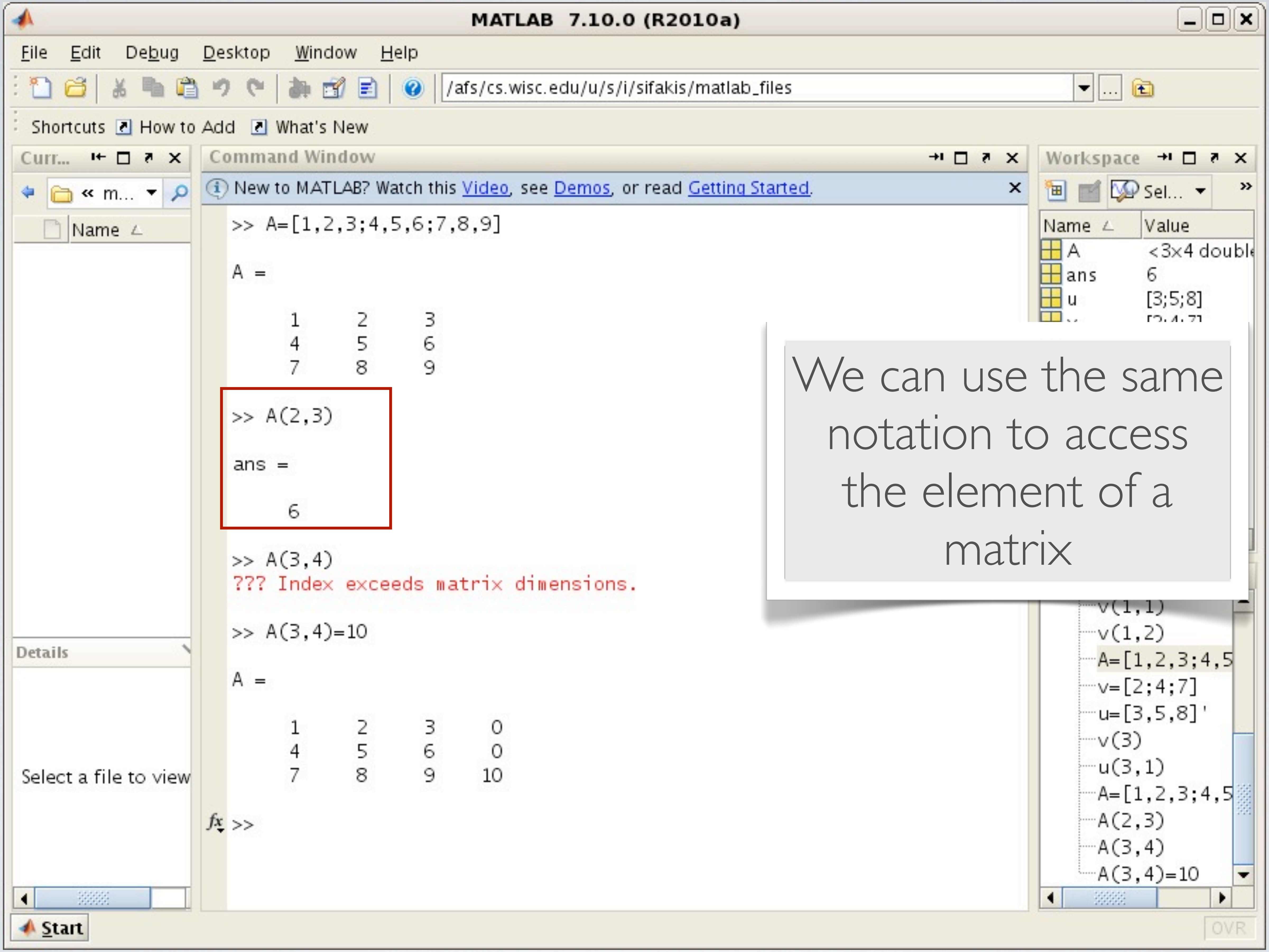
```
%-- 2/14/11  9:0
   x=3
   x=4;
   v=[2 4 7]
   u=[3,5,8]
   A=[1,2,3;4,5,6
   v=[2;4;7]
   u=[3,5,8]'
```


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
Command Window
Workspace
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
« m...
Name
Sel...
Details
Select a file to view
Start
OVR

```
>> v=[2;4;7]

v =

     2
     4
     7

>> u=[3,5,8]'

u =

     3
     5
     8

>> v(3)

ans =

     7

>> u(3,1)

ans =

     8

>>
```

```
v(1,1)
v(1,2)
A=[1,2,3;4,5
v=[2;4;7]
u=[3,5,8]'
v(3)
u(3,1)
```

We can access components of matrices and vectors using the parenthesis operator ().
In the case of a column vector, an element can be accessed either with a single index, or in matrix notation (with a column index of 1)

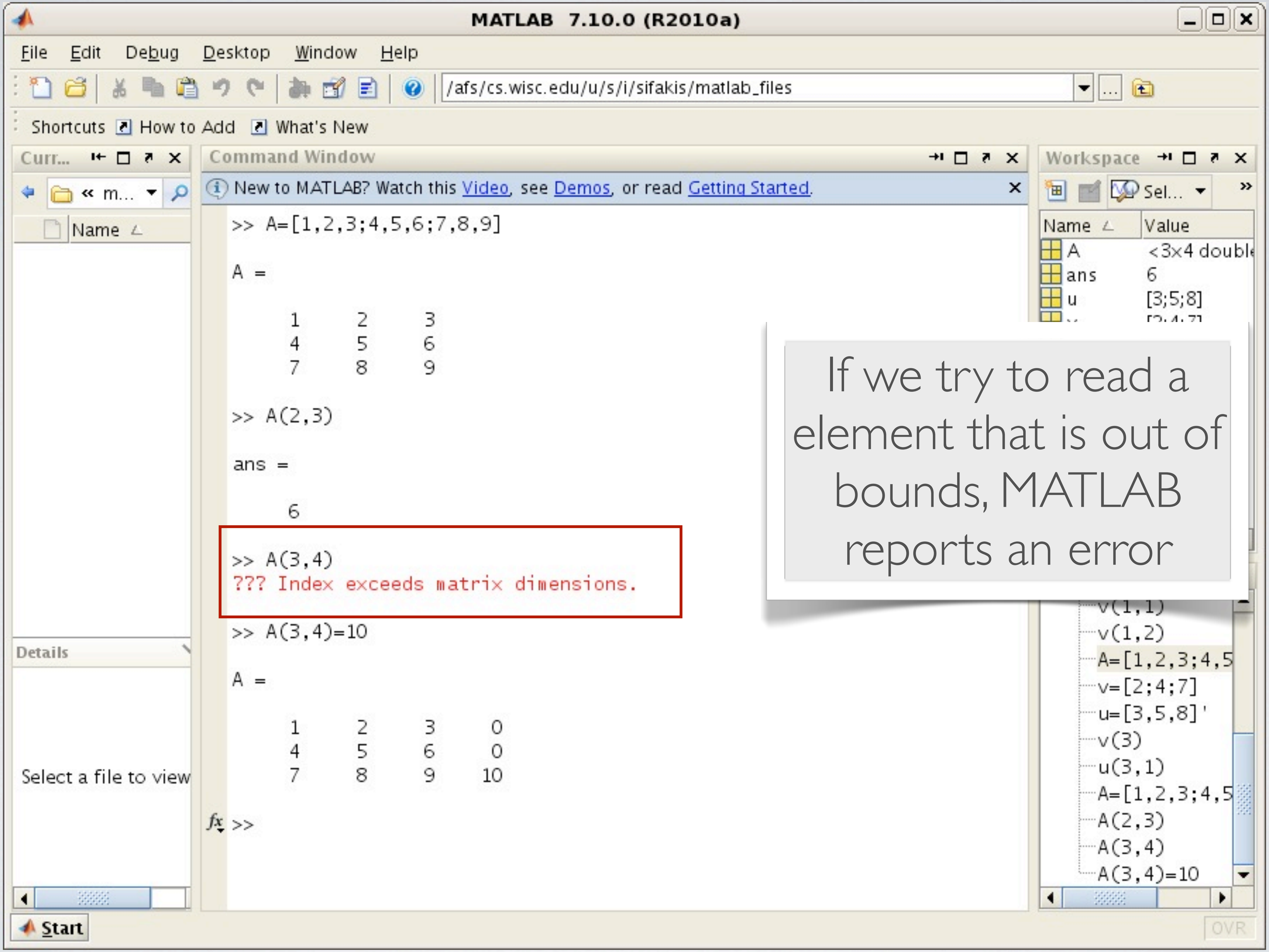

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Start
OVR

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

A =

     1     2     3
     4     5     6
     7     8     9

>> A(2,3)

ans =

     6

>> A(3,4)
??? Index exceeds matrix dimensions.

>> A(3,4)=10

A =

     1     2     3     0
     4     5     6     0
     7     8     9    10

>>
```

Name	Value
A	<3x4 double
ans	6
u	[3;5;8]

```
v(1,1)
v(1,2)
A=[1,2,3;4,5
v=[2;4;7]
u=[3,5,8]'
v(3)
u(3,1)
A=[1,2,3;4,5
A(2,3)
A(3,4)
A(3,4)=10
```

We can use the same notation to access the element of a matrix

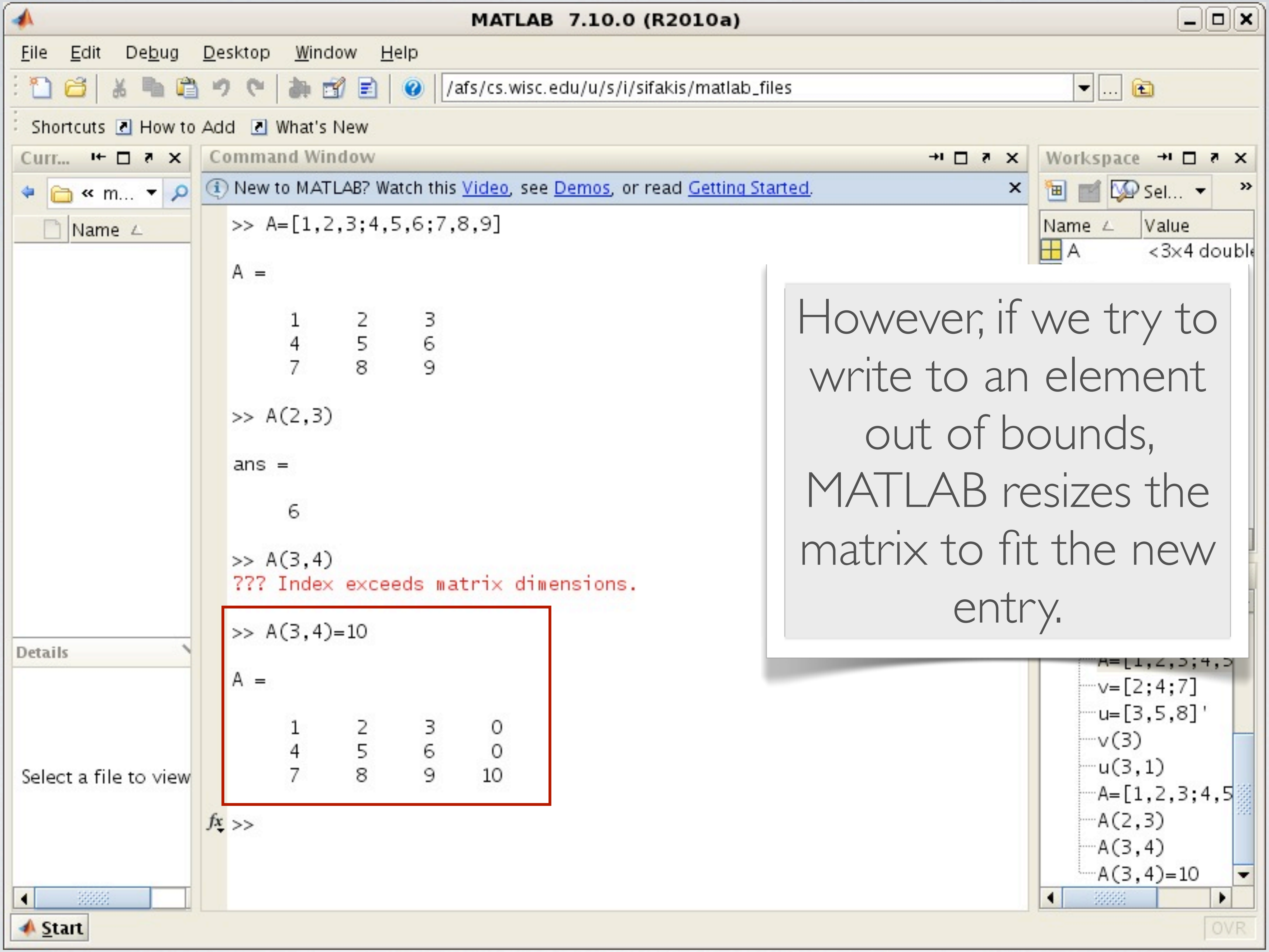

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Start
OVR

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

A =

     1     2     3
     4     5     6
     7     8     9

>> A(2,3)

ans =

     6

>> A(3,4)
??? Index exceeds matrix dimensions.

>> A(3,4)=10

A =

     1     2     3     0
     4     5     6     0
     7     8     9    10

>>
```

Name	Value
A	<3x4 double
ans	6
u	[3;5;8]

```
v(1,1)
v(1,2)
A=[1,2,3;4,5
v=[2;4;7]
u=[3,5,8]'
v(3)
u(3,1)
A=[1,2,3;4,5
A(2,3)
A(3,4)
A(3,4)=10
```

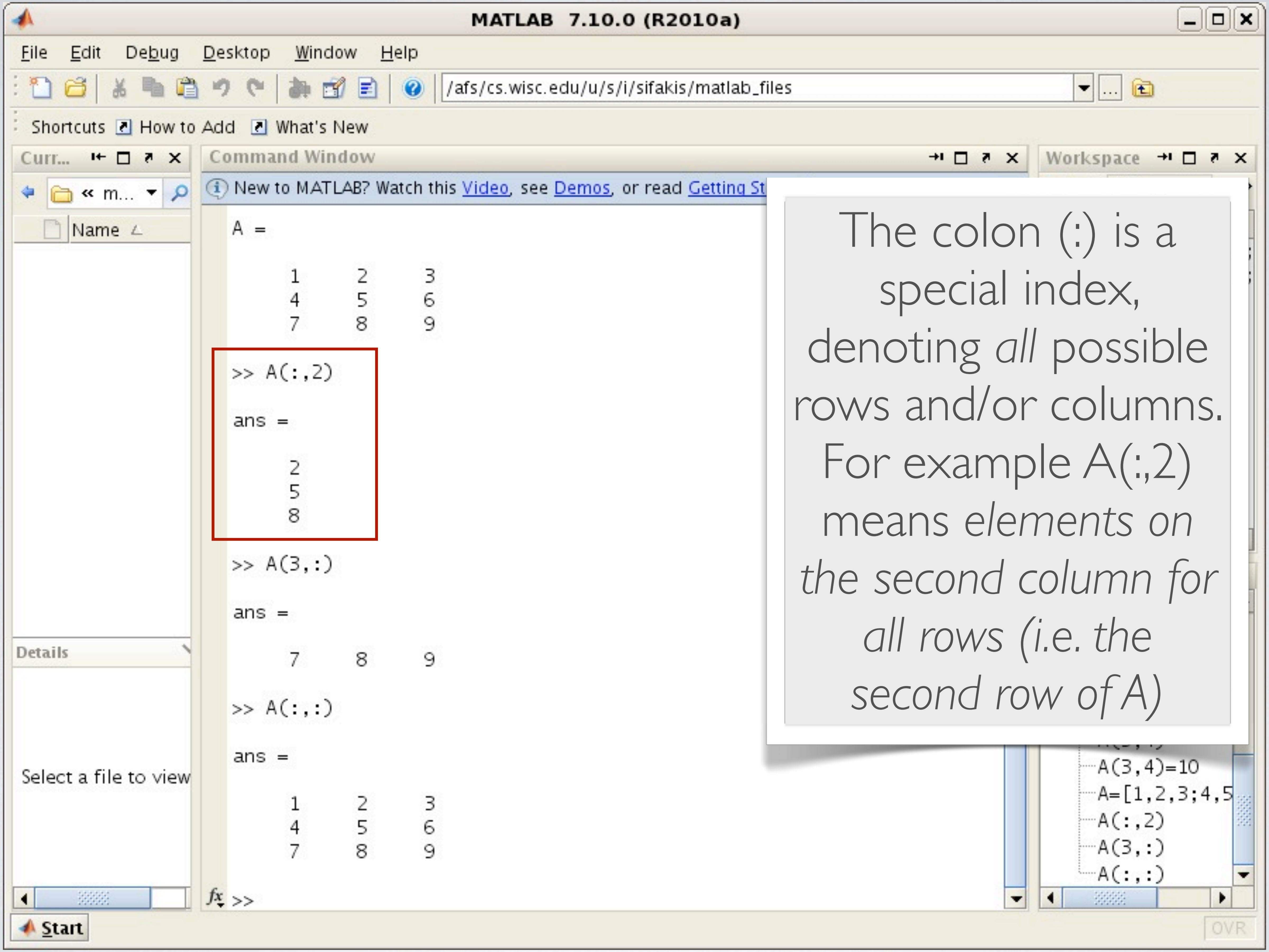
If we try to read a element that is out of bounds, MATLAB reports an error


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Name Value
A <3x4 double
Start
OVR

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

A =

     1     2     3
     4     5     6
     7     8     9

>> A(2,3)

ans =

     6

>> A(3,4)
??? Index exceeds matrix dimensions.
>> A(3,4)=10

A =

     1     2     3     0
     4     5     6     0
     7     8     9    10

>>
```

```
A=[1,2,3;4,5
v=[2;4;7]
u=[3,5,8]'
v(3)
u(3,1)
A=[1,2,3;4,5
A(2,3)
A(3,4)
A(3,4)=10
```

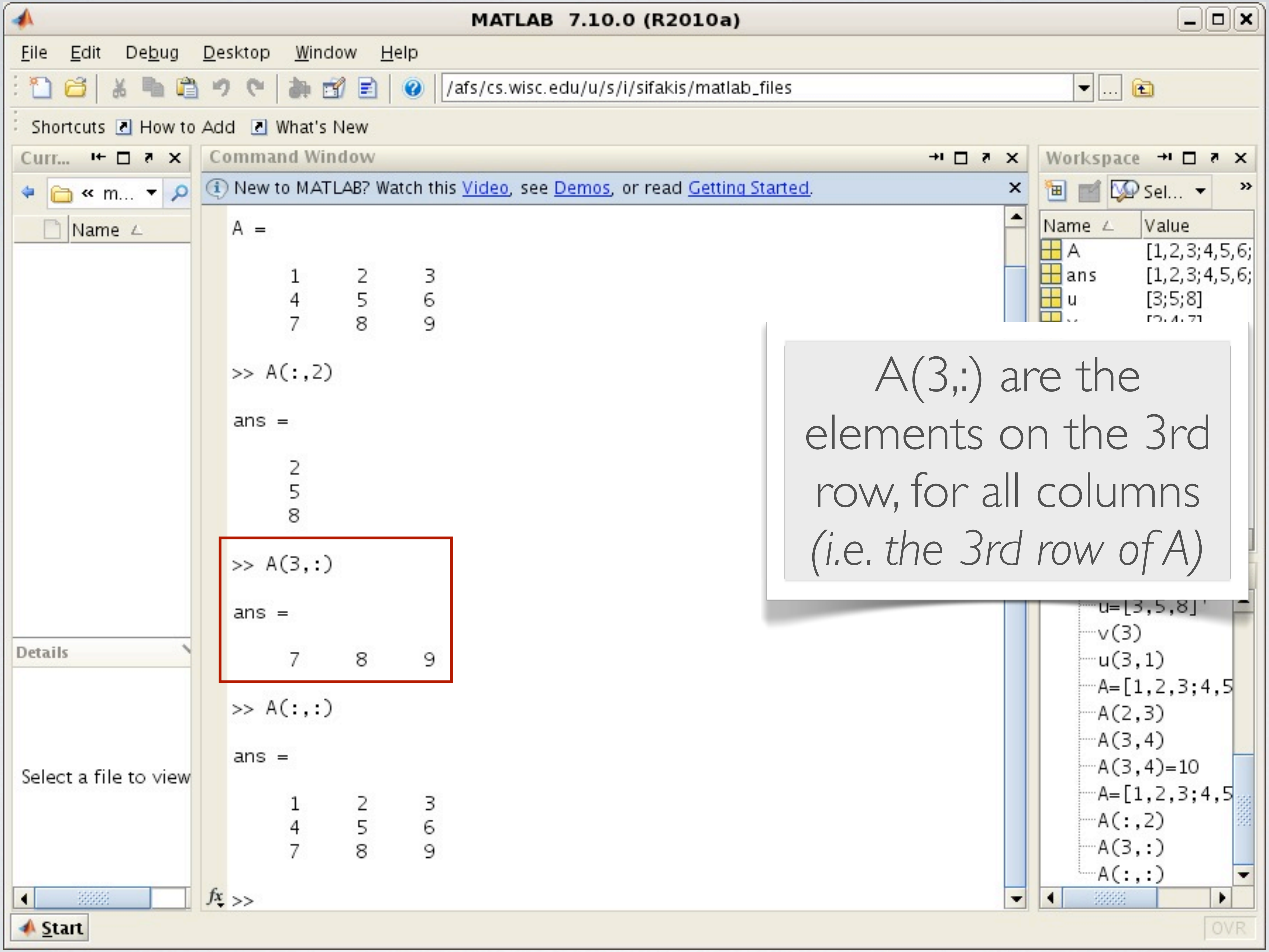
However, if we try to write to an element out of bounds, MATLAB resizes the matrix to fit the new entry.


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
Command Window
Workspace
New to MATLAB? Watch this Video, see Demos, or read Getting St
« m...
Name
Details
Select a file to view
Start
OVR

```
A =

     1     2     3
     4     5     6
     7     8     9

>> A(:,2)

ans =

     2
     5
     8

>> A(3,:)

ans =

     7     8     9

>> A(:,:)

ans =

     1     2     3
     4     5     6
     7     8     9

>>
```

```
A(3,4)=10
A=[1,2,3;4,5
A(:,2)
A(3,:)
A(:,:)
```

The colon (:) is a special index, denoting *all* possible rows and/or columns. For example A(:,2) means *elements on the second column for all rows (i.e. the second row of A)*

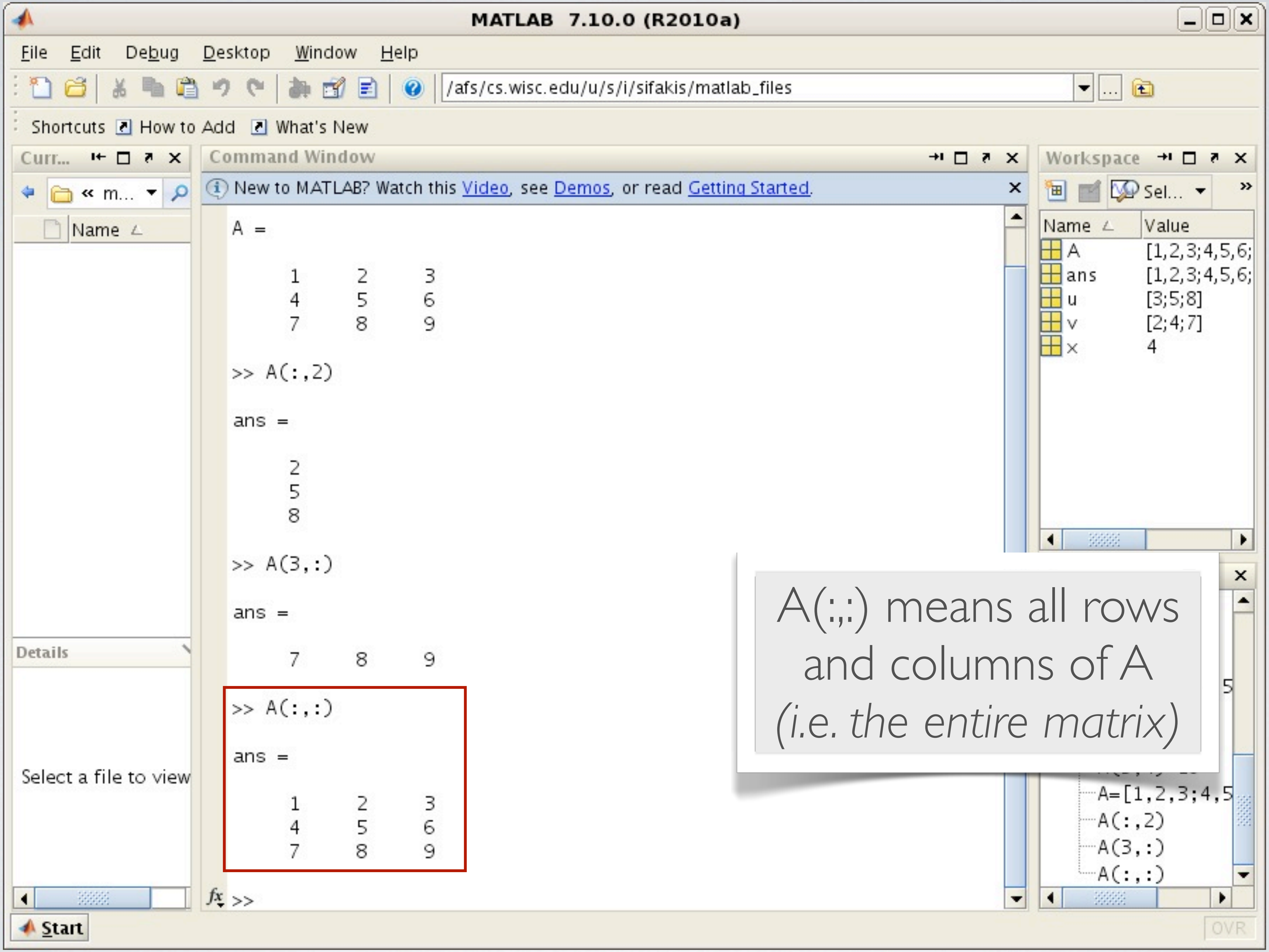

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Start
OVR

```
A =

     1     2     3
     4     5     6
     7     8     9

>> A(:,2)

ans =

     2
     5
     8

>> A(3,:)

ans =

     7     8     9

>> A(:,:)

ans =

     1     2     3
     4     5     6
     7     8     9

>>
```

Name	Value
A	[1,2,3;4,5,6;
ans	[1,2,3;4,5,6;
u	[3;5;8]

```
u=[3,5,8]'
v(3)
u(3,1)
A=[1,2,3;4,5
A(2,3)
A(3,4)
A(3,4)=10
A=[1,2,3;4,5
A(:,2)
A(3,:)
A(:,:)
```

A(3,:) are the elements on the 3rd row, for all columns *(i.e. the 3rd row of A)*

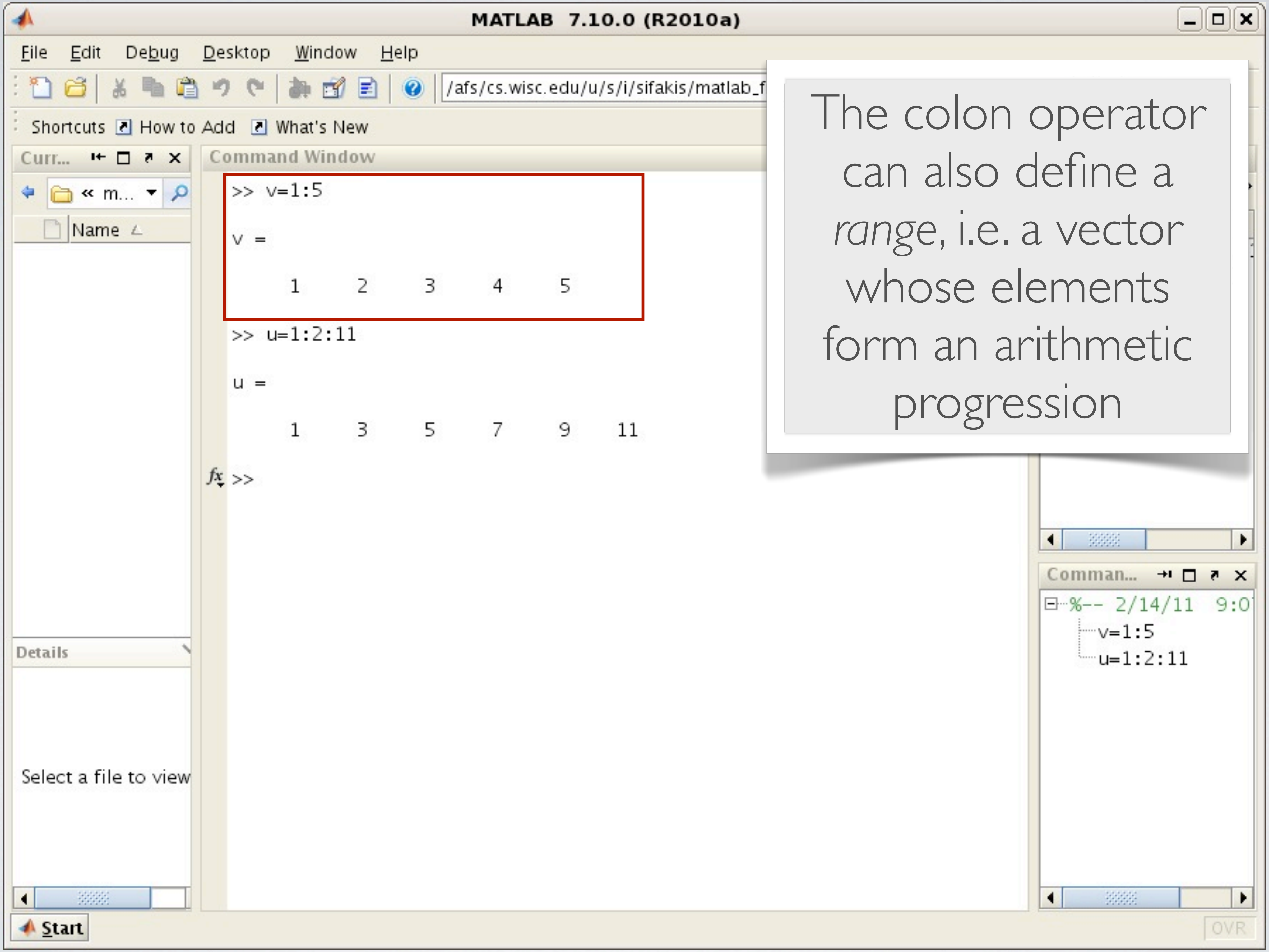

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
New to MATLAB? Watch this Video, see Demos, or read Getting Started.
Workspace
Sel...
Start
OVR

```
A =

     1     2     3
     4     5     6
     7     8     9

>> A(:,2)

ans =

     2
     5
     8

>> A(3,:)

ans =

     7     8     9

>> A(:,:)

ans =

     1     2     3
     4     5     6
     7     8     9

>>
```

Name	Value
A	[1,2,3;4,5,6;
ans	[1,2,3;4,5,6;
u	[3;5;8]
v	[2;4;7]
x	4

```
A=[1,2,3;4,5
A(:,2)
A(3,:)
A(:,:)
```

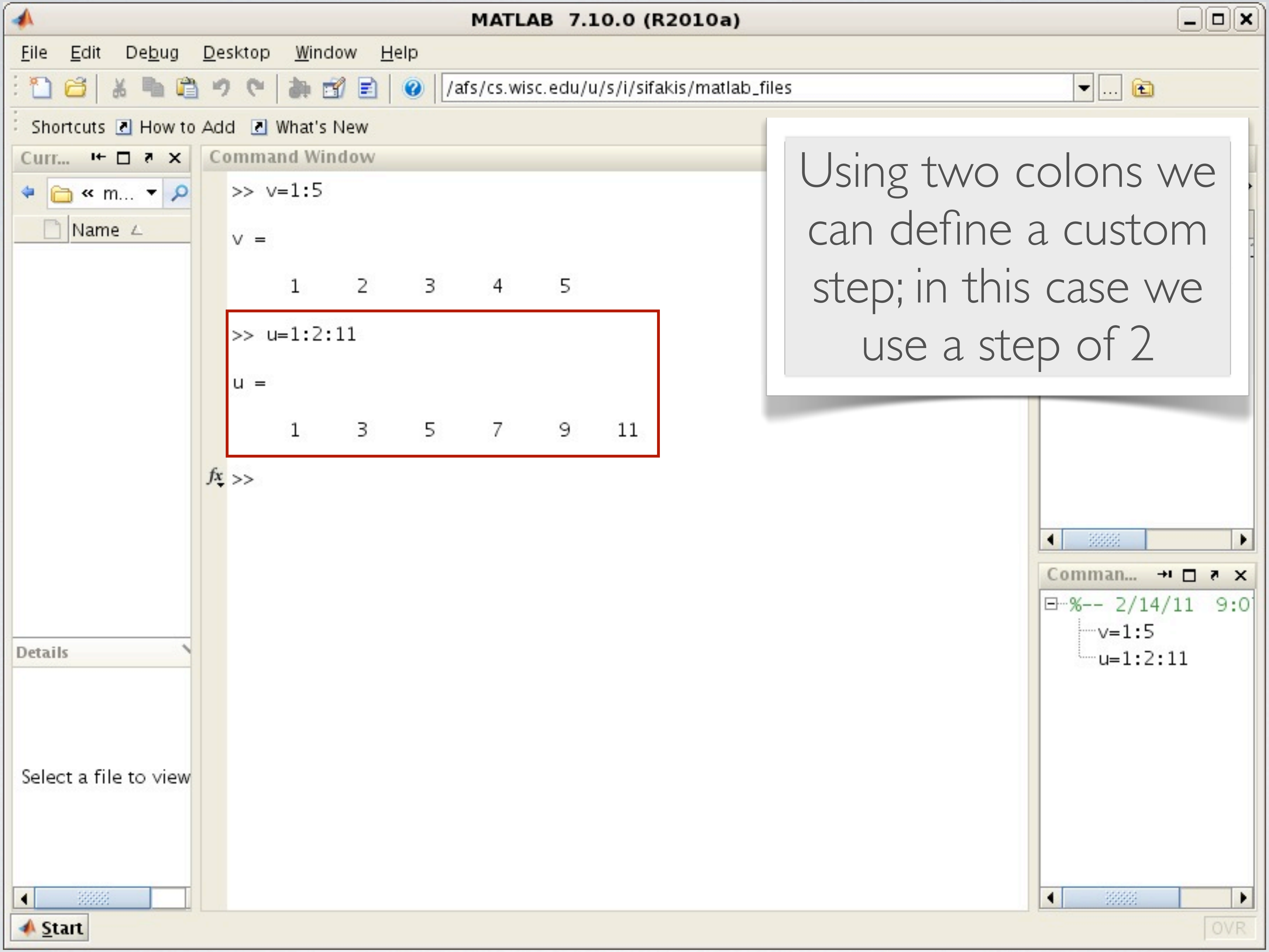
A(:,:) means all rows and columns of A *(i.e. the entire matrix)*


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_f
Shortcuts How to Add What's New
Curr...
« m...
Name
Command Window
Details
Select a file to view
Comman...
Start
OVR

```
>> v=1:5

v =

     1     2     3     4     5

>> u=1:2:11

u =

     1     3     5     7     9    11

>>
```

```
%-- 2/14/11  9:0
v=1:5
u=1:2:11
```

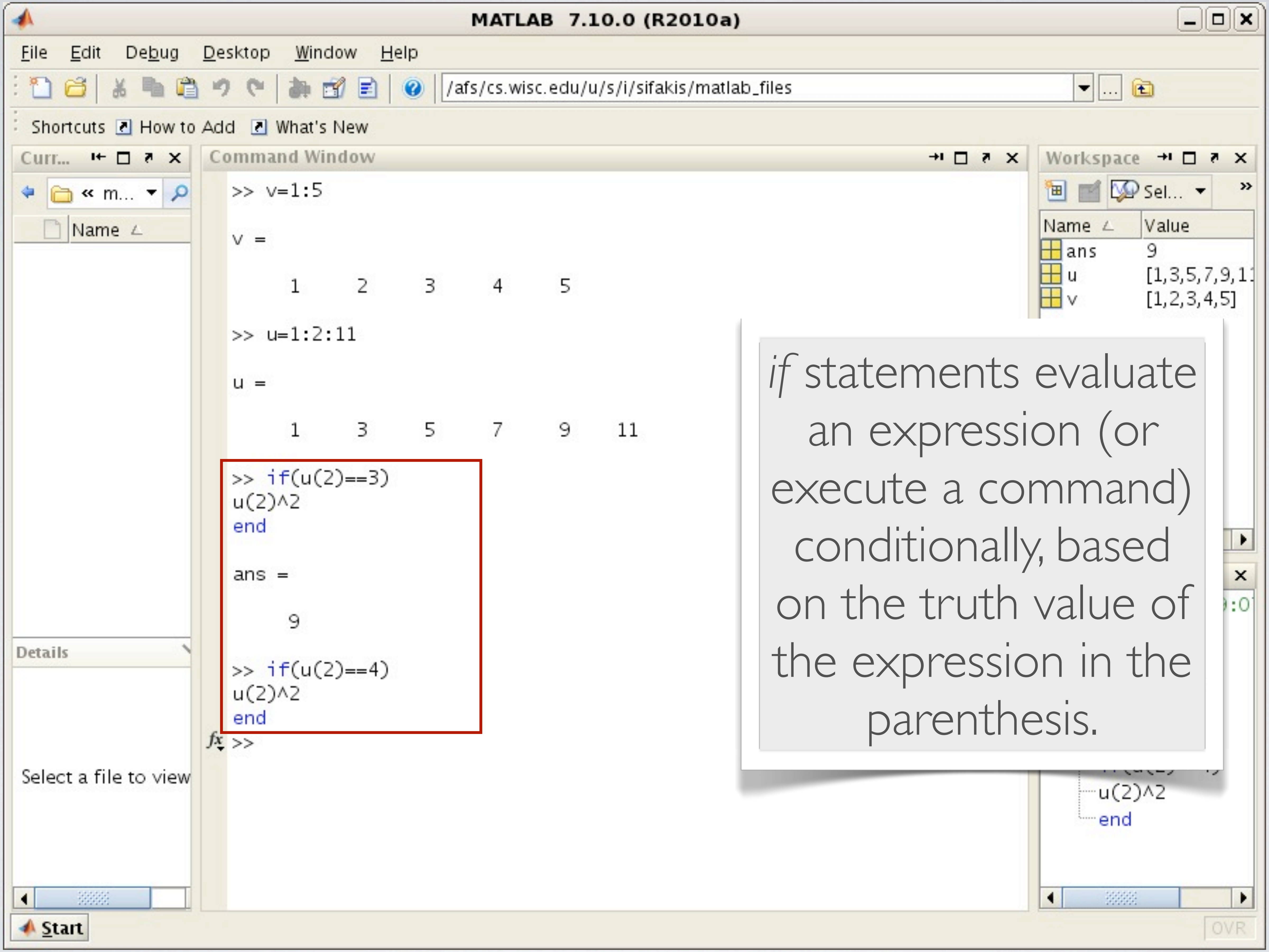
The colon operator can also define a *range*, i.e. a vector whose elements form an arithmetic progression


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
Comman...
Start
OVR

```
>> v=1:5

v =

     1     2     3     4     5

>> u=1:2:11

u =

     1     3     5     7     9    11

>>
```

```
%-- 2/14/11  9:0
v=1:5
u=1:2:11
```

Using two colons we can define a custom step; in this case we use a step of 2

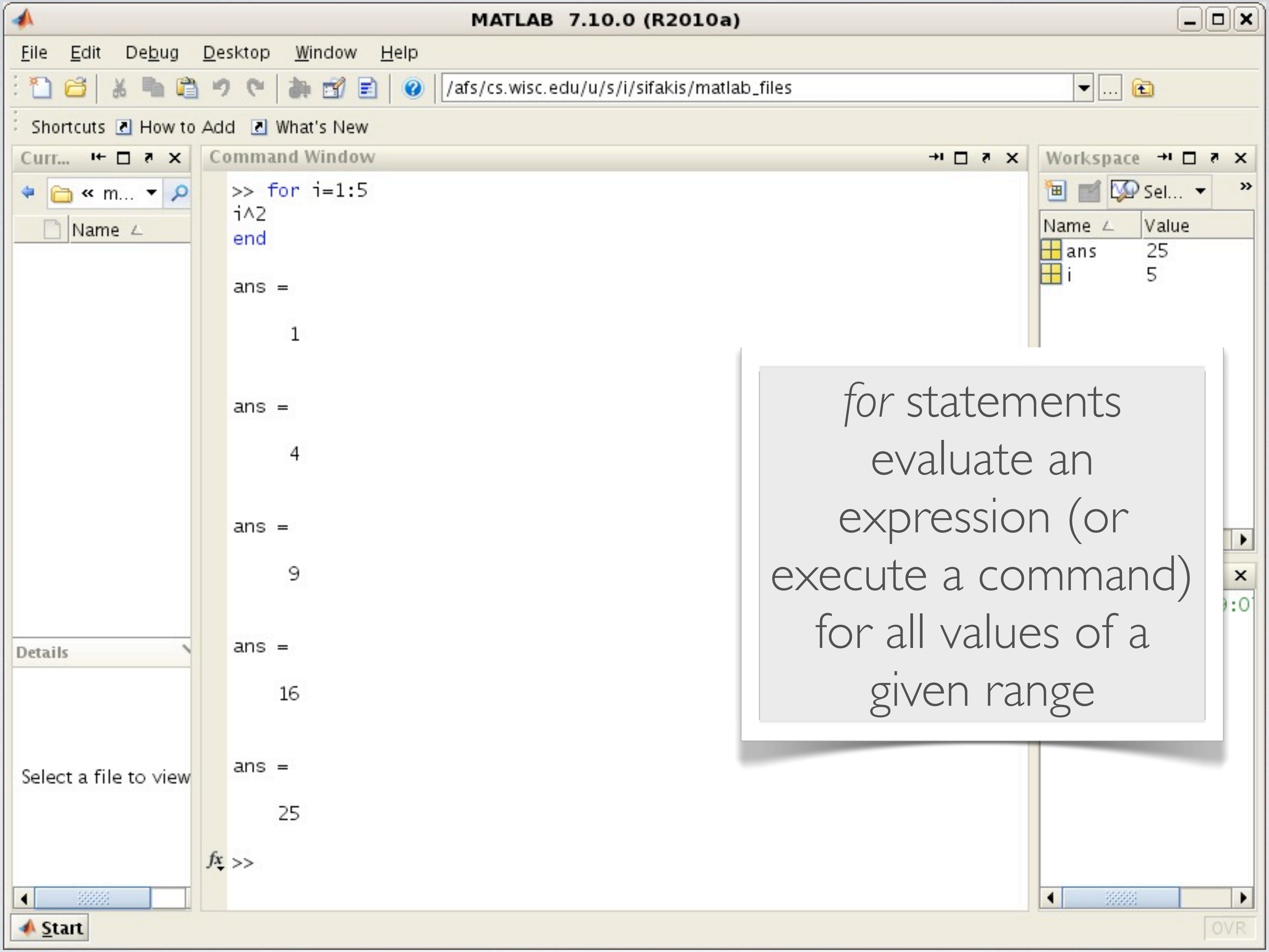

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
Workspace
Sel...
Start
OVR

```
>> v=1:5

v =

     1     2     3     4     5

>> u=1:2:11

u =

     1     3     5     7     9    11

>> if(u(2)==3)
u(2)^2
end

ans =

     9

>> if(u(2)==4)
u(2)^2
end
>>
```

Name	Value
ans	9
u	[1,3,5,7,9,1
v	[1,2,3,4,5]

```
u(2)^2
end
```

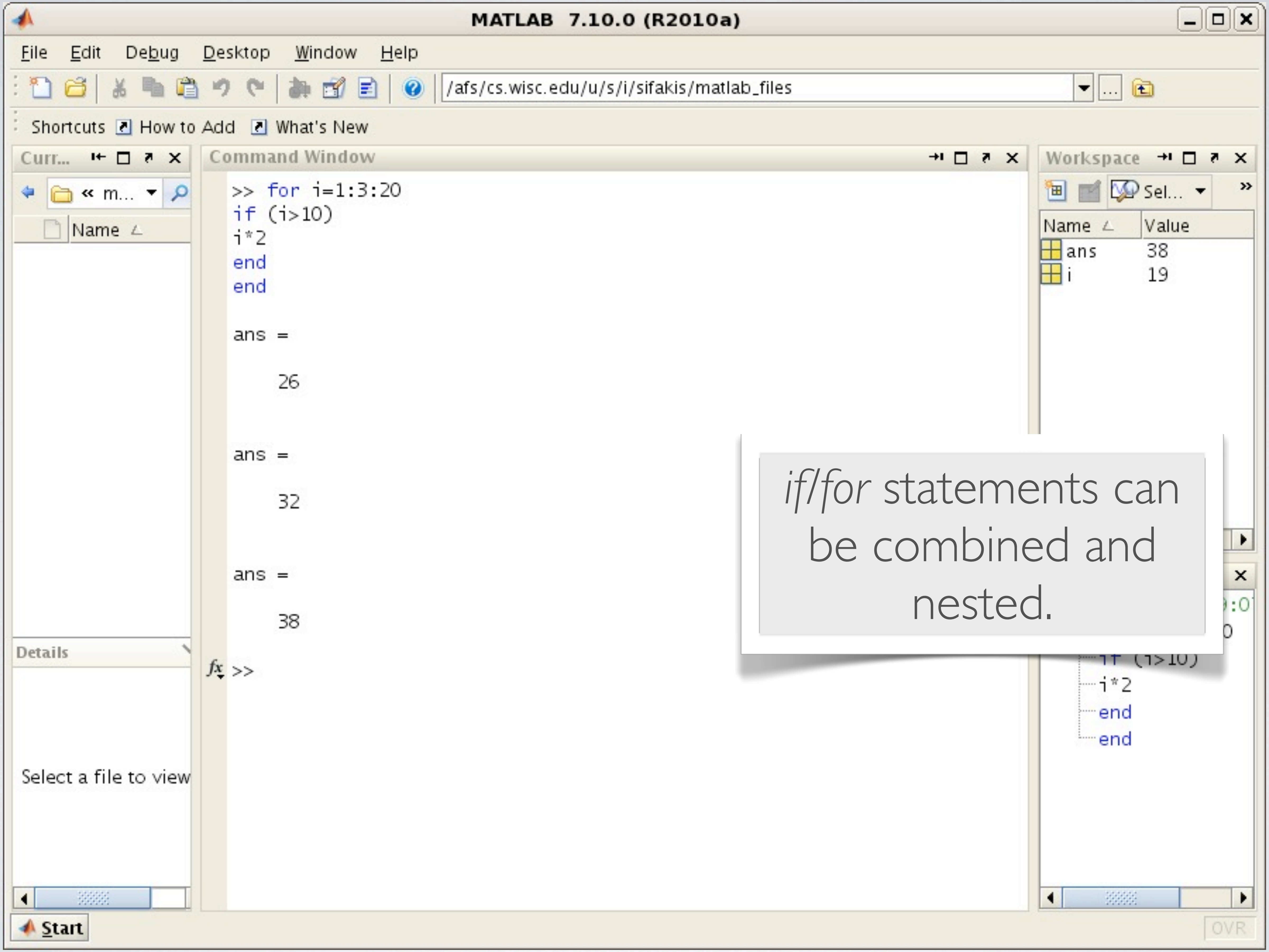
if statements evaluate an expression (or execute a command) conditionally, based on the truth value of the expression in the parenthesis.


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
Name
Details
Select a file to view
Command Window
Workspace
Sel...
Start
OVR

```
>> for i=1:5
i^2
end

ans =

     1


ans =

     4


ans =

     9


ans =

    16


ans =

    25

>>
```

Name	Value
ans	25
i	5

for statements evaluate an expression (or execute a command) for all values of a given range

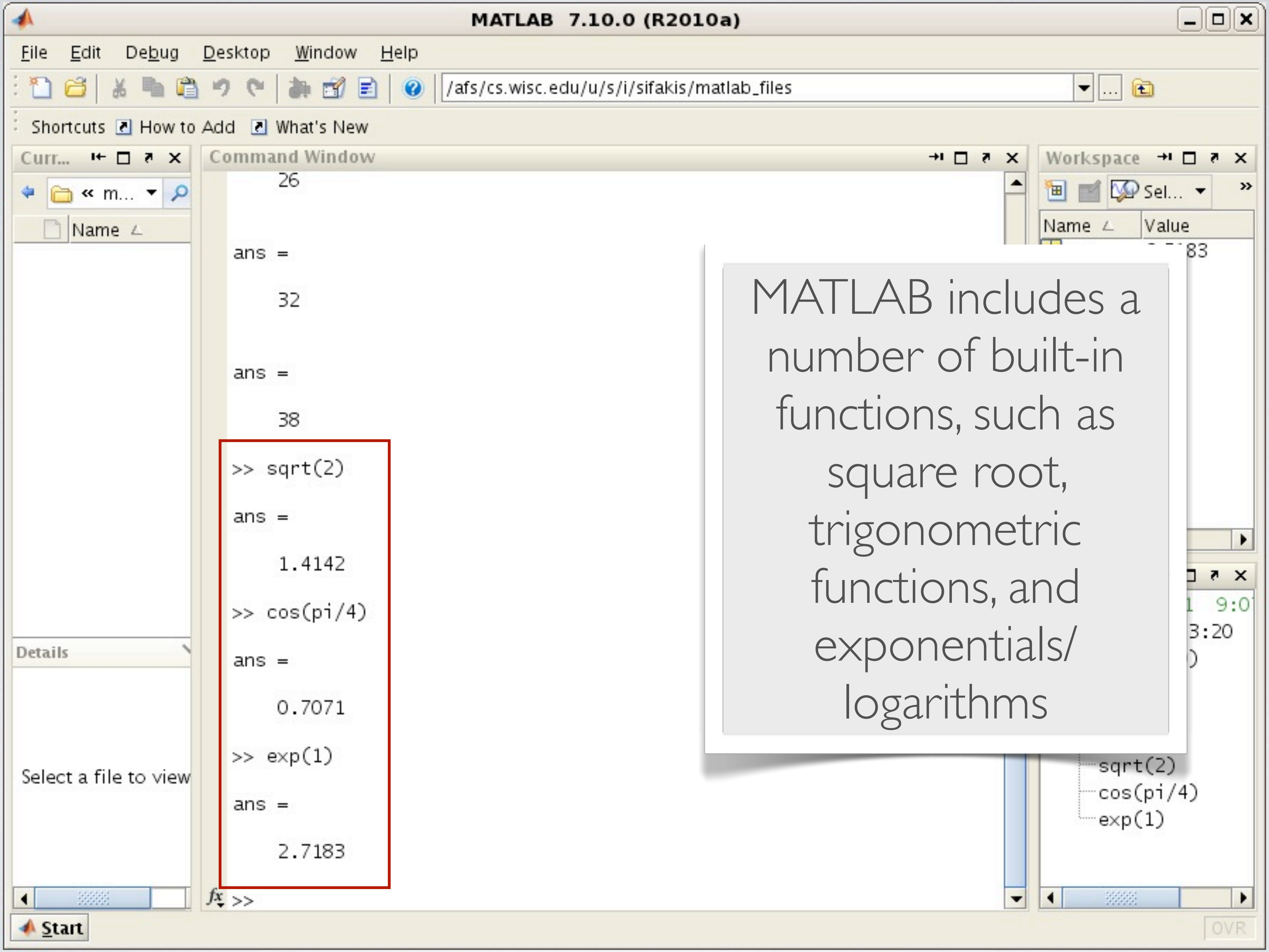

MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Name
Details
Select a file to view
Command Window
Workspace
Sel...
Start
OVR

```
>> for i=1:3:20
if (i>10)
i*2
end
end

ans =

    26


ans =

    32


ans =

    38

>>
```

Name	Value
ans	38
i	19

```
if (i>10)
i*2
end
end
```

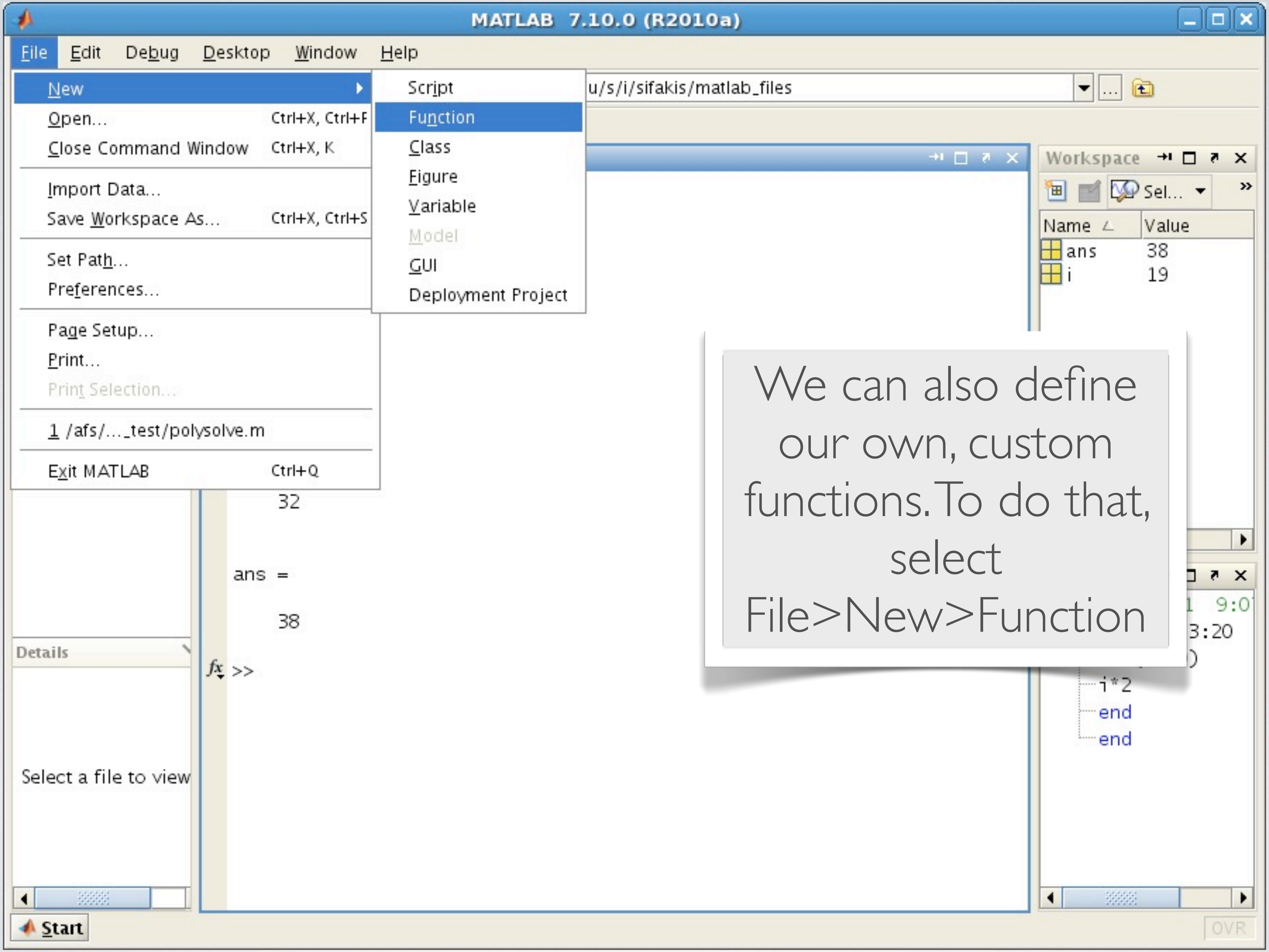
if/for statements can be combined and nested.


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
Name
Details
Select a file to view
Command Window
Workspace
Sel...
Name Value
Start
OVR

```
    26


ans =

    32


ans =

    38

>> sqrt(2)

ans =

    1.4142

>> cos(pi/4)

ans =

    0.7071

>> exp(1)

ans =

    2.7183

>>
```

```
sqrt(2)
cos(pi/4)
exp(1)
```

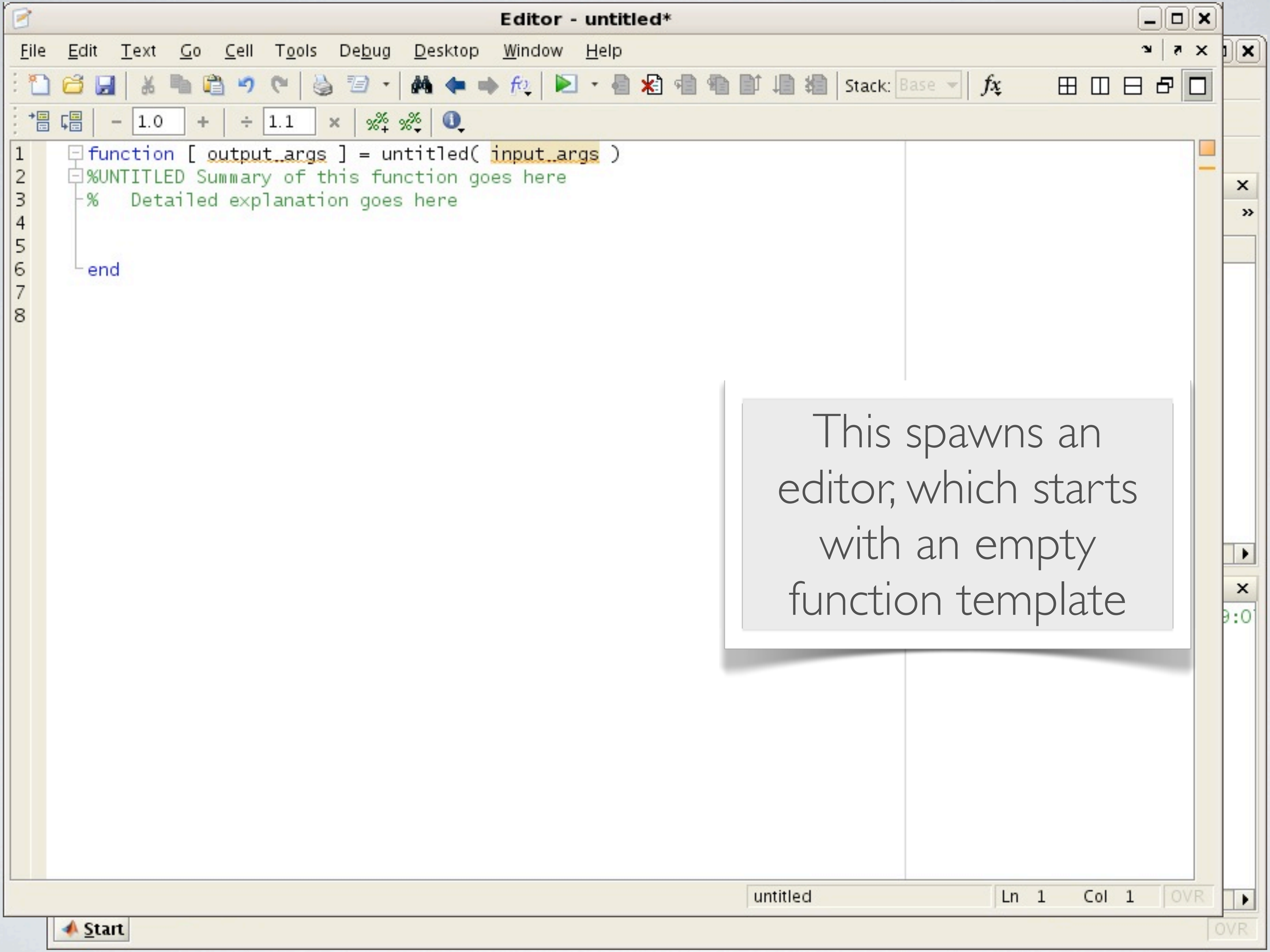
MATLAB includes a number of built-in functions, such as square root, trigonometric functions, and exponentials/ logarithms



```
     32


ans =

     38

>>
```

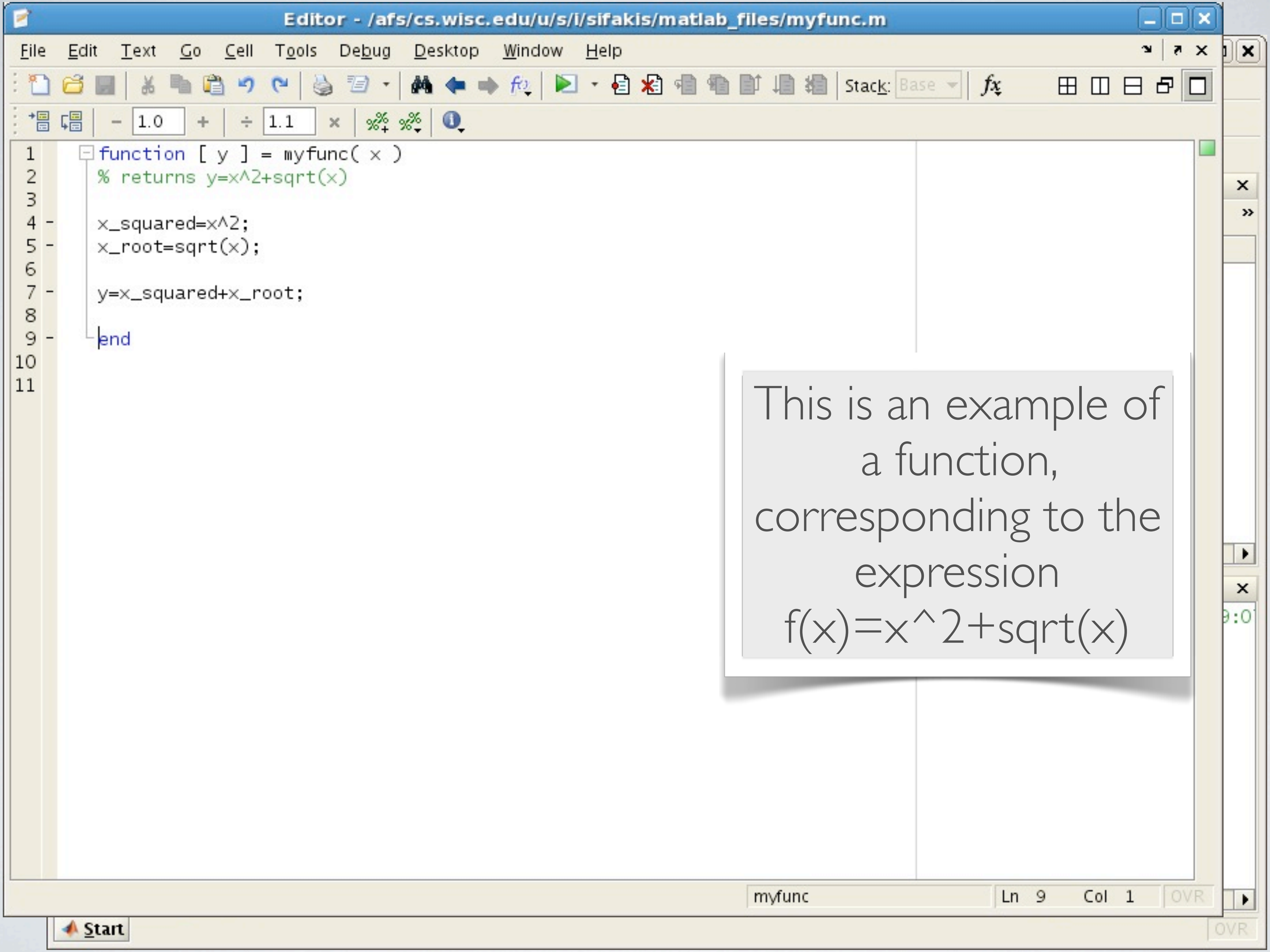
```
i*2
end
end
```

We can also define our own, custom functions. To do that, select File>New>Function



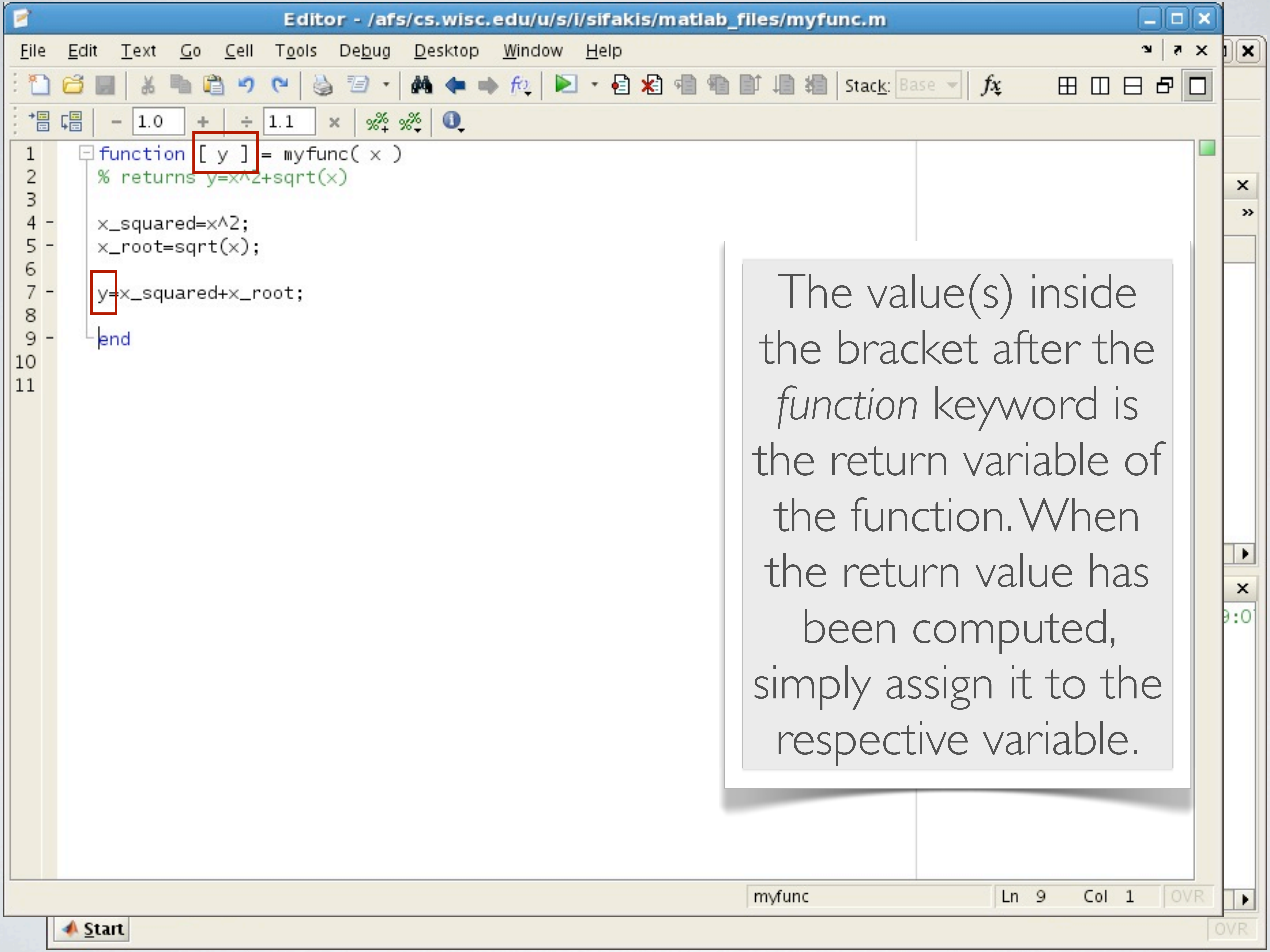
```
function [ output_args ] = untitled( input_args )
%UNTITLED Summary of this function goes here
%   Detailed explanation goes here


end
```

This spawns an editor, which starts with an empty function template


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc
Ln 9 Col 1
OVR
Start

```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

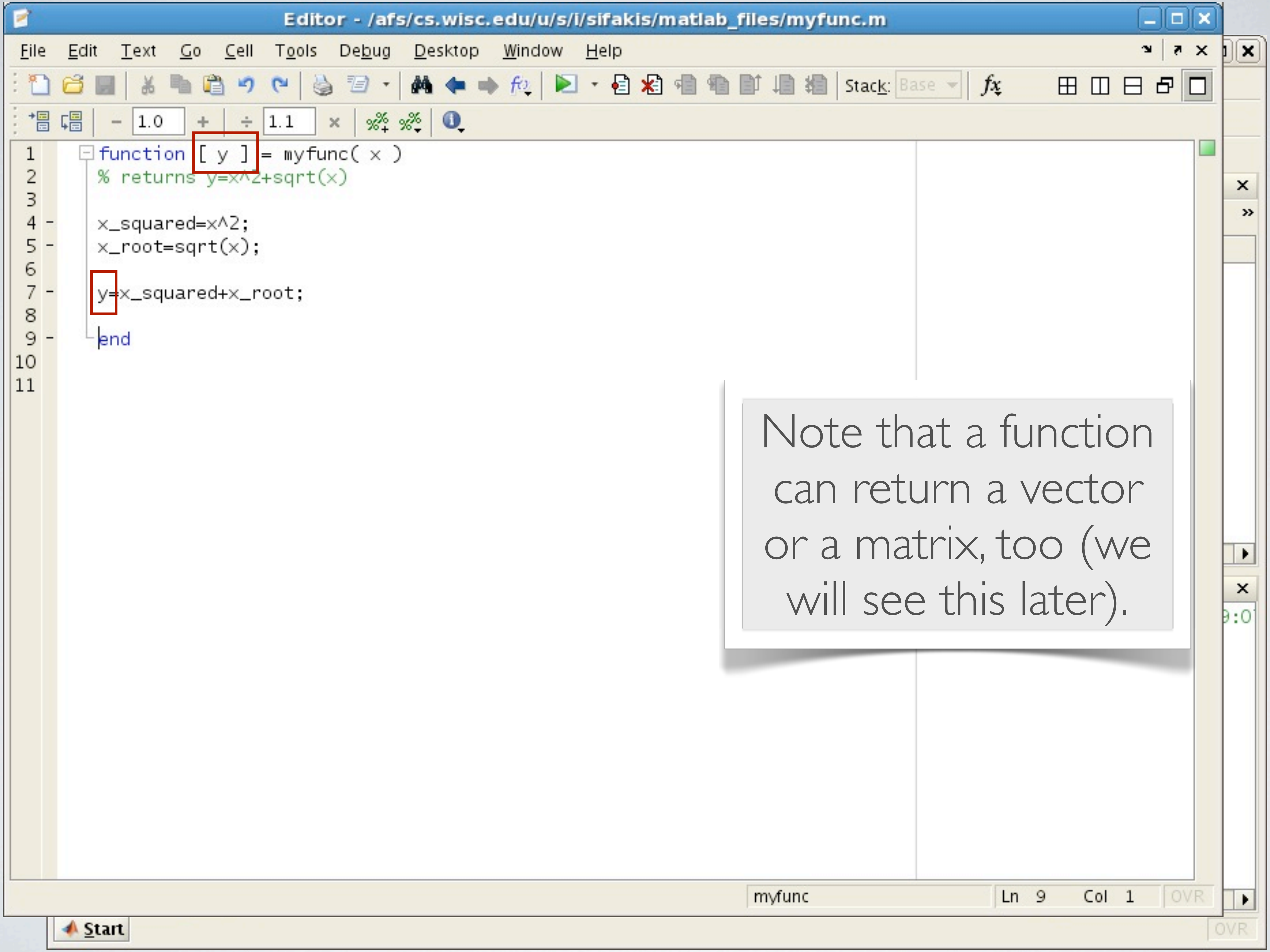
This is an example of a function, corresponding to the expression f(x)=x^2+sqrt(x)



Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc
Ln 9
Col 1
OVR
Start

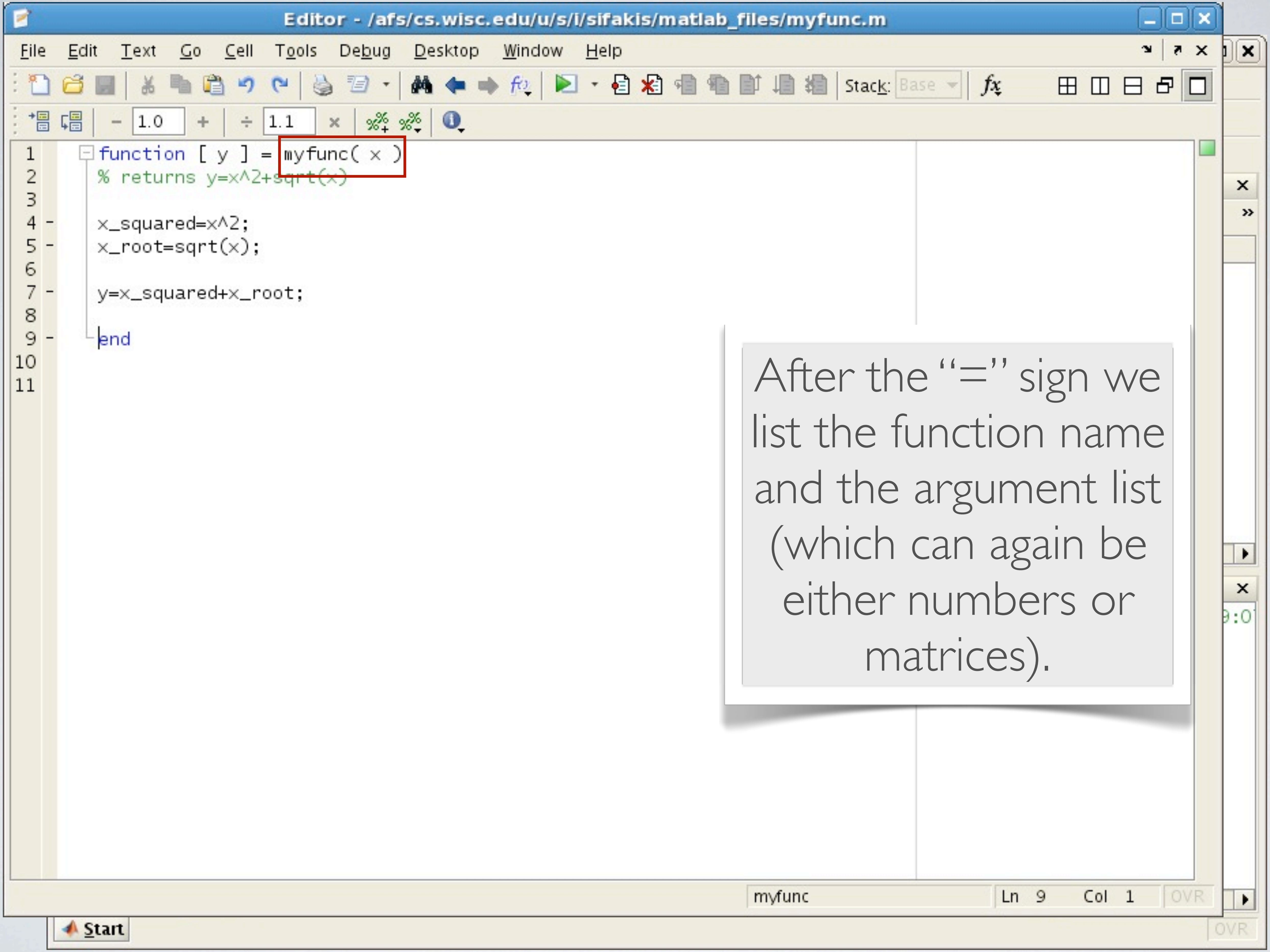
```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

The value(s) inside the bracket after the *function* keyword is the return variable of the function. When the return value has been computed, simply assign it to the respective variable.


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc
Ln 9 Col 1
OVR
Start

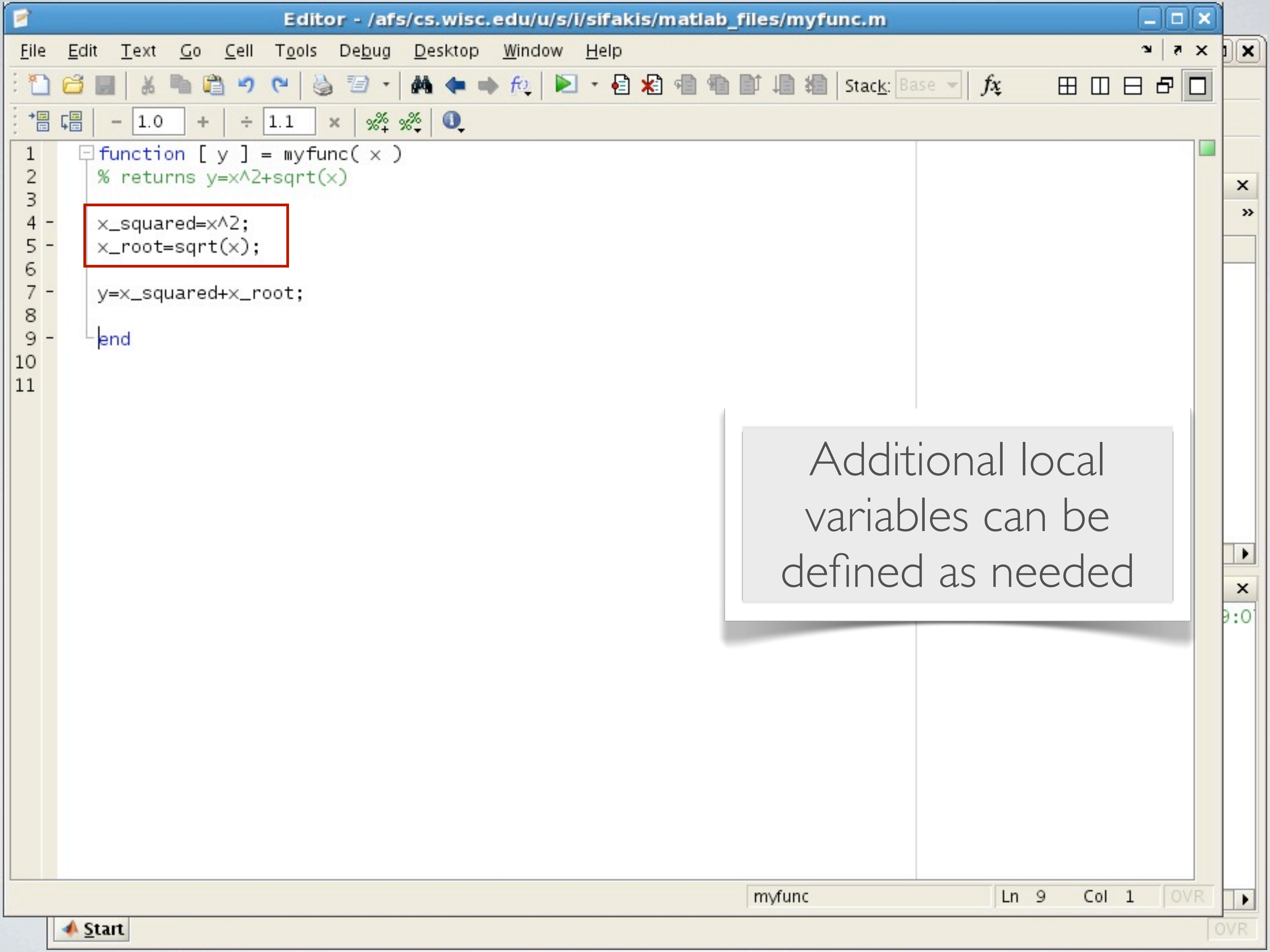
```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

Note that a function can return a vector or a matrix, too (we will see this later).


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc
Ln 9 Col 1
OVR
Start

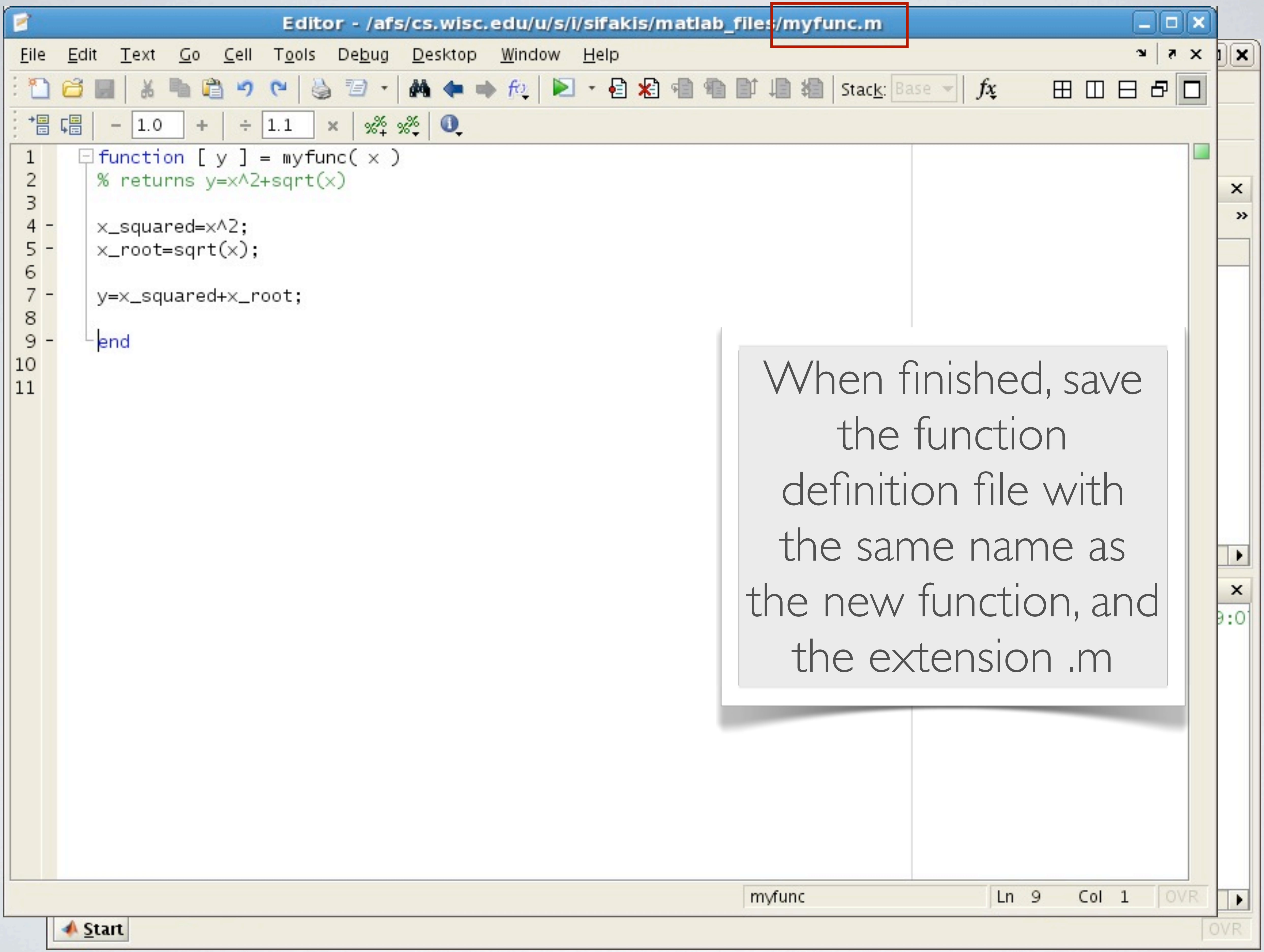
```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

After the "=" sign we list the function name and the argument list (which can again be either numbers or matrices).



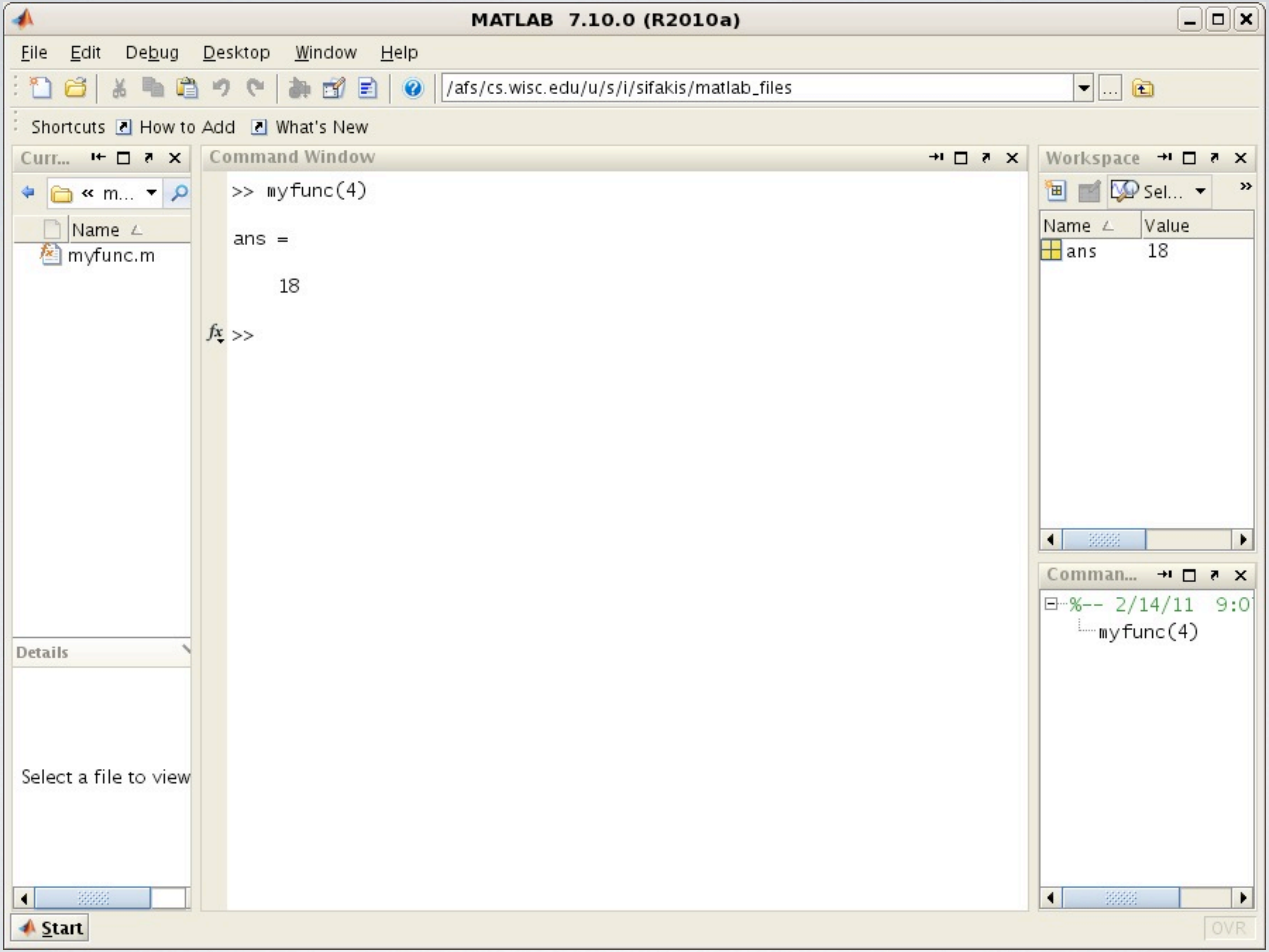
```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

Additional local variables can be defined as needed


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc
Ln 9 Col 1
OVR
Start

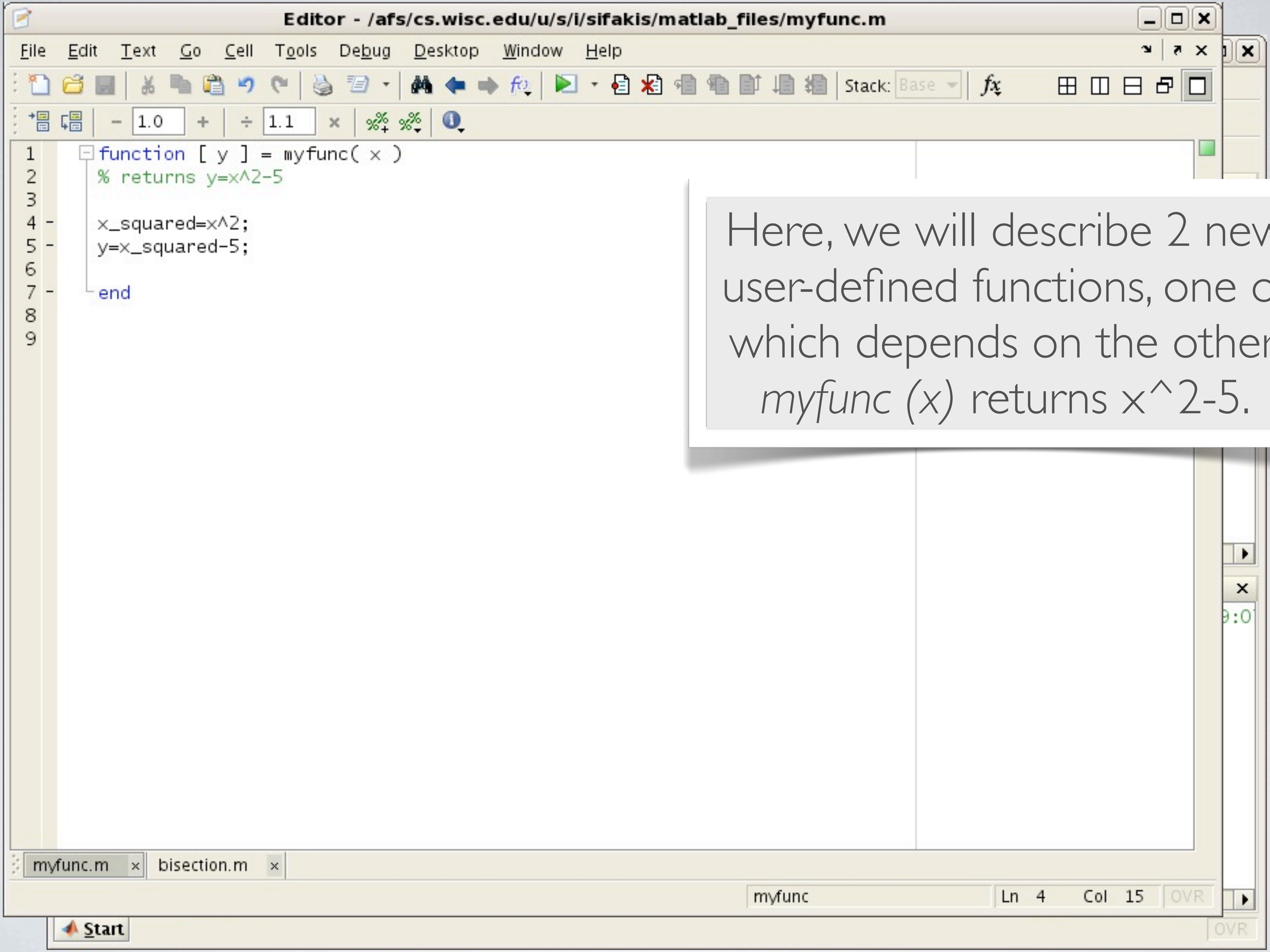
```
function [ y ] = myfunc( x )
% returns y=x^2+sqrt(x)

x_squared=x^2;
x_root=sqrt(x);

y=x_squared+x_root;

end
```

When finished, save the function definition file with the same name as the new function, and the extension .m



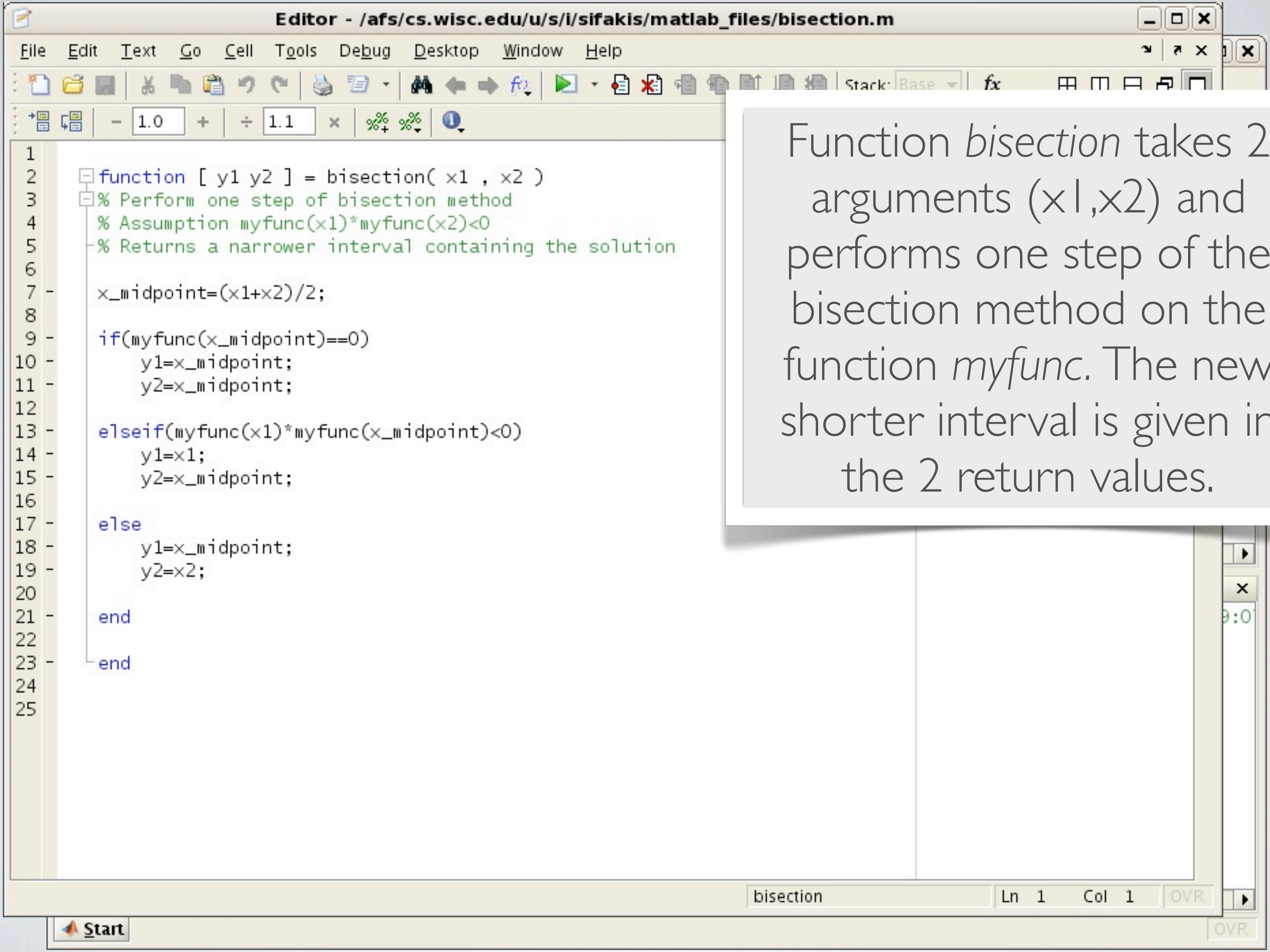
```
>> myfunc(4)

ans =

    18

>>
```

```
%-- 2/14/11  9:0
  myfunc(4)
```


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/myfunc.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
myfunc.m
bisection.m
myfunc
Ln 4 Col 15
OVR
Start

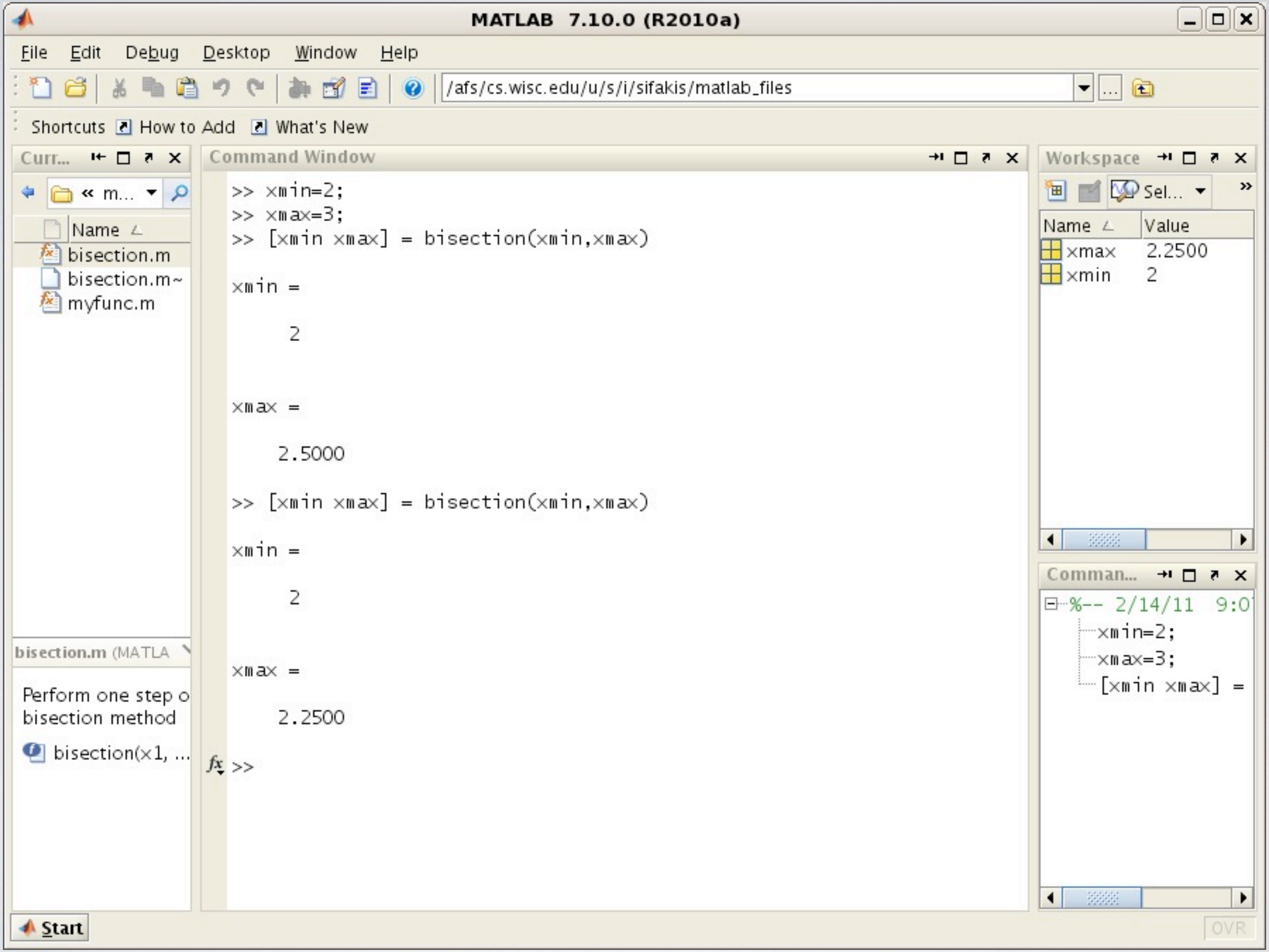
```
function [ y ] = myfunc( x )
% returns y=x^2-5

x_squared=x^2;
y=x_squared-5;

end
```

Here, we will describe 2 new user-defined functions, one of which depends on the other. *myfunc (x)* returns x^2-5.


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/bisection.m
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
fx
1.0
1.1
bisection
Ln 1 Col 1
OVR
Start

```
function [ y1 y2 ] = bisection( x1 , x2 )
% Perform one step of bisection method
% Assumption myfunc(x1)*myfunc(x2)<0
% Returns a narrower interval containing the solution

x_midpoint=(x1+x2)/2;

if(myfunc(x_midpoint)==0)
    y1=x_midpoint;
    y2=x_midpoint;

elseif(myfunc(x1)*myfunc(x_midpoint)<0)
    y1=x1;
    y2=x_midpoint;

else
    y1=x_midpoint;
    y2=x2;

end

end
```

Function *bisection* takes 2 arguments (x1,x2) and performs one step of the bisection method on the function *myfunc*. The new shorter interval is given in the 2 return values.



Name
bisection.m
bisection.m~
myfunc.m

bisection.m (MATLA

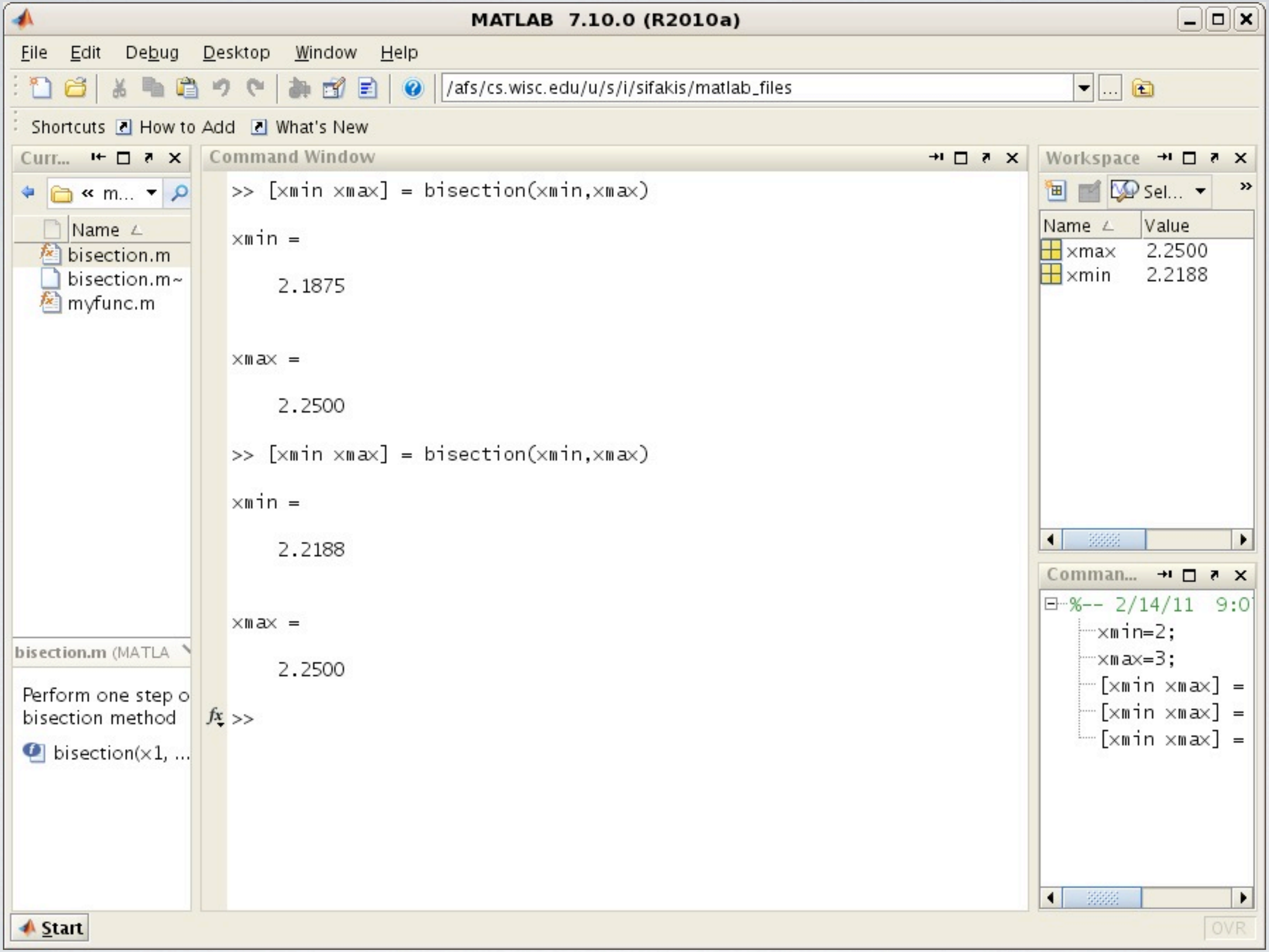
Perform one step o
bisection method

bisection(x1, ...

```
>> xmin=2;
>> xmax=3;
>> [xmin xmax] = bisection(xmin,xmax)

xmin =

     2


xmax =

    2.5000

>> [xmin xmax] = bisection(xmin,xmax)

xmin =

     2


xmax =

    2.2500

>>
```

Name	Value
xmax	2.2500
xmin	2

```
%-- 2/14/11  9:0
   xmin=2;
   xmax=3;
   [xmin xmax] =
```



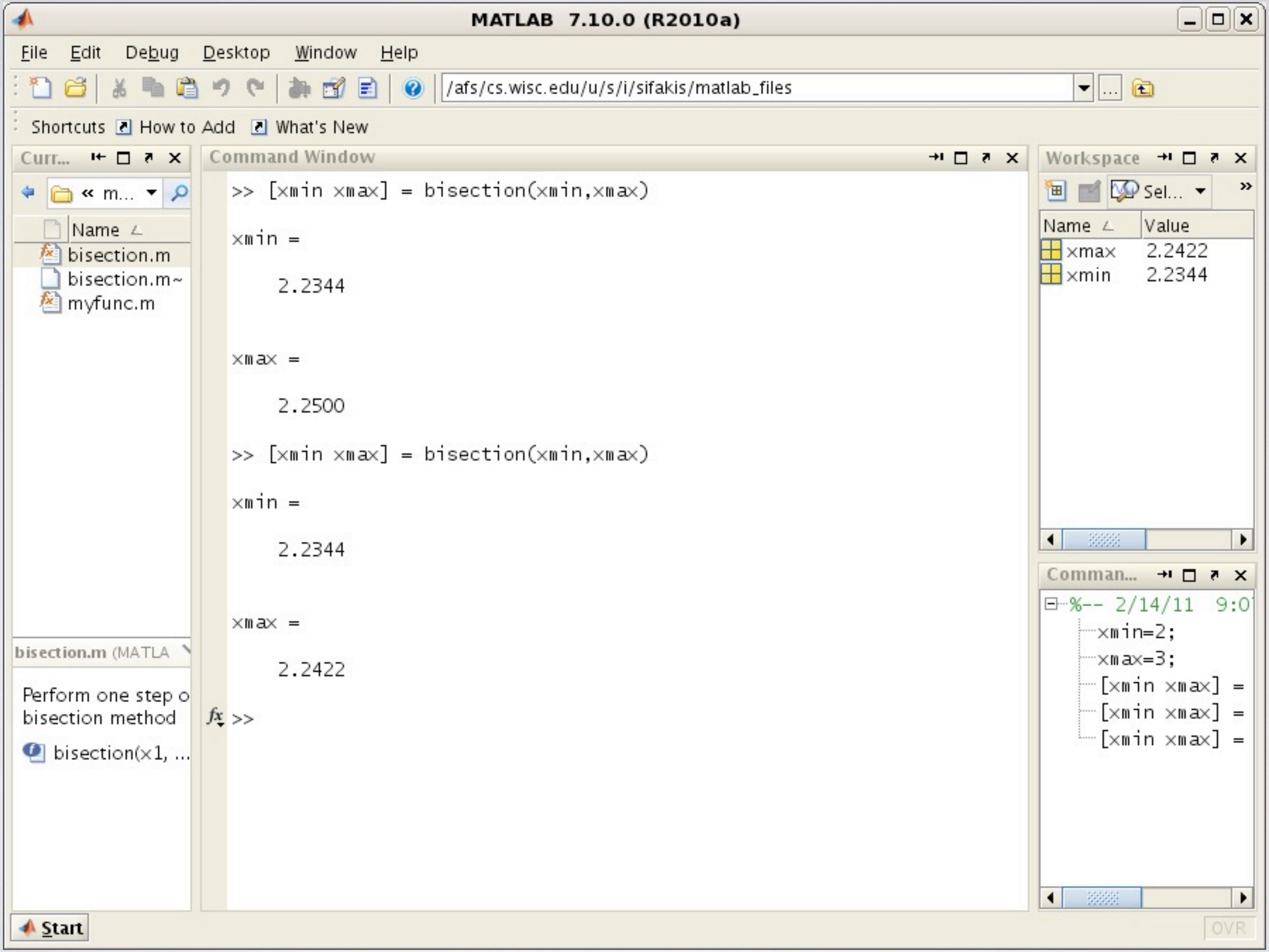
```
>> [xmin xmax] = bisection(xmin,xmax)

xmin =

    2.1875


xmax =

    2.2500

>> [xmin xmax] = bisection(xmin,xmax)

xmin =

    2.2188


xmax =

    2.2500

>>
```

Perform one step o
bisection method

bisection(x1, ...

Name	Value
xmax	2.2500
xmin	2.2188

```
%-- 2/14/11   9:0
  xmin=2;
  xmax=3;
  [xmin xmax] =
  [xmin xmax] =
  [xmin xmax] =
```


MATLAB 7.10.0 (R2010a)
File Edit Debug Desktop Window Help
/afs/cs.wisc.edu/u/s/i/sifakis/matlab_files
Shortcuts How to Add What's New
Curr...
« m...
Command Window
Workspace
Sel...
Comman...
Start
OVR

Name
bisection.m
bisection.m~
myfunc.m

bisection.m (MATLA

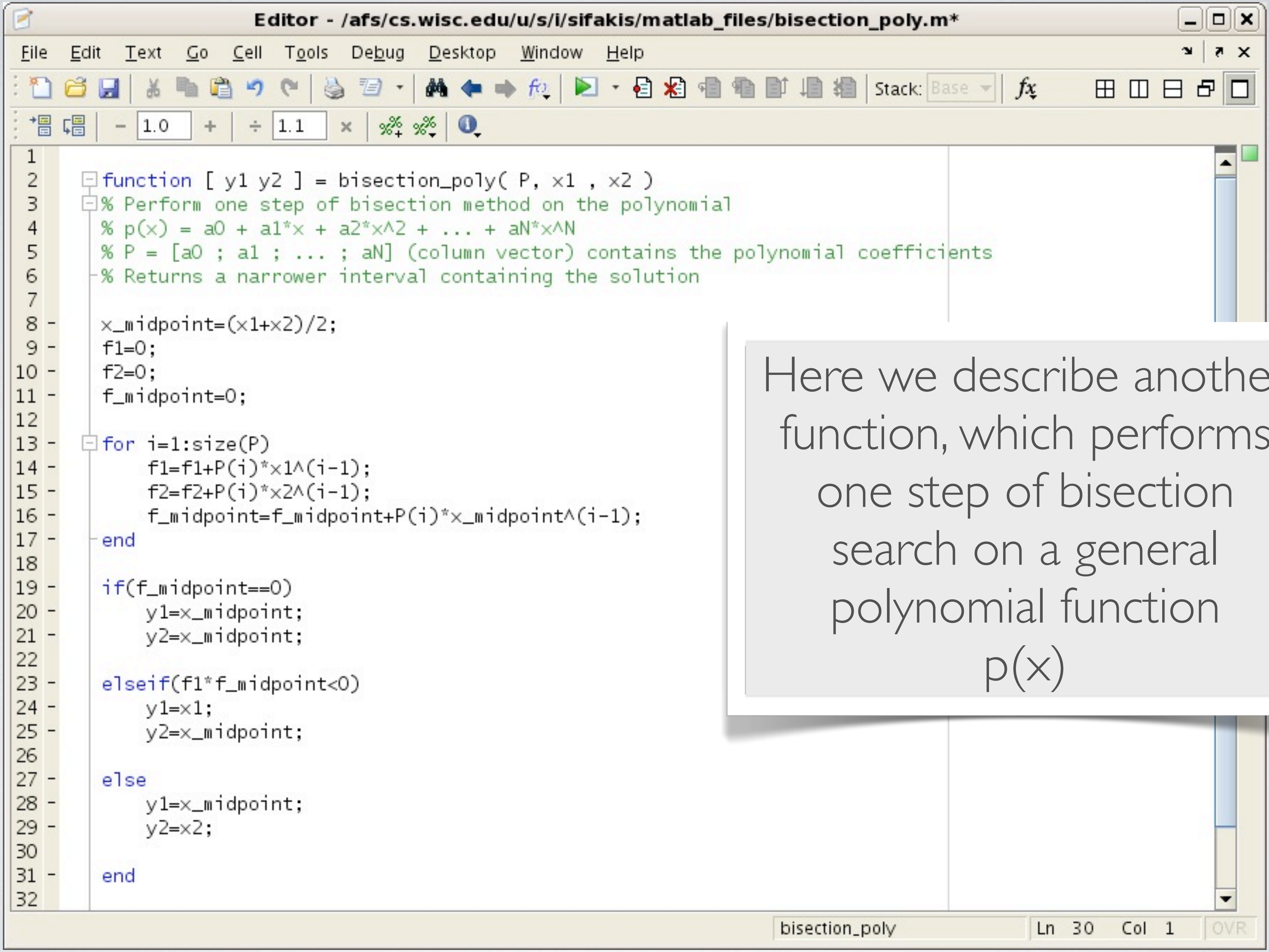
Perform one step o
bisection method

bisection(x1, ...

```
>> [xmin xmax] = bisection(xmin,xmax)

xmin =

    2.2344


xmax =

    2.2500

>> [xmin xmax] = bisection(xmin,xmax)

xmin =

    2.2344


xmax =

    2.2422

>>
```

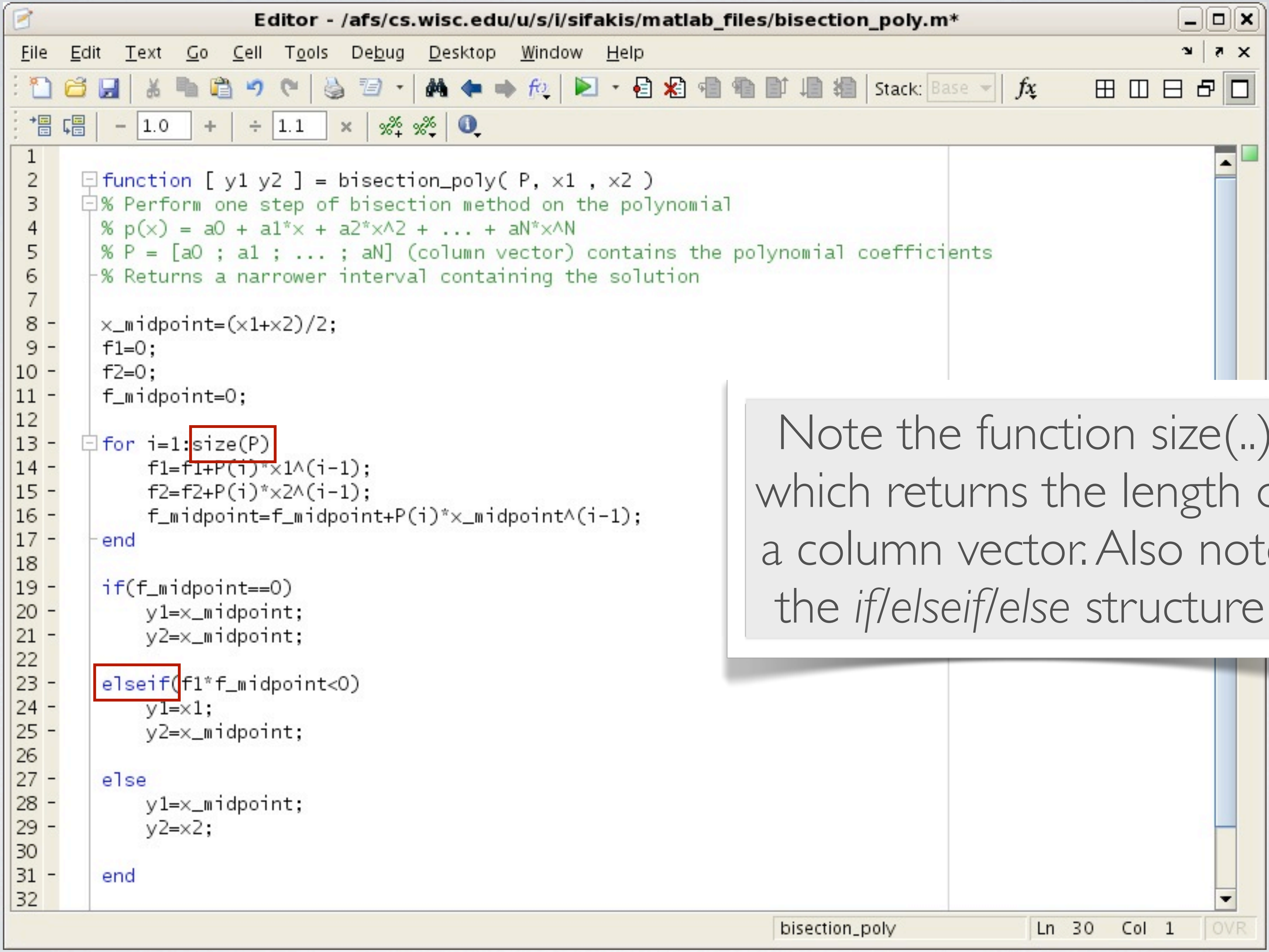
Name	Value
xmax	2.2422
xmin	2.2344

```
%-- 2/14/11   9:0
   xmin=2;
   xmax=3;
   [xmin xmax] =
   [xmin xmax] =
   [xmin xmax] =
```


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/bisection_poly.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
bisection_poly
Ln 30 Col 1
OVR

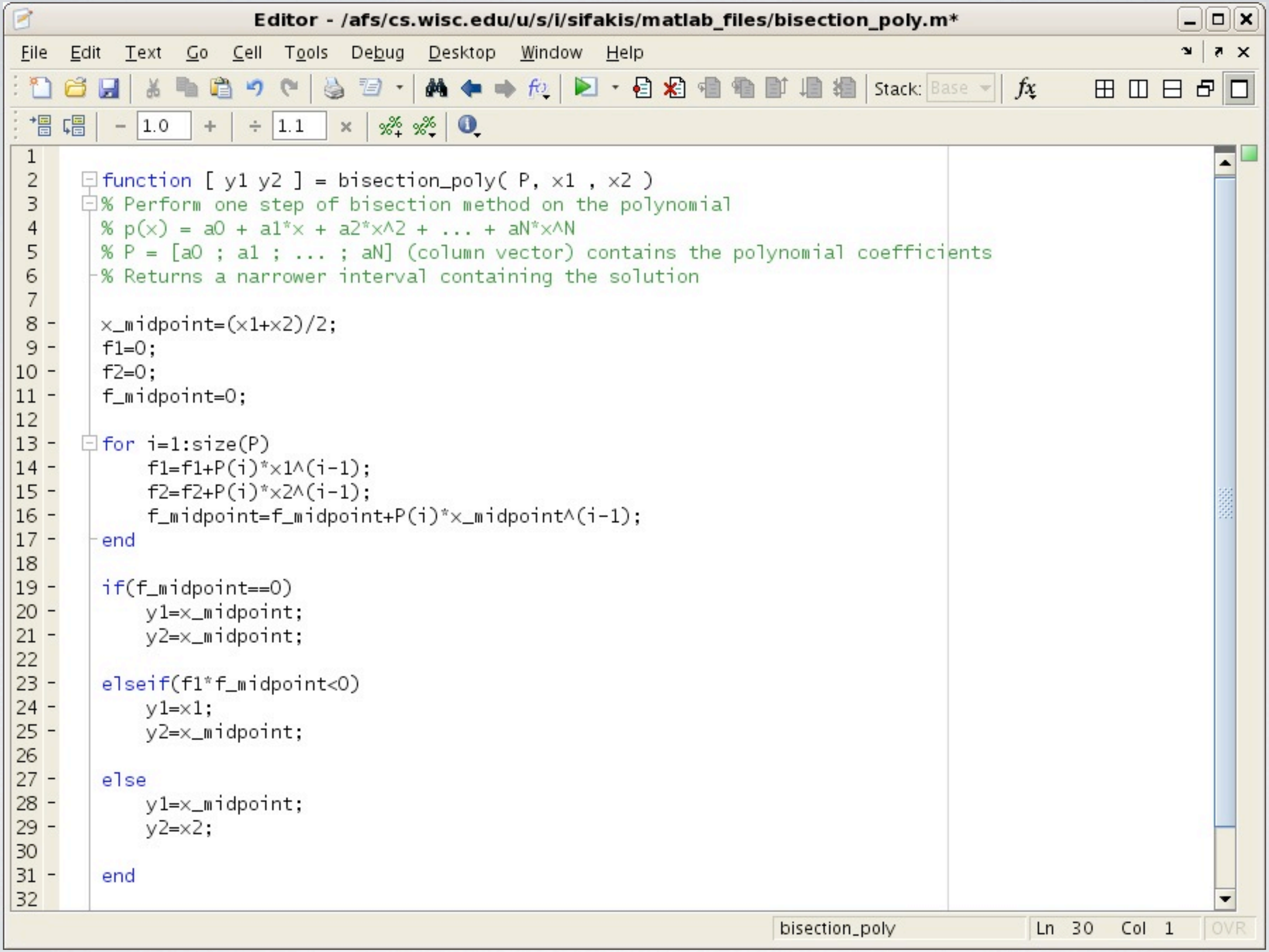
```matlab

function [ y1 y2 ] = bisection_poly( P, x1 , x2 )
% Perform one step of bisection method on the polynomial
% p(x) = a0 + a1*x + a2*x^2 + ... + aN*x^N
% P = [a0 ; a1 ; ... ; aN] (column vector) contains the polynomial coefficients
% Returns a narrower interval containing the solution

x_midpoint=(x1+x2)/2;
f1=0;
f2=0;
f_midpoint=0;

for i=1:size(P)
    f1=f1+P(i)*x1^(i-1);
    f2=f2+P(i)*x2^(i-1);
    f_midpoint=f_midpoint+P(i)*x_midpoint^(i-1);
end

if(f_midpoint==0)
    y1=x_midpoint;
    y2=x_midpoint;

elseif(f1*f_midpoint<0)
    y1=x1;
    y2=x_midpoint;

else
    y1=x_midpoint;
    y2=x2;

end

```

Here we describe another function, which performs one step of bisection search on a general polynomial function p(x)

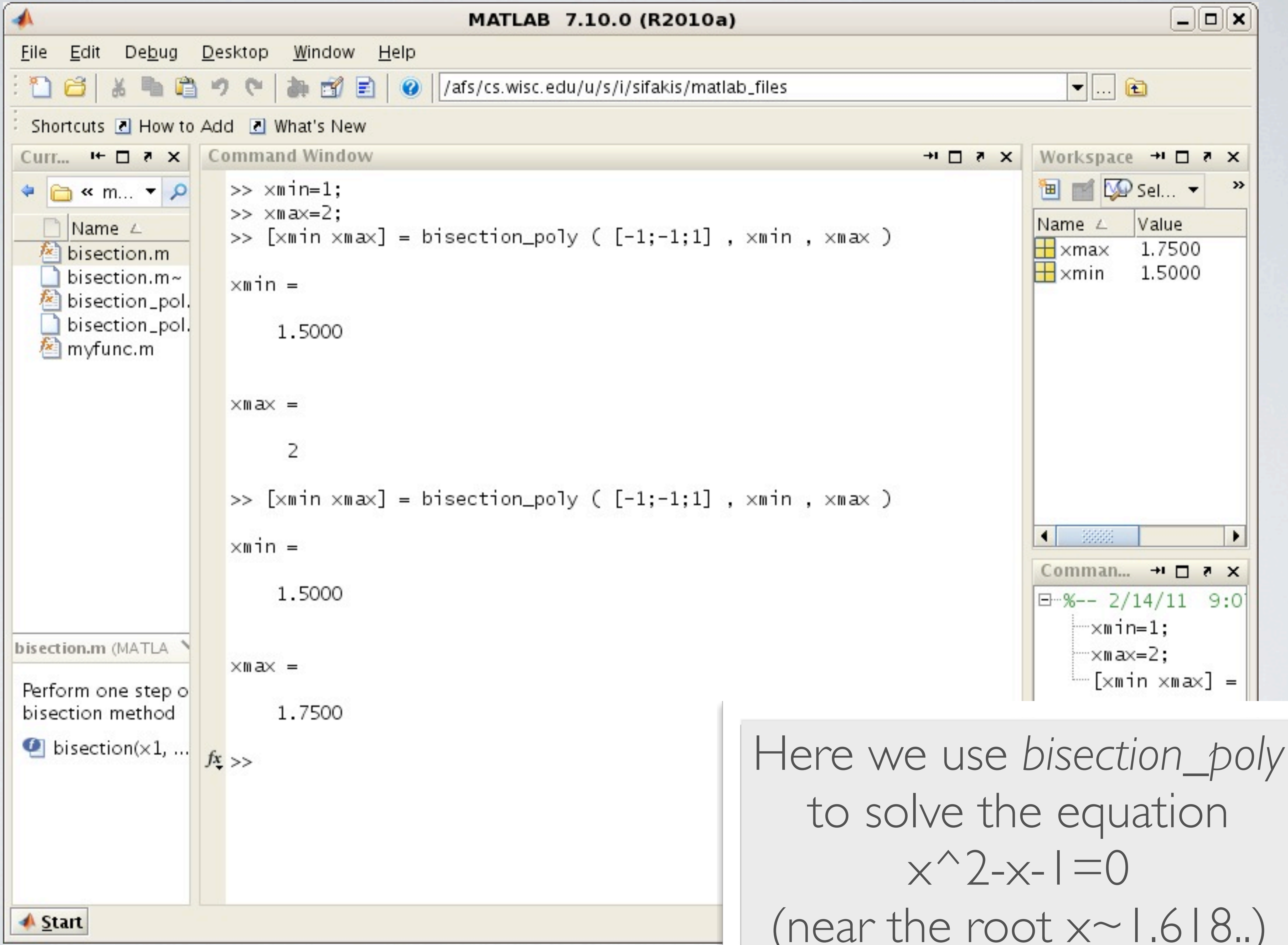

Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/bisection_poly.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
bisection_poly
Ln 30 Col 1
OVR

```matlab

function [ y1 y2 ] = bisection_poly( P, x1 , x2 )
% Perform one step of bisection method on the polynomial
% p(x) = a0 + a1*x + a2*x^2 + ... + aN*x^N
% P = [a0 ; a1 ; ... ; aN] (column vector) contains the polynomial coefficients
% Returns a narrower interval containing the solution

x_midpoint=(x1+x2)/2;
f1=0;
f2=0;
f_midpoint=0;

for i=1:size(P)
    f1=f1+P(i)*x1^(i-1);
    f2=f2+P(i)*x2^(i-1);
    f_midpoint=f_midpoint+P(i)*x_midpoint^(i-1);
end

if(f_midpoint==0)
    y1=x_midpoint;
    y2=x_midpoint;

elseif(f1*f_midpoint<0)
    y1=x1;
    y2=x_midpoint;

else
    y1=x_midpoint;
    y2=x2;

end

```

Note the function size(..) which returns the length of a column vector. Also note the *if/elseif/else* structure.


Editor - /afs/cs.wisc.edu/u/s/i/sifakis/matlab_files/bisection_poly.m*
File Edit Text Go Cell Tools Debug Desktop Window Help
Stack: Base
1.0
1.1
bisection_poly
Ln 30 Col 1
OVR

```matlab

function [ y1 y2 ] = bisection_poly( P, x1 , x2 )
% Perform one step of bisection method on the polynomial
% p(x) = a0 + a1*x + a2*x^2 + ... + aN*x^N
% P = [a0 ; a1 ; ... ; aN] (column vector) contains the polynomial coefficients
% Returns a narrower interval containing the solution

x_midpoint=(x1+x2)/2;
f1=0;
f2=0;
f_midpoint=0;

for i=1:size(P)
    f1=f1+P(i)*x1^(i-1);
    f2=f2+P(i)*x2^(i-1);
    f_midpoint=f_midpoint+P(i)*x_midpoint^(i-1);
end

if(f_midpoint==0)
    y1=x_midpoint;
    y2=x_midpoint;

elseif(f1*f_midpoint<0)
    y1=x1;
    y2=x_midpoint;

else
    y1=x_midpoint;
    y2=x2;

end

```

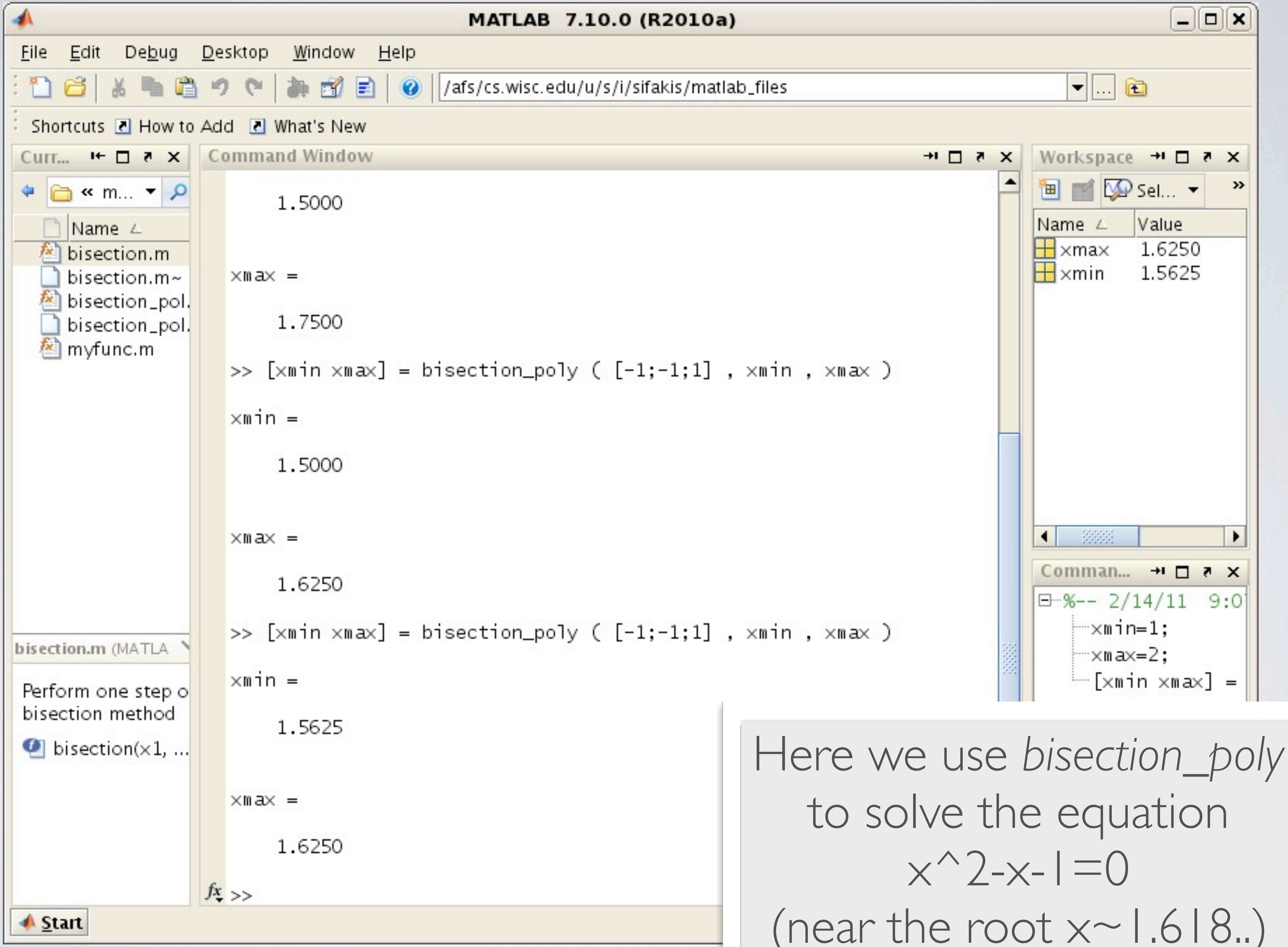


```
>> xmin=1;
>> xmax=2;
>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.5000


xmax =

     2

>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.5000


xmax =

    1.7500

>>
```

Name	Value
xmax	1.7500
xmin	1.5000

```
%-- 2/14/11  9:0
xmin=1;
xmax=2;
[xmin xmax] =
```

Here we use *bisection_poly* to solve the equation x^2-x-1=0 (near the root x~1.618..)

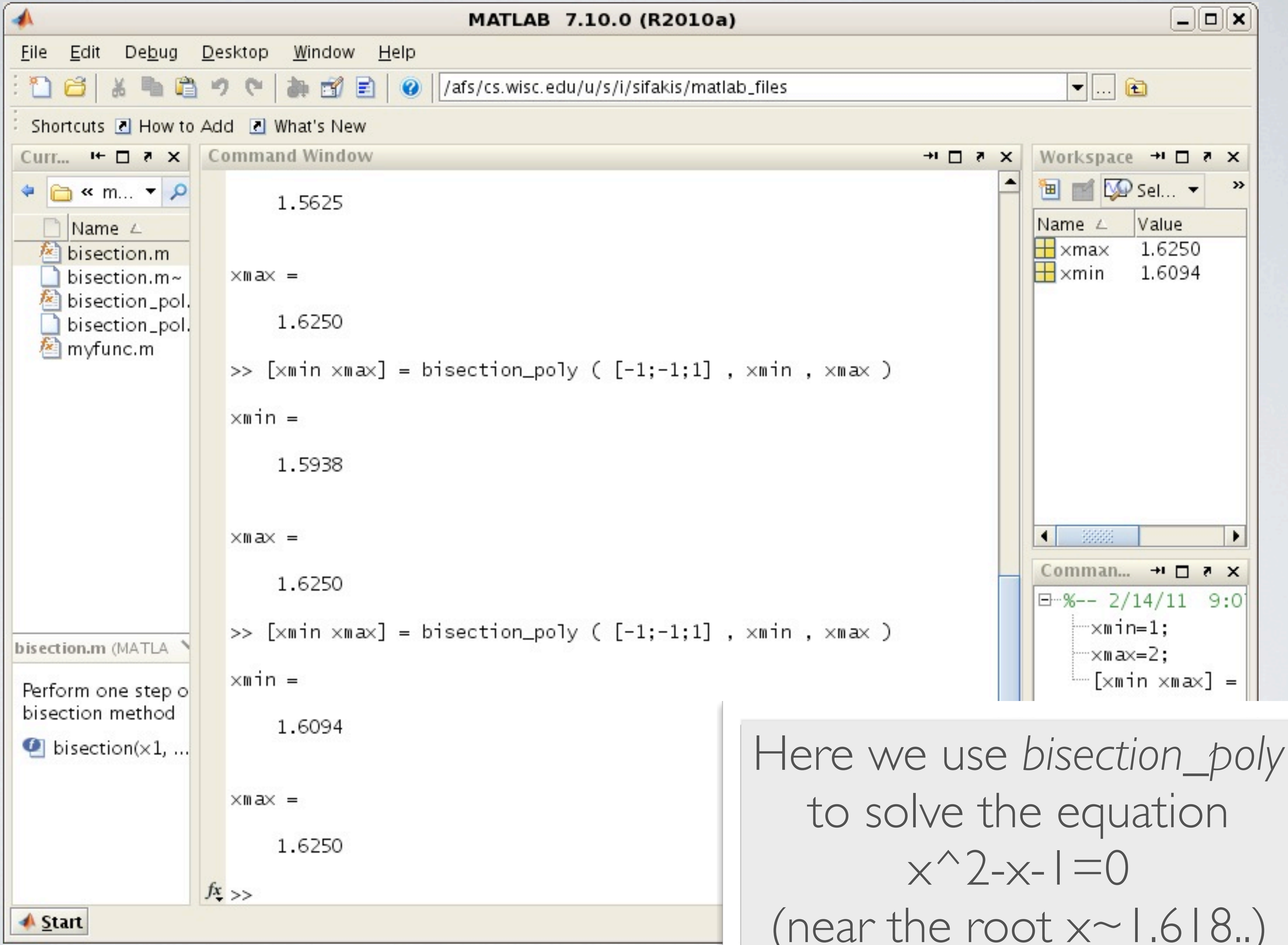


```
    1.5000


xmax =

    1.7500

>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.5000


xmax =

    1.6250

>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.5625


xmax =

    1.6250

>>
```

Name	Value
xmax	1.6250
xmin	1.5625

```
%-- 2/14/11  9:0
  xmin=1;
  xmax=2;
  [xmin xmax] =
```

Here we use *bisection_poly* to solve the equation x^2-x-1=0 (near the root x~1.618..)



```
    1.5625


xmax =

    1.6250

>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.5938


xmax =

    1.6250

>> [xmin xmax] = bisection_poly ( [-1;-1;1] , xmin , xmax )

xmin =

    1.6094


xmax =

    1.6250

>> 
```

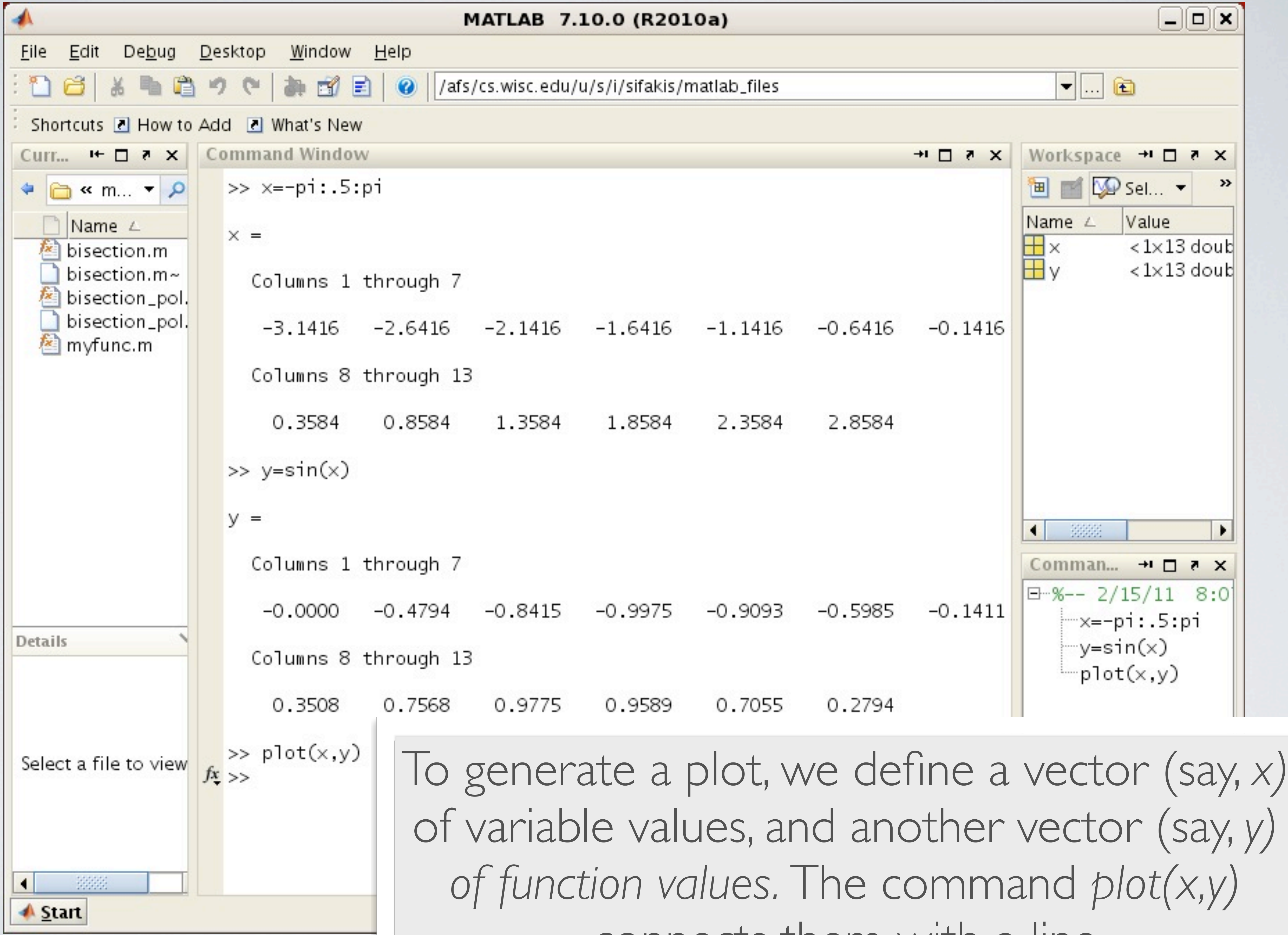
Name	Value
xmax	1.6250
xmin	1.6094

```
%-- 2/14/11  9:0
  xmin=1;
  xmax=2;
  [xmin xmax] =
```

Perform one step o
bisection method

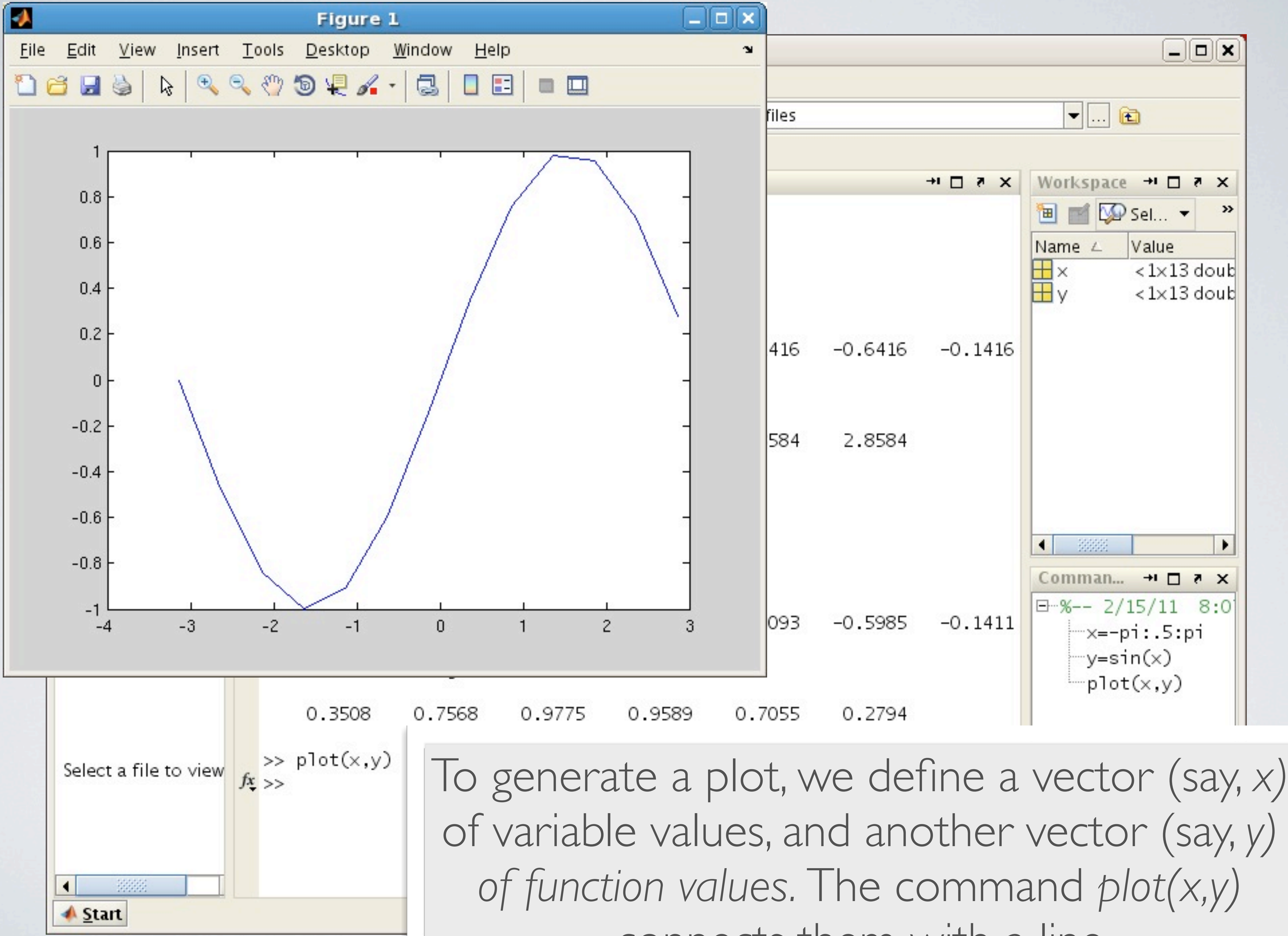
bisection(x1, ...

Here we use *bisection_poly*
to solve the equation
x^2-x-1=0
(near the root x~1.618..)



```
>> x=-pi:.5:pi

x =

  Columns 1 through 7

   -3.1416   -2.6416   -2.1416   -1.6416   -1.1416   -0.6416   -0.1416

  Columns 8 through 13

    0.3584    0.8584    1.3584    1.8584    2.3584    2.8584

>> y=sin(x)

y =

  Columns 1 through 7

   -0.0000   -0.4794   -0.8415   -0.9975   -0.9093   -0.5985   -0.1411

  Columns 8 through 13

    0.3508    0.7568    0.9775    0.9589    0.7055    0.2794

>> plot(x,y)
>>
```

To generate a plot, we define a vector (say, *x*) of variable values, and another vector (say, *y*) *of function values.* The command *plot(x,y)* connects them with a line

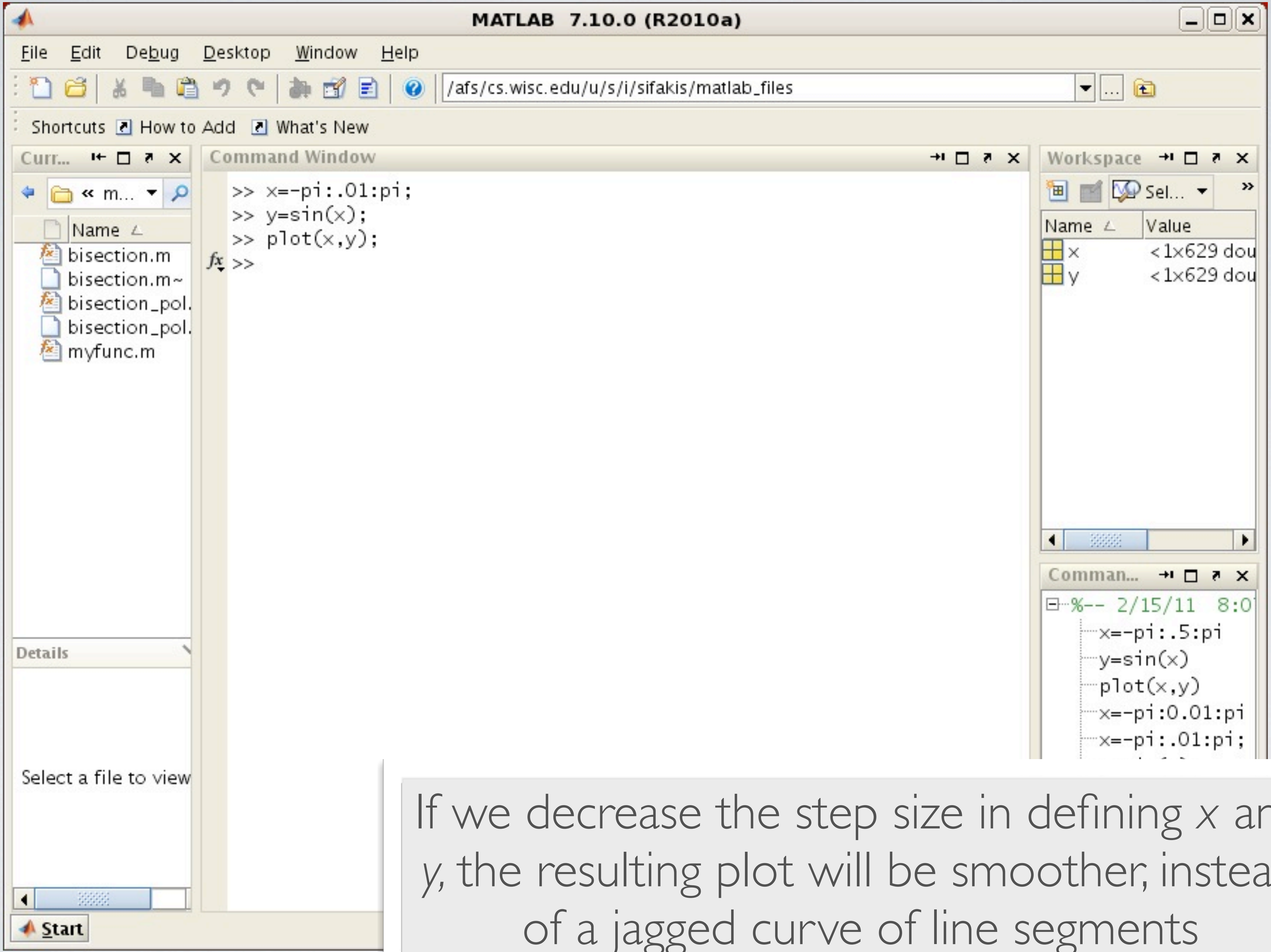


```
416     -0.6416     -0.1416

584      2.8584

093     -0.5985     -0.1411

    0.3508    0.7568    0.9775    0.9589    0.7055    0.2794

>> plot(x,y)
>>
```

Name	Value
x	<1x13 doub
y	<1x13 doub

```
%-- 2/15/11  8:0
   x=-pi:.5:pi
   y=sin(x)
   plot(x,y)
```

To generate a plot, we define a vector (say, *x*) of variable values, and another vector (say, *y*) *of function values.* The command *plot(x,y)* connects them with a line

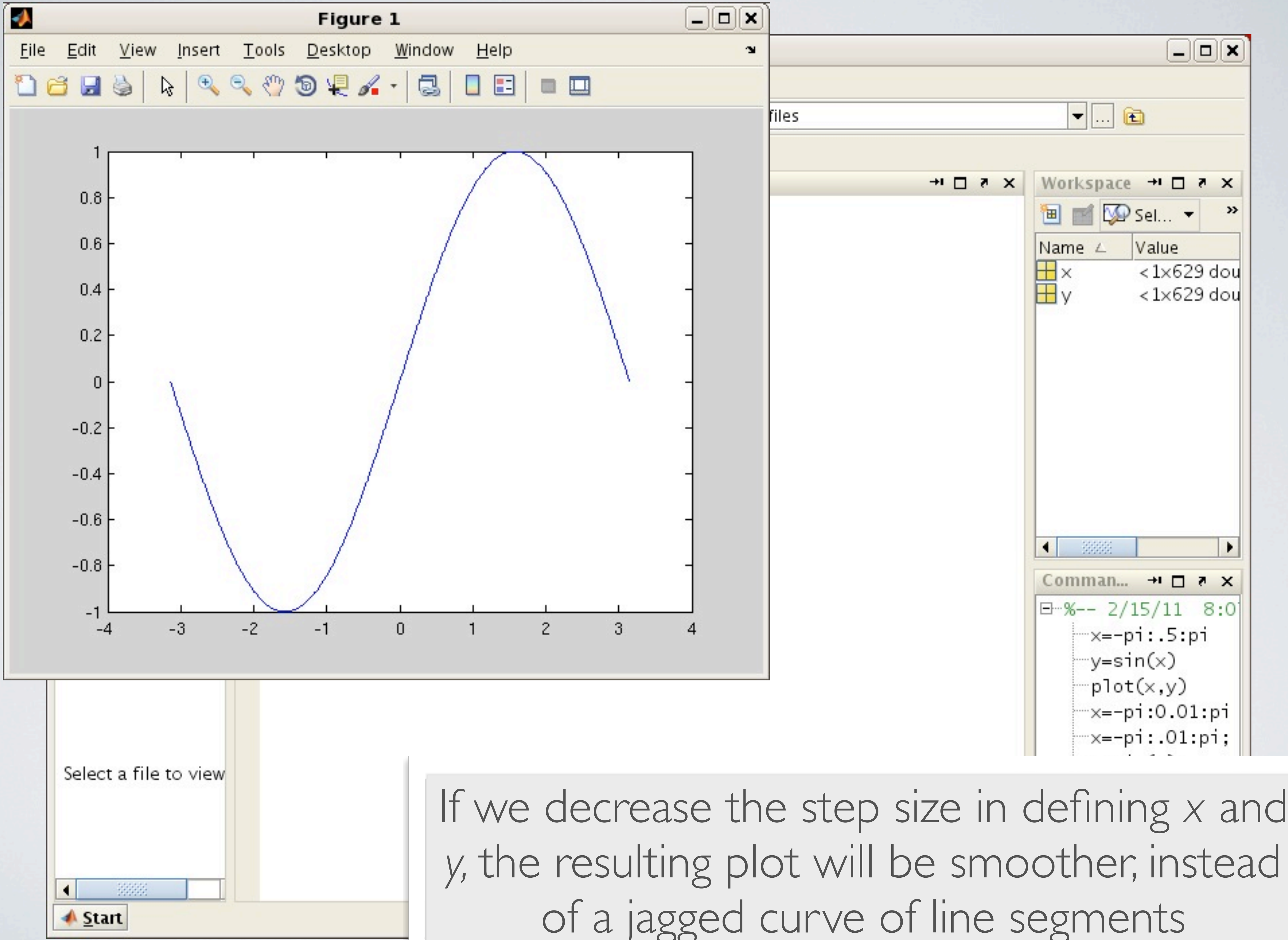


```
>> x=-pi:.01:pi;
>> y=sin(x);
>> plot(x,y);
>> 
```

Name	Value
x	<1x629 dou
y	<1x629 dou

```
%-- 2/15/11  8:0
   x=-pi:.5:pi
   y=sin(x)
   plot(x,y)
   x=-pi:0.01:pi
   x=-pi:.01:pi;
```

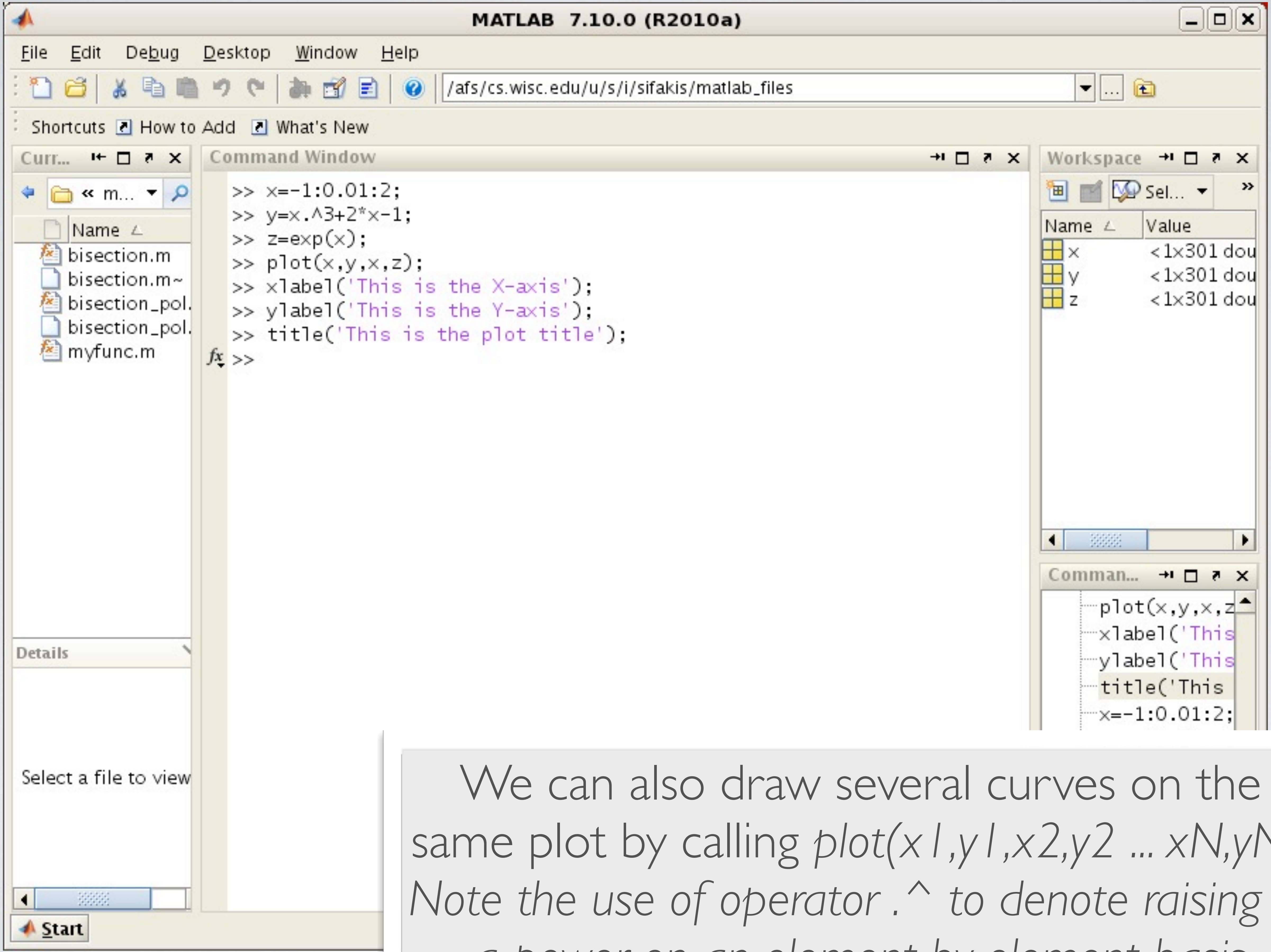
If we decrease the step size in defining *x* and *y*, the resulting plot will be smoother, instead of a jagged curve of line segments



Name	Value
x	<1x629 dou
y	<1x629 dou

```
%-- 2/15/11  8:0
x=-pi:.5:pi
y=sin(x)
plot(x,y)
x=-pi:0.01:pi
x=-pi:.01:pi;
```

If we decrease the step size in defining *x* and *y*, the resulting plot will be smoother, instead of a jagged curve of line segments

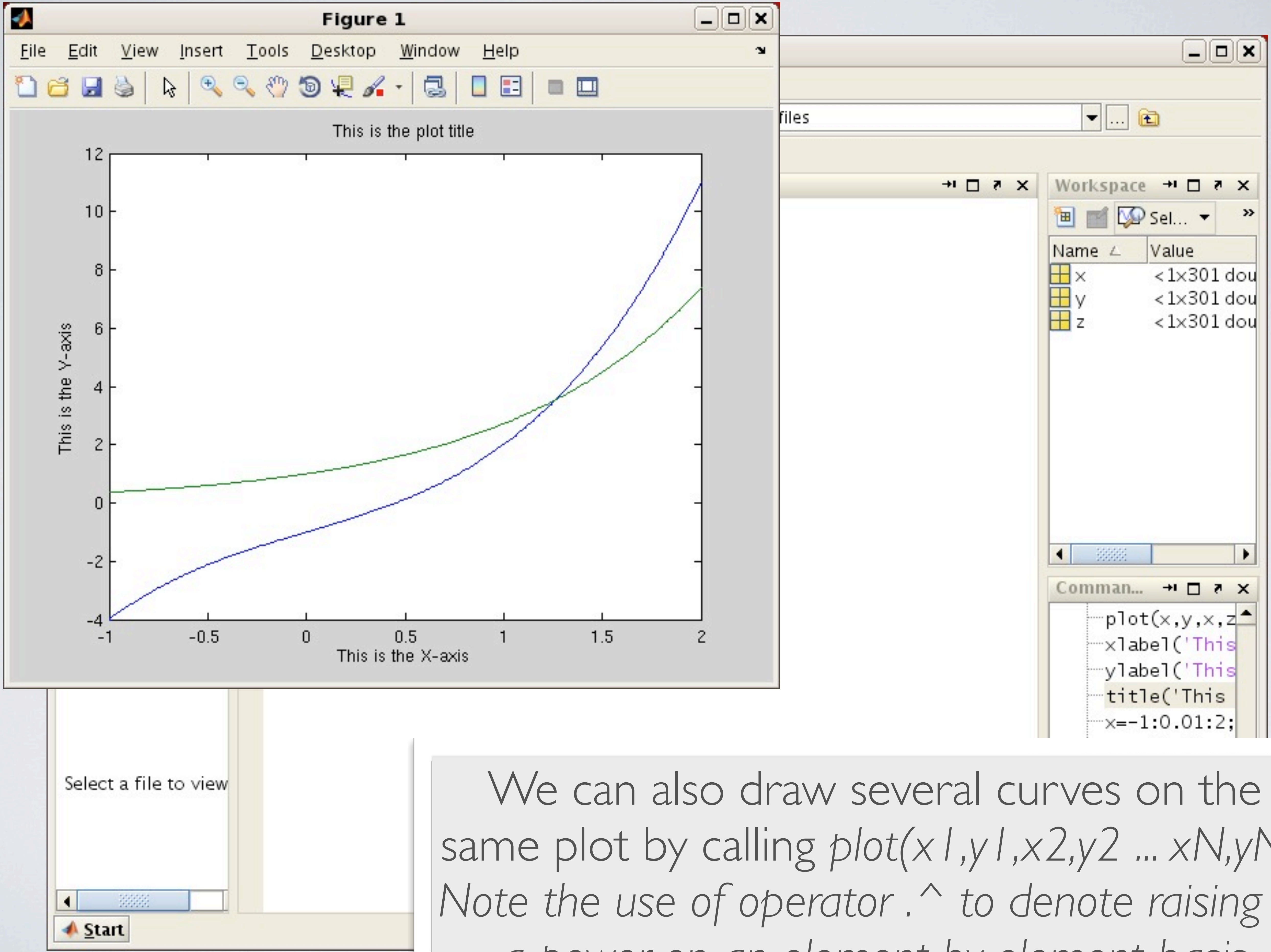


```
>> x=-1:0.01:2;
>> y=x.^3+2*x-1;
>> z=exp(x);
>> plot(x,y,x,z);
>> xlabel('This is the X-axis');
>> ylabel('This is the Y-axis');
>> title('This is the plot title');
>>
```

Name	Value
x	<1x301 dou
y	<1x301 dou
z	<1x301 dou

```
plot(x,y,x,z
xlabel('This
ylabel('This
title('This
x=-1:0.01:2;
```

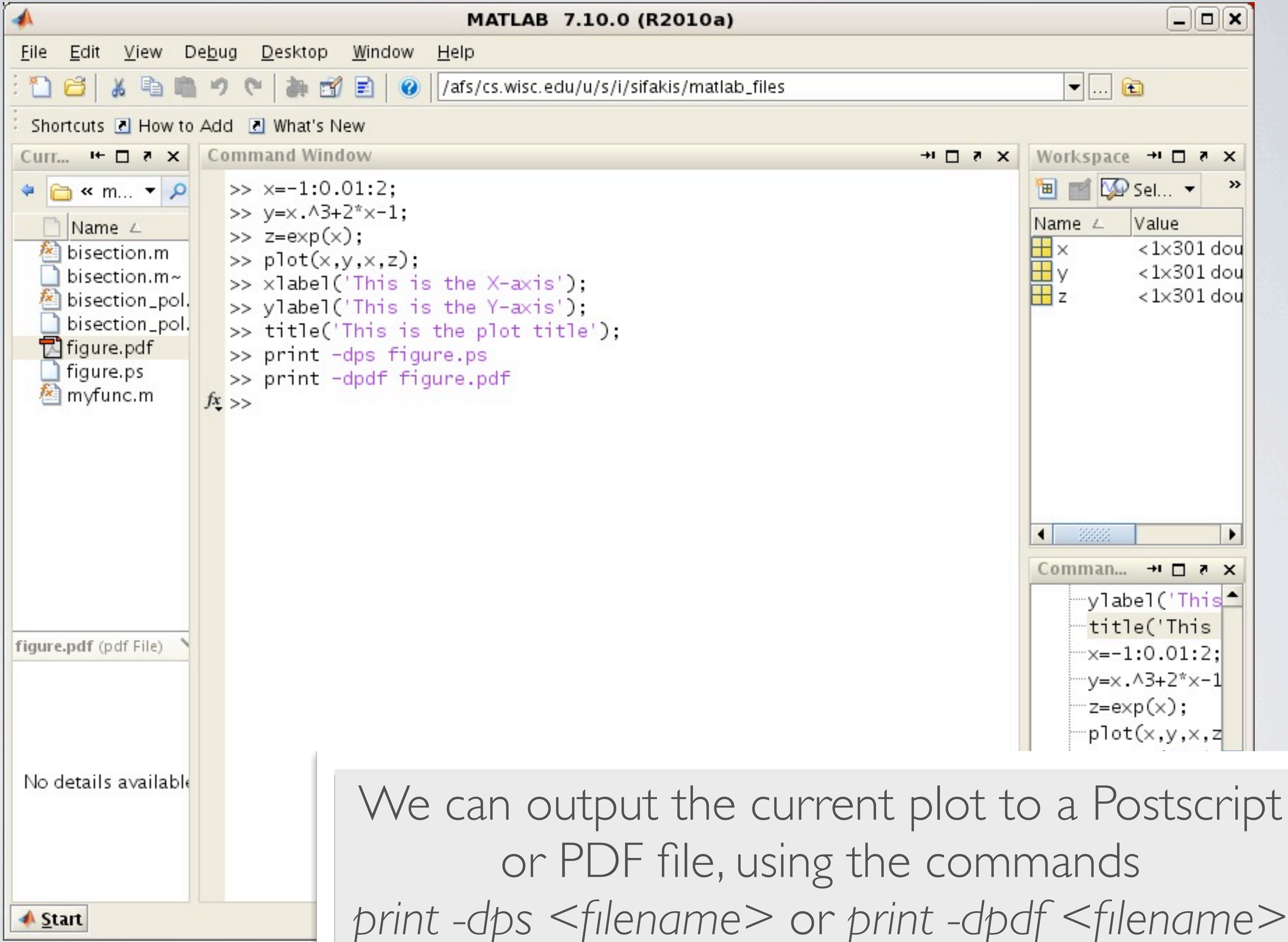
We can also draw several curves on the same plot by calling *plot(x1,y1,x2,y2 ... xN,yN). Note the use of operator .^ to denote raising to a power on an element-by-element basis*



Name	Value
x	<1x301 dou
y	<1x301 dou
z	<1x301 dou

```
plot(x,y,x,z
xlabel('This
ylabel('This
title('This
x=-1:0.01:2;
```

We can also draw several curves on the same plot by calling *plot(x1,y1,x2,y2 ... xN,yN)*. *Note the use of operator .^ to denote raising to a power on an element-by-element basis*



```
>> x=-1:0.01:2;
>> y=x.^3+2*x-1;
>> z=exp(x);
>> plot(x,y,x,z);
>> xlabel('This is the X-axis');
>> ylabel('This is the Y-axis');
>> title('This is the plot title');
>> print -dps figure.ps
>> print -dpdf figure.pdf
>>
```

```
ylabel('This
title('This
x=-1:0.01:2;
y=x.^3+2*x-1
z=exp(x);
plot(x,y,x,z
```

We can output the current plot to a Postscript or PDF file, using the commands *print -dps <filename>* or *print -dpdf <filename>*