HPFBU

Introduction to MatLab™

Dr. Öznur METE
University of Manchester
The Cockcroft Institute of Accelerator Science and Technology
İletişim Bilgileri
oznur.mete@cockcroft.ac.uk
oznur.mete@manchester.ac.uk
www.cern.ch/omete

This document includes material about MatLab to be discussed in HPFBU 2014 school.
More info about the product is provided by MatWorks on http://www.mathworks.com/

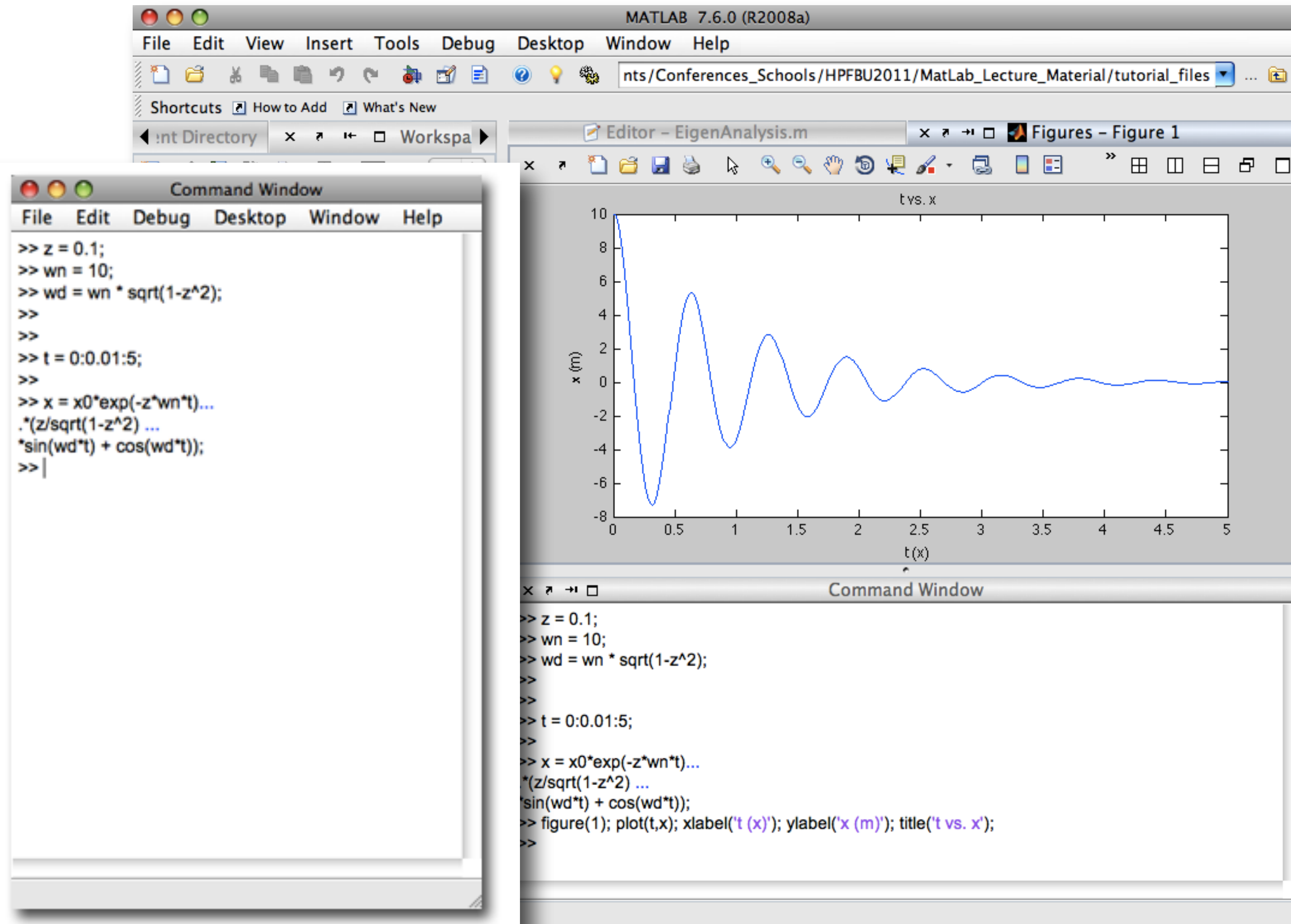
- MatLab is a powerful graphical calculator.
- Its built-in functions and libraries can be used for complicated calculations on large data sets.
- The results is visualised in the form of graphs and plots.

Given:

$$\xi = 0.1$$
$$\omega_n = 10$$
$$\omega_d = \omega_n \sqrt{1-\xi^2}$$
$$x_0 = 10$$

Plot the following function for t=0 to 5 s.

$$x(t) = x_0 e^{-\xi \omega_n t} \left(\frac{\xi}{\sqrt{(1-\xi^2)}} sin\omega_d t + cos\omega_d t \right)$$



MathWorks

MATLAB family

- **Math, statistics,** optimisation
- Control systems design and analyses
- Signal processing and communications
- Image processing and computer vision
- Test and measurement
- Computational finance
- Computational biology
- Code generation and verification
- Database connectivity and reporting

SIMULINK family

- Event based modelling
- Physical modelling
- Control system design and analysis
- …

POLYSPACE family

- Debug
- Prove
- Review
- …

http://www.mathworks.com/products/

PART 1 - Fundamentals of MATLAB

Basic Calculations in MATLAB

- MATLAB as a calculator
- Creating variables
- Locating data in MATLAB
- Inspecting contents of variables

Creating arrays

- Creating vectors
- Creating matrices

Manipulating arrays

- Indexing into arrays
- The colon (:) operator

Computing with arrays

- Matrix operations
- Eigenvalue analysis
- Array operations

Visualising mathematical functions

Writing your function in MATLAB

PART 2 - Hands-on Practice Session

Projects

- Graphical User Interface: Building a calculator
- Under-dumped string-mass system
- Gaussian fit to a given data set (on command line and by using Fitting Toolbox)
- Quadrupole scan analysis for emittance measurement (online analysis HW after diagnostics lecture)

PART I

FUNDAMENTALS of MATLAB

MATLAB as a calculator

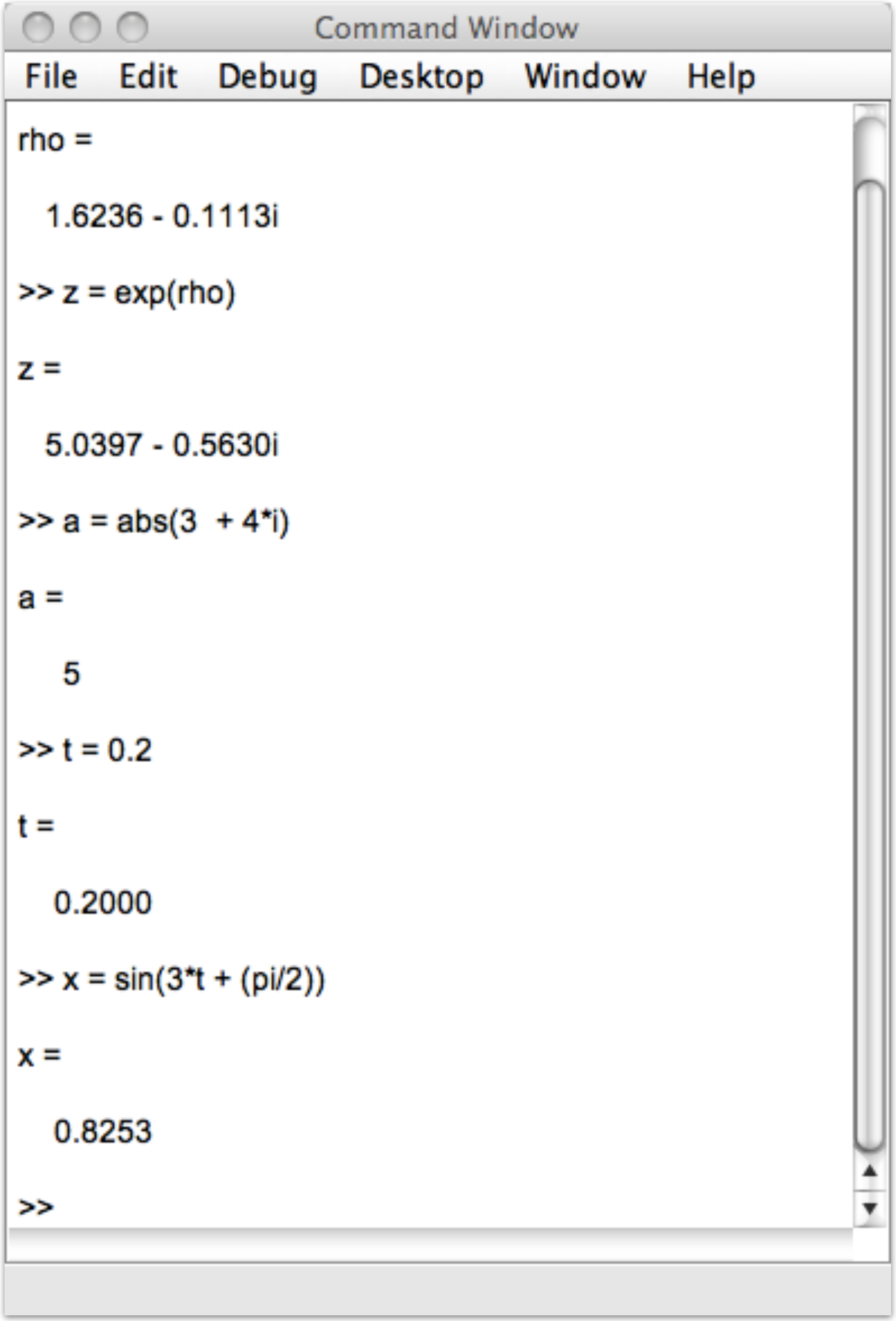
$$\rho = \frac{1 + \sqrt{(5 - i)}}{2}$$

$$z = e^{\rho}$$

$$a = |3 + 4i|$$

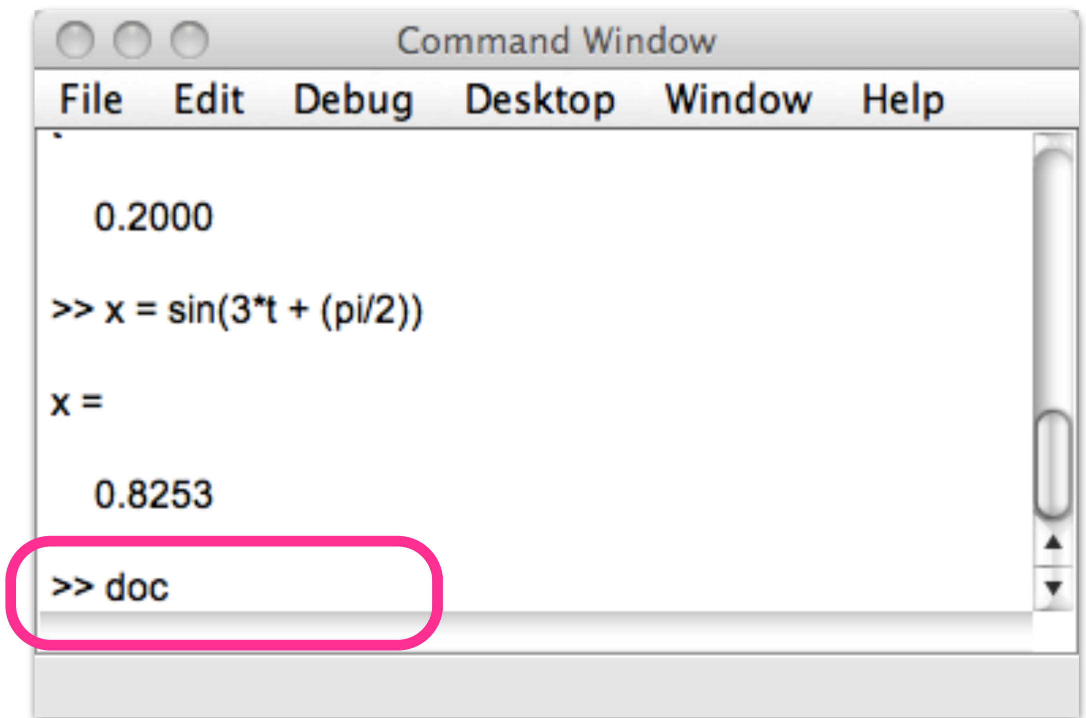
$$t = 0.2$$

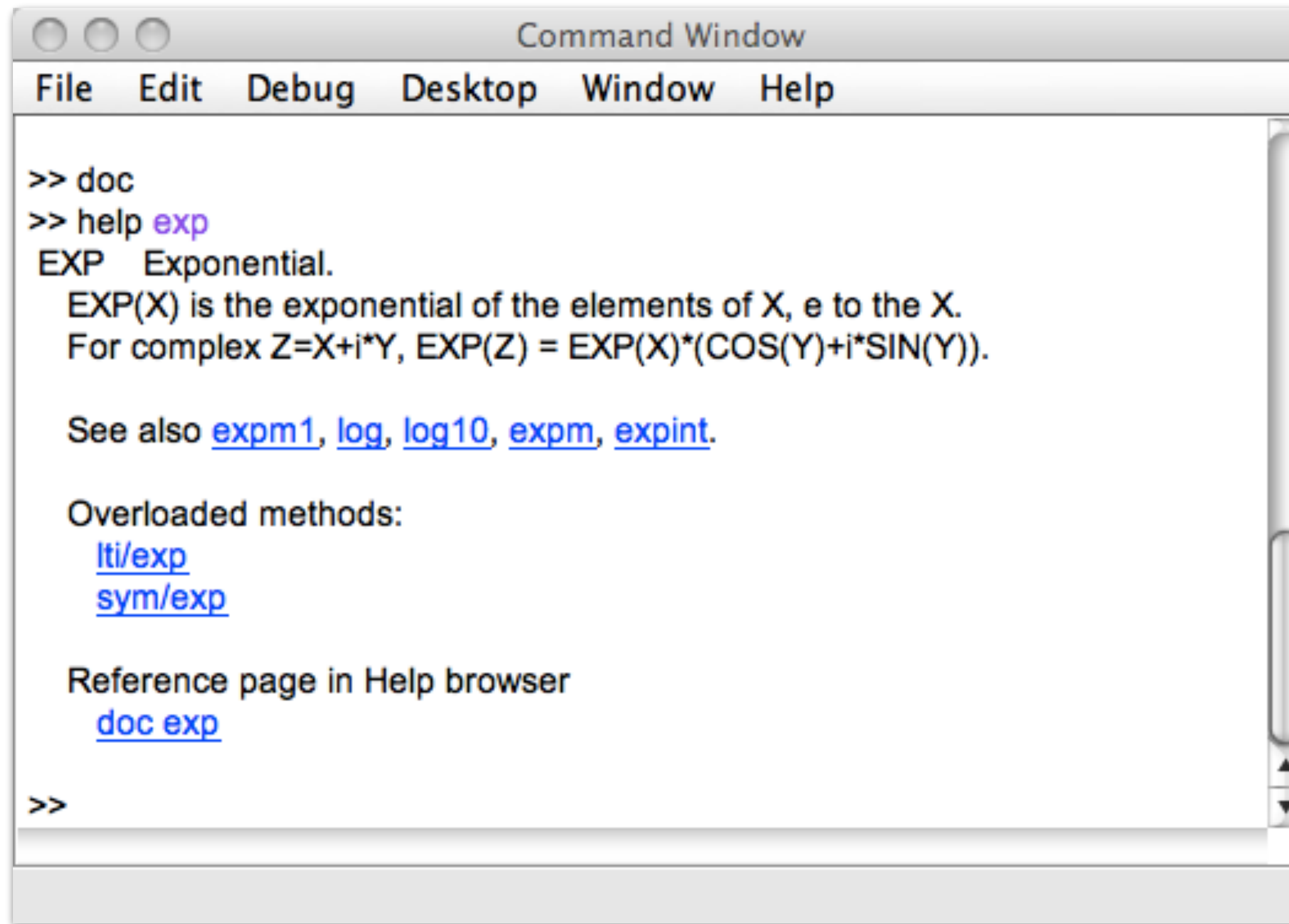
$$x = sin(3t + \frac{\pi}{2})$$



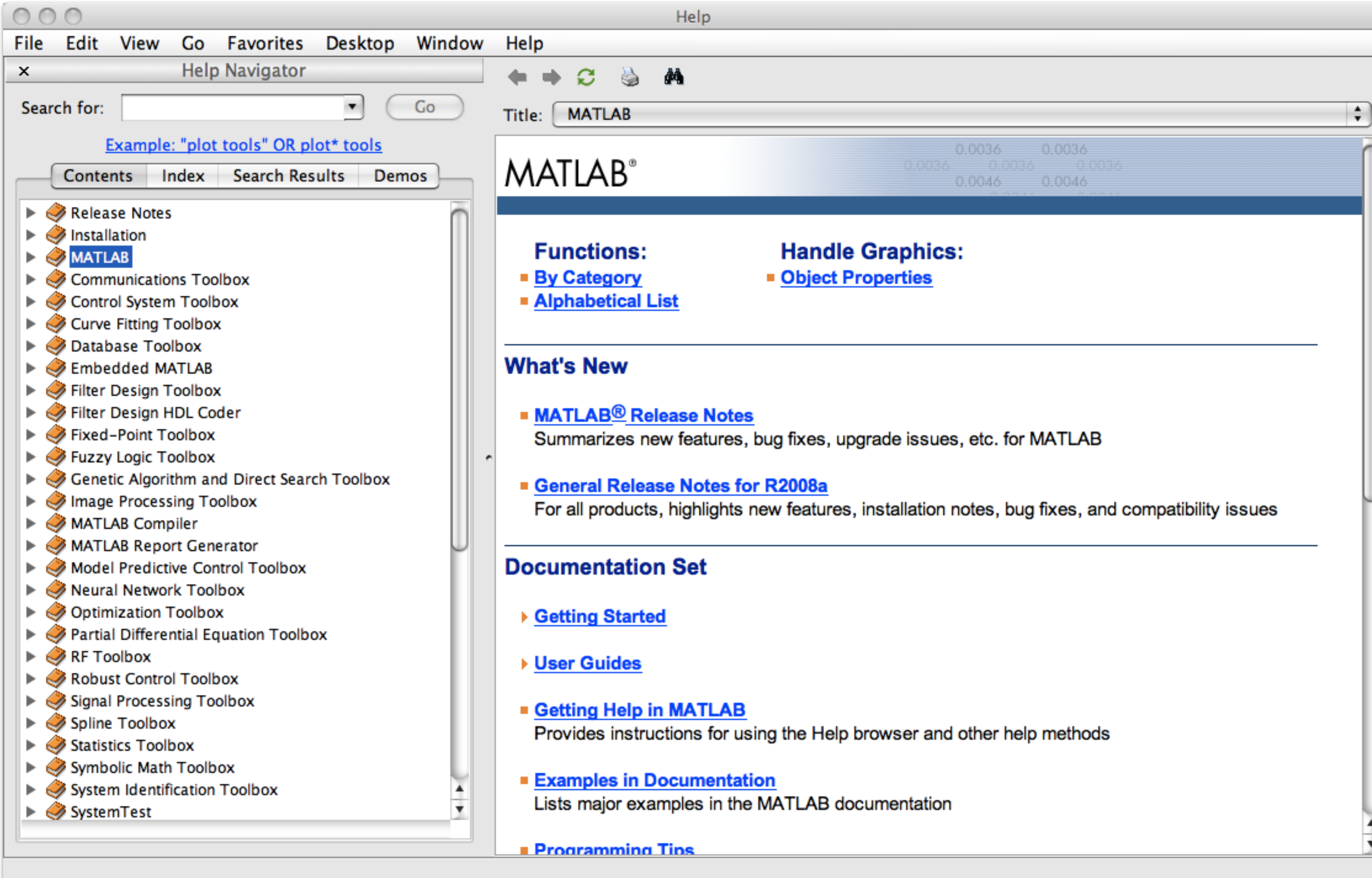
Mathematical functions

- MATLAB has many built-in functions.
- Information on MATLAB programming and the built-in functions can be found in the MATLAB documentation.

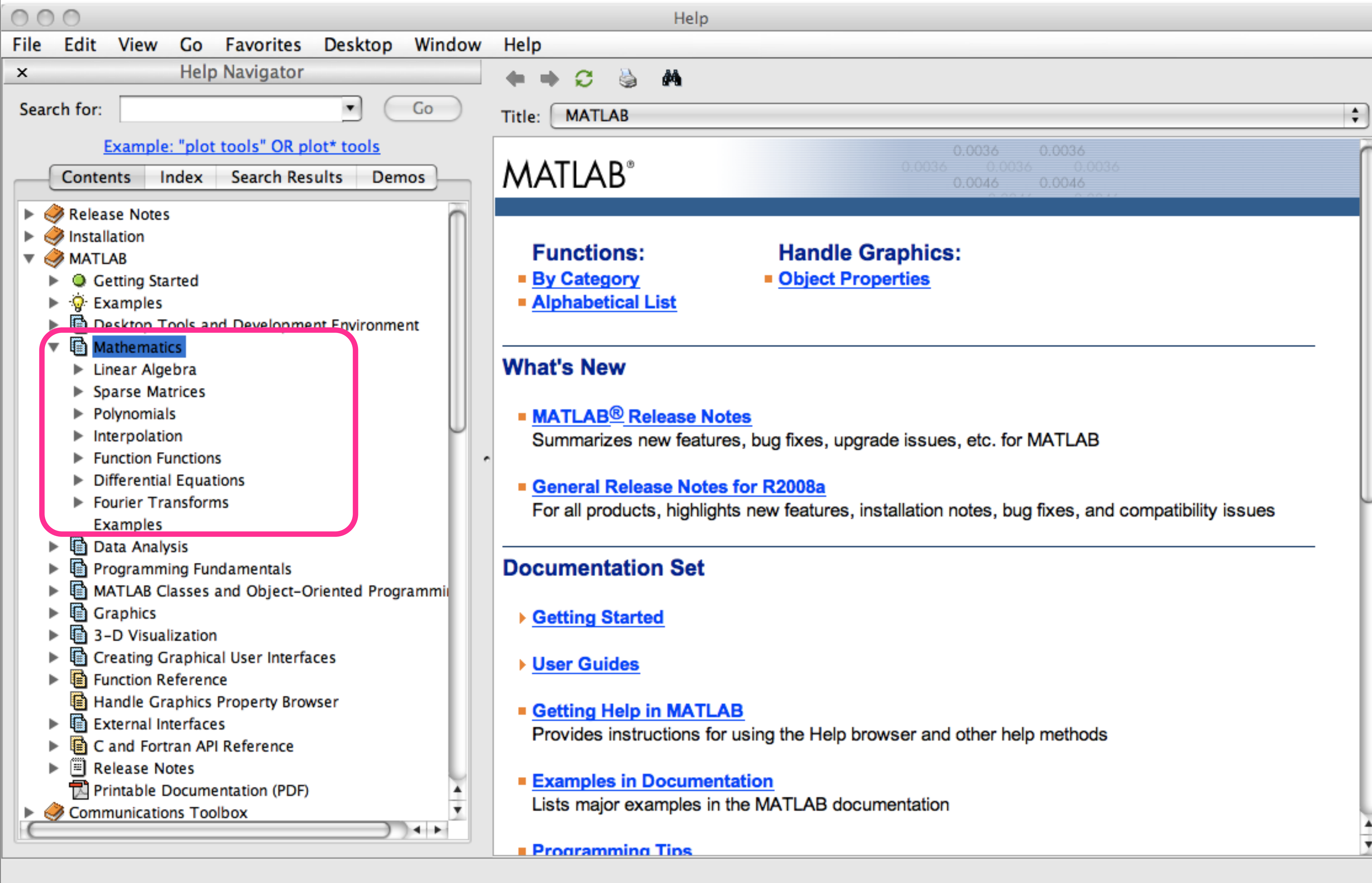




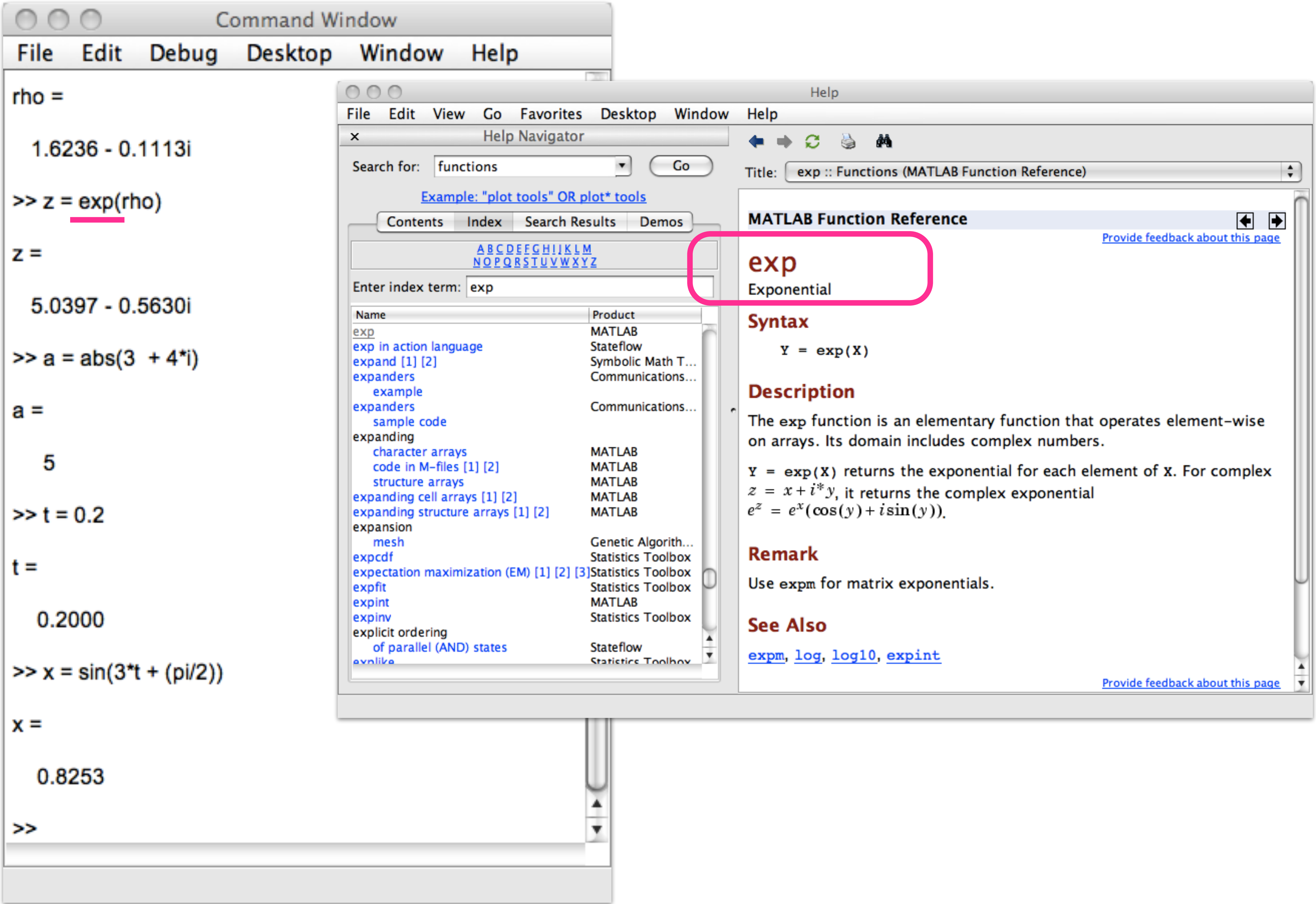
Mathematical functions



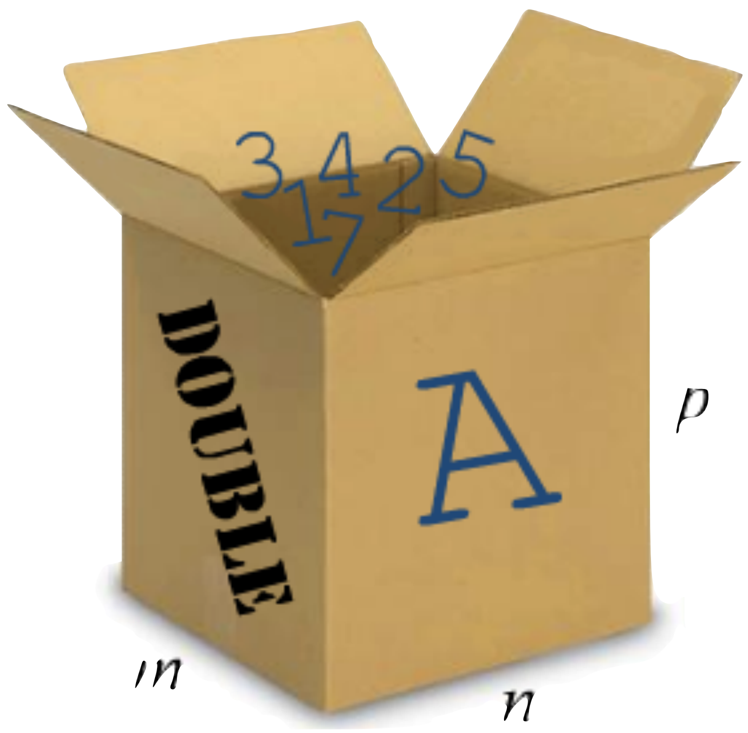
Mathematical functions



Mathematical functions



Data Containers



- MATLAB variables are data containers
- All variables are **arrays**
- Variables come in different **sizes** mxnxp ...
- Variables come in different **types** double, single, cell, ...

Nota Bene:

- In MATLAB, fundamental data type is matrix.
- Even scalar variables are treated as 1x1 arrays.
- The default numerical data type is double.

Creating Variables

Assign Operator

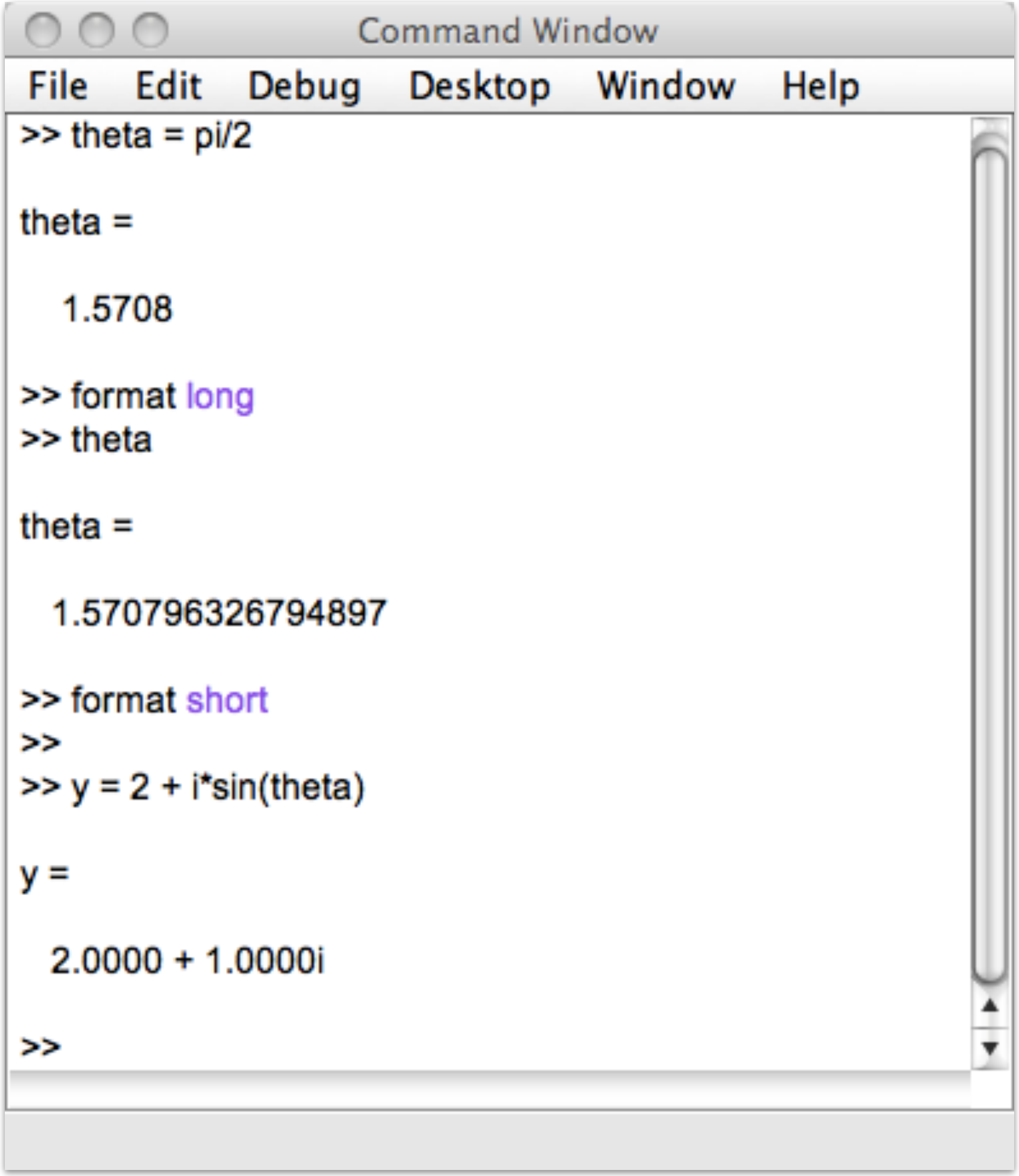
lumi_CLIC = 5.9e34

Variable Name

Value

$$\theta = \frac{\pi}{2}$$

$$y = 2 + isin(\theta)$$



Creating Variables

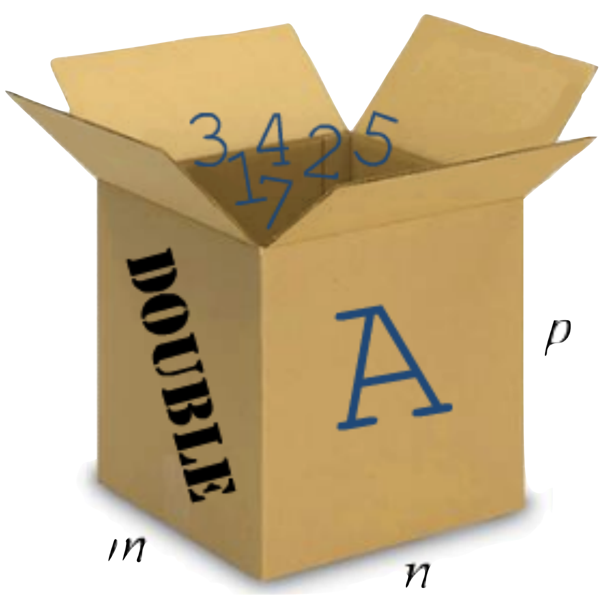
Exercise 02 - Creating variables in MATLAB

- A variable is a container for the data in MATLAB. True or false?

☑ True
☐ False

That's right!

In MATLAB, data can be placed in areas like containers, also referred to as **variables.**



Once created, the **name** of a variable is used as a **tag**, allowing access and manipulation of the data assigned to it.

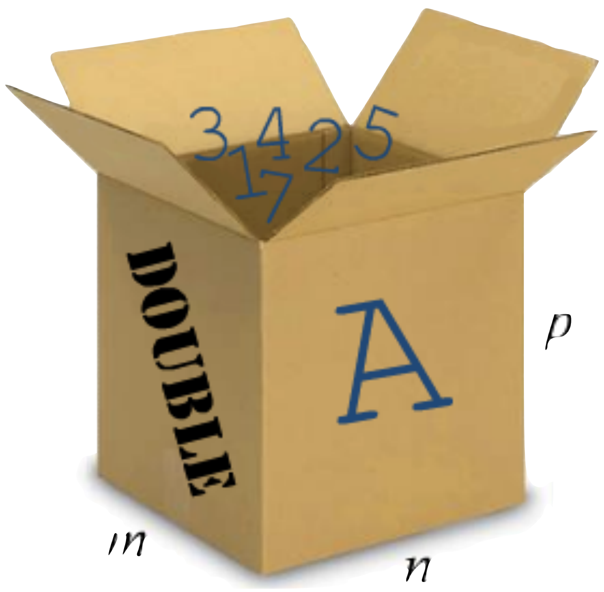
Creating Variables

Exercise 02 - Creating variables in MATLAB

- Which of the following are legitimate ways of assigning data to a variable?

Please try them in the MATLAB command line!

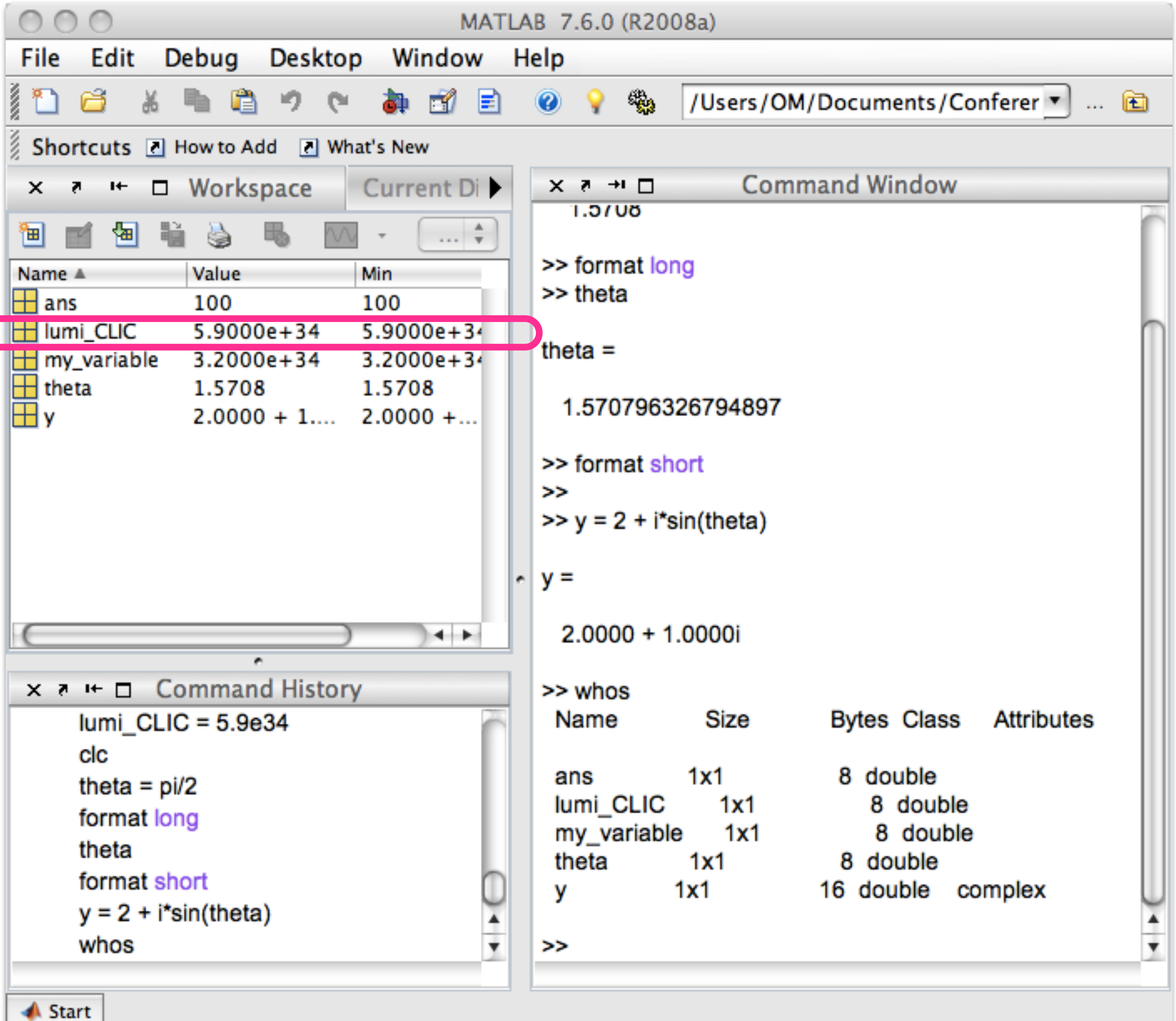
- ☐ a) a = b = 1
- ☐ b) 8*x + 2 = y
- ☑ c) temp_variable = (a + 1)/2



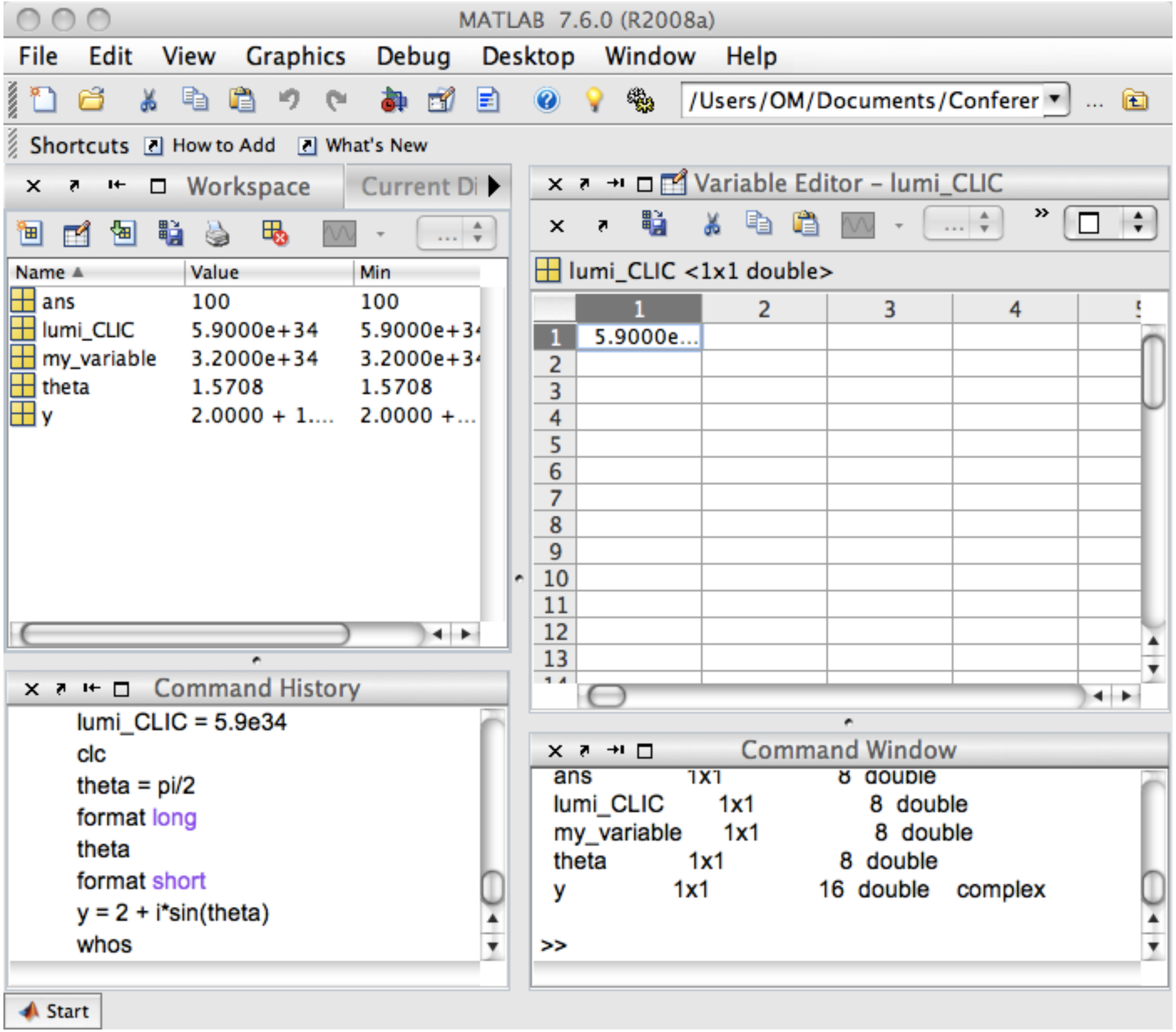
That's right!
The right-hand-side of the equals sign can be a value, another variable or the result of a calculation.
Also, multiple assignments are not allowed in a single command.

Accessing Data in MATLAB

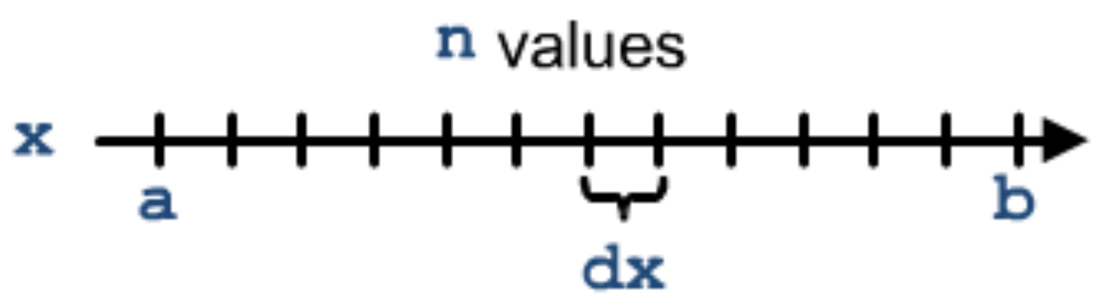
click on
a variable
within the
workspace



Accessing Data in MATLAB



Creating Vectors

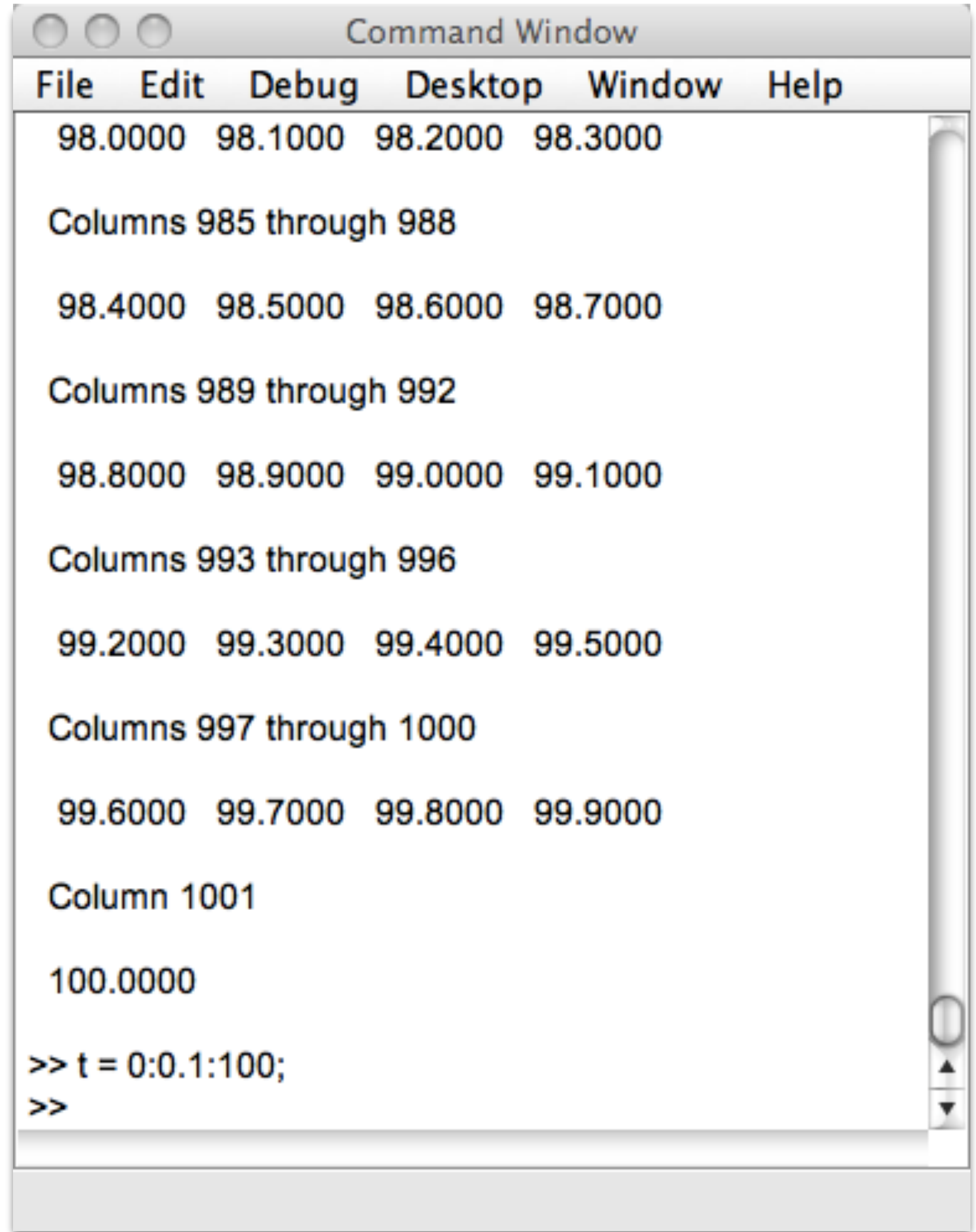



```
>> x = a:dx:b

>> x = linspace(a,b,n)
```

} row vectors

```
>> x = x'
```

← transpose to column vector



Creating Matrices

$$\begin{pmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{pmatrix}$$

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

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

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

} data entry mode

Column separator – , or space

Row separator – ; or enter

Command Window

File Edit Debug Desktop Window Help

```
>> ones(3)

ans =

     1     1     1
     1     1     1
     1     1     1

>> zeros(3)

ans =

     0     0     0
     0     0     0
     0     0     0

>> rand(4)

ans =

    0.8147    0.6324    0.9575    0.9572
    0.9058    0.0975    0.9649    0.4854
    0.1270    0.2785    0.1576    0.8003
    0.9134    0.5469    0.9706    0.1419

>>
```

Creating Arrays

Exercise 03 - Creating arrays in MATLAB

- Create the array below in MATLAB:

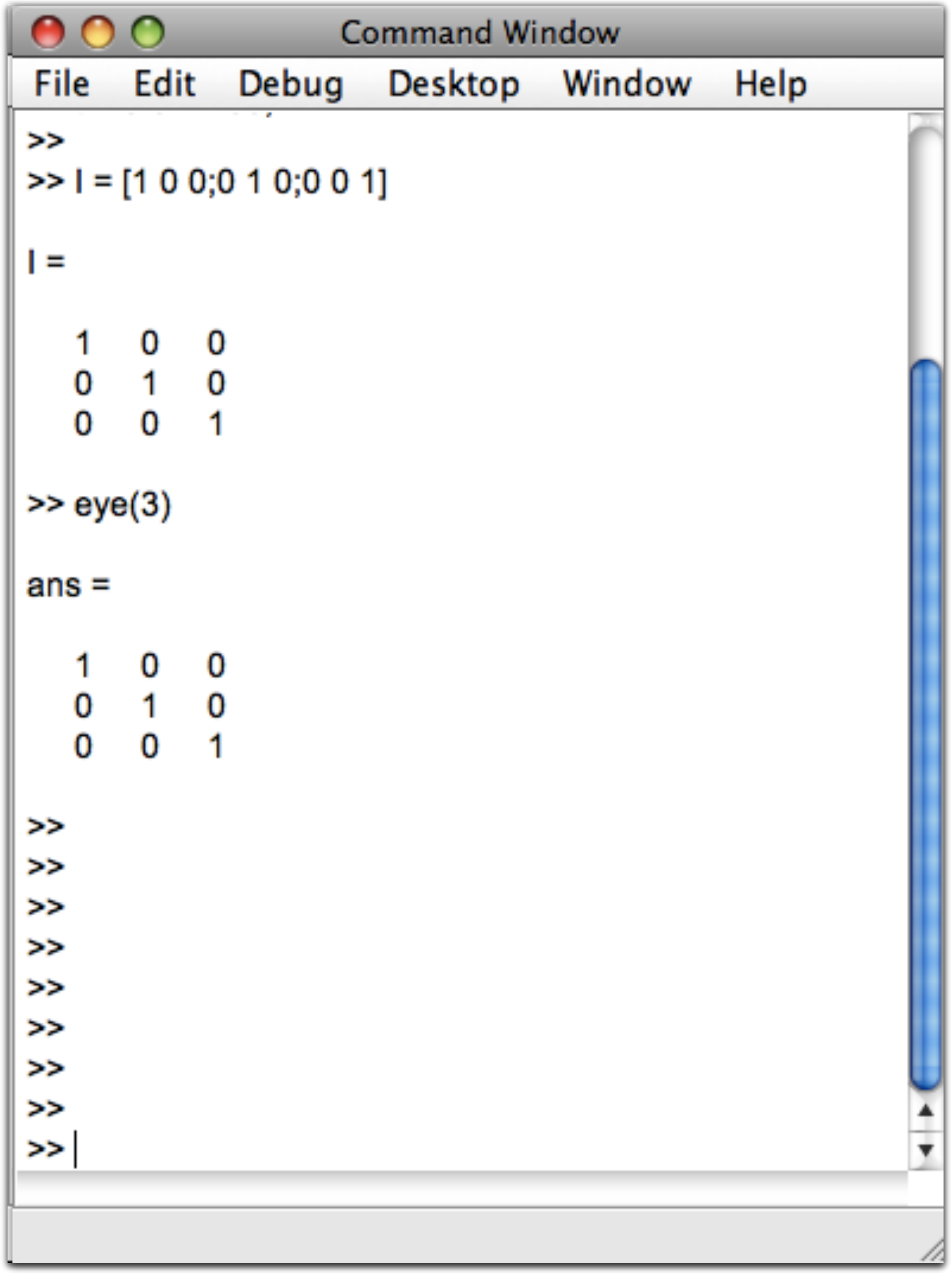
$$x = [2 \quad 4 \quad 6 \quad 8]$$

- Complete the command to suppress the command line output when the vector t is created.

$$>> t = 0 : 0.1 : 100$$

- Create this matrix:

$$I = \begin{pmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}$$



Manipulating Arrays

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

- Indexing

```
>> k = A(2,3)
>> block1 = A(2, [1 2])
```

- Colon operator

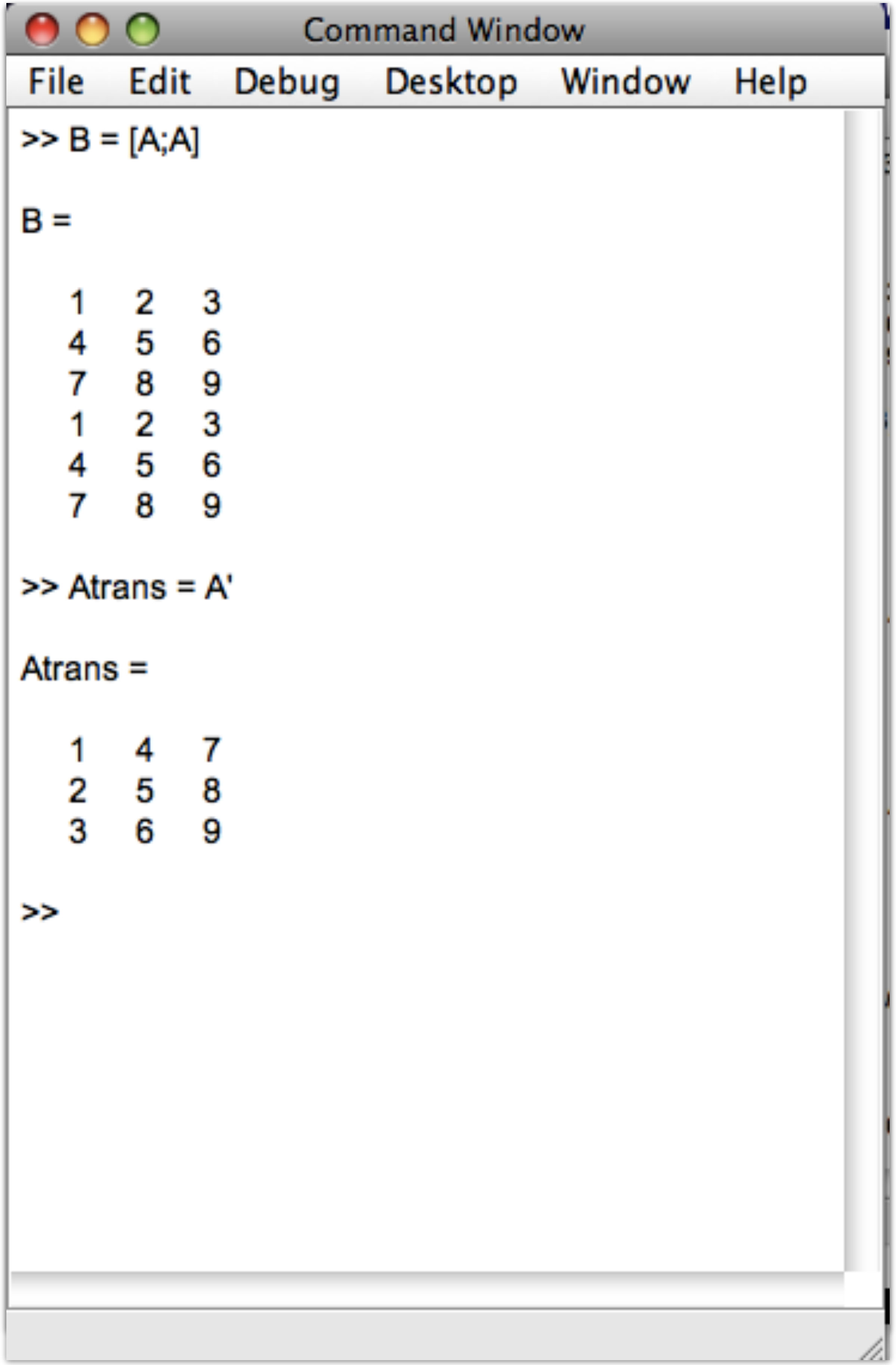
```
>> block2 = A(2, 1:2)
>> row2 = A(2,:)
```

- Concatenating matrices

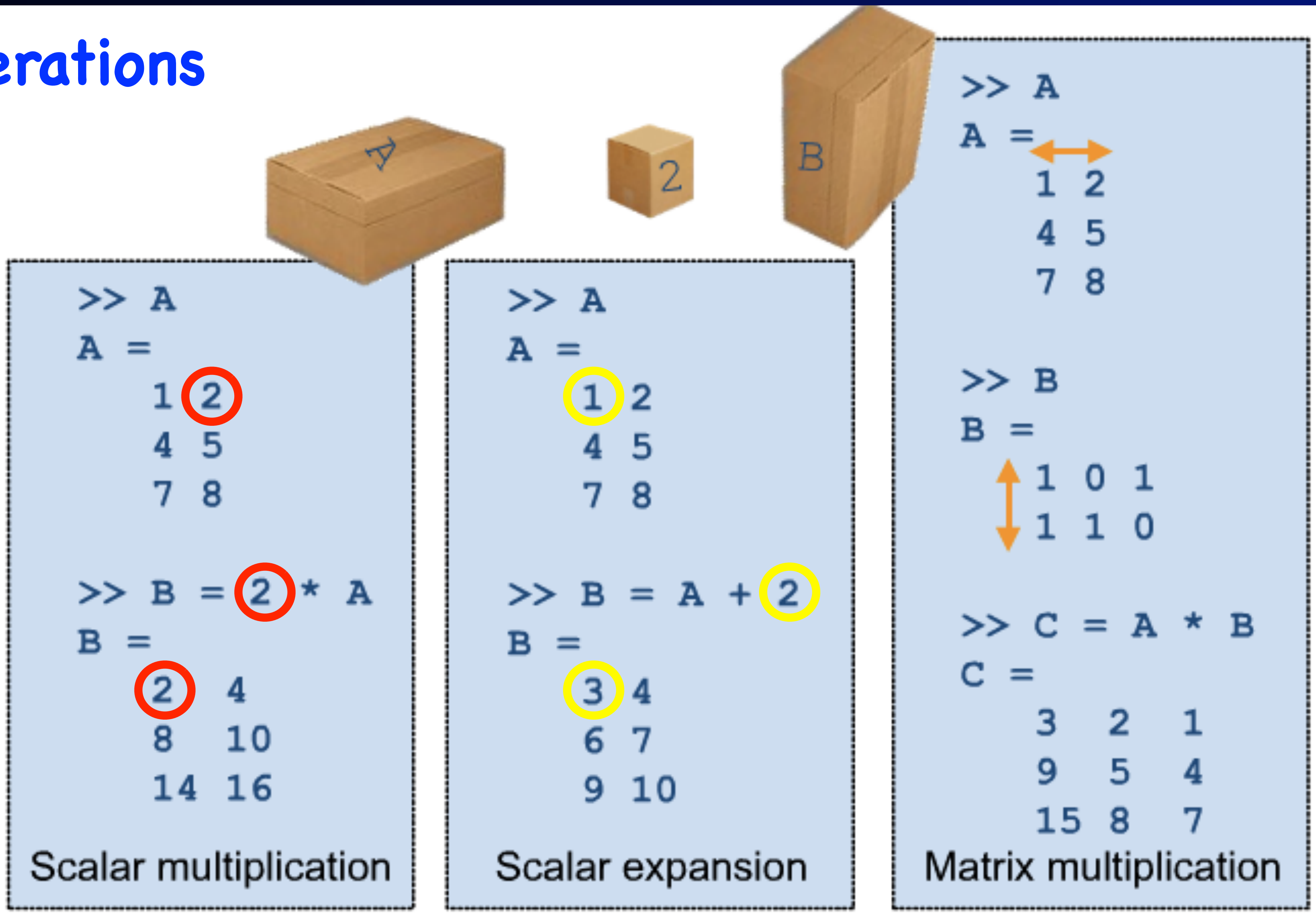
```
>> B = [A;A]
```

- Transposing

```
>> Atrans = A'
```



Matrix Operations



- MATLAB considers operands as matrices. (regular matrix algebra is valid)
- However multiplication with a scalar is a special case.
- For multiplication of two matrices the inner dimensions must agree.
- During addition/subtraction both matrices must have the same dimensions.
- For addition/subtraction with a scalar, the scalar expansion is automatically performed.

System of Linear Equations

- We have a set of linear equations and we want to find out the variables of this system.

$$\begin{aligned} x_1 + x_2 - x_3 &= 0 \\ 2x_1 + x_2 + x_3 &= 1 \\ x_1 - x_3 &= -1 \end{aligned}$$

$$\underbrace{\begin{pmatrix} 1 & 1 & -1 \\ 2 & 1 & 1 \\ 1 & 0 & -1 \end{pmatrix}}_{\textbf{A}} \underbrace{\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}}_{\textbf{x}} = \underbrace{\begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}}_{\textbf{b}}$$

$$Ax = b$$

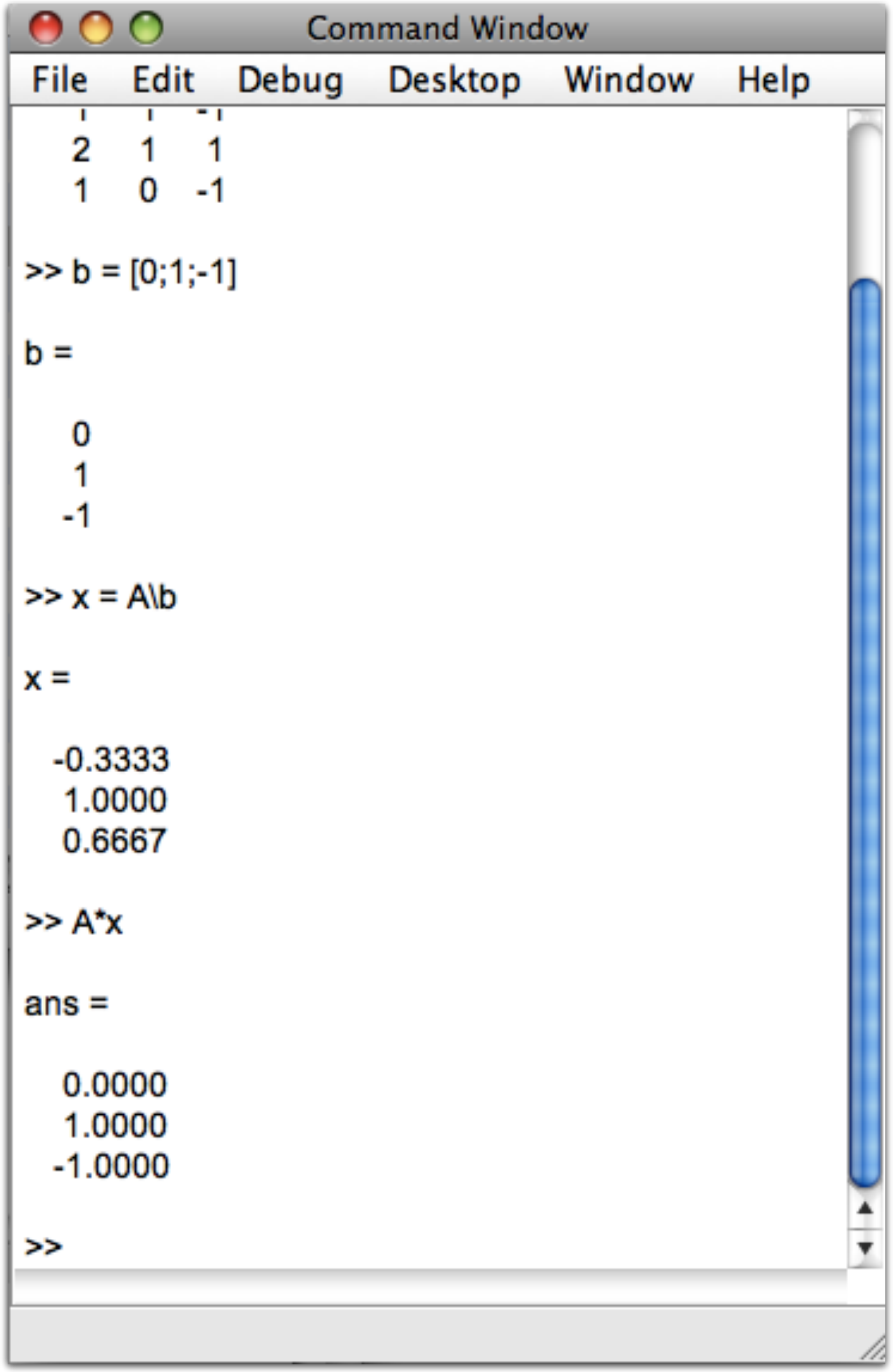
System of Linear Equations

- How we calculate this by using MATLAB?

$$x_1 + x_2 - x_3 = 0$$
$$2x_1 + x_2 + x_3 = 1$$
$$x_1 - x_3 = -1$$

$$\begin{pmatrix} 1 & 1 & -1 \\ 2 & 1 & 1 \\ 1 & 0 & -1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ -1 \end{pmatrix}$$

$$Ax = b$$



Eigenvalue Analysis

- Eigenvalue decomposition is a type of matrix operation that can be carried out to determine the eigenvalues and the eigenvectors of a matrix.

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

- Obtain the characteristic polynomial by extending the characteristic equation.

$$det(A - \lambda I) = 0$$

- Obtain the coefficients of the characteristic polynomial by using "poly" function.

$$c_n\lambda^n + ... + c_2\lambda^2 + c_1\lambda + c_0 = 0$$

- The eigenvalues can be computed by obtaining the roots of the function.

- One can also use **the "eig" function** in MATLAB, which returns the **eigenvalues** and the **eigenvectors** of a matrix.

Command Window

File Edit Debug Desktop Window Help

```
>> p = poly(A)

p =

    1.0000    1.0000   -5.0000

>> r = roots(p)

r =

   -2.7913
    1.7913

>> [eVec eVal] = eig(A)

eVec =

    0.9789    0.2043
    0.2043    0.9789


eVal =

    1.7913         0
         0   -2.7913

>>
```

Eigenvalue Analysis

- Eigenvalue decomposition is a type of matrix operation that can be carried out to determine the eigenvalues and the eigenvectors of a matrix.

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

↓

- Obtain the characteristic polynomial by extending the characteristic equation.

$$det(A - \lambda I) = 0$$

↓

- Obtain the coefficients of the characteristic polynomial by using "poly" function.

$$c_n\lambda^n + ... + c_2\lambda^2 + c_1\lambda + c_0 = 0$$

- The eigenvalues can be computed by obtaining the roots of the function.

- One can also use **the "eig" function** in MATLAB, which returns the **eigenvalues** and the **eigenvectors** of a matrix.

Command Window

File Edit Debug Desktop Window Help

```
>> p = poly(A)

p =

    1.0000    1.0000   -5.0000

>> r = roots(p)

r =

   -2.7913
    1.7913

>> [eVec eVal] = eig(A)

eVec =

    0.9789    0.2043
    0.2043    0.9789


eVal =

    1.7913         0
         0   -2.7913

>>
```

Array Operations

- Operands have to be in the same size and shape.
- The array operators operate element by element.


```
>> A
A =
    1 2 4
    5 7 8

>> B
B =
    1 0 1
    1 1 0

>> C = A + B
C =
    2 2 5
    6 8 8
```

Array addition

```
>> A
A =
    1 2 4
    5 7 8

>> B
B =
    1 0 1
    1 1 0

>> C = A .* B
C =
    1 0 4
    5 7 0
```

Array multiplication

```
>> A
A =
    1 2 4
    5 7 8

>> B = A.^2
B =
    1  4  16
   25 49  64
```

Array power

Array Operations

- Operands have to be in the same size and shape.
- The array operators operate element by element.


```
>> A
A =
    1  2  4
    5  7  8

>> B
B =
    1  0  1
    1  1  0

>> C = A + B
C =
    2  2  5
    6  8  8
```

Array addition

```
>> A
A =
    1  2  4
    5  7  8

>> B
B =
    1  0  1
    1  1  0

>> C = A .* B
C =
    1  0  4
    5  7  0
```

Array multiplication

```
>> A
A =
    1  2  4
    5  7  8

>> B = A.^2
B =
    1   4  16
   25  49  64
```

Array power

Array Operations

- Operands have to be in the same size and shape.
- The array operators operate element by element.


```
>> A
A =
    1 2 4
    5 7 8

>> B
B =
    1 0 1
    1 1 0

>> C = A + B
C =
    2 2 5
    6 8 8
```

Array addition

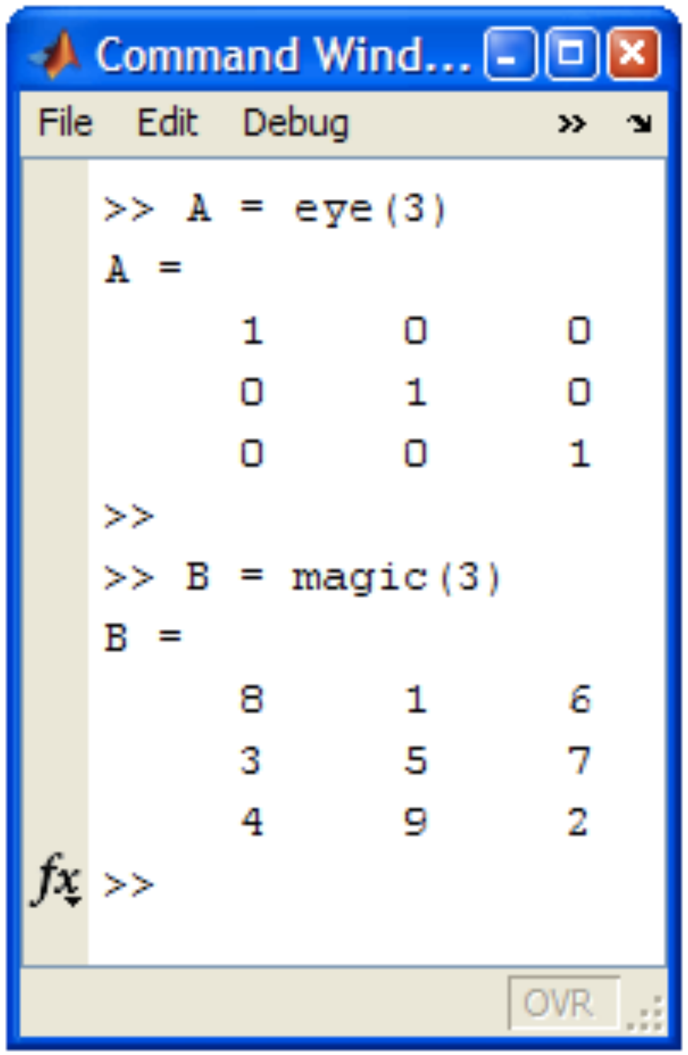
```
>> A
A =
    1 2 4
    5 7 8

>> B
B =
    1 0 1
    1 1 0

>> C = A .* B
C =
    1 0 4
    5 7 0
```

Array multiplication

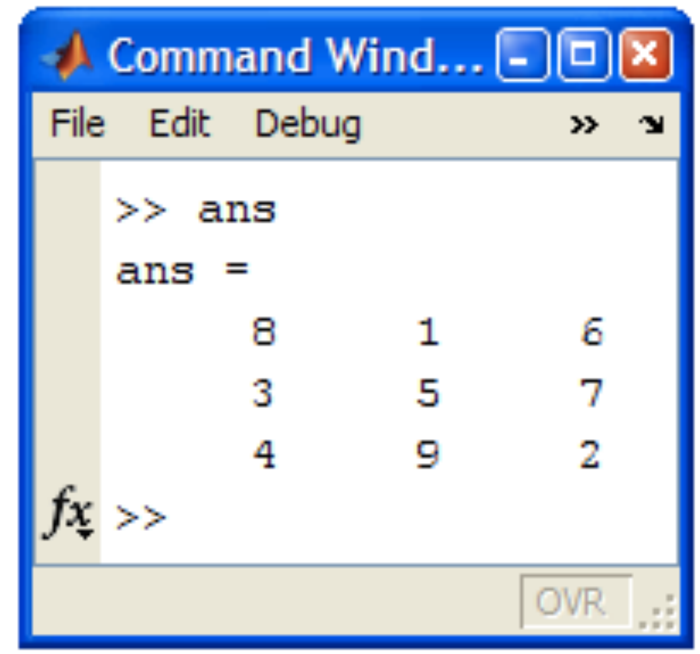
```
>> A
A =
    1 2 4
    5 7 8

>> B = A.^2
B =
    1   4  16
   25  49  64
```

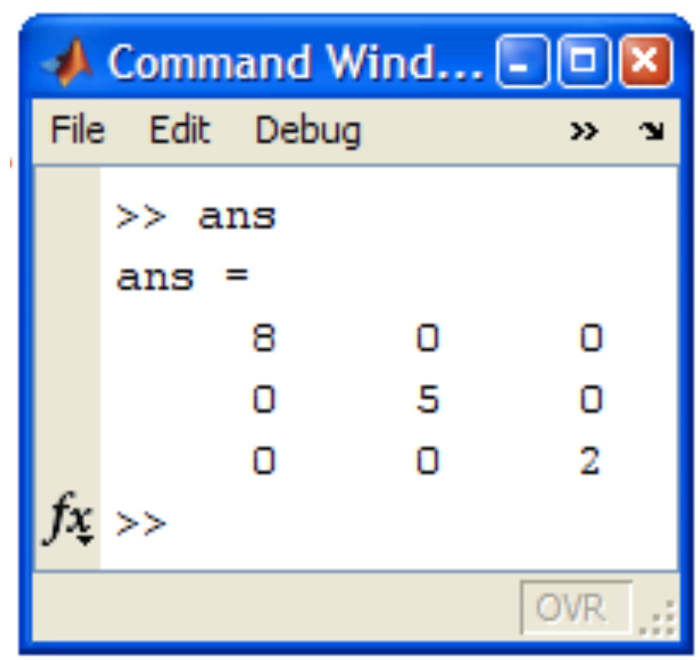
Array power

Exercise 04 - Match the expected outcome to the operators used.




```
>> A .* B

>> A * B
```





Visualising the Mathematical Functions

- Displacement of an under-damped spring-mass system.

Given:

$$\xi = 0.1 \quad \textit{damping coefficient}$$
$$\omega_n = 10 \quad \textit{simple harmonic oscillation frequency}$$
$$\omega_d = \omega_n\sqrt{1-\xi^2} \quad \textit{damped oscillation frequency}$$
$$x_0 = 10 \quad \textit{initial position}$$

Plot the following function for t=0 to 5 s.

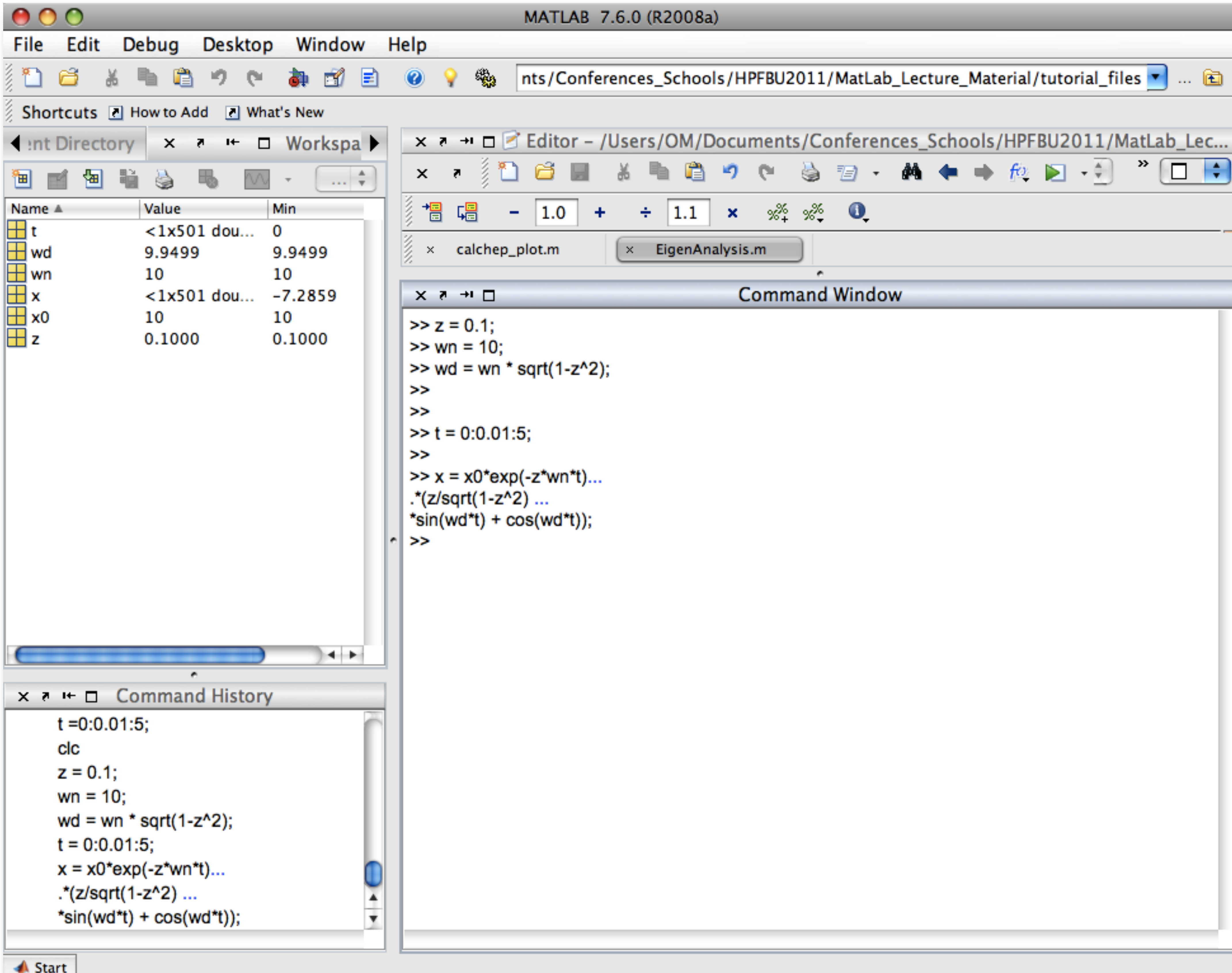
$$x(t) = x_0 e^{-\xi\omega_n t}\left(\frac{\xi}{\sqrt{(1-\xi^2)}} sin\omega_d t + cos\omega_d t\right)$$

Command Window

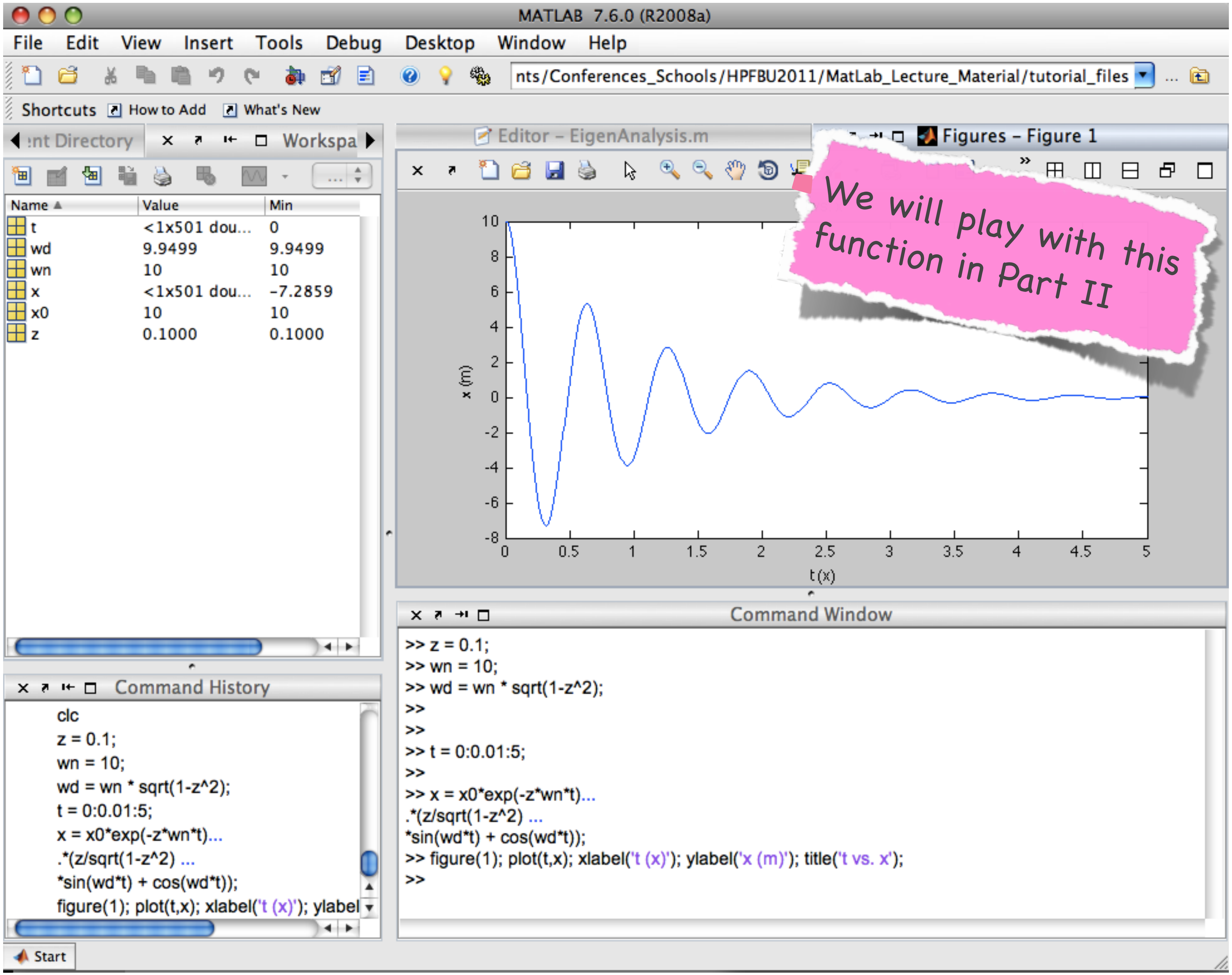
File Edit Debug Desktop Window Help

```
>> z = 0.1;
>> wn = 10;
>> wd = wn * sqrt(1-z^2);
>>
>>
>> t = 0:0.01:5;
>>
>> x = x0*exp(-z*wn*t)...
.*(z/sqrt(1-z^2) ...
*sin(wd*t) + cos(wd*t));
>> |
```

Visualising the Mathematical Functions



Visualising the Mathematical Functions



Writing your function in MATLAB

- We can write a function in order to perform specific jobs in MATLAB.
- It makes our life easier.
- One function - One Task!
- Let's repeat the previous exercise by using functions...

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%          Displacement of an under-damped spring-mass system.
%
%                    HPFBU 2011 - MATLAB Tutorial
%                     Help/Questions  --> O. Mete
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

function [x, t] = damped_oscillator(z)

% Parameters
%z = 0.1;
wn = 10;
x0 =  10;
wd = wn * sqrt(1-z^2);

% Time range
t= 0:0.01:5;

% Position function of the spring-mass system
x = x0*exp(-z*wn*t).*(z/sqrt(1-z^2)*sin(wd*t) + cos(wd*t));

end
```

- Instead writing the all commands and assignments by hand into the command line, we can gather them all inside a ".m" file.
- We can relate them with a function.
- Functions are called by their attributes.
- Their outputs can be assigned to variables.

PART II

HANDS-ON PRACTICE SESSION

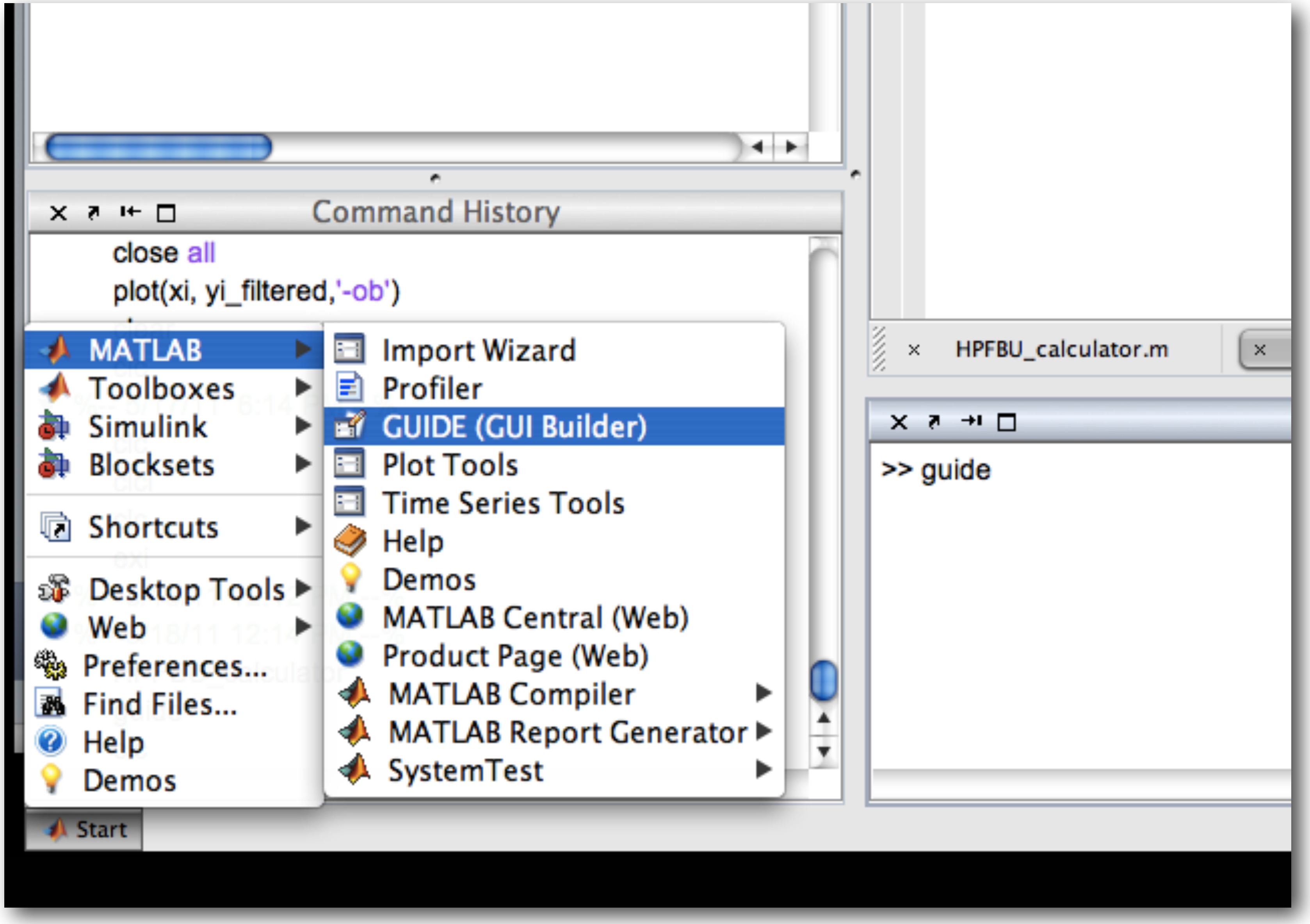
PART 2 - Hands-on Practice Session

Projects

- **Graphical User Interface: Building a calculator**
- Under-dumped string-mass system
- Gaussian fit to a given data set (on command line and by using Fitting Toolbox)
- Quadrupole scan analysis for emittance measurement

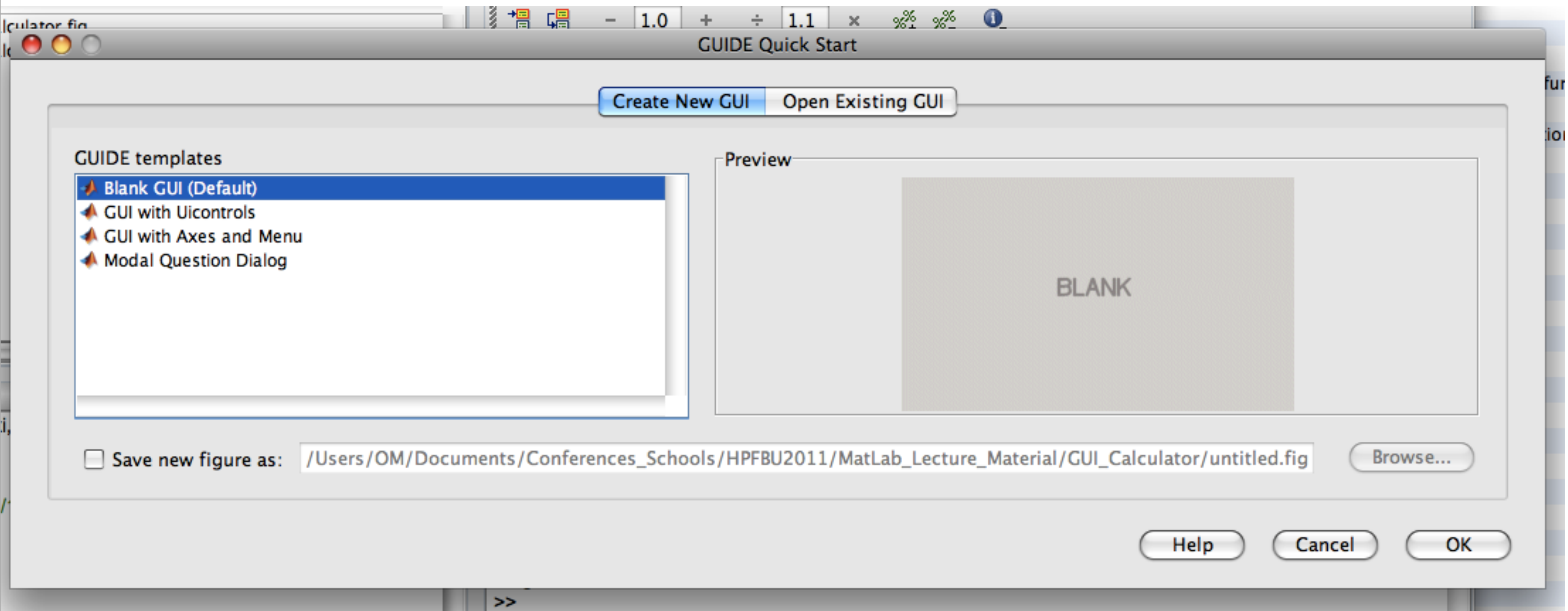
Graphical User Interface: Building a Calculator

- Let's create a GUI that does basic mathematical calculations, interactively.
- Call the MATLAB GUI builder by typing "guide" in the command window,
- or from MATLAB start menu as shown:



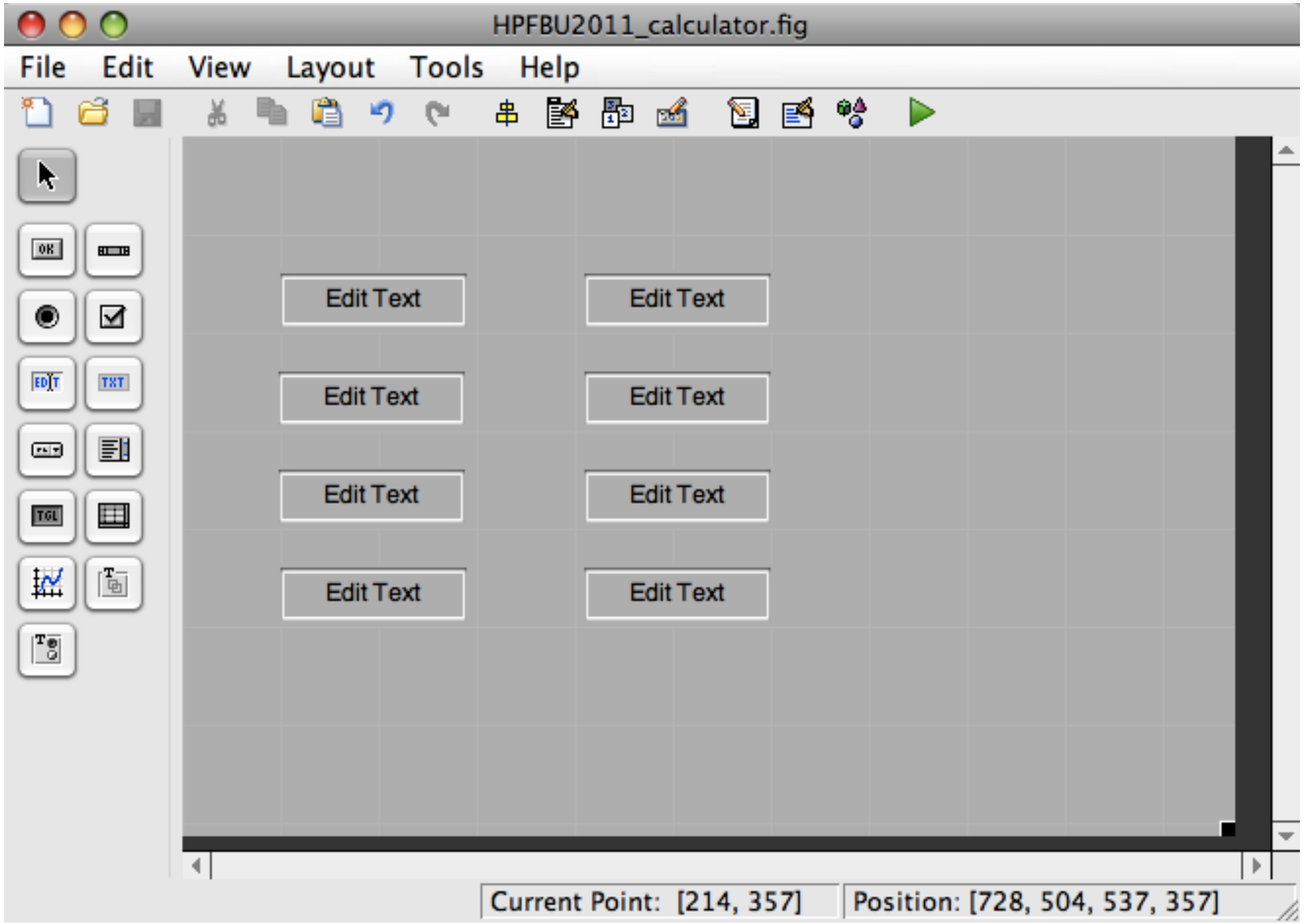
Graphical User Interface: Building a Calculator

- Choose one of the templates of the GUI builder.



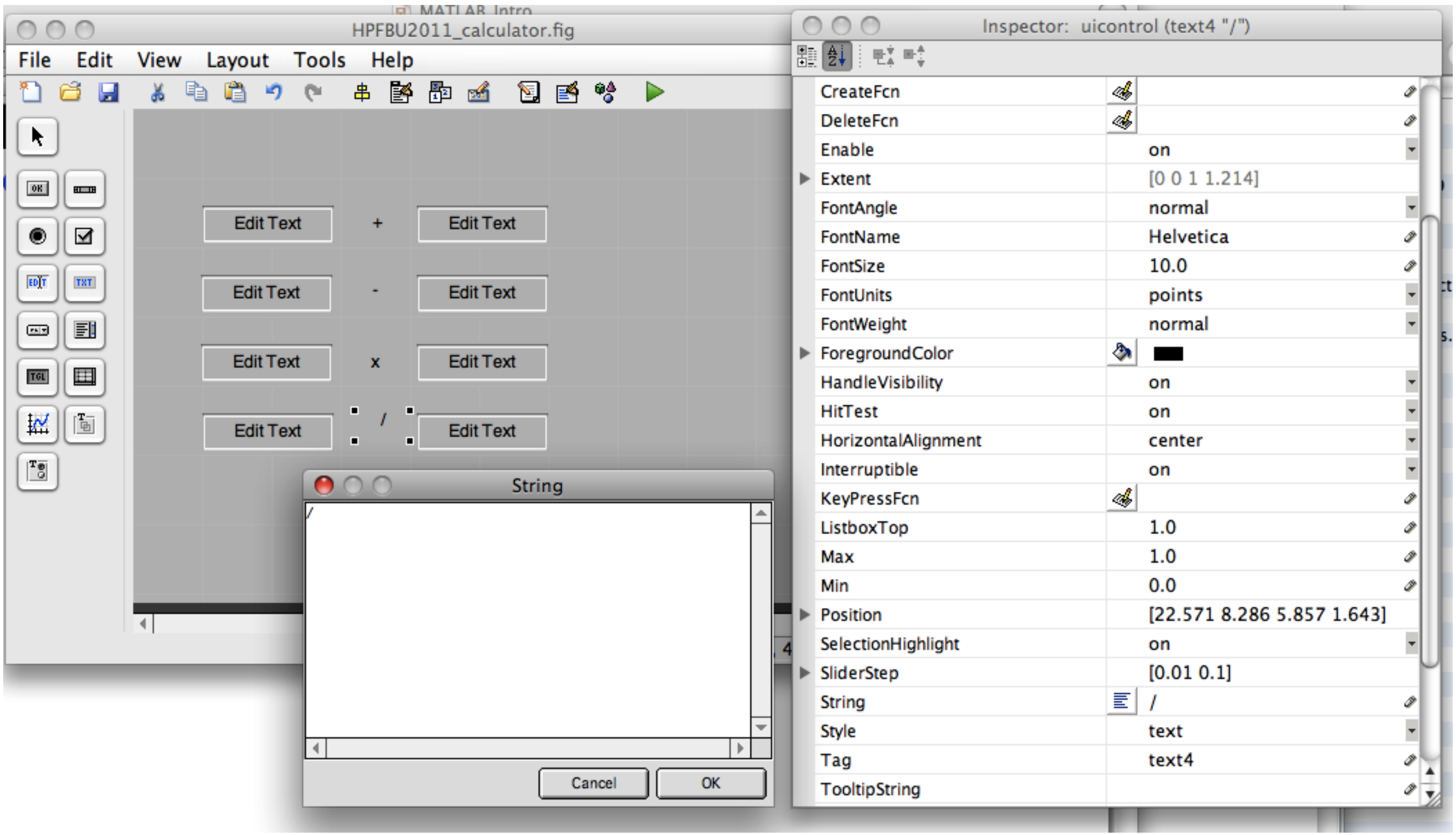
Graphical User Interface: Building a Calculator

- Add 8 “Edit Text” objects on the GUI panel to form our input boxes.



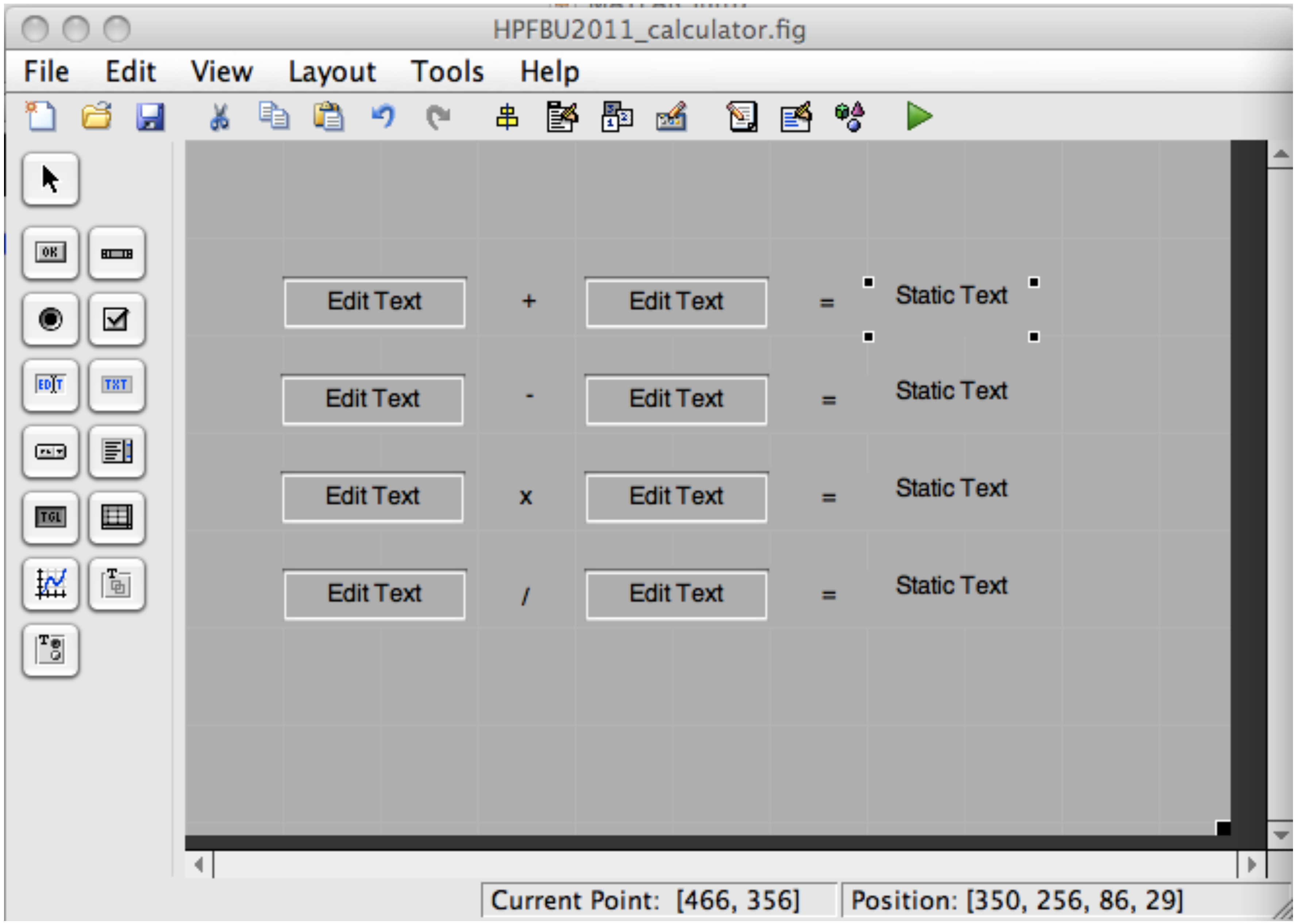
Graphical User Interface: Building a Calculator

- Add 4 “static text” objects on the panel to indicate the mathematical operators.
- You can edit each text box by using the “String” property from the “Inspector”.



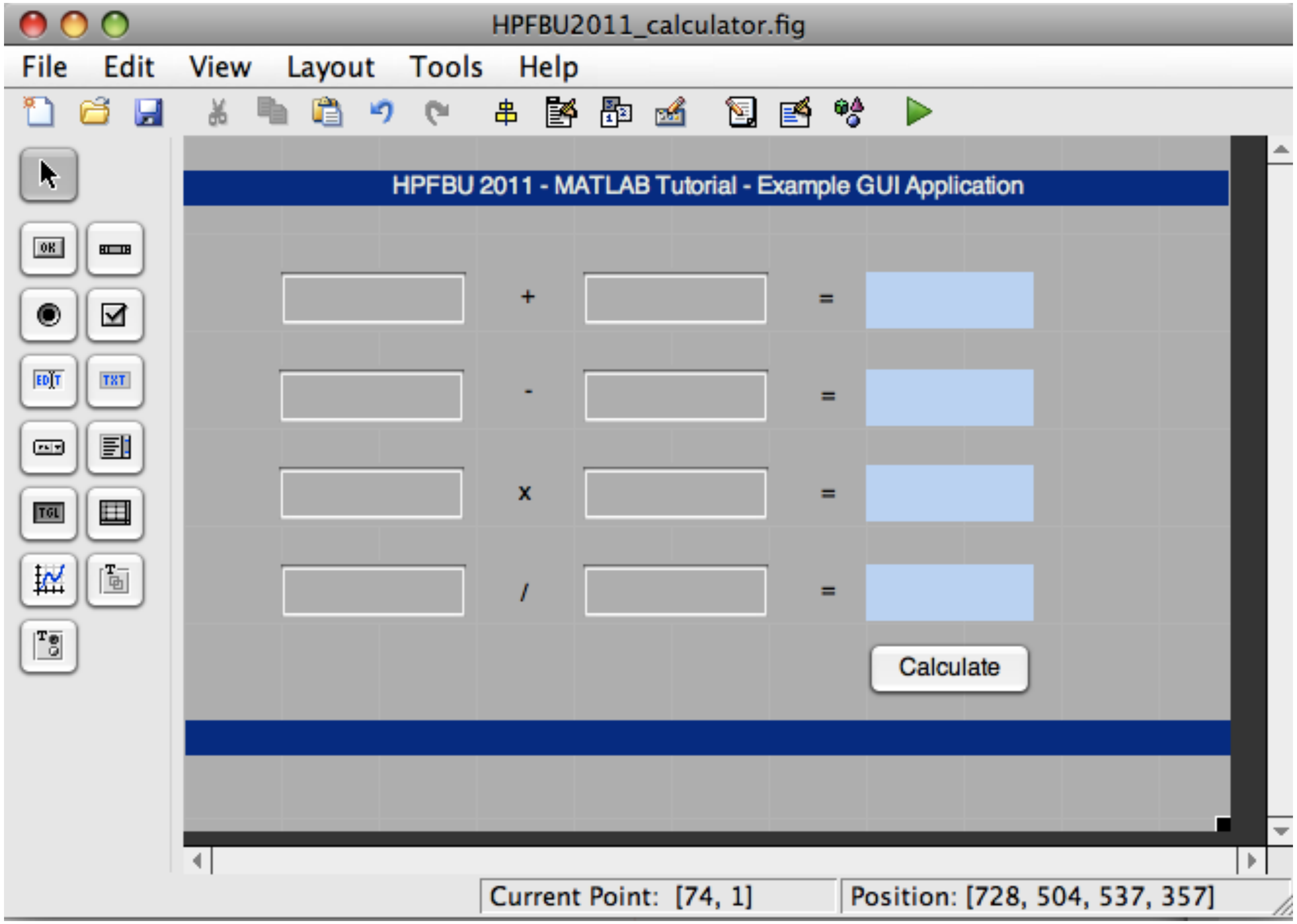
Graphical User Interface: Building a Calculator

- Add "equals" signs and 4 additional "static text" boxes to display the results of the calculations.



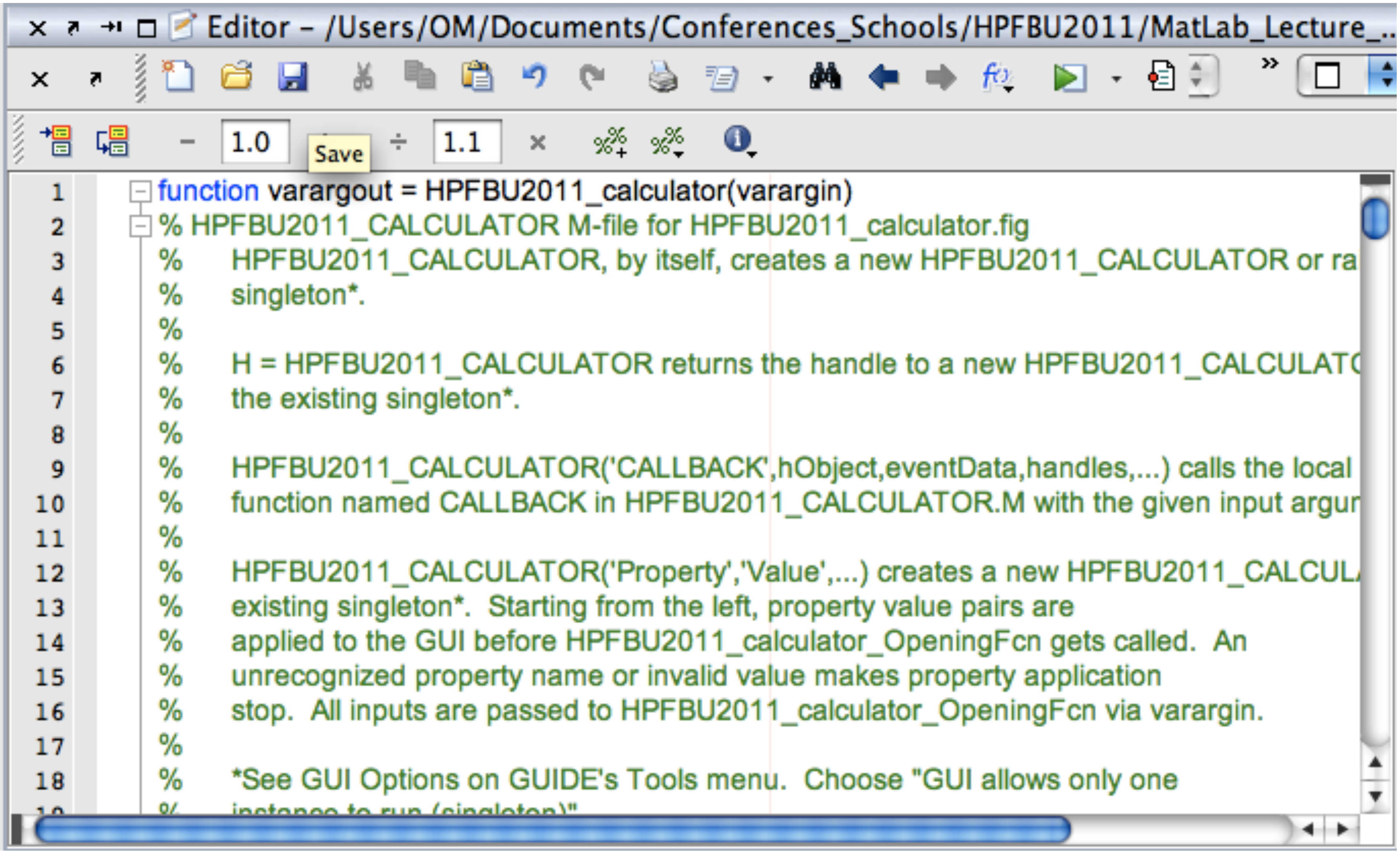
Graphical User Interface: Building a Calculator

- Some make-up for your panel :)



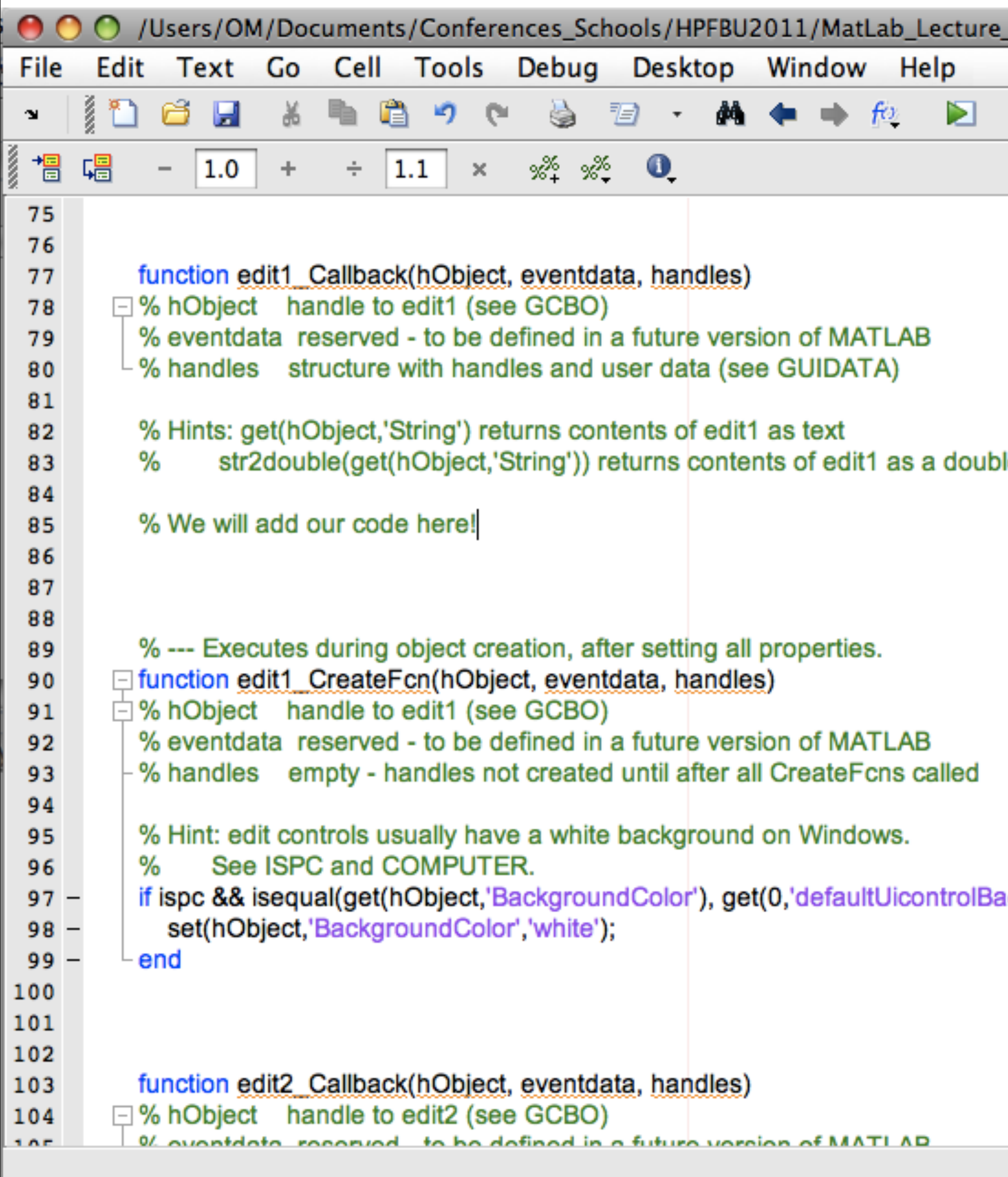
Graphical User Interface: Building a Calculator

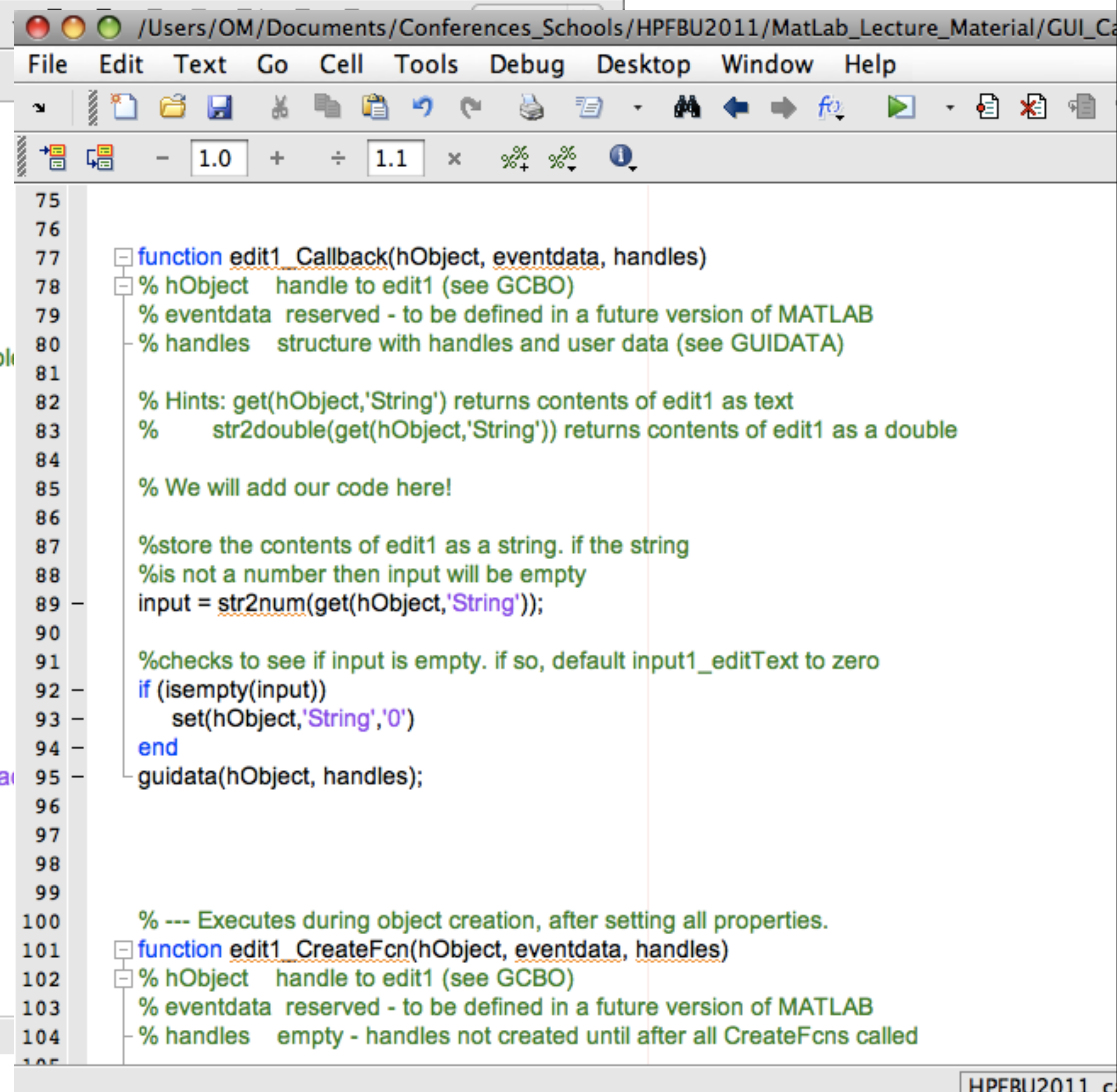
- The script for the GUI will be automatically generated when we save our project.



Graphical User Interface: Building a Calculator

- First, we will edit the "callback" functions of the objects.
- Repeat the same for all edit box callback functions that will be used for the data entry.





Graphical User Interface: Building a Calculator

- Edit the callback function for the "Calculate" "pushbutton" object.

/Users/OM/Documents/Conferences_Schools/HPFBU2011/MatLab_Lecture_Material/GUI_Calculator/HPFBU2011_calculator.m

File Edit Text Go Cell Tools Debug Desktop Window Help

```
% Hint: edit controls usually have a white b
%       See ISPC and COMPUTER.
if ispc && isequal(get(hObject,'Background
    set(hObject,'BackgroundColor','white');
end


% --- Executes on button press in calculate
function calculate_pushbutton1_Callback(h
% hObject    handle to calculate_pushbutto
% eventdata  reserved - to be defined in a
% handles    structure with handles and us

% We will add our code here!
```

/Users/OM/Documents/Conferences_Schools/HPFBU2011/MatLab_Lecture_Material/GUI_Calculator/HPFBU2011_calculator.m*

File Edit Text Go Cell Tools Debug Desktop Window Help

```
% eventdata  reserved - to be defined in a future versio
% handles    structure with handles and user data (see

% We will add our code here!

% Toplama islemi
sayi1 = get(handles.edit1,'String');
sayi2 = get(handles.edit2,'String');
% sayi1 and sayi2 are variables of Strings type, and nee
% to variables of Number type before they can be adde

toplam = str2num(sayi1) + str2num(sayi2);
c = num2str(toplam);
% need to convert the answer back into String type to d
set(handles.text9,'String',c);
guidata(hObject, handles);

% Cikarma islemi
cikar1 = get(handles.edit3,'String');
cikar2 = get(handles.edit4,'String');
% cikar1 and cikar2 are variables of Strings type, and n
% to variables of Number type before they can be adde

cikarma_sonucu = str2num(cikar1) - str2num(cikar2);
e = num2str(cikarma_sonucu);
% need to convert the answer back into String type to d
set(handles.text10,'String',e);
guidata(hObject, handles);
```

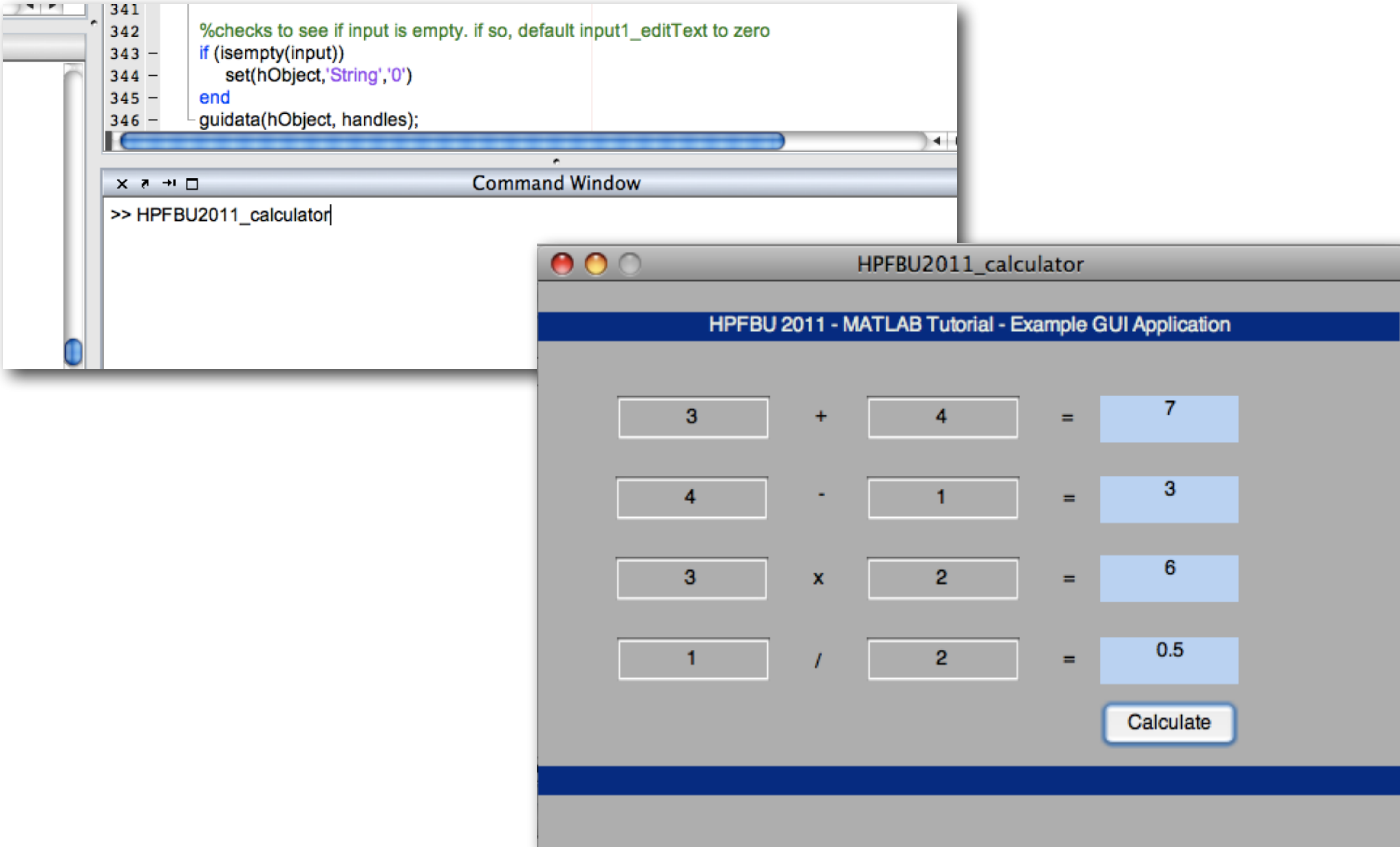
/Users/OM/Documents/Conferences_Schools/HPFBU2011/MatLab_Lect

File Edit Text Go Cell Tools Debug Desktop Window Help

```
set(handles.text10,'String',e);
guidata(hObject, handles);

% Carpma islemi
carp1 = get(handles.edit5,'String');
carp2 = get(handles.edit6,'String');
% a and b are variables of Strings type, and need to be converted
% to variables of Number type before they can be added together

carpim = str2num(carp1) * str2num(carp2);
carp3 = num2str(carpim);
% need to convert the answer back into String type to display it
set(handles.text11,'String',carp3);
guidata(hObject, handles);


% Bolme islemi
bol1 = get(handles.edit7,'String');
bol2 = get(handles.edit8,'String');
% a and b are variables of Strings type, and need to be converted
% to variables of Number type before they can be added together

bolum = str2num(bol1) / str2num(bol2);
bol3 = num2str(bolum);
% need to convert the answer back into String type to display it
set(handles.text12,'String',bol3);
guidata(hObject, handles);
```

Graphical User Interface: Building a Calculator

- Call your GUI by using its name in the command window.
- And, try a few calculations!



PART 2 - Hands-on Practice Session

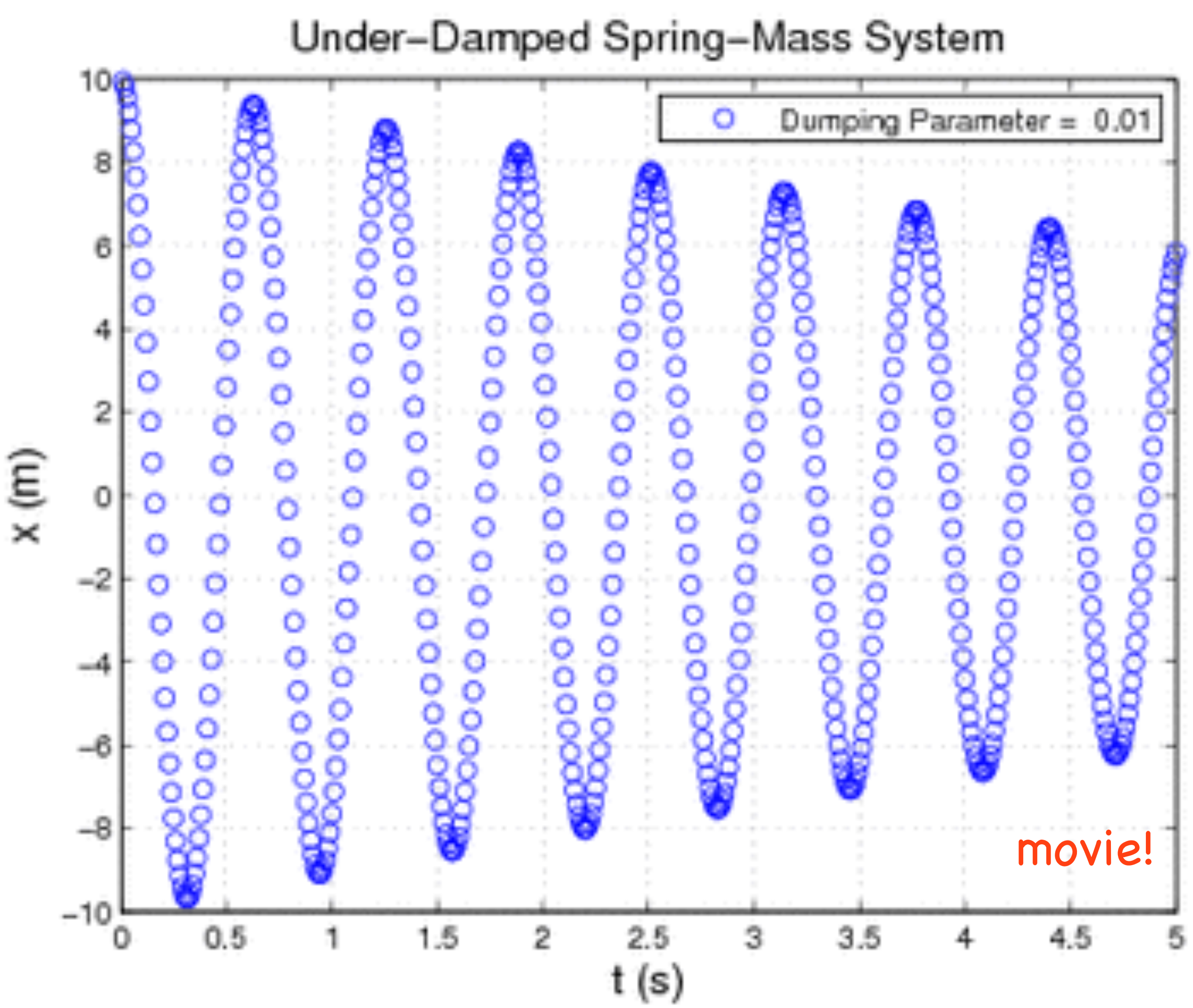
Projects

- Graphical User Interface: Building a calculator
- **Under-dumped string-mass system**
- Gaussian fit to a given data set (on command line and by using Fitting Toolbox)
- Quadrupole scan analysis for emittance measurement

Under-damped Spring-Mass System

Homework

- Write a program;
  - that calls the function "damped_oscillator" recursively for different z values,
  - and draws the x-t plots on the same figure.
  - Therefore, one could monitor the behavior of the system for different z values.



PART 2 - Hands-on Practice Session

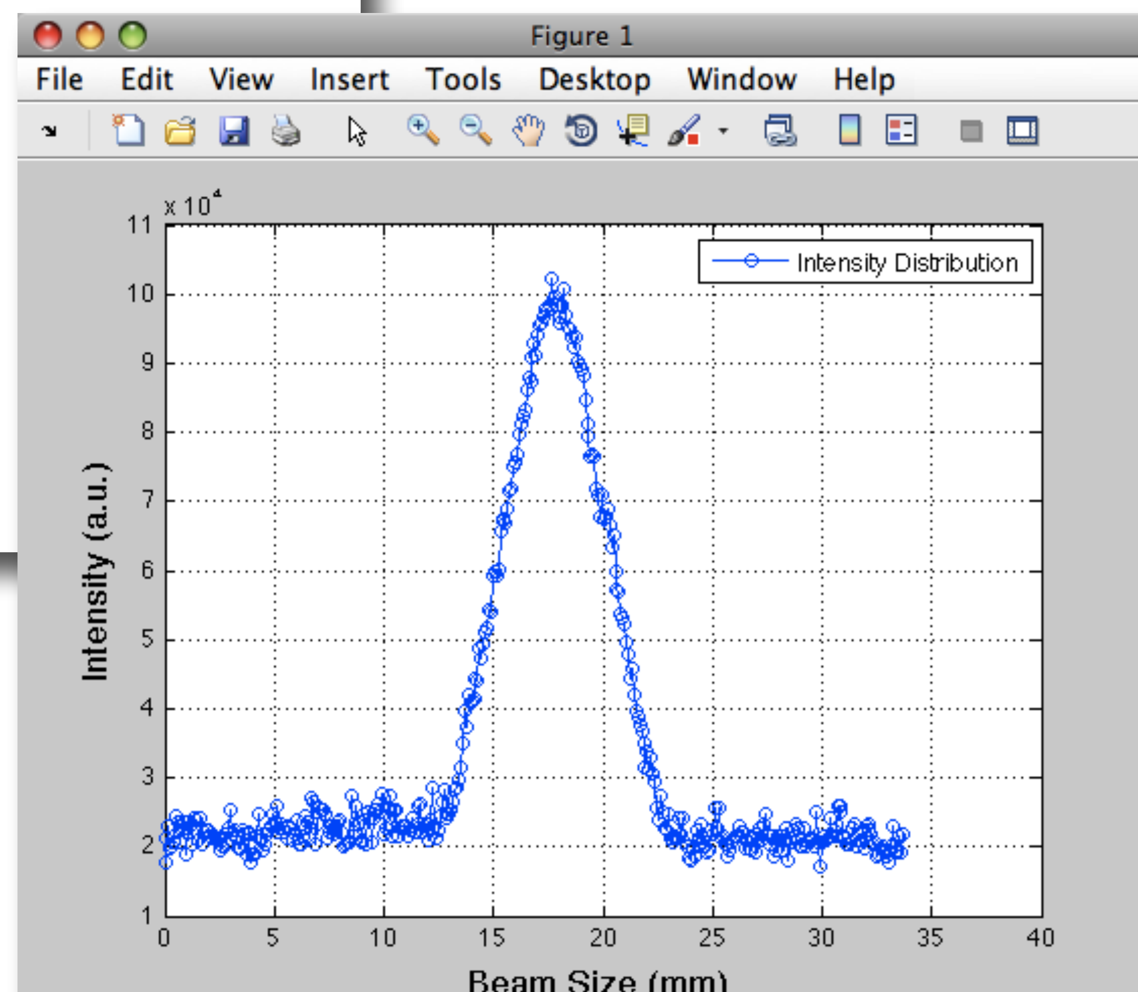
Projects

- Graphical User Interface: Building a calculator
- Under-dumped string-mass system
- **Gaussian fit to a given data set (on command line and by using Fitting Toolbox)**
- Quadrupole scan analysis for emittance measurement

Gaussian fit to a given data set by using MATLAB

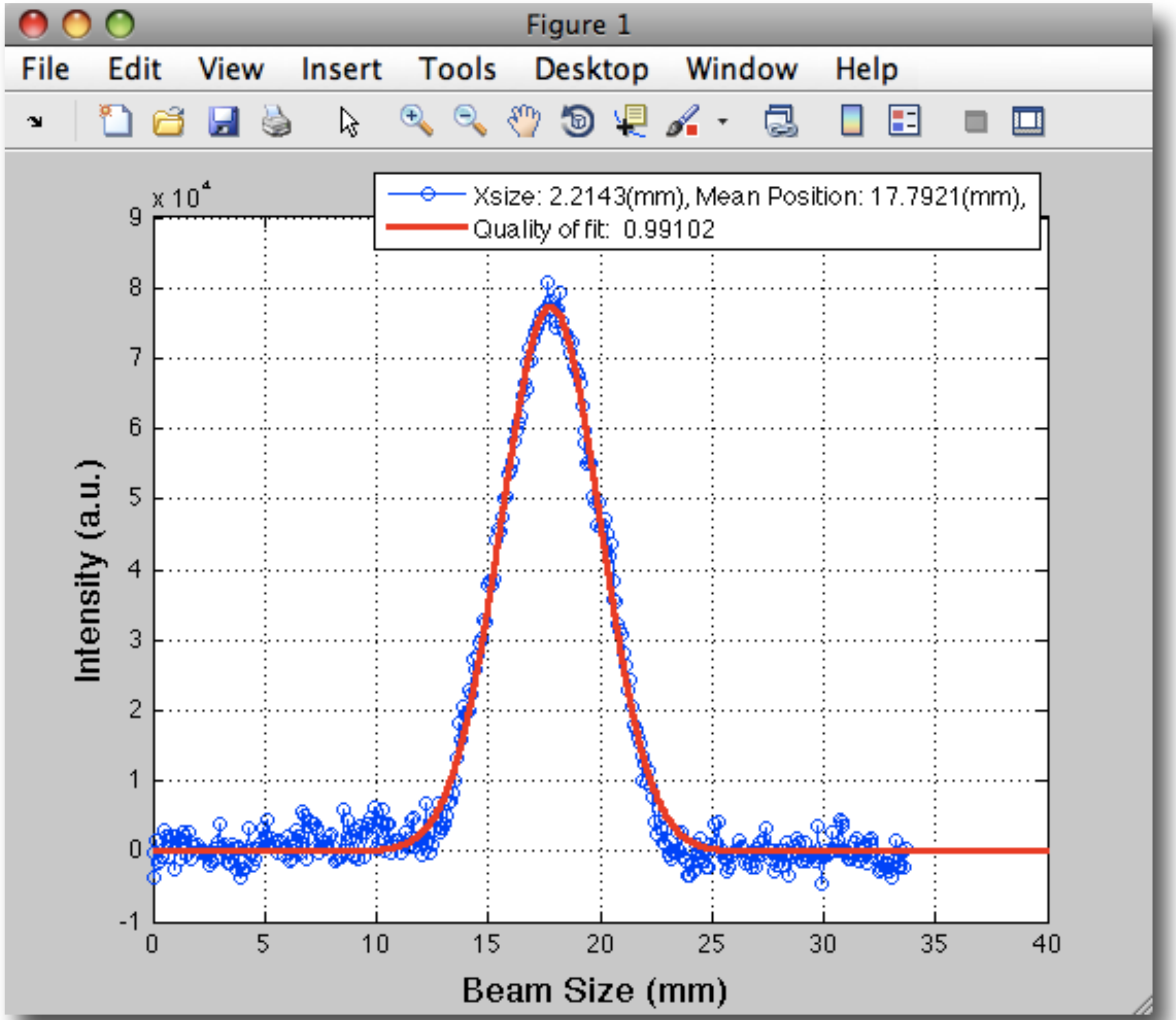
- Load a data set into the MATLAB workspace.
- Visualise the data set to be fit.

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%                Loading and Plotting a Data Set with MATLAB
%
%                HPFBU 2011 - MATLAB Tutorial
%
%                Help/Questions O.Mete
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

load('data_to_be_fit.mat');

figure(1)
plot(x_ax-x_ax(1),y_ax,'-ob');
xlabel('Beam Size (mm) ','fontsize',14);
ylabel('Intensity (a.u.)','fontsize',14);
legend('Intensity Distribution')
grid on;
xlim([0 40]);
```



Gaussian fit to a given data set by using MATLAB

- Load a data set into the MATLAB workspace.
- Visualise the data set to be fit.
- How is the "fit" built-in function used in MATLAB? Please search within the documentation.



- Determine the initial fit parameters for the fit.
- Find the background to be subtracted before the fit (in this case zeroth order polynomial).
- Fit the data to a Gaussian curve.
- Extract the fit parameters.
- Plot the data and the Gaussian fit curve on top of each other.
- Transform your fitting script into a MATLAB function. Use the x and y data as the function arguments. Function should return the mean and 1sigma of the distribution as well as the Chi^2 value.

Gaussian fit to a given data set by using MATLAB

- Load a data set into the MATLAB workspace.
- Visualize the data set to be fit.
- How is the "fit" built-in function used in MATLAB? Please search within the documentation.

```
%%%%%% Gauss fit     %%%%%%%%%%%%%%%%%%%%%

% Let's try to fit a polynomial background to the distribution.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
ff_=polyfit( datx(1:50),daty(1:50),0);


% Subtract the background from the data.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
daty = daty - ff_;

% Fit the noise-free data.
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[cf1_,gof1]=fit(datx(),daty(),'gauss1','Startpoint',[max(daty) mean0 sigma0],'MaxFunEvals',6000);

% Create the fit curve
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
for i=1:10000
nc1_(i)=(cf1_.a1)*exp(-((i/100-cf1_.b1)/cf1_.c1).^2);
axn1_(i)=i/100;
end

% Retrive fit parameters
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
xsize = (cf1_.c1)/sqrt(2);
posx = cf1_.b1;
gofx = gof1.rsquare;
```

Homework - Please implement and plot the data and fit model together.

PART III

EXTRAS

one step further

- How to make your plots visually more representable? :)
- How to increase computing speed in MATLAB?
- MATLAB toolboxes: plots, statistics, image processing, signal processing, neural network...
- Importing c++ codes
- Object-oriented programming
- GPU programming
- MATLAB - Simulink
- System optimisation
- ...