MATLAB Programming

(Lecture 7)

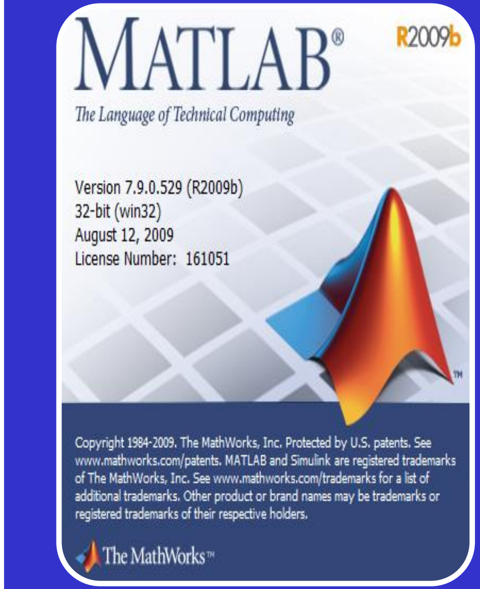
Dr. Sun Bing

School of EIE

Beihang University

Contents

Graphical User Interface



MATLAB® R2009b

The Language of Technical Computing

Version 7.9.0.529 (R2009b)
32-bit (win32)
August 12, 2009
License Number: 161051

Copyright 1984-2009. The MathWorks, Inc. Protected by U.S. patents. See www.mathworks.com/patents. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

The MathWorks™

- Introduction to GUI
- GUIDE
- GUIDE Templates
- Setting Properties for GUI Components
- Programming the GUI M-File
- Dialog Boxes
- Generate GUI by m code
- Demos

7.1 Introduction to Graphical User Interface

- **A graphical user interface provides the user with a familiar environment in which to work.**
- **This environment contains pushbuttons, toggle buttons, list box, edit box, text boxes, slider, and so forth, all of which are already familiar to the user, so that he or she can concentrate on using the application rather than on the mechanics involved in doing things.**

7.1 Introduction to Graphical User Interface

- **GUIs are harder for the programmer because a GUI-based program must be prepared for mouse clicks for any GUI element at any time. Such inputs are known as events, and a program that responds to events is said to be *event driven.***
- **The three principal elements required to create a MATLAB GUI are:**
  - Components
  - Figures
  - Callbacks

7.1 Introduction to Graphical User Interface

(1) Components.

- **Each item on a MATLAB GUI (pushbuttons, labels, edit boxes, etc.) is a graphical component. The types of components include: graphical controls (pushbuttons, edit boxes, list box, sliders, etc.), static elements (frames and text strings), menus, and axes.**
- **Graphical controls and static elements are created by the function uicontrol, and menus are created by the functions uimenu and uicontextmenu.**
- **Axes, which are used to display graphical data, are created by the function axes.**

7.1 Introduction to Graphical User Interface

(2) Figures

- The components of a GUI must be arranged within a figure, which is a window on the computer screen.
- Empty figures can be created with the function *figure* and can be used to hold any combination of components.

7.1 Introduction to Graphical User Interface

(3) Callbacks

- There must be some way to perform an action if a user clicks a mouse on a button. A mouse click is an event, and the MATLAB program must respond to each event if the program is to perform its function.
- For example, if a user clicks on a button, that event must cause the MATLAB code that implements the function of the button to be executed.
- The code executed in response to an event is known as a call back.

7.2 GUIDE

- **MATLAB GUIs are created using a tool called guide, the GUI Development Environment. This *tool* allows a programmer to layout the GUI, selecting and aligning the GUI components to be placed in it.**
- **Once the components are in place, the programmer can edit their properties: name, color, size, font, text to display, and so forth.**
- **When guide saves the GUI, it creates working program including skeleton functions that the programmer can modify to implement the behavior of the GUI.**

7.2 GUIDE

The basic steps required to create a MATLAB GUI

- 1. Decide what elements are required for the GUI and what the function of each element will be.
- 2. Use a MATLAB tool called guide (GUI Development Environment) to layout the Components on a figure.
- 3. Use a MATLAB tool called the Property Inspector (built into guide) to give each component a name (a "tag") and to set the characteristics of each component, such as its color, the text it displays, and so on.

7.2 GUIDE

- **4. Save the figure to a file. When the figure is saved, two files will be created on disk with the same name but different extents. The fig file contains the actual GUI that you have created, and the M-file contains the code to load the figure and skeleton call backs for each GUI element.**
- **5. Write code to implement the behavior associated with each callback function.**

7.2 GUIDE

- **GUIDE, the MATLAB® Graphical User Interface development environment, provides a set of tools for creating graphical user interfaces (GUIs). These tools greatly simplify the process of designing and building GUIs. You can use the GUIDE tools to Lay out the GUI.**
- **Using the GUIDE Layout Editor, you can lay out a GUI easily by clicking and dragging GUI components -- such as panels, buttons, text boxes, sliders, Pop up menus, and so on -- into the layout area.**

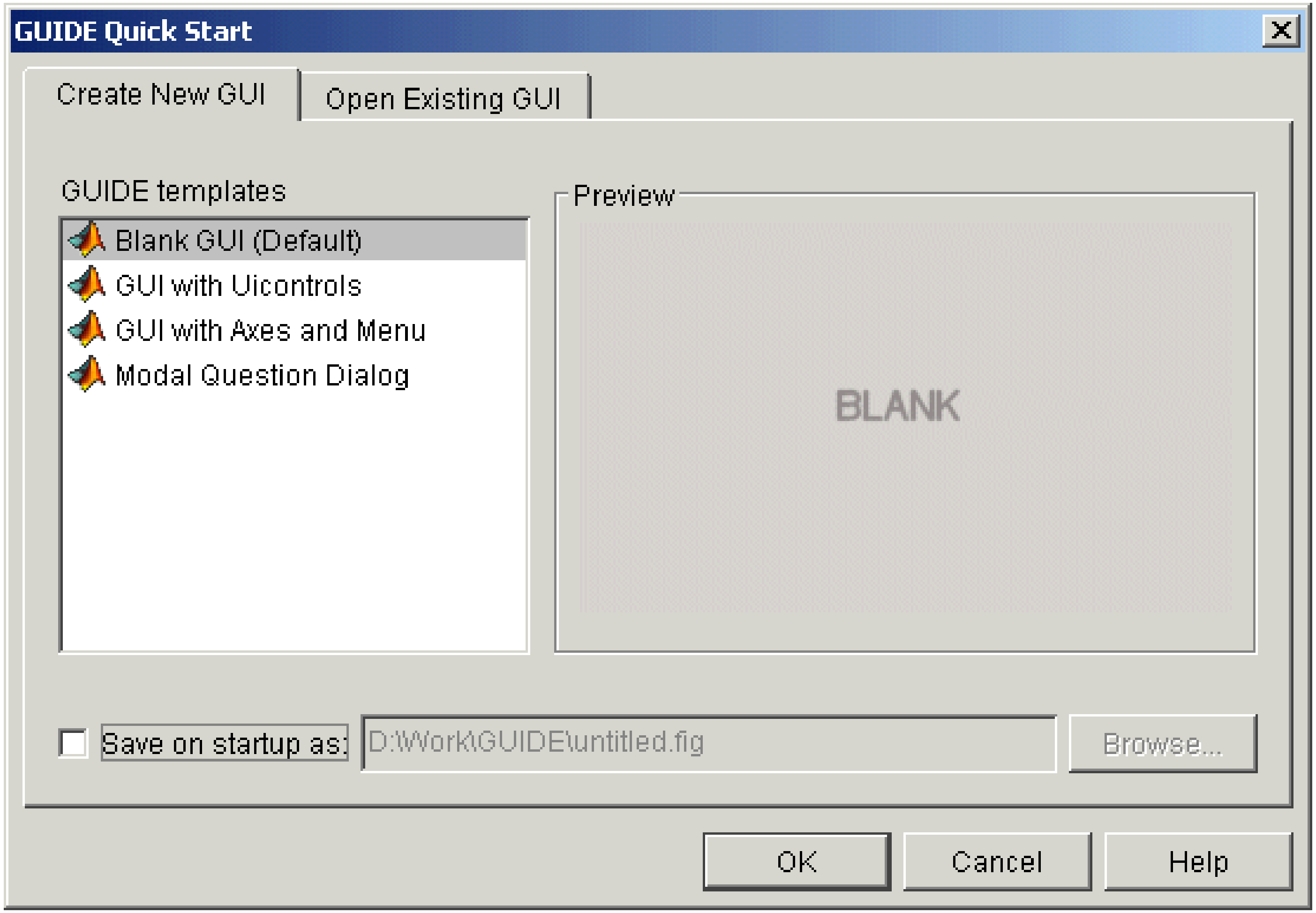
7.2 GUIDE

- **GUIDE automatically generates an M-file that controls how the GUI operates. The M-file initializes the GUI and contains a framework for all the GUI callbacks -- the commands that are executed when a user clicks a GUI component(or called controls). Using the M-file editor, you can add code to the callbacks to perform the functions you want them to.**

7.2.1 Starting GUIDE

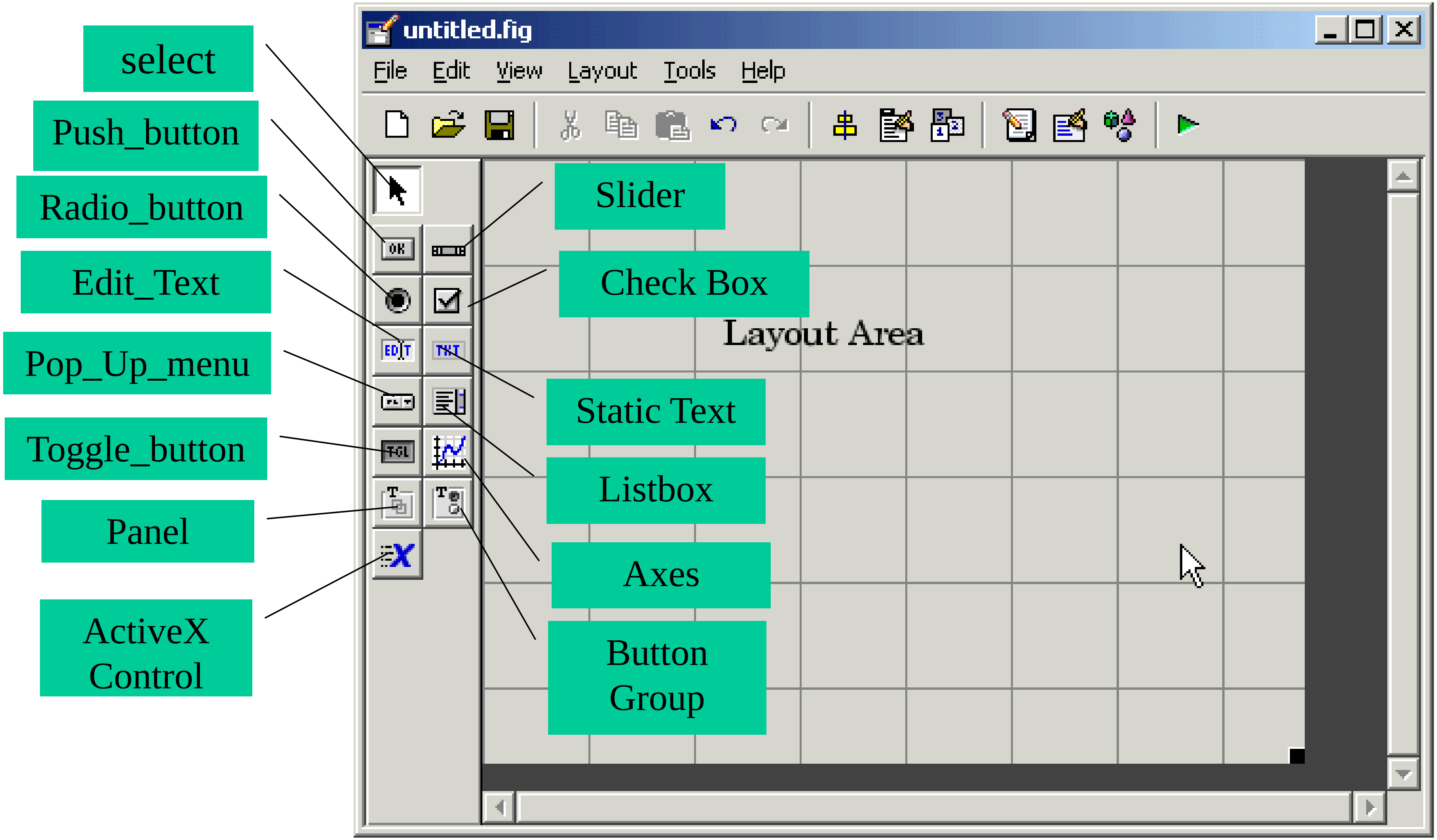
- **To start GUIDE, enter guide at the MATLAB prompt. This displays the GUIDE Quick Start dialog, as shown in the following figure.**
- **From the Quick Start dialog, you can Create a new GUI from one of the GUIDE templates -- prebuilt GUIs that you can modify for your own purposes. Open an existing GUI.**

7.2.1 Starting GUIDE



□ Save on startup as: D:\Work\GUIDE\untitled.fig

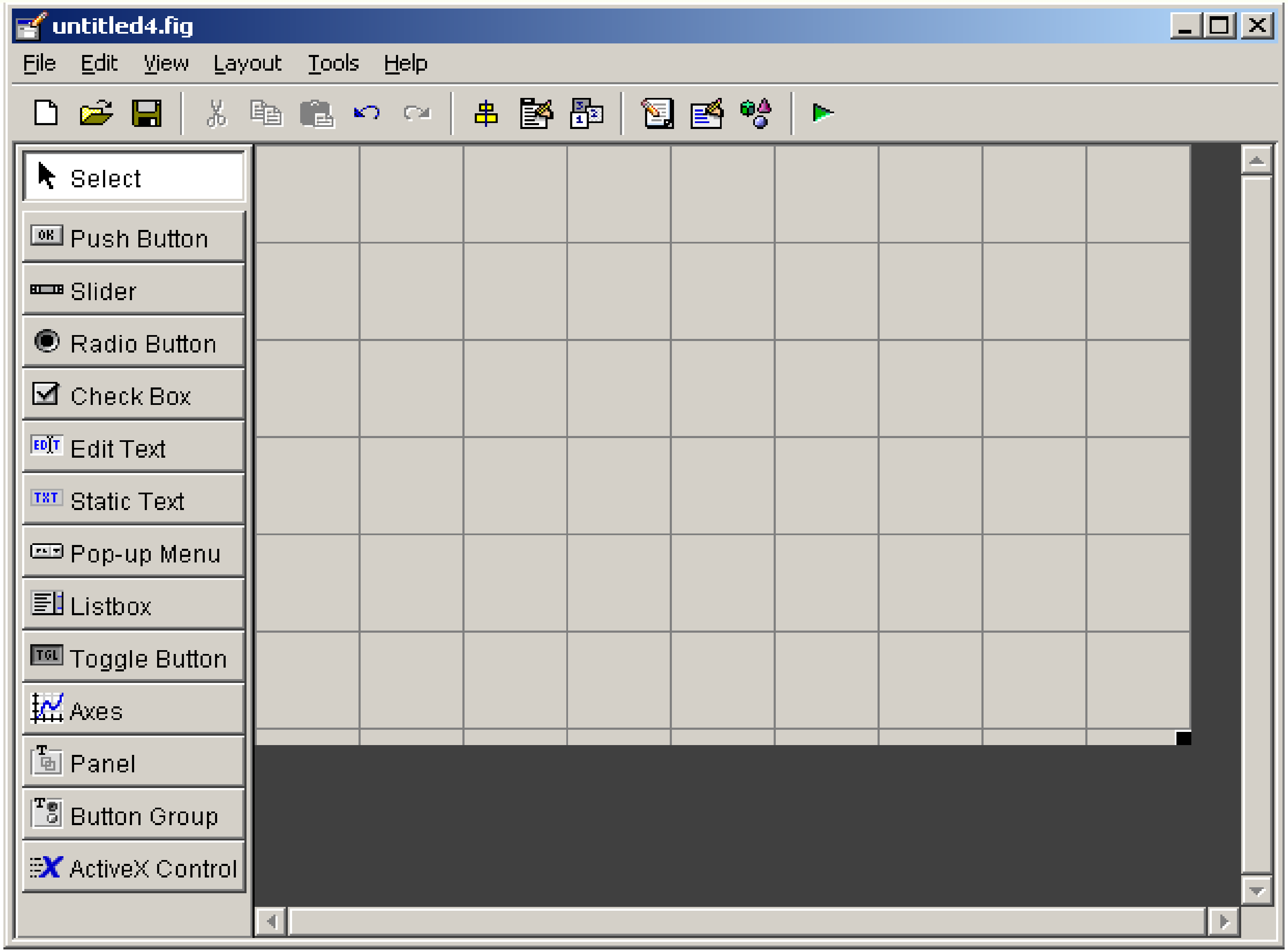
7.2.2 The Layout Editor



7.2.3 Display the component name

- **To display the names of the GUI components in the component palette, click ‘File’→ ‘Preferences’ selection, check the box next to Show names in component palette, and click OK.**
- **The Layout Editor shows the component name as in the following slide.**

7.2.3 The Layout Editor



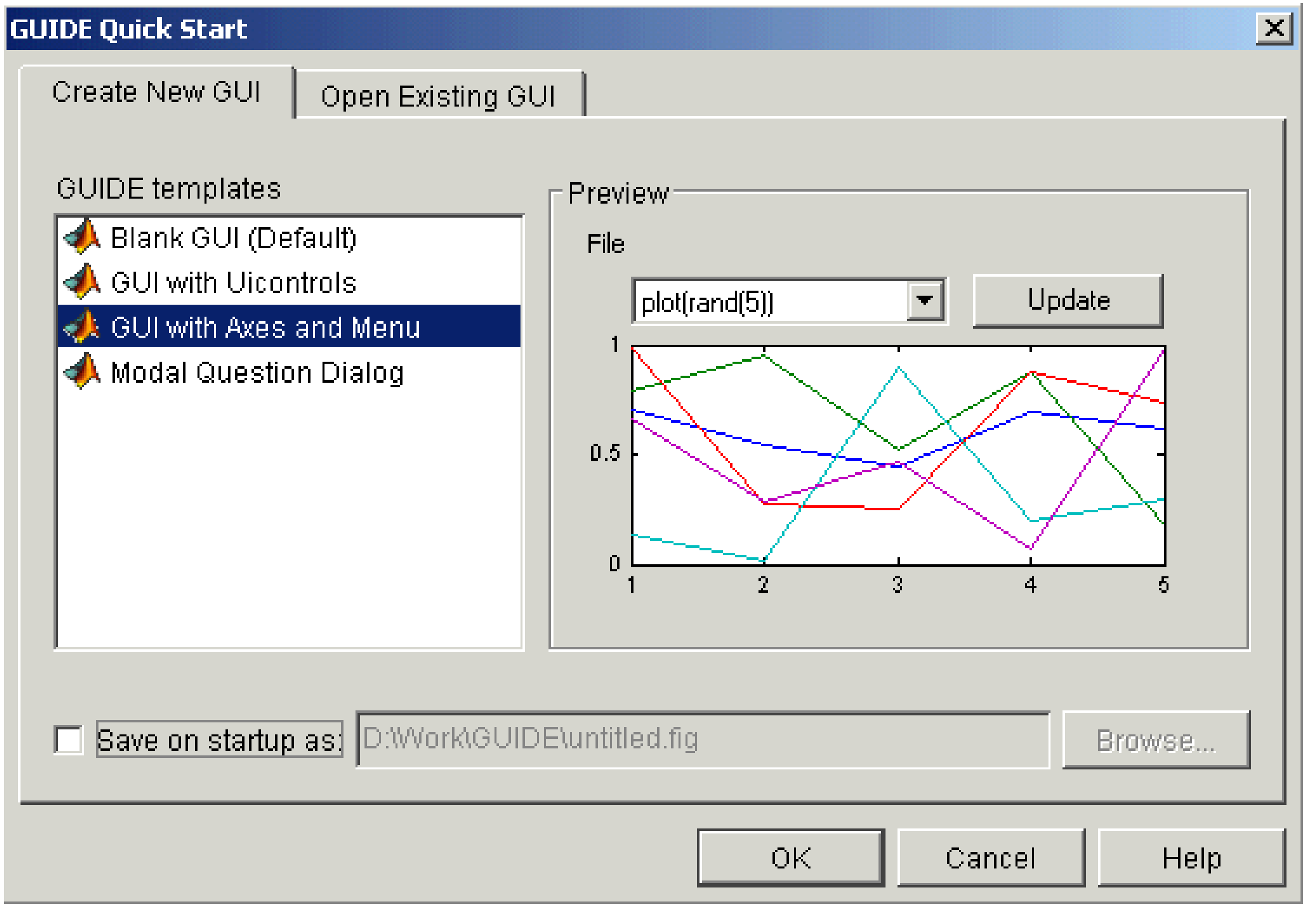
7.2.4 Using Layout Editor

- **You can lay out your GUI by dragging components, such as push buttons, pop-up menus, or axes, from the component palette, at the left side of the Layout Editor, into the layout area.**

7.3 GUIDE Templates

- **The GUIDE Quick Start dialog provides templates for several basic types of GUIs. The advantage of using templates is that often you can modify a template more quickly and easily than by starting from a blank GUI.**
- **For example, when you select the GUI with Axes and Menu, the Quick Start dialog appears as in the following figure.**

7.3 GUIDE Templates



GUIDE Quick Start

GUIDE templates

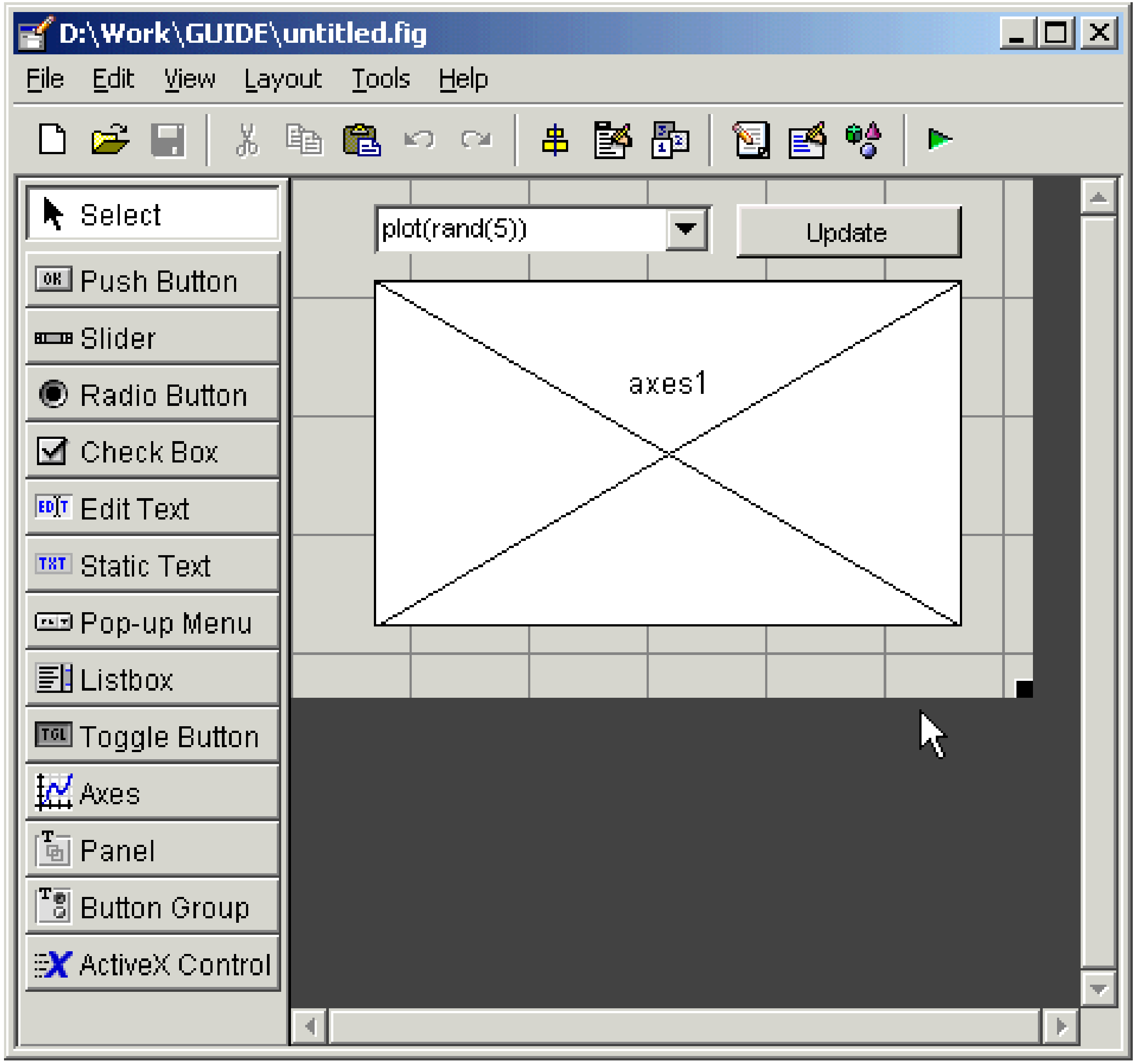
- Blank GUI (Default)
- GUI with Uicontrols
- GUI with Axes and Menu
- Modal Question Dialog

Preview

File: plot(rand(5))

□ Save on startup as: D:\Work\GUIDE\untitled.fig

7.3.1 Create GUIDE Templates



7.3.2 Running the GUI Templates

- **To run a GUI, select Run from the Tools menu, or click the run button ▶ on the toolbar. This displays the functioning GUI outside the Layout Editor.**

7.3.3 GUI FIG-Files and M-Files

- **GUIDE stores a GUI in two files, which are generated the first time you save or run the GUI: A FIG-file, with extension .fig, which contains a complete description of the GUI layout and the components of the GUI: pushbuttons, menus, axes, and so on. An M-file, with extension .m, that contains the code that controls the GUI, including the callbacks for its components.**
- **These two files correspond to the tasks of laying out and programming the GUI. When you lay out of the GUI in the Layout Editor, your work is stored in the FIG-file. When you program the GUI, your work is stored in the M-file.**

Summary : The steps of creating GUI

- **1. Open a New GUI in the Layout Editor**
- **2. Set the GUI Figure Size**
- **3. Add the Components(or controls) and Align the Components**
- **4. Setting Properties for GUI Components**
- **5. Programming the GUI**
- **6. Run and test the GUI**

7.4 Setting Properties for GUI Components

- To set the properties of each GUI component, select the Property Inspector from the View menu to display the Property Inspector dialog box. Or click the icon on the Layout tool bar.
- When you select a component in the Layout Editor, the Property Inspector displays that component's properties that you can change or set.

7.4 Setting Properties for GUI Components

- **1. The Tag Property: The Tag property provides a string as a unique identifier for each component. GUIDE uses this identifier to construct unique callback names for the different components in the GUI.**
- **2. String Property for Push Buttons and Static Text: You can set the label in some user interface controls, such as push buttons, by using the String property.**

7.4 Setting Properties for GUI Components

- **3. String Property for Pop-Up Menus : A pop-up menu's String property controls the list of menu items. To set the pop-up menu items, select the pop-up menu in the Layout Editor. In the Property Inspector, click the icon ☰ next to String. This opens the String property edit box. Delete 'Pop-up Menu 'in the String property edit box, and type the selections each on the separate lines.**

7.5 Programming the GUI M-File

- **After laying out your GUI, and setting component properties, the next step is to program it.**
- **To open the M-file, click the M-file Editor icon 📝 on the Layout Editor toolbar.**
- **you can program the GUI M-file using the M-file editor.**
- **GUIDE automatically generates this file from your layout the first time you save or run the GUI.**

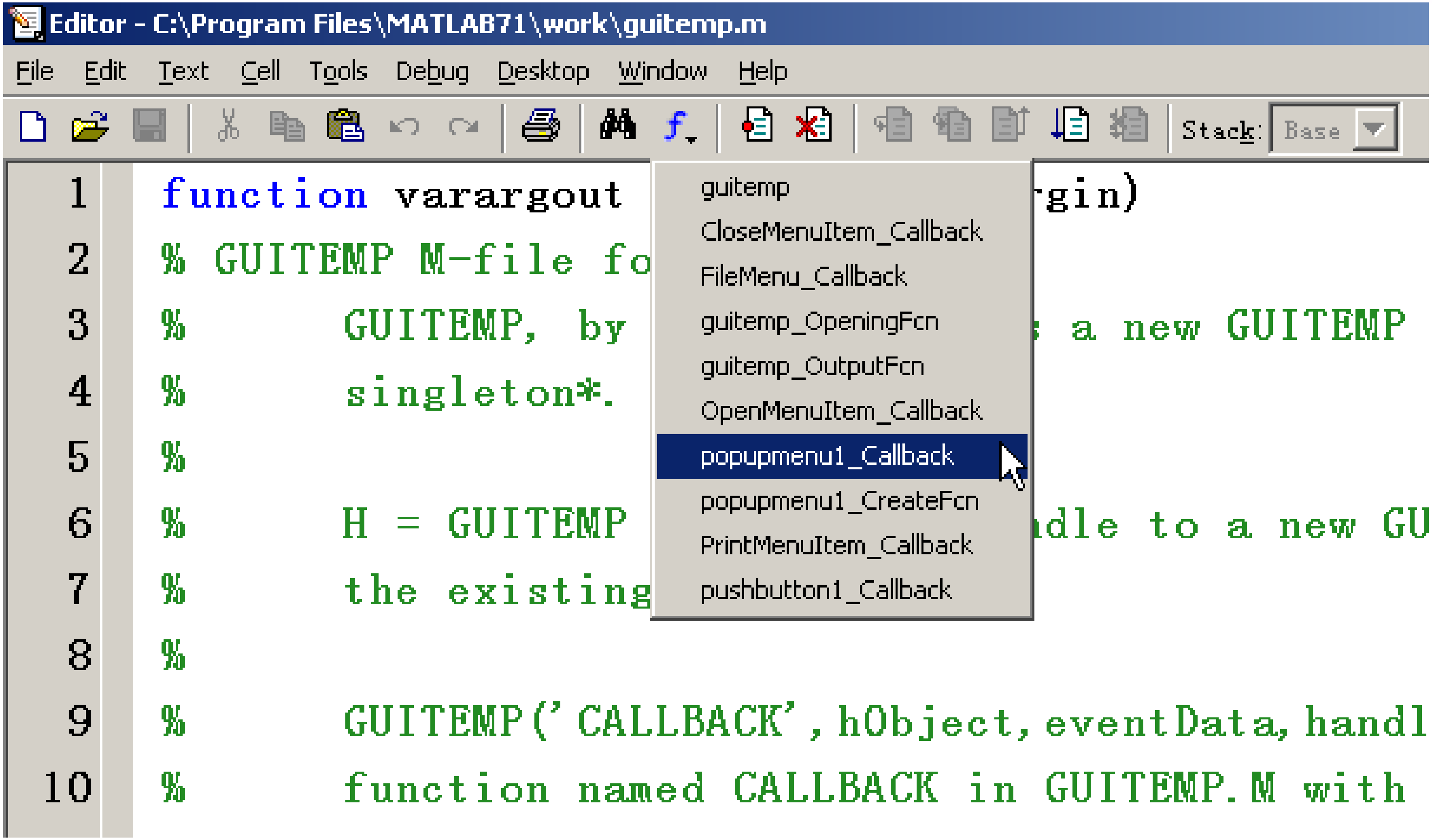
7.5.1 The GUI M-File

- **The GUI M-file**
  - (1) Initializes the GUI ,
  - (2) Contains code to perform tasks before the GUI appears on the screen, such as creating data or graphics
  - (3) Contains the callback functions that are executed each time a user clicks a GUI component
- **Initially, each callback contains just a function definition line. You then use the M-file editor to add code that makes the component function the way you want it to.**

7.5.1 The GUI M-File

- **You can view the callback for any of the GUI components by clicking the function icon *f* on the toolbar. This displays a list of all the callbacks, as shown in the following figure.**
- **Clicking a callback on the list displays the section of the M-file containing the selected callback, where you can edit it.**

7.5.1 The GUI M-File



Editor - C:\Program Files\MATLAB71\work\guitemp.m

```
function varargout                     rgin)
% GUITEMP M-file fo
%       GUITEMP, by                    a new GUITEMP
%       singleton*.
%
%       H = GUITEMP                    dle to a new GU
%       the existing
%
%       GUITEMP('CALLBACK',hObject,eventData,handl
%       function named CALLBACK in GUITEMP.M with
```

7.5.2 Sharing Data Between Callbacks

- **You can share data between callbacks by storing the data in the MATLAB *handles* structure. All components in a GUI share the same handles structure. It is passed as an input argument to all callbacks generated by GUIDE.**

7.5.2 Sharing Data Between Callbacks

- **For example, to store data contained in vector X in the handles structure, you Choose a name for the field of the handles structure where you want to store the data, for example, handles.my_data Add the field to the handles structure and set it equal to X with the following statement:**

```
handles.my_data = X;
```

- **Save the handles structure with the *guidata* function:**

```
guidata(hObject,handles)
```

- **Here, hObject is the handle to the component object that executes the callback. The component's object handle is passed as the input argument, hObject, to each of its callbacks that is generated by GUIDE.**

7.5.2 Sharing Data Between Callbacks

- **To retrieve X in another callback, use the command**

  X = handles.my_data;

- **You can access the data in the handles structure in any callback because hObject and handles are input arguments for all the callbacks generated by GUIDE.**

7.5.3 Adding Code to the Opening Function

The Opening Function

- **The opening function is the first callback in every GUI M-file. For Initializing the GUI**
- **You can use it to perform tasks that need to be done before the user has access to the GUI, for example, to create data or to read data from an external source.The code in the opening function is executed just before the GUI is made visible to the user, but after all the components have been created.**

7.5.4 The callback frame

1. Toggle Button

Get the toggle button information;

```
 state = get(handles.togglebutton1,’value’);
 if state == 1
    % toggle button is pressed take appropriate action.
else
    % toggle button is not pressed. Take another action.
end
```

2. The Radio Button is same as above.

7.5.4 The callback frame

3.Check Boxes

Get the check box information:

```
check_state = get(hobject,’value’);
if check_state ==get (hobject,’max’)
    % The check box is clicked, take appropriate action.
else
    % The check box is not clicked. Take another action.
end
```

7.5.4 The callback frame

4. The Edit text

(1) Get the string in the edit text box

```
str = get(handles.edit1, ‘string’);
```

(2)

```
user_val =str2double(get(handles.edit1, ‘string’));
 if isnan(user_val)
   error(‘invalid value’,’Bad Input’,’modal’);
end
% put the action codes here.
```

7.5.4 The callback frame

5. Sliders

- **In other callback function get the slider val:**

```
slider_val = get(handles.slider1, ‘value’);
```

- **6. Pop_Up menu**
- **In other callback function get the pop_up index:**

```
popup_index = get(handles.popupmenu1, 'Value');
 switch popup_index
        case 1
            % Put code here;
        case 2
             %put code here
  end
```

7.6 Dialog Boxes

- **A dialog box is a special type of figure that is uesd to display information or to get input from a user.**
- **Dialog boxes may be modal or non-modal. A modal dialog box does not allow any other window in the application to be accessed until it is dismissed, whereas a non-modal dialog box does not block access to other windows.**
- **A modal dialog box is typically used for warning and error messages.**

7.6 Dialog Boxes

In the list there are some most useful dialog boxes.

- **Dialog creates a generic dialog box.**
- **Errordlg displays an error messages in a dialog box.**
- **Inputdlg display a request for input data and accepts the user’s input values.**
- **Listdlg allows user to make one or more selections from a list.**
- **Printdlg displays a printer selection dialog box.**
- **Questdlg Asks a question. This dialog box can contain either two or three button, which by default are labeled Yes,No, and Cancel.**

7.6.1 Error and Warning Dialog Boxes

- **errordlg(error_string,box_title,create_mode);**

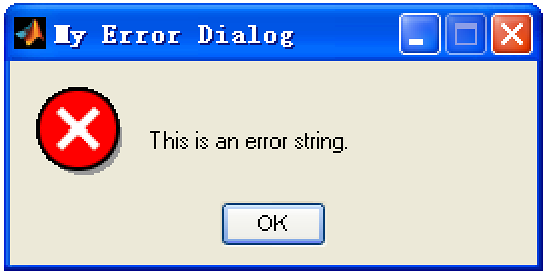
  warndlg(warning_string,box_title,create_mode);

error_string or warning_string is the message to display to the user.

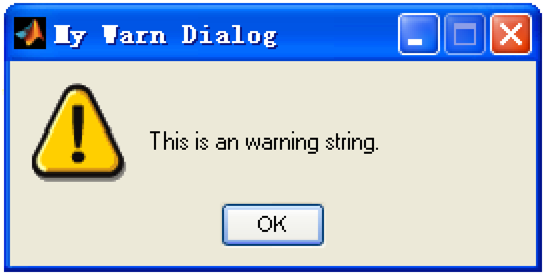
box_title is the title of the dialog box.

create_mode is a string that can be 'modal' or 'nonmodal'

```
% errordlg  & warndlg example
errordlg('This is an error string.', 'My Error Dialog', 'modal');
warndlg('This is an warning string.', 'My Warn Dialog', 'modal');
```



This is an error string.



This is an warning string.

7.6.2 Input dialog boxes

```
ans = inputdlg(prompt)
ans = inputdlg(prompt,title)
ans = inputdlg(prompt,title,line-no)
ans = inputdlg(prompt,title,line-no,default_ans)
```

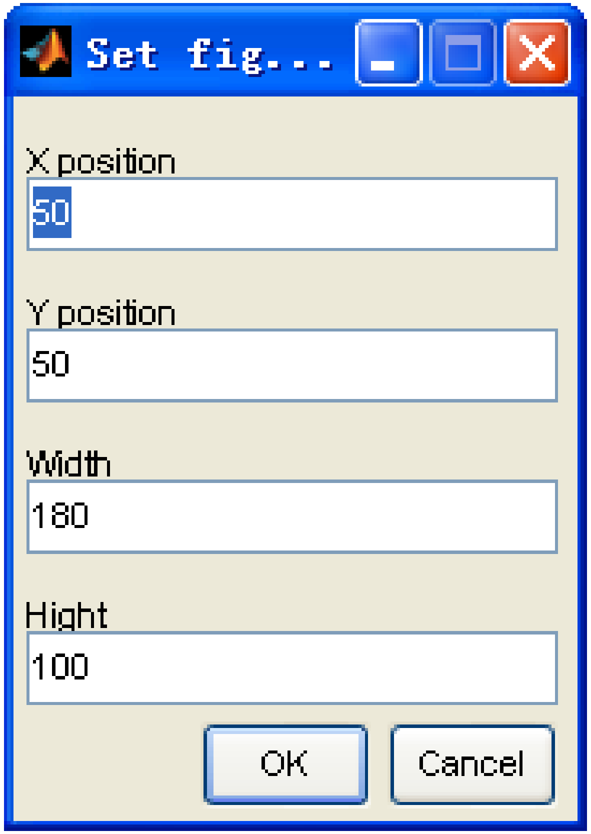
7.6.2 Input dialog boxes

- **Prompt : a cell array of string, with each element of the array corresponding to one value that the user will be asked to enter.**
- **Title : The title of the box.**
- **line-no : specifies the number of lines to be allowed for each answer.**
- **default_ans : a cell array containning the default answers.**

7.6.2 Input dialog boxes

- **For example’**

```
prompt{1} = 'X position';
prompt{2} = 'Y position';
prompt{3} = 'Width';
prompt{4} = 'Hight';
title = 'Set figure Position';
default = {'50','50','180','100'}
ans = inputdlg(prompt,title,1,default);
```



X position

50

Y position

50

Width

180

Hight

100

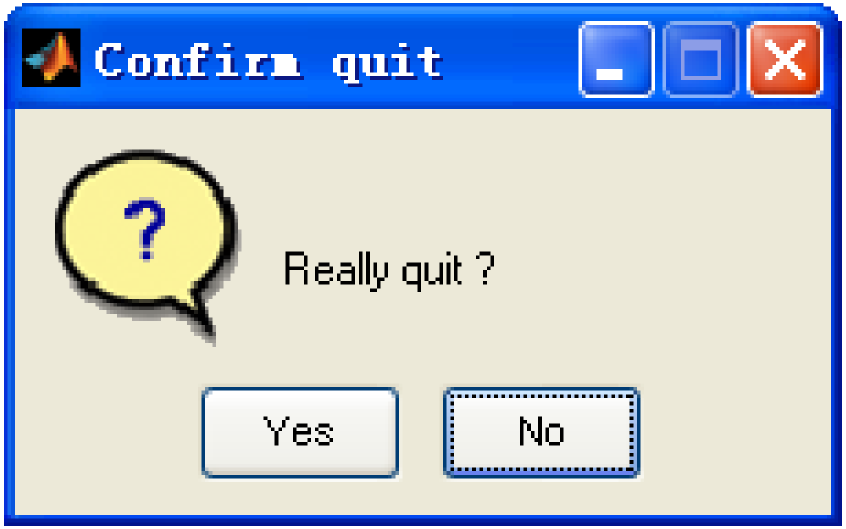
7.6.3 questdlg

- **questdlg() Asks a question.**
- **button=questdlg('qstring','title','str1','str2','default') creates a question dialog box with two push buttons labeled 'str1' and 'str2'. 'default' specifies the default button selection and must be 'str1' or 'str2'.**

7.6.3 questdlg

- **For example:**

```
a = questdlg('Really quit ?', 'Confirm quit ', 'Yes', 'No','No');
if strcmp(a,'Yes')
   delete(gcbf);
end
```

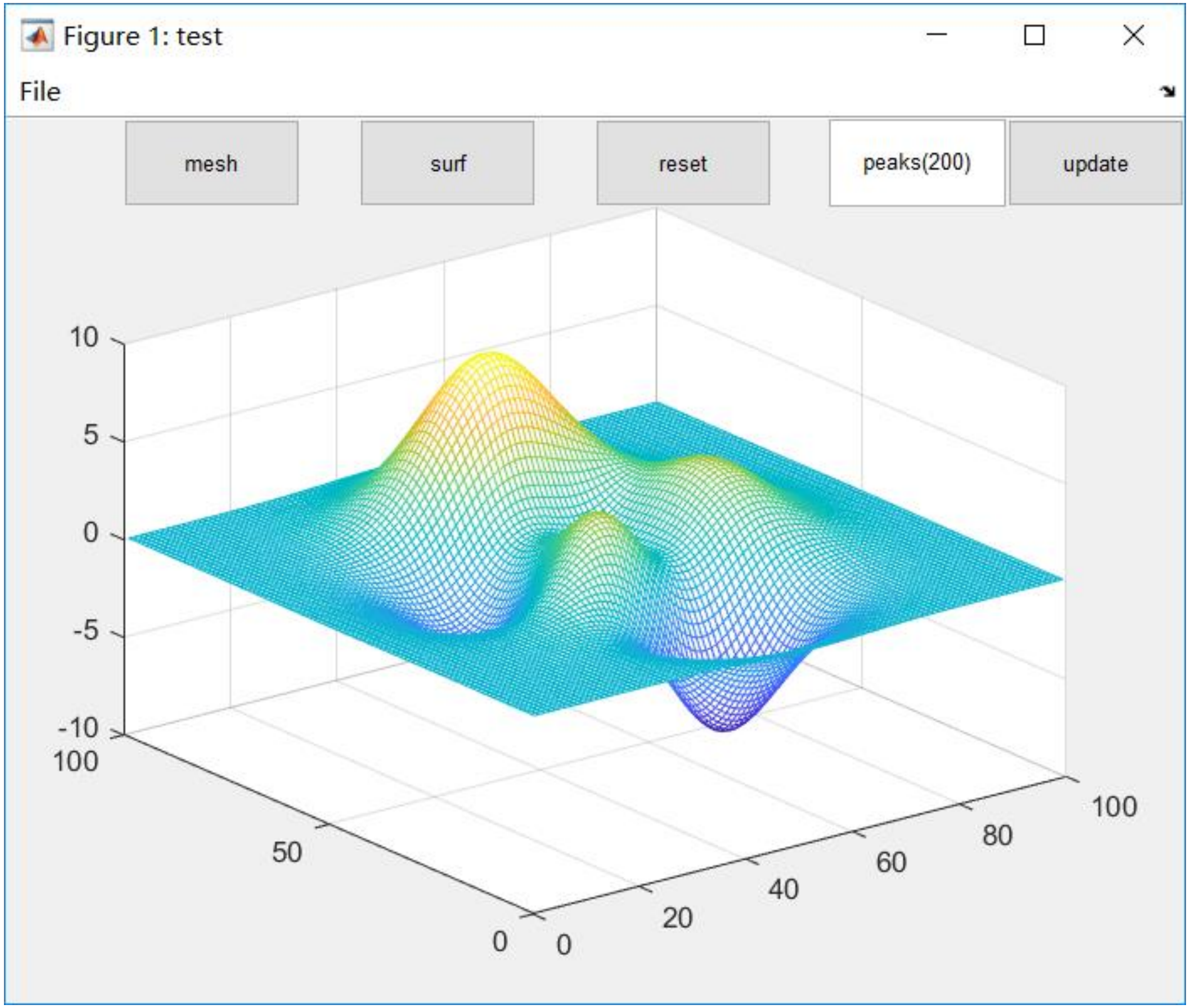


Really quit ?

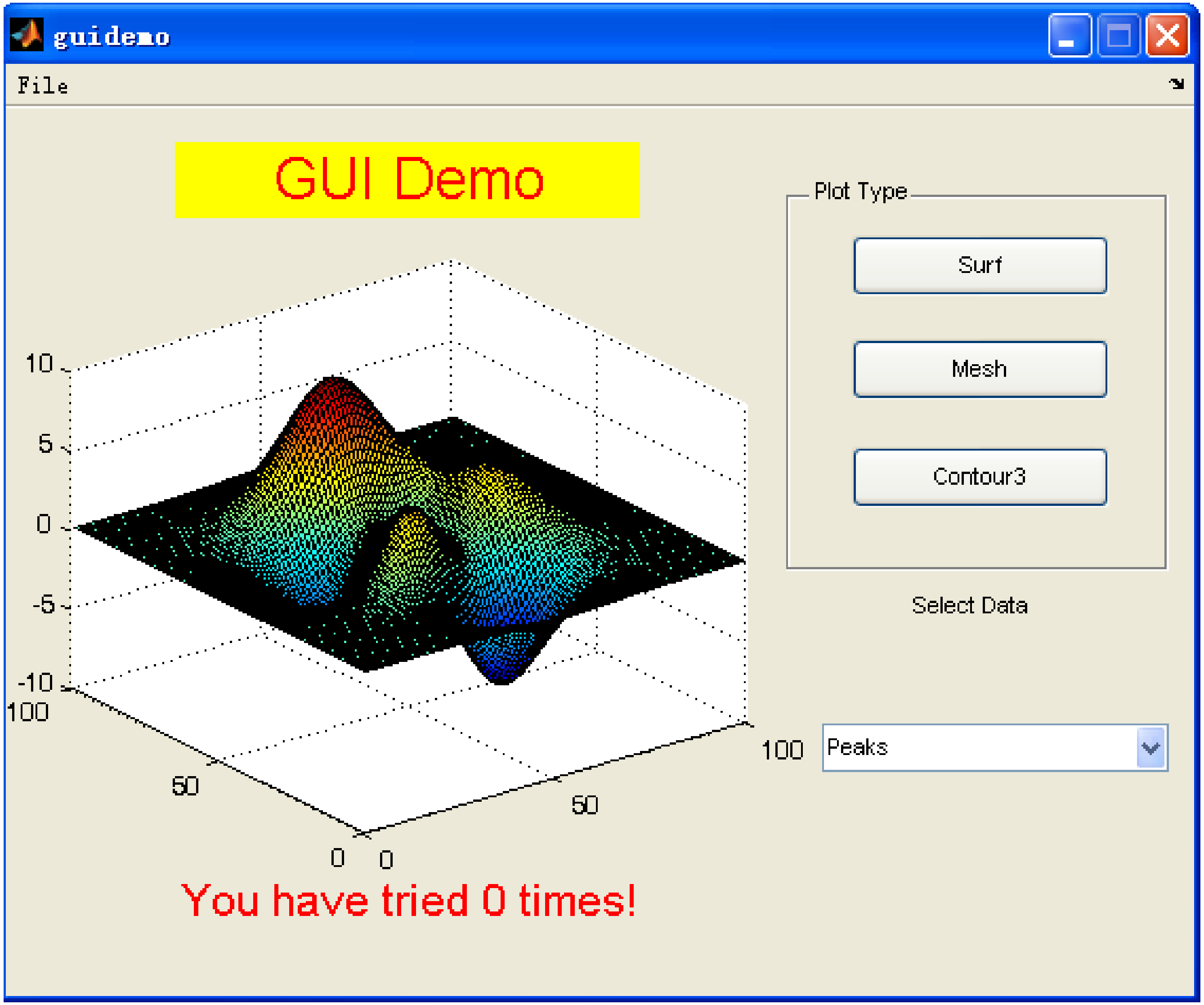
7.7 Generate GUI by m code

- **See test.m**

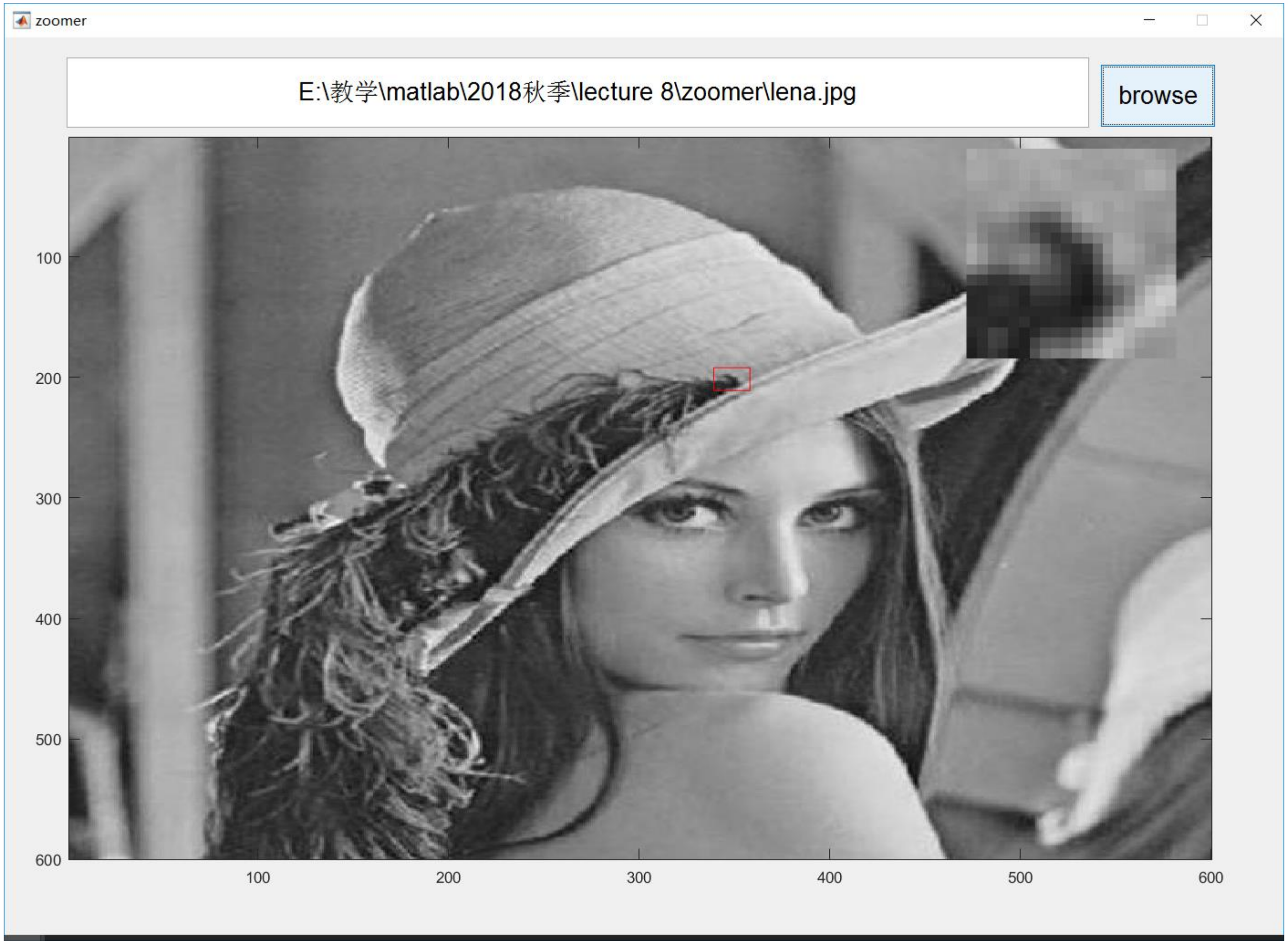
- ➢ figure
- ➢ axes
- ➢ uicontrol
- ➢ uimenu
- ➢ uicontextmenu
- ➢ set
- ➢ get
- ➢ guidata



GUI demo1

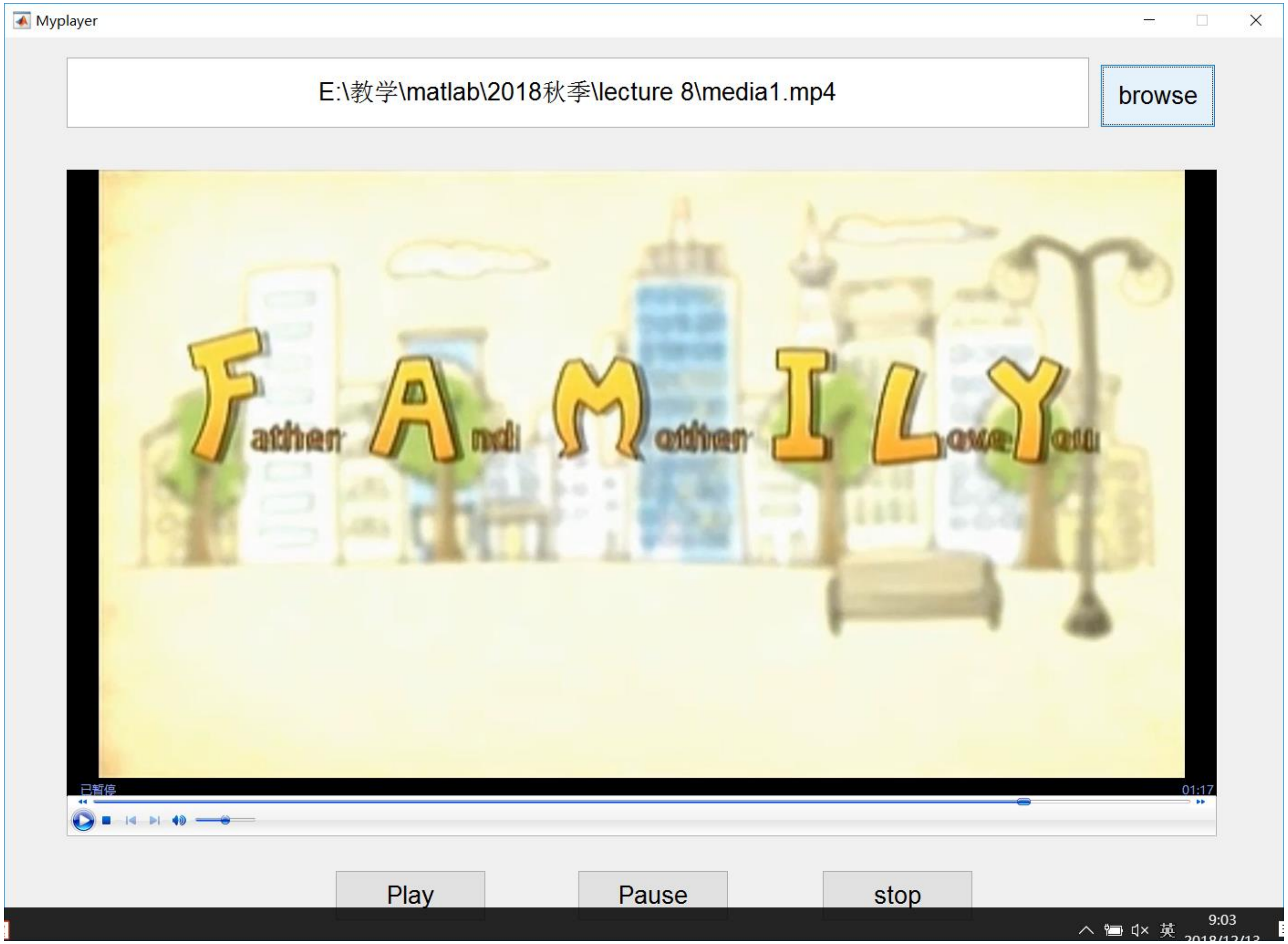

guidemo
File
GUI Demo
Plot Type
Surf
Mesh
Contour3
Select Data
Peaks
10
5
0
-5
-10
100
50
0
0
50
100
You have tried 0 times!

GUI demo2



zoomer
E:\教学\matlab\2018秋季\lecture 8\zoomer\lena.jpg
browse
100
200
300
400
500
600
100
200
300
400
500
600

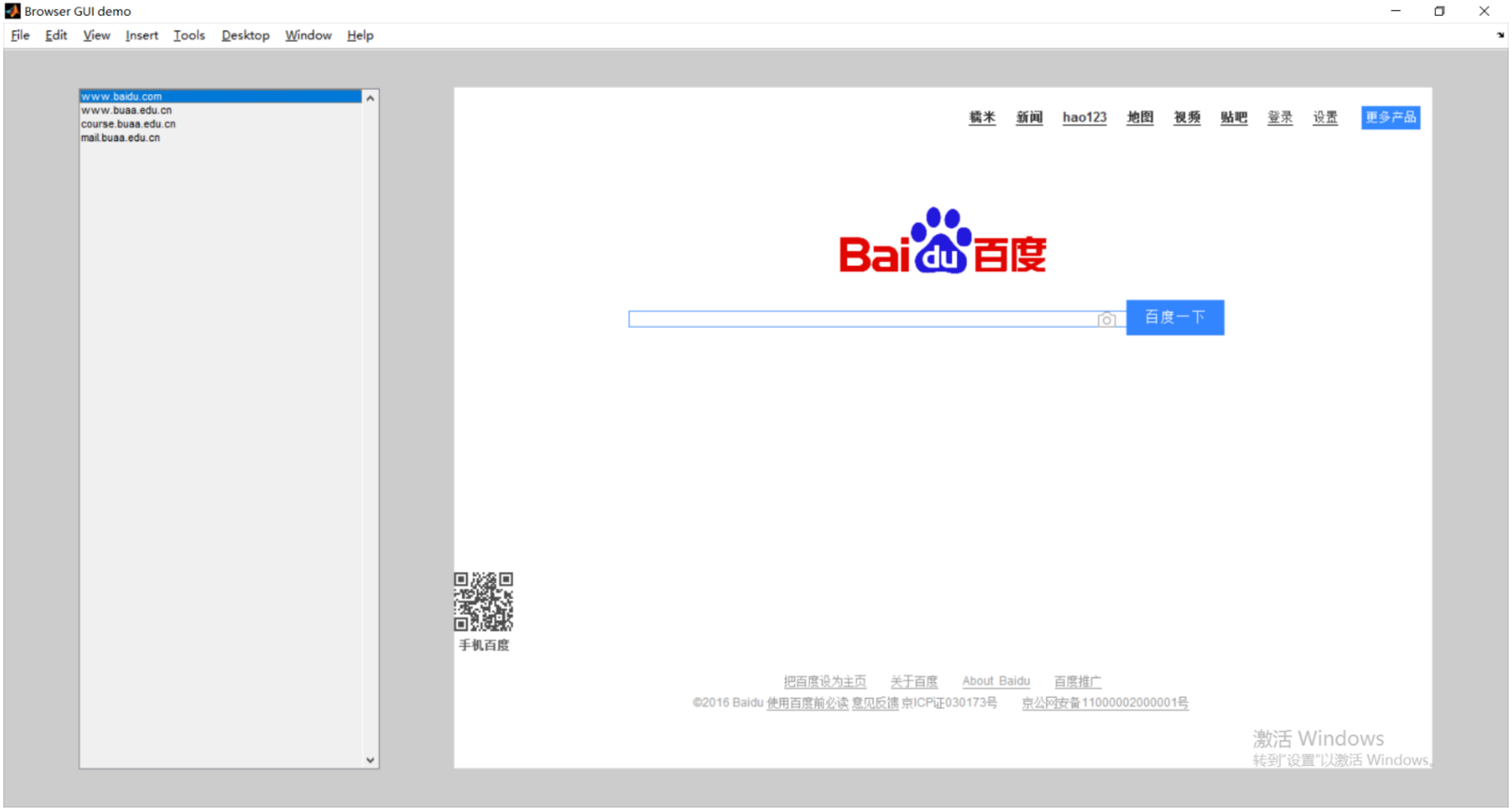
GUI demo3



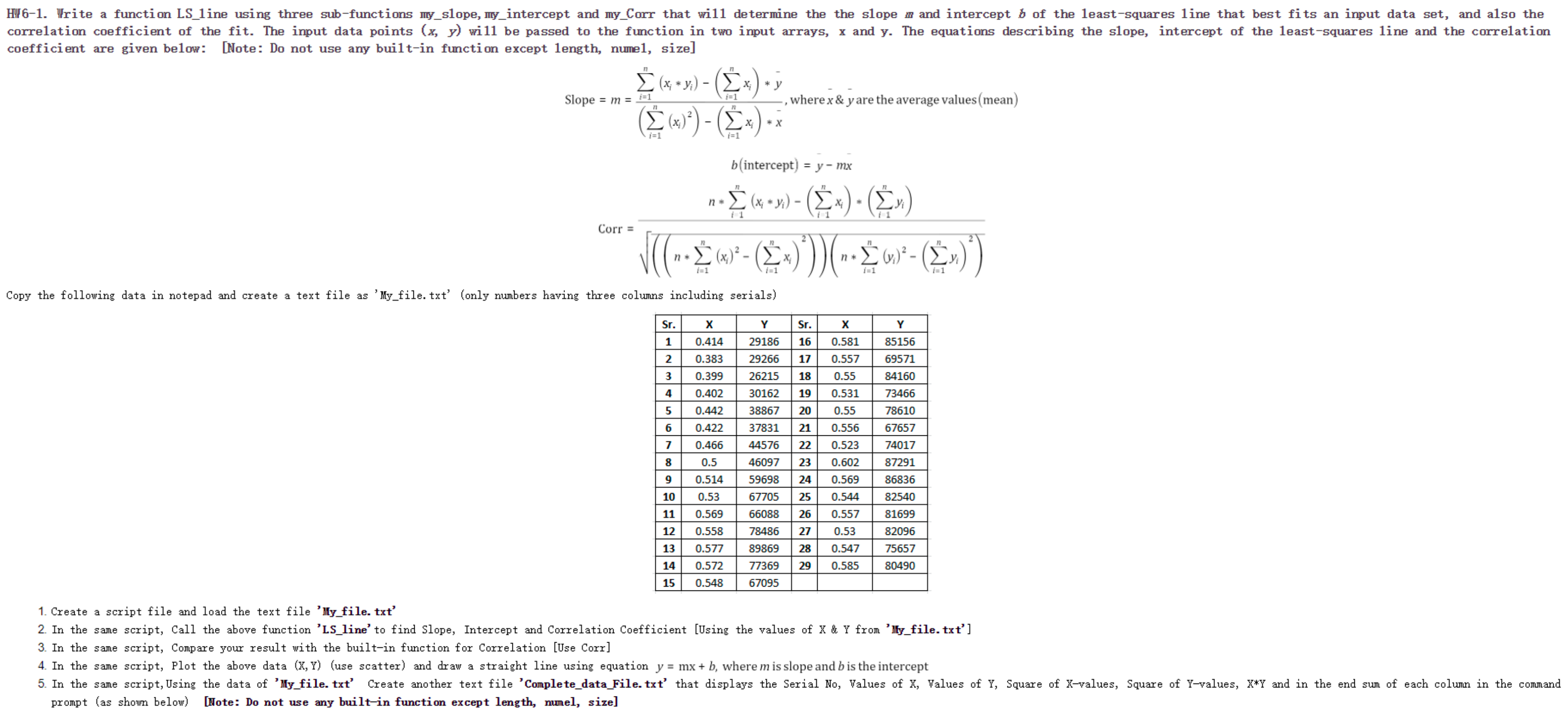
Myplayer

E:\教学\matlab\2018秋季\lecture 8\media1.mp4

GUI demos


Browser GUI demo
File
Edit
View
Insert
Tools
Desktop
Window
Help
www.baidu.com
www.buaa.edu.cn
course.buaa.edu.cn
mail.buaa.edu.cn
糯米
新闻
hao123
地图
视频
贴吧
登录
设置
更多产品
Bai du 百度
百度一下
手机百度
把百度设为主页
关于百度
About Baidu
百度推广
©2016 Baidu 使用百度前必读 意见反馈 京ICP证030173号 京公网安备11000002000001号
激活 Windows
转到"设置"以激活 Windows。

Homework 6



HW6-1. Write a function LS_line using three sub-functions my_slope, my_intercept and my_Corr that will determine the the slope *m* and intercept *b* of the least-squares line that best fits an input data set, and also the correlation coefficient of the fit. The input data points (*x*, *y*) will be passed to the function in two input arrays, x and y. The equations describing the slope, intercept of the least-squares line and the correlation coefficient are given below: [Note: Do not use any built-in function except length, numel, size]

$$\text{Slope} = m = \frac{\sum_{i=1}^{n}(x_i * y_i) - \left(\sum_{i=1}^{n} x_i\right) * \bar{y}}{\left(\sum_{i=1}^{n}(x_i)^2\right) - \left(\sum_{i=1}^{n} x_i\right) * \bar{x}}, \text{where } \bar{x} \,\&\, \bar{y} \text{ are the average values (mean)}$$

$$b(\text{intercept}) = \bar{y} - m\bar{x}$$

$$\text{Corr} = \frac{n * \sum_{i=1}^{n}(x_i * y_i) - \left(\sum_{i=1}^{n} x_i\right) * \left(\sum_{i=1}^{n} y_i\right)}{\sqrt{\left(\left(n * \sum_{i=1}^{n}(x_i)^2 - \left(\sum_{i=1}^{n} x_i\right)^2\right)\right)\left(n * \sum_{i=1}^{n}(y_i)^2 - \left(\sum_{i=1}^{n} y_i\right)^2\right)}}$$

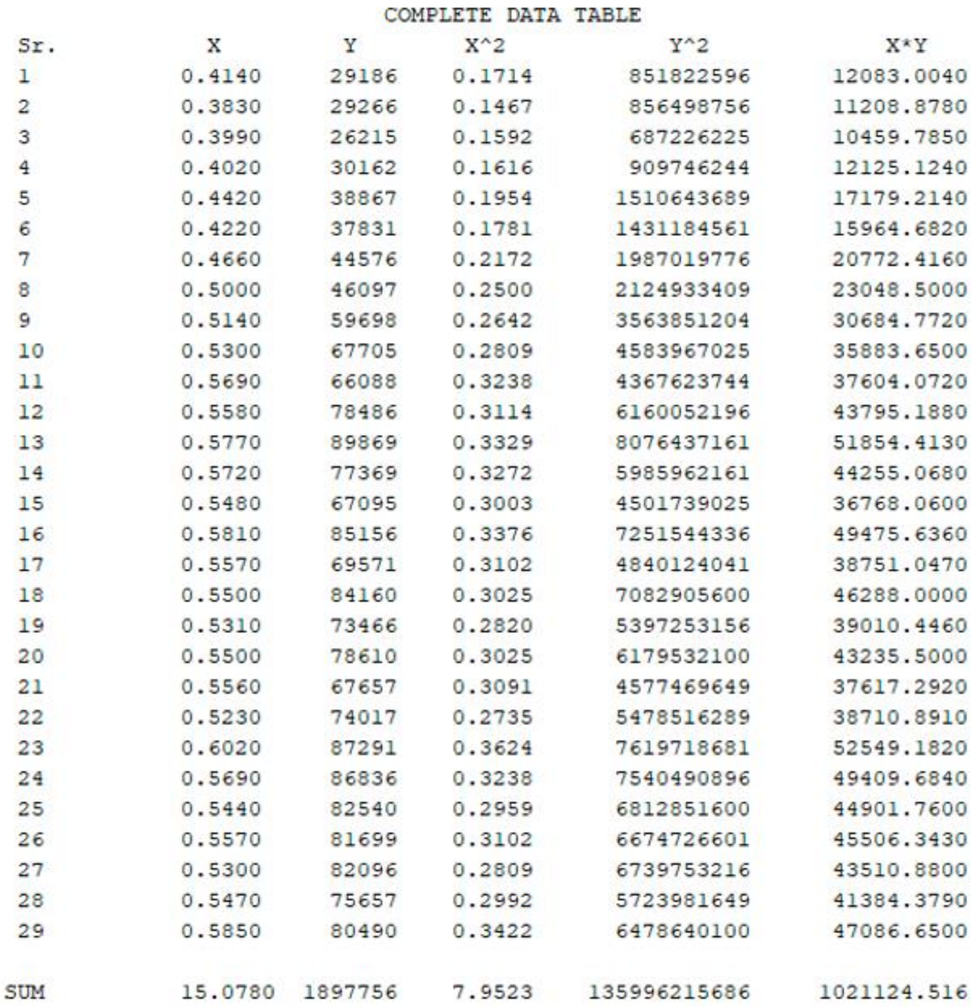
Copy the following data in notepad and create a text file as 'My_file.txt' (only numbers having three columns including serials)

Sr.	X	Y	Sr.	X	Y
1	0.414	29186	16	0.581	85156
2	0.383	29266	17	0.557	69571
3	0.399	26215	18	0.55	84160
4	0.402	30162	19	0.531	73466
5	0.442	38867	20	0.55	78610
6	0.422	37831	21	0.556	67657
7	0.466	44576	22	0.523	74017
8	0.5	46097	23	0.602	87291
9	0.514	59698	24	0.569	86836
10	0.53	67705	25	0.544	82540
11	0.569	66088	26	0.557	81699
12	0.558	78486	27	0.53	82096
13	0.577	89869	28	0.547	75657
14	0.572	77369	29	0.585	80490
15	0.548	67095			

1. Create a script file and load the text file **'My_file.txt'**
2. In the same script, Call the above function **'LS_line'** to find Slope, Intercept and Correlation Coefficient [Using the values of X & Y from **'My_file.txt'**]
3. In the same script, Compare your result with the built-in function for Correlation [Use Corr]
4. In the same script, Plot the above data (X,Y) (use scatter) and draw a straight line using equation $y = mx + b$, where m is slope and b is the intercept
5. In the same script,Using the data of **'My_file.txt'** Create another text file **'Complete_data_File.txt'** that displays the Serial No, Values of X, Values of Y, Square of X-values, Square of Y-values, X*Y and in the end sum of each column in the command prompt (as shown below) **[Note: Do not use any built-in function except length, numel, size]**

Submit homework via online before Nov 12，2019

Homework 6



COMPLETE DATA TABLE

Sr.	X	Y	X^2	Y^2	X*Y
1	0.4140	29186	0.1714	851822596	12083.0040
2	0.3830	29266	0.1467	856498756	11208.8780
3	0.3990	26215	0.1592	687226225	10459.7850
4	0.4020	30162	0.1616	909746244	12125.1240
5	0.4420	38867	0.1954	1510643689	17179.2140
6	0.4220	37831	0.1781	1431184561	15964.6820
7	0.4660	44576	0.2172	1987019776	20772.4160
8	0.5000	46097	0.2500	2124933409	23048.5000
9	0.5140	59698	0.2642	3563851204	30684.7720
10	0.5300	67705	0.2809	4583967025	35883.6500
11	0.5690	66088	0.3238	4367623744	37604.0720
12	0.5580	78486	0.3114	6160052196	43795.1880
13	0.5770	89869	0.3329	8076437161	51854.4130
14	0.5720	77369	0.3272	5985962161	44255.0680
15	0.5480	67095	0.3003	4501739025	36768.0600
16	0.5810	85156	0.3376	7251544336	49475.6360
17	0.5570	69571	0.3102	4840124041	38751.0470
18	0.5500	84160	0.3025	7082905600	46288.0000
19	0.5310	73466	0.2820	5397253156	39010.4460
20	0.5500	78610	0.3025	6179532100	43235.5000
21	0.5560	67657	0.3091	4577469649	37617.2920
22	0.5230	74017	0.2735	5478516289	38710.8910
23	0.6020	87291	0.3624	7619718681	52549.1820
24	0.5690	86836	0.3238	7540490896	49409.6840
25	0.5440	82540	0.2959	6812851600	44901.7600
26	0.5570	81699	0.3102	6674726601	45506.3430
27	0.5300	82096	0.2809	6739753216	43510.8800
28	0.5470	75657	0.2992	5723981649	41384.3790
29	0.5850	80490	0.3422	6478640100	47086.6500
SUM	15.0780	1897756	7.9523	135996215686	1021124.516

Submit homework via online before Nov 12，2019

Attention

- **The presentation time is Nov 26, 2019.**
- **Please send me an email to bingsun@buaa.edu.cn if you want to give us presentation before Nov 19, 2019.**
- **You’d better to show a slide.**

Thanks