

# Creación de interfaces de usuario con MATLAB

Pedro Corcuera

Dpto. Matemática Aplicada y  
Ciencias de la Computación

**Universidad de Cantabria**

**corcuerp@unican.es**



# Objetivos

---

- Aprender a crear interfaces de usuario desde el entorno Matlab
- Utilizar la capacidad de generación de código de Matlab para distribuir aplicaciones



# Indice

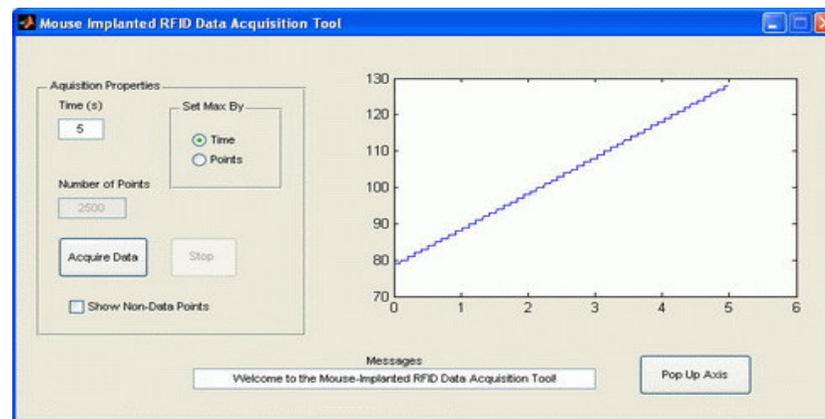
---

- Interfaces de usuario (GUI)
- Creación de GUIs con GUIDE
- Ejemplo de desarrollo GUI
- Generación de ejecutable



## Interfaces gráficas de usuario (GUI)

- Una GUI debe ser consistente y fácilmente entendida por el usuario
- Proporciona al usuario con la habilidad de usar un programa sin tener que preocuparse sobre los comandos para ejecutar un programa
- Los componentes habituales de una GUI son
  - Pushbuttons
  - Sliders
  - List boxes
  - Menus, ..etc





# Características esenciales de una GUI

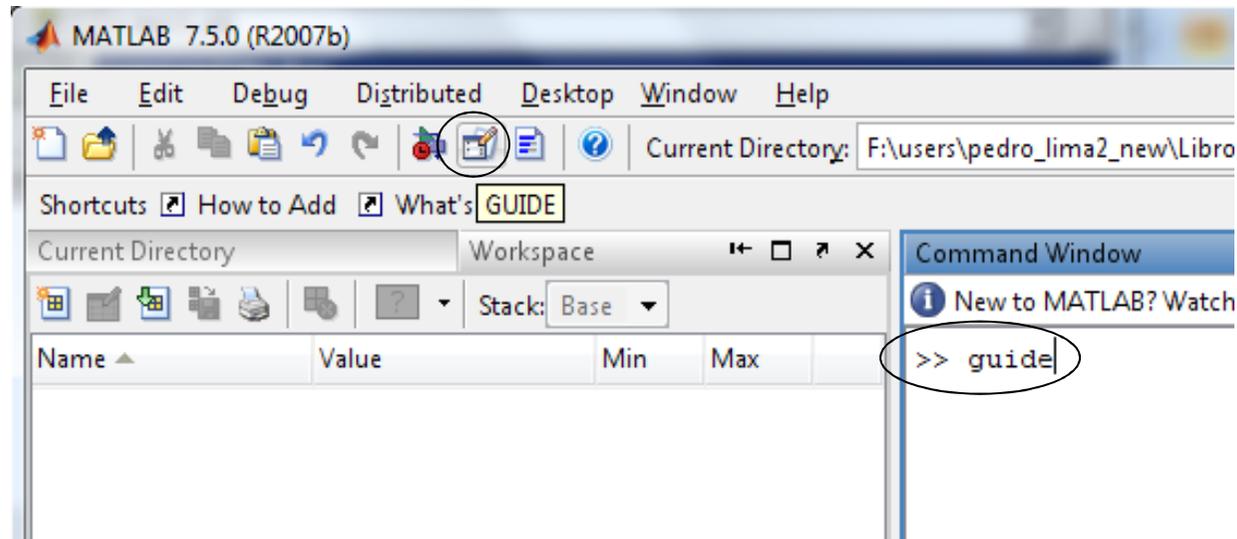
---

- Componentes
  - Gráficos
    - pushbuttons, edit boxes, sliders, labels, menus, etc...
  - Estáticos
    - Frames, text strings,...
  - Ambos se crean usando la función uicontrol
- Figuras – los componentes están contenidos en figuras
- Callbacks – son funciones que realizan las acciones requeridas cuando un componente se activa



# Creación de GUIs con GUIDE

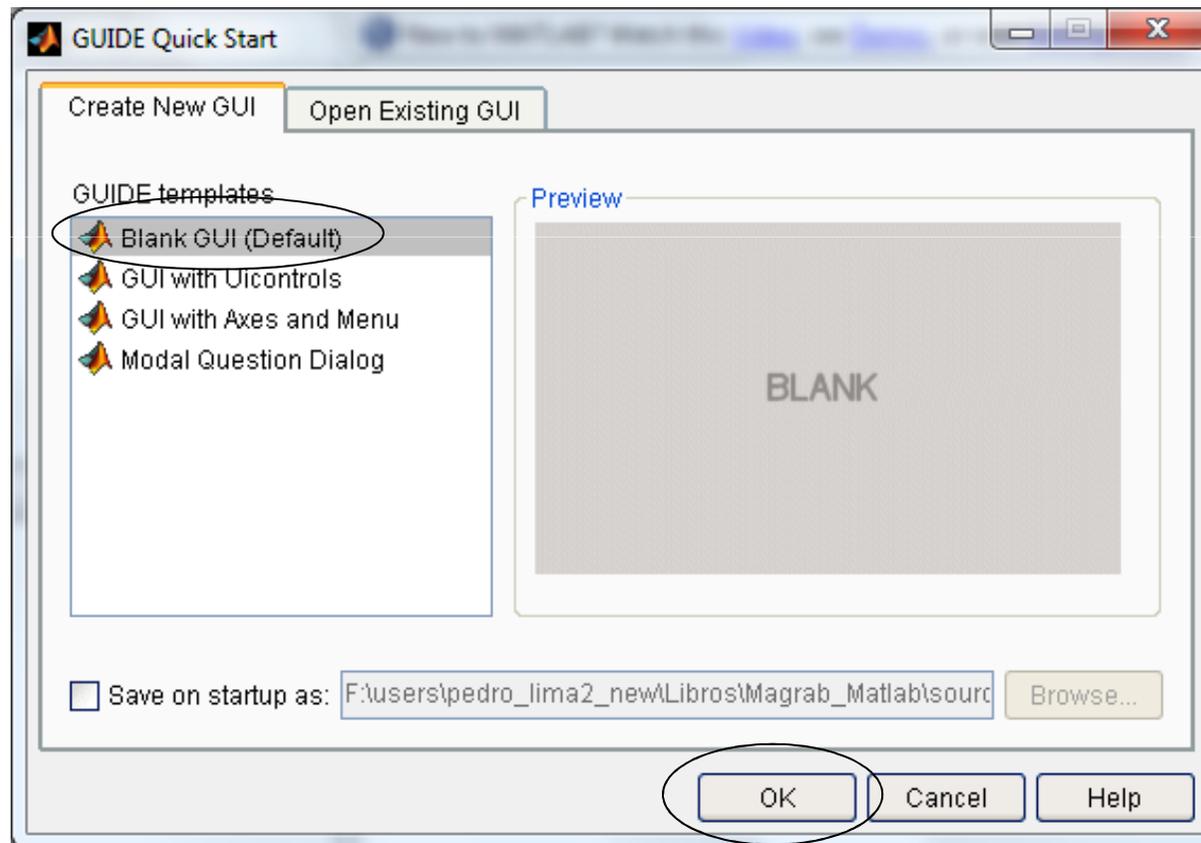
- MATLAB dispone de una utilidad para el desarrollo interactivo de interfaces de usuario (GUI) llamado GUIDE
- Para ello se escribe el comando `guide` o se pulsa sobre su icono en el entorno de desarrollo Matlab





## Entorno de desarrollo GUIDE

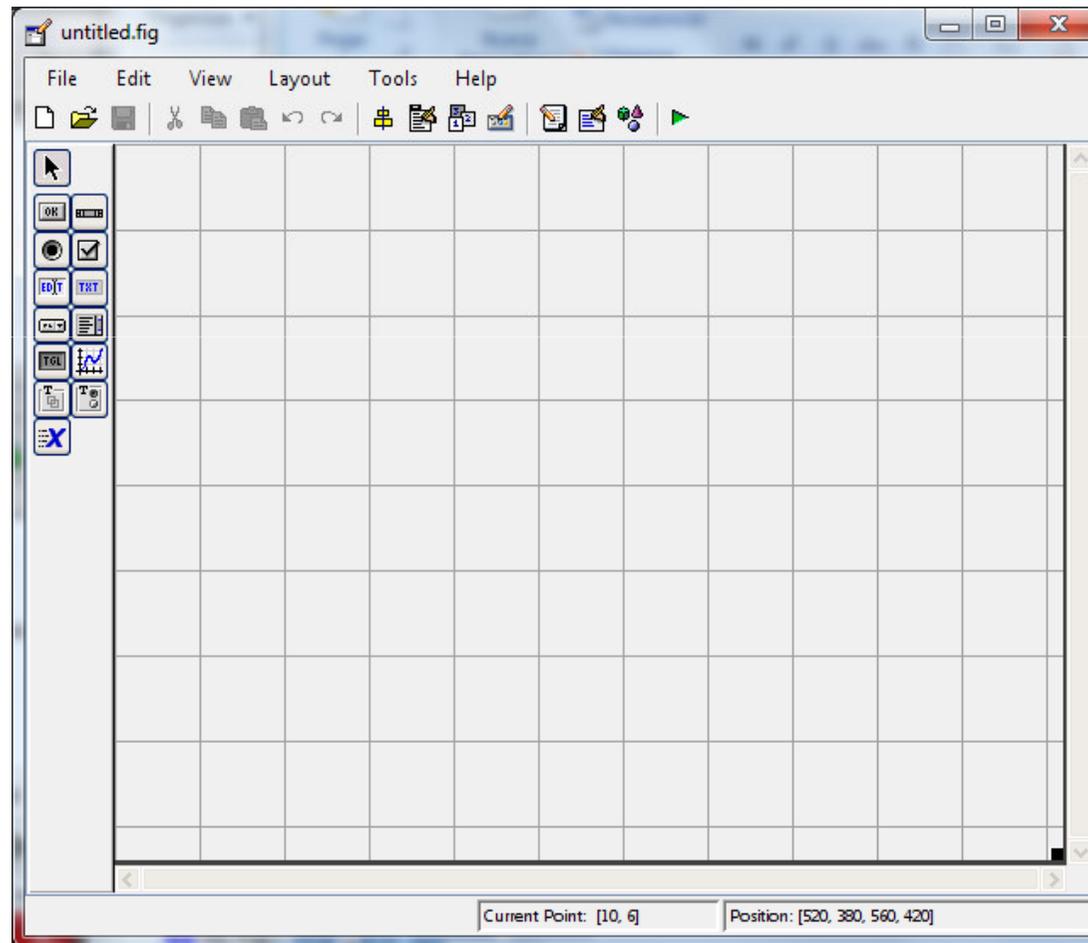
- Aparece una ventana para seleccionar plantillas o abrir GUIs existentes





# Entorno de desarrollo GUIDE

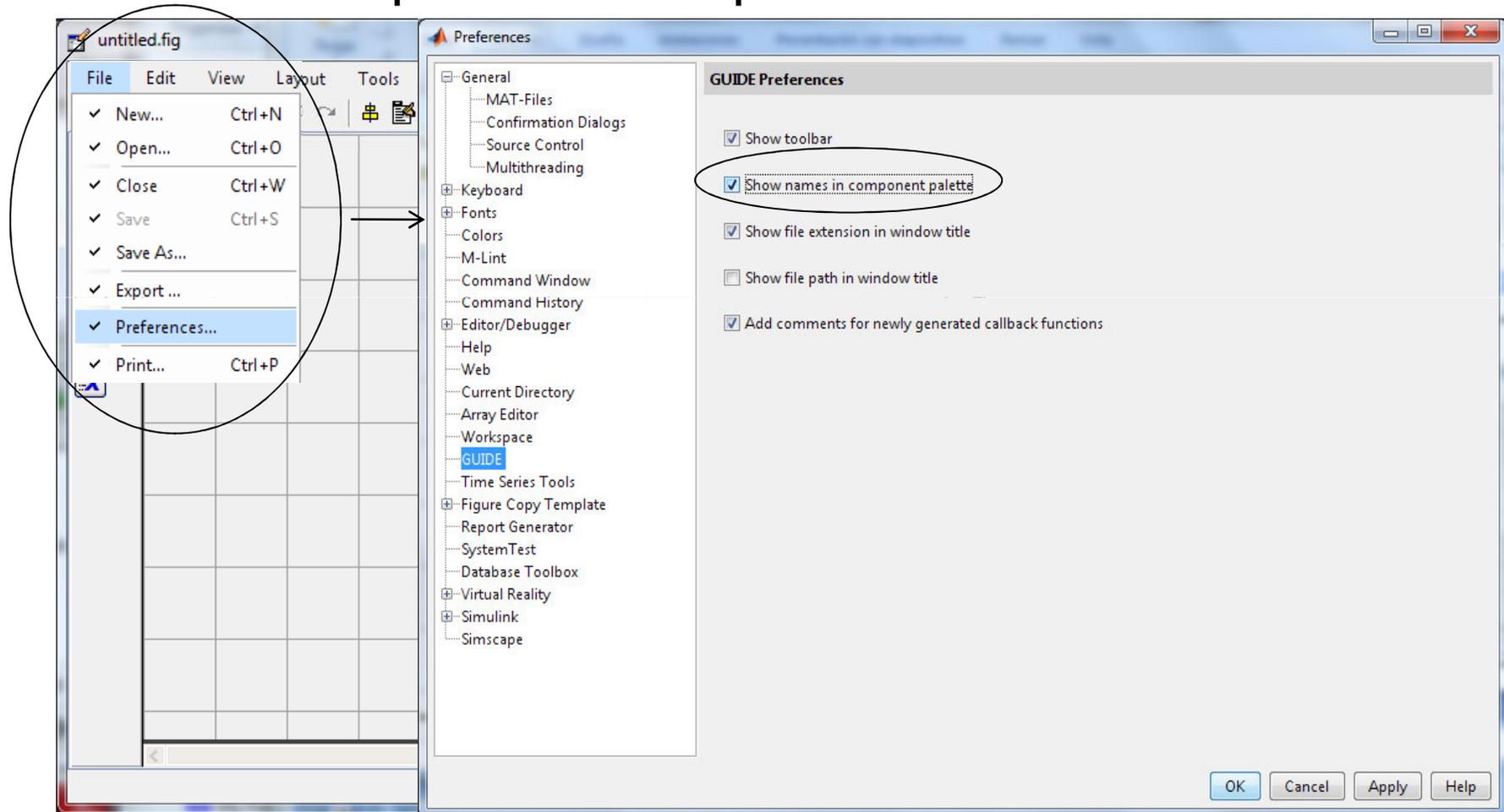
- Ventana de desarrollo GUI





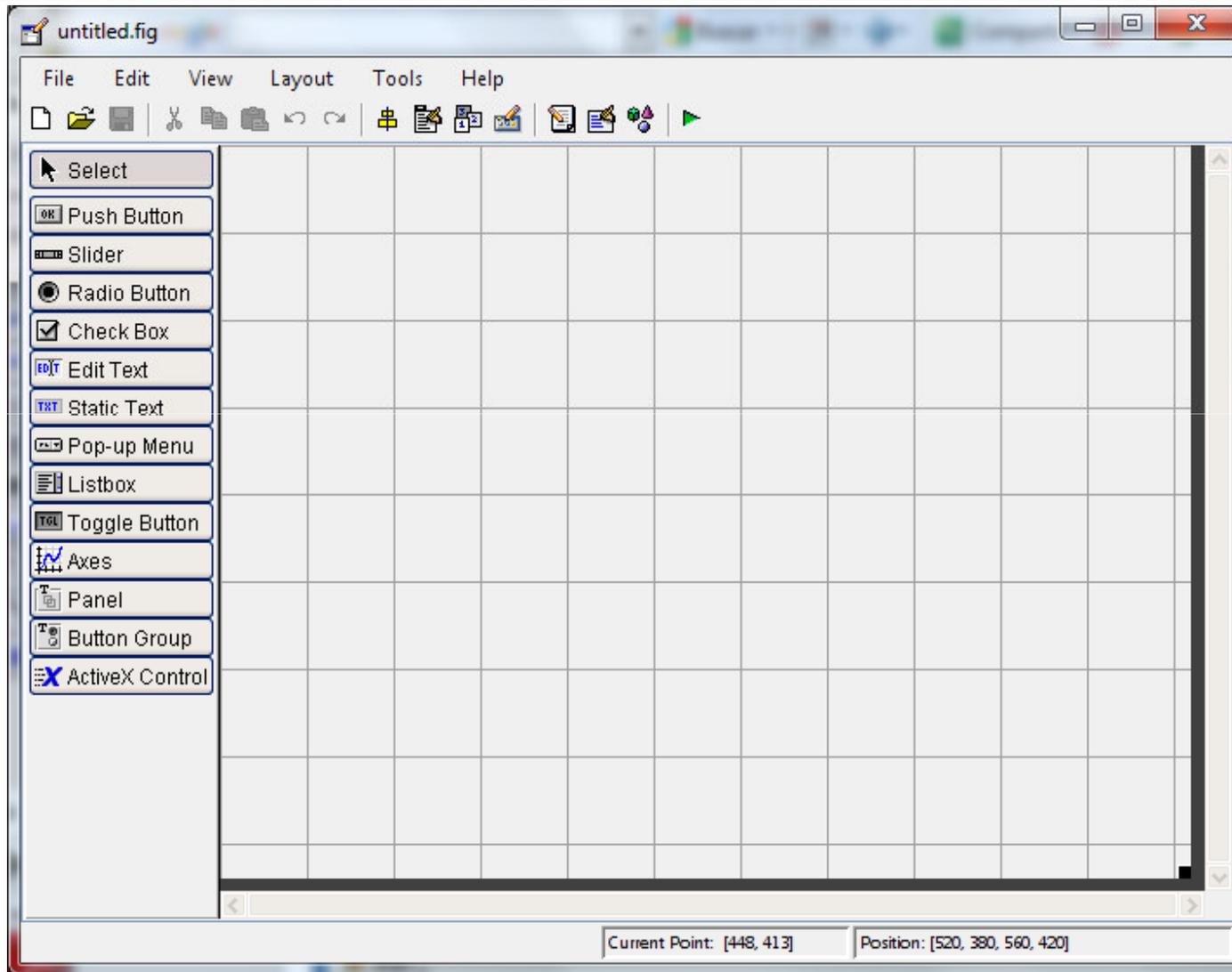
# Entorno de desarrollo GUIDE

- Cambio de preferencias para mostrar nombres





# Entorno de desarrollo GUIDE





## Entorno de desarrollo GUIDE

---

- El editor permite construir interfaces arrastrando y soltando componentes en el área de diseño de la GUI
- Todas las GUIs creadas con guide empiezan con una función inicial (callback) que se invoca cuando se invoca la interfaz
- La operación automática de guardado (save) genera un fichero .m y un fichero .fig
- El fichero .fig contiene el diseño del GUI en binario y el fichero .m contiene el código que controla el GUI



# Entorno de desarrollo GUIDE

---

- Componentes disponibles en la paleta





## Ejemplo de GUI con GUIDE

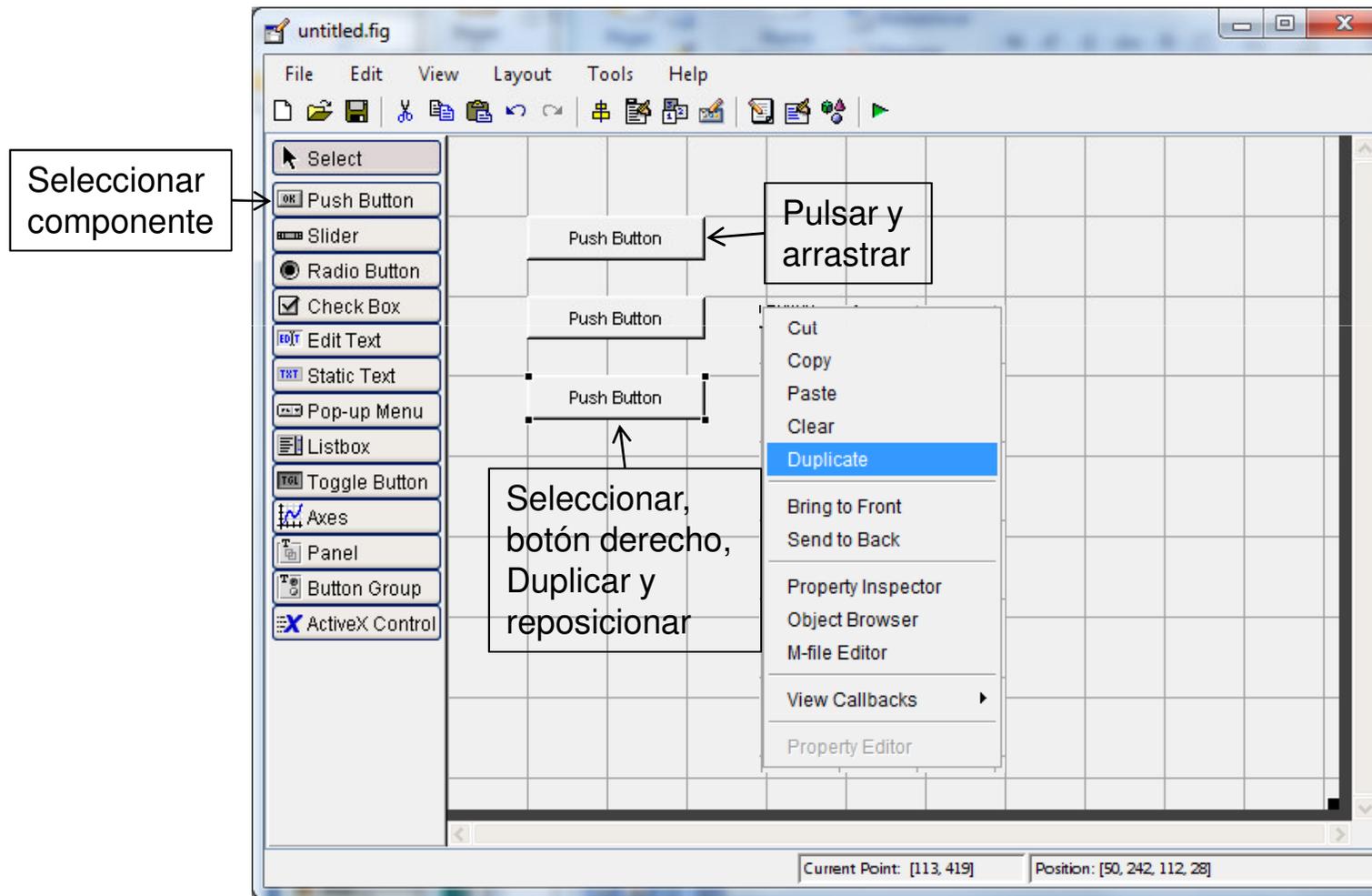
---

- La interfaz que se desarrollará consiste en permitir al usuario seleccionar entre tres conjuntos de datos y mostrarlos según tres tipos de gráficos
- Para ello se insertarán los componentes adecuados desde la paleta de componentes (Push Buttons, Panel, Static Text box, Pop-up Menu, y Axes)
- Se puede redimensionar el tamaño del canvas de la interfaz en modo diseño, aunque después se puede modificar haciendo uso de Tools



# Ejemplo de GUI con GUIDE

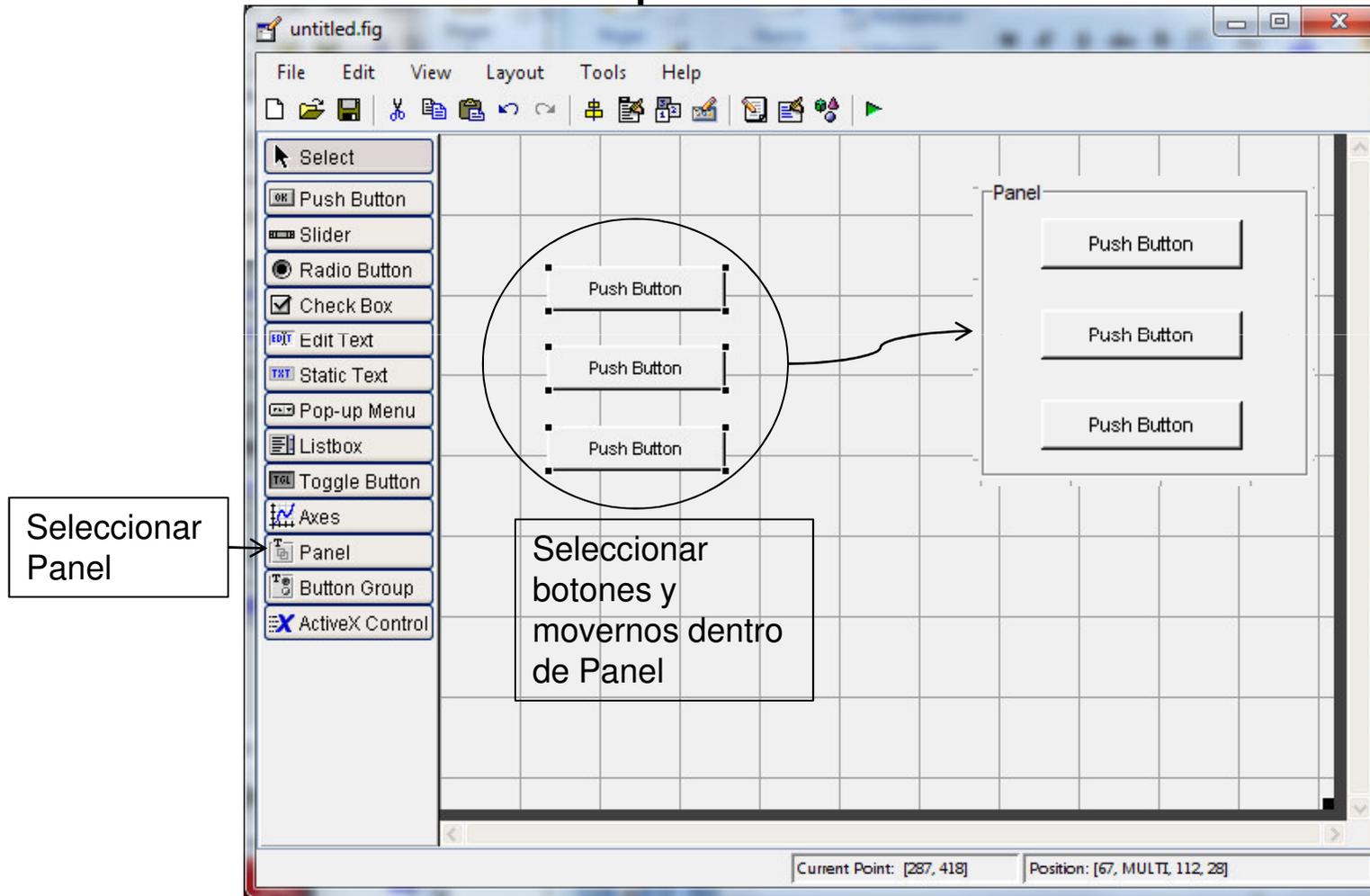
- Se inserta un Push Button y por duplicación dos más





# Ejemplo de GUI con GUIDE

- Se añade un Panel para colocar los botones dentro





# Ejemplo de GUI con GUIDE

- Añadir Static Text, Pop-up Menu y Axis

Seleccionar componentes



# Ejemplo de GUI con GUIDE

- Alineamiento de objetos

Seleccionar componentes a alinear

Panel

Push Button

Push Button

Push Button

Static Text

Pop-up Menu

Align Objects

Vertical

Align

Distribute

Set spacing 20 pixels

Horizontal

Align

Distribute

Set spacing 20 pixels

OK Apply Cancel

Current Point: [145, 48] Position: [37, MULTI, 112, 28]



# Ejemplo de GUI con GUIDE

- Modificación de propiedades con Property Inspector

The screenshot displays the MATLAB GUIDE interface. The main window, titled 'untitled.fig', shows a GUI design with a central axes object labeled 'axes1'. The 'View' menu is open, and 'Property Inspector' is selected. The Property Inspector window on the right shows the properties of the selected 'axes1' object, including 'BeingDeleted', 'BusyAction', 'ButtonDownFcn', 'Clipping', 'CloseRequestF...', 'Color', 'CreateFcn', 'CurrentCharac...', 'CurrentPoint', 'DeleteFcn', 'DockControls', 'DoubleBuffer', 'FileName', 'FixedColors', 'HandleVisibility', 'HitTest', 'IntegerHandle', 'Interruptible', 'InvertHardcopy', 'KeyPressFcn', and 'KeyReleaseFcn'. A text box at the bottom of the GUI design area contains the text 'Seleccionar componente a modificar propiedad'.



## Ejemplo de GUI con GUIDE

---

- Modificación de propiedades con Property Inspector:
    - Figure, Name: GUI Simple
    - Panel , Title: Tipos de gráficos
    - Push Button, String: Surf, Tag: surf\_pb
    - Push Button, String: Mesh, Tag: mesh\_pb
    - Push Button, String: Contour, Tag: contour\_pb
    - Static Text, String: Seleccionar datos
    - Pop-up Menu, String: (editor) peaks, membrane, sinc  
Tag: plot\_popup
  - Grabar GUI. File - Save As: ejm\_gui (sufijo .fig)
-



# Ejemplo de GUI con GUIDE

- Se puede activar (Run) la GUI y editar el código

The screenshot displays the MATLAB environment with two windows. The top-left window is the 'untitled.fig' window, showing the 'Run' button circled in red. A label 'Editor código' points to the 'Run' button. The bottom-left window is the 'Editor - F:\users\pedro\_lima2\_new\Libros\Magrab\_Matlab\source\ejm\_gui.m' window, showing the MATLAB code for the GUI. The code includes comments and function definitions. The right window is the 'ejm\_gui' window, which is a GUI with a plot area and control buttons. The plot area shows a 2D plot with axes ranging from 0 to 1. The control panel on the right has buttons for 'Surf', 'Mesh', and 'Contour', and a dropdown menu labeled 'Seleccionar datos' with 'peaks' selected.

```
1 function varargout = ejm_gui(varargin)
2 % EJM_GUI M-file for ejm_gui.fig
3 % EJM_GUI, by itself, creates a new EJM_GUI or raises
4 % singleton*.
5 %
6 % H = EJM_GUI returns the handle to a new EJM_GUI or to
7 % the existing singleton*.
8 %
9 % EJM_GUI('CALLBACK',hObject,eventData,handles,...) calls
10 % function named CALLBACK in EJM_GUI.M with the given
11 %
12 % EJM_GUI('Property','Value',...) creates a new EJM_GUI
13 % existing singleton*. Starting from the left, proper
14 % applied to the GUI before ejm_gui_OpeningFcn gets called.
15 % unrecognized property name or invalid value makes pr
16 % stop. All inputs are passed to ejm_gui_OpeningFcn v
17 %
18 % *See GUI Options on GUIDE's Tools menu. Choose "GUI
19 % instance to run (singleton)".
```



## Ejemplo de GUI con GUIDE

- Se puede adaptar el código generado por GUIDE para ejecutarse cuando se seleccionan los controles
- El código de usuario introducido se llama *funciones callback*. El editor M facilita la navegación por estas funciones con el icono Show Functions

```
Editor - F:\users\pedro_lima2_new\Libros\Magrab_Matlab\source\ejm_gui.m
File Edit Text Go Cell Tools Debug Desktop Window Help
fx
Stack: Base
- 1.0 + ÷ 1.1 x
1 function varargout = ejm_gui
2 % EJM_GUI M-file for ejm_gui
3 % EJM_GUI, by itself, c
4 % singleton*.
5 %
6 % H = EJM_GUI returns t
7 % the existing singleto
8 %
9 % EJM_GUI('CALLBACK',hO
    ✓ ejm_gui
    ✓ contour_pb_Callback
    ✓ ejm_gui_OpeningFcn
    ✓ ejm_gui_OutputFcn
    ✓ mesh_pb_Callback
    ✓ plot_popup_Callback
    ✓ plot_popup_CreateFcn
    ✓ surf_pb_Callback
    or raises the exist
    M_GUI or the handle
    es,...) calls the 1
```



## Ejemplo de GUI con GUIDE

---

- Primero se modifica la función OpeningFcn que se ejecuta al inicio de ejecutar el GUI
- En esta función se puede cargar o crear datos a usar en el GUI y realizar algunos gráficos iniciales
- Para compartir datos dentro del GUI se usa la estructura handles que se pasa entre funciones



# Ejemplo de GUI con GUIDE

```
function ejm_gui_OpeningFcn(hObject, eventdata, handles, varargin)
% This function has no output args, see OutputFcn.
% hObject    handle to figure
% eventdata  reserved - to be defined in a future version of MATLAB
% handles    structure with handles and user data (see GUIDATA)
% varargin   command line arguments to ejm_gui (see VARARGIN)
% Create the data to plot
handles.peaks = peaks(35);
handles.membrane = membrane;
[x,y]=meshgrid(-8:0.5:8);
r=sqrt(x.^2 + y.^2) + eps;
sinc = sin(r)./r;
handles.sinc = sinc;
handles.current_data = handles.peaks;
surf(handles.current_data);
% Choose default command line output for ejm_gui
handles.output = hObject;
% Update handles structure
guidata(hObject, handles);
```

Código añadido

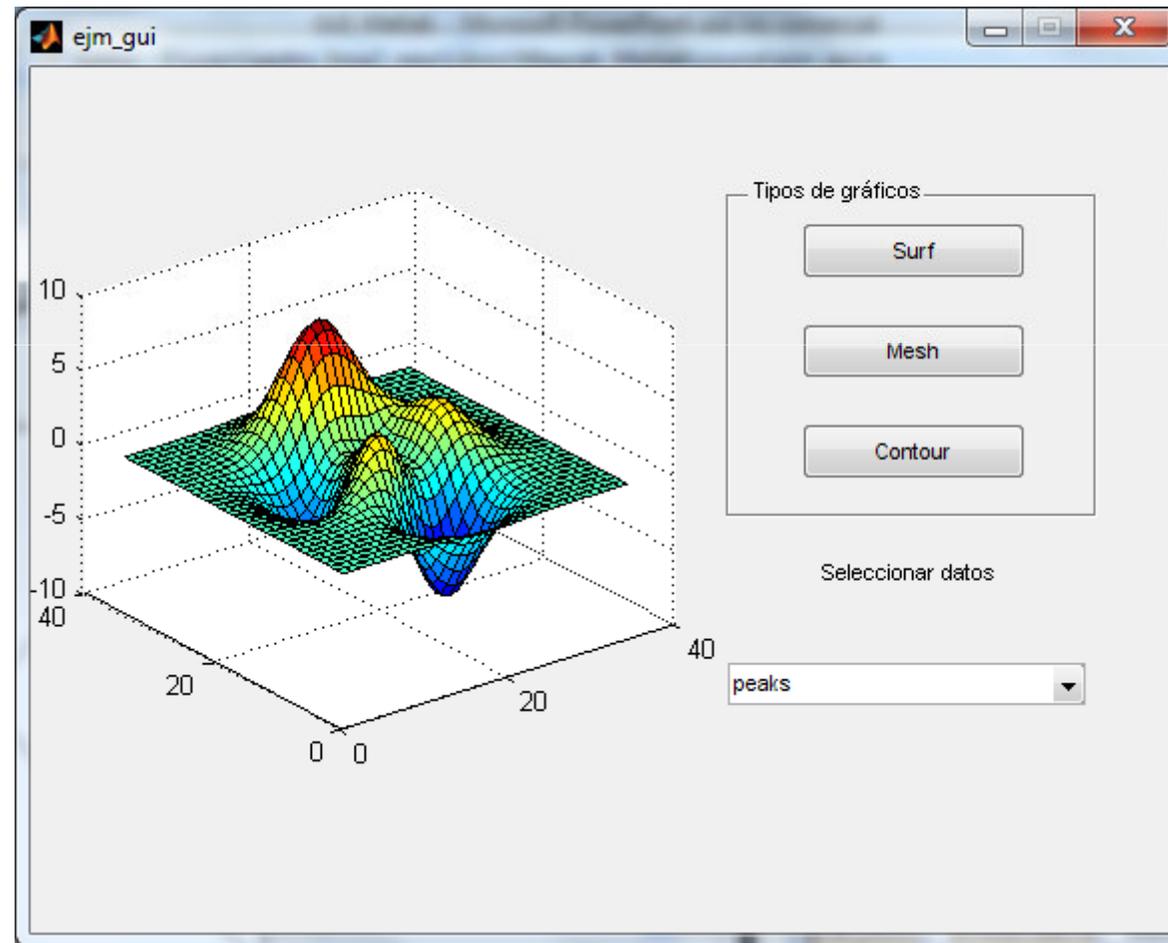
Argumento retornado cuando se invoca el GUI

Línea que actualiza la GUI



# Ejemplo de GUI con GUIDE

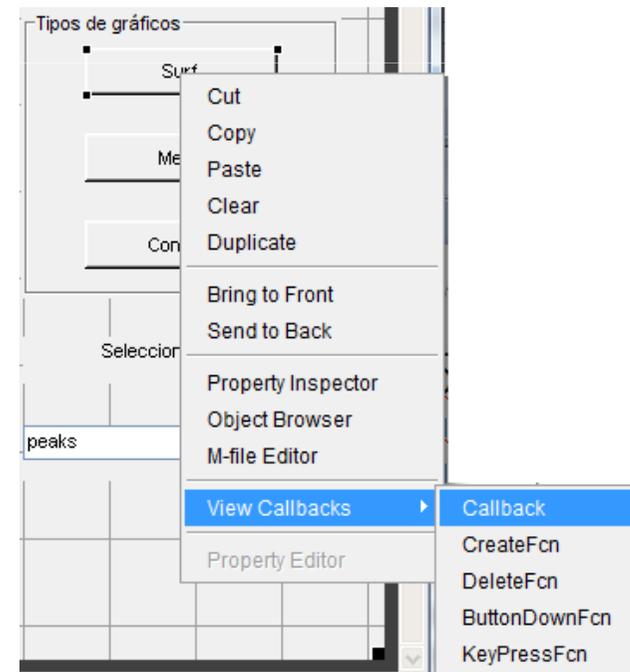
- Después de introducir el código se activa la GUI





## Ejemplo de GUI con GUIDE

- Cuando una GUI está completa y en ejecución, y el usuario pulsa sobre un control de la interfaz de usuario, como un botón, Matlab ejecuta la función callback del control
- Otra forma de acceder a la función callback de un control es usar el editor de GUI, seleccionar el control y hacer click en el botón derecho





## Ejemplo de GUI con GUIDE

---

- Se agregan los callbacks de los botones

```
% --- Executes on button press in surf_pb.  
function surf_pb_Callback(hObject, eventdata, handles)  
% hObject    handle to surf_pb (see GCBO)  
% eventdata  reserved - to be defined in a future version of MATLAB  
% handles    structure with handles and user data (see GUIDATA)  
% Display surf plot of the currently selected data  
surf(handles.current_data);  
% --- Executes on button press in mesh_pb.  
function mesh_pb_Callback(hObject, eventdata, handles)  
% hObject    handle to mesh_pb (see GCBO)  
% eventdata  reserved - to be defined in a future version of MATLAB  
% handles    structure with handles and user data (see GUIDATA)  
mesh(handles.current_data);  
% --- Executes on button press in contour_pb.  
function contour_pb_Callback(hObject, eventdata, handles)  
% hObject    handle to contour_pb (see GCBO)  
% eventdata  reserved - to be defined in a future version of MATLAB  
% handles    structure with handles and user data (see GUIDATA)  
contour(handles.current_data);
```



## Ejemplo de GUI con GUIDE

---

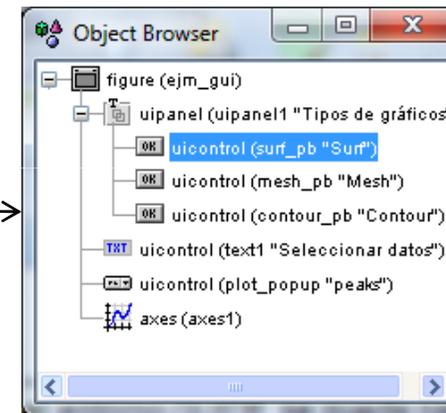
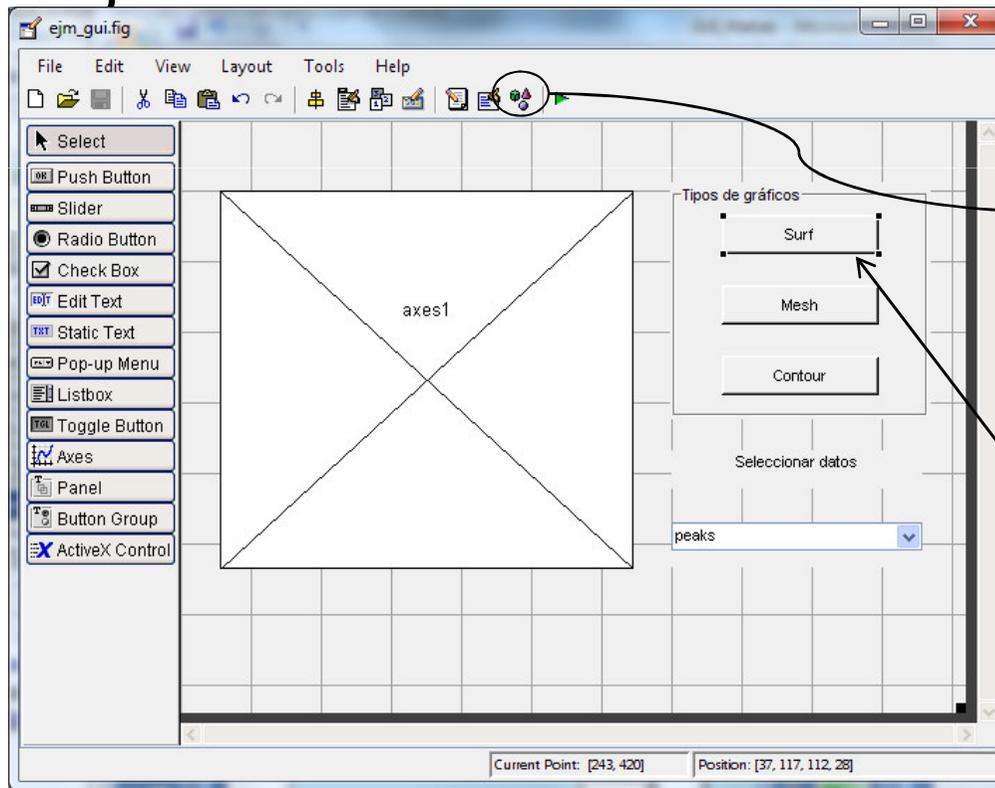
- Se agrega el callback para el popup menu

```
% --- Executes on selection change in plot_popup.  
function plot_popup_Callback(hObject, eventdata, handles)  
% hObject      handle to plot_popup (see GCBO)  
% eventdata    reserved - to be defined in a future version of MATLAB  
% handles      structure with handles and user data (see GUIDATA)  
% Hints: contents = get(hObject,'String') returns plot_popup contents as cell array  
%            contents{get(hObject,'Value')} returns selected item from plot_popup  
val = get(hObject,'Value');  
str = get(hObject,'String');  
switch str{val}  
    case 'peaks' % User selects peaks  
        handles.current_data = handles.peaks;  
    case 'membrane'  
        handles.current_data = handles.membrane;  
    case 'sinc'  
        handles.current_data = handles.sinc;  
end  
guidata(hObject,handles);
```



# Ejemplo de GUI con GUIDE

- En el entorno GUIDE se puede navegar por los componentes e identificar sus callbacks usando el Object Browser

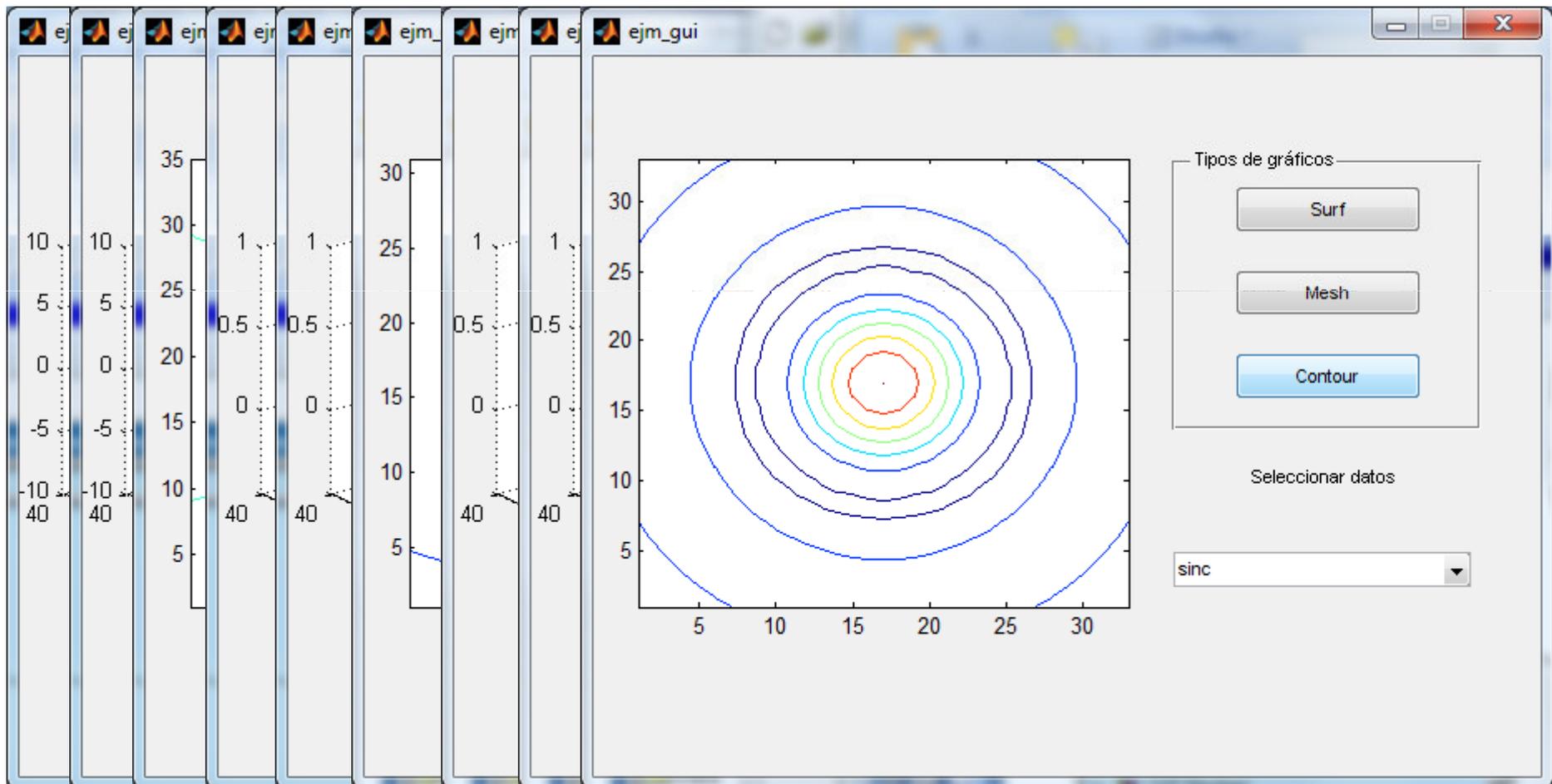


Mientras se navega se resalta el objeto



# Ejemplo de GUI con GUIDE

- Se activa la GUI





## Generación de ejecutables

---

- Matlab proporciona los comandos y herramientas necesarias para la generación de código ejecutable con el *Matlab Compiler* (sin requerir el entorno Matlab) pudiendo ser distribuído a usuarios finales
- Otras opciones son la integración de Matlab en aplicaciones C o C++, crear paquetes de librerías Matlab, incorporar algoritmos creados en Matlab en aplicaciones desarrolladas con otros lenguajes y tecnologías, encriptar y proteger el código Matlab



## Generación de ejecutables

---

- Para invocar el compilador Matlab se ejecuta la herramienta GUI *deploytool* o el comando *mcc*
- En los ejecutables o librerías creadas se puede incluir el MATLAB Compiler Runtime (MCR), que permite la ejecución de los mismos en ordenadores que no tienen una versión instalada de Matlab
- Una aplicación o librería generada por Matlab tiene dos partes: un fichero binario dependiente de la plataforma y un fichero con el código y datos Matlab encriptado



## Generación de ejecutables

---

- Matlab Compiler requiere la instalación de un compilador C o C++ compatible (también Fortran)
- Un compilador, gratuito, soportado es Microsoft Visual C++ Express

<http://www.microsoft.com/express/Downloads/#2010-Visual-CPP>

- Para seleccionar el compilador para usar con Matlab Compiler se usa el comando

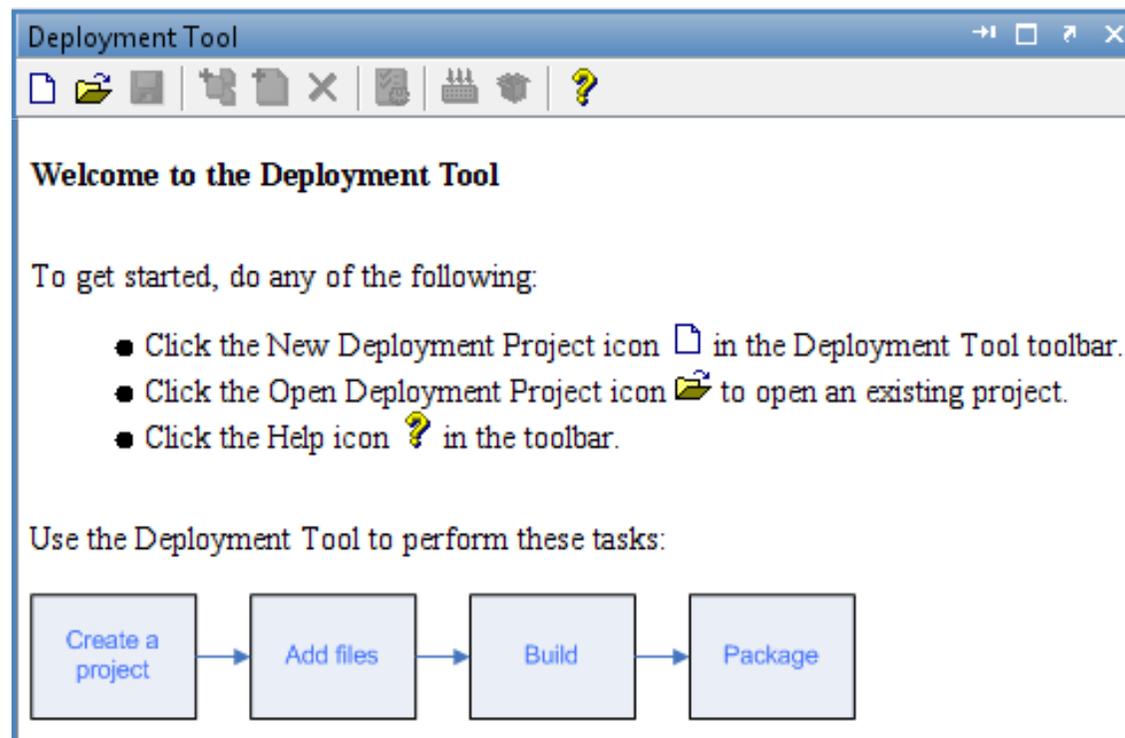
```
>> mbuild -setup
```

que pedirá el path donde se encuentra instalado MS VC sino lo detecta automáticamente



# Generación de ejecutables

- Invocar la herramienta de generación de ejecutables  
`>> deploytool`  
aparece la ventana de la herramienta en el entorno





# Generación de ejecutables

- Creación de un proyecto de despliegue indicando Name, Location Target

**Deployment Tool**

**Nuevo proyecto**

Welcome to the Deployment Tool

To get started, do any of the following:

- Click the New Deployment Project icon  in the D
- Click the Open Deployment Project icon  to open
- Click the Help icon  in the toolbar.

Use the Deployment Tool to perform these tasks:

Create a project → Add files → Build → Package

**New Deployment Project**

MATLAB Compiler  
MATLAB Builder for Excel  
MATLAB Builder for .NET  
MATLAB Builder for Java

Standalone Application  
Windows Standalone Application  
C Shared Library  
C++ Shared Library

Target

Name: GUIexe.prj

Location: F:\users\pedro\_lima2\_new\Libros\Magrab\_Matlab\source



# Generación de ejecutables

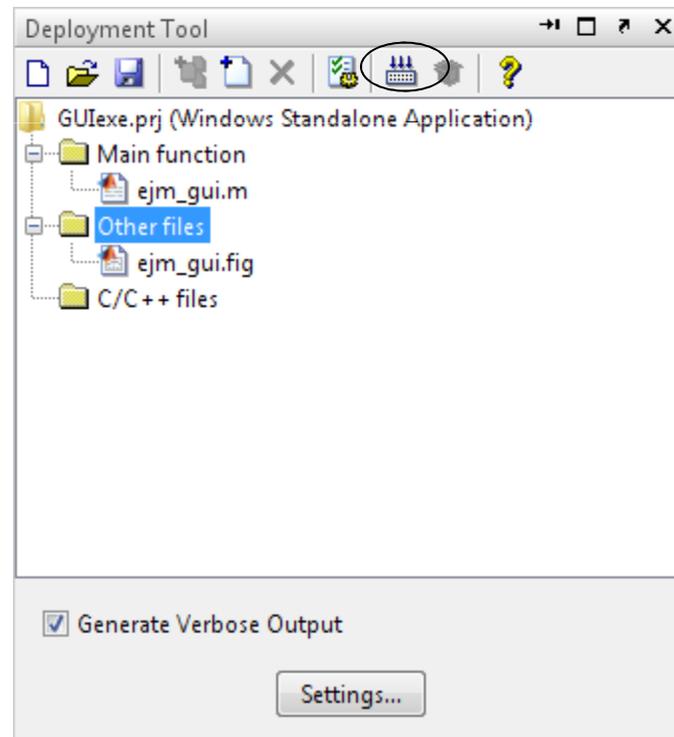
- Se agrega la función principal (.m) y los ficheros necesarios desde la ventana de directorios o con add

The screenshot shows the MATLAB 7.5.0 (R2007b) interface. The main window displays the 'Current Directory' with files like 'Ejemploode.m', 'ejm\_gui.asv', 'ejm\_gui.fig', and 'ejm\_gui.m'. The 'Deployment Tool' window shows a project named 'GUIexe.prj (Windows Standalone Application)' with subfolders 'Main function', 'Other files', and 'C/C++ files'. The 'Other files' folder is highlighted, indicating the process of adding files to the deployment project.



## Generación de ejecutables

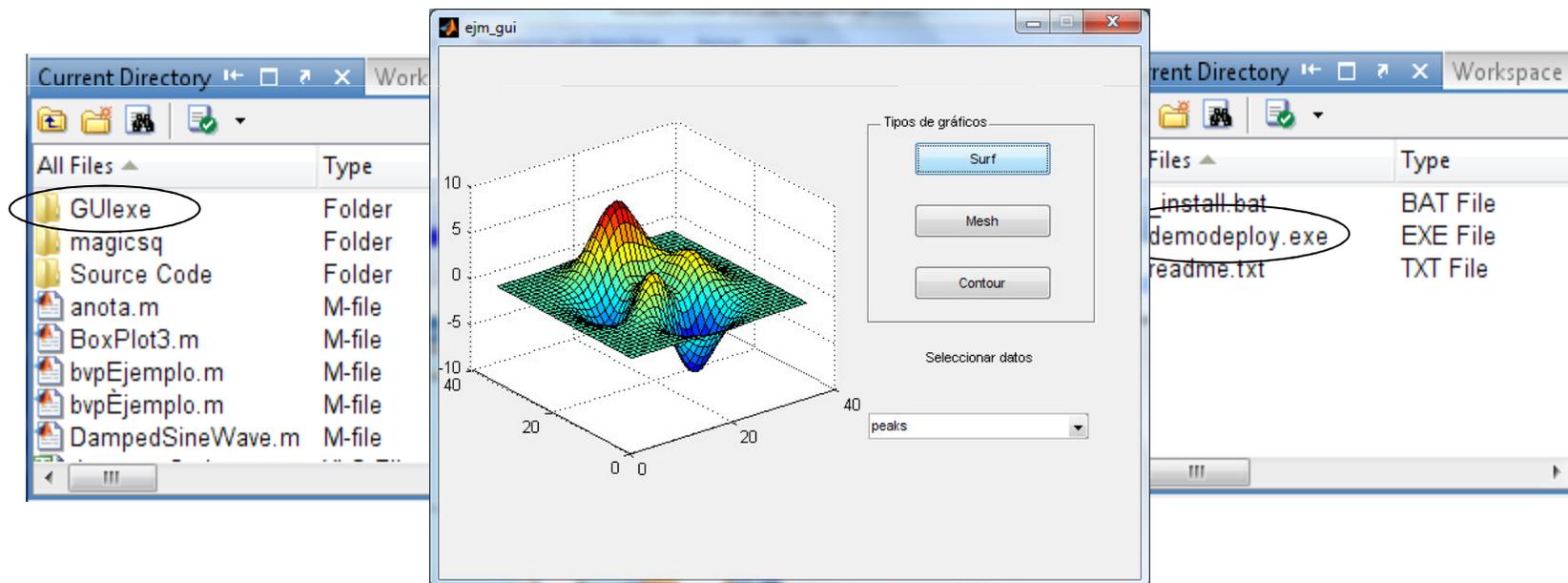
- Se genera el ejecutable pulsando sobre el icono Build the project  . En la ventana output de deploytool se visualiza el proceso de generación





# Generación de ejecutables

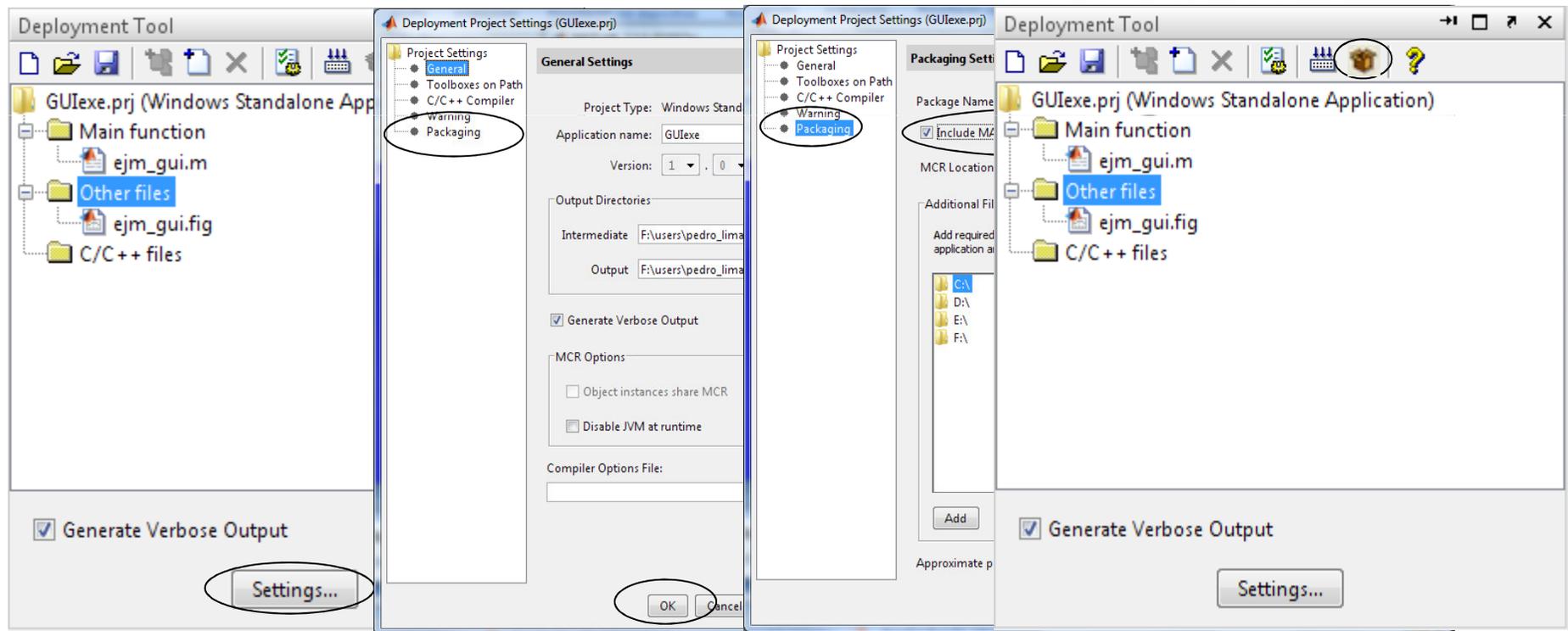
- Se puede probar el ejecutable generado, navegando por el directorio distrib del proyecto. Este ejecutable no incluye el MCR





# Generación de ejecutables

- Para empaquetar el programa, incluir la MCR y todos los ficheros necesarios, se requiere configurar el proyecto y pulsar sobre el icono de empaquetado 





# Generación de ejecutables

- En el directorio distrib del proyecto se genera el fichero empaquetado (.exe)

