# *8 Biomedical signal processing environment: Graphical User Interface*

# Chapter 8
# Biomedical Signal Processing Environment: Graphical User Interface

## 8.1 Introduction

A graphical user interface (GUI) is a user interface built with graphical objects; the components of the GUI; such as buttons, text fields, sliders, and menus. A well designed GUI is intuitively obvious to the user regarding the operation of the system. Applications that provide GUIs are generally easier to learn and use. The action that results from a particular user action can be made clear by design of the interface.

MATLAB implements GUIs as figure windows containing various styles of ui-control objects. We can program each object to perform the intended action when activated by user of the GUI. MATLAB's Graphical User Interface Development Environment (GUIDE) makes design easier.

A GUI can be created that sets the parameters of Simulink model. The Simulink diagram of the application must be open before running its GUI. If it is closed then GUI reopens it whenever it requires the model to execute. In addition, giving the facilities of obtaining simulation and saving the results and also plotting the results, the GUI can run the simulation and plot the results.

## 8.2 Implementation of GUI

Creating a GUI involves following basic tasks:

- ✓ Laying out the GUI components.
- ✓ Programming the GUI components.
- ✓ Saving and running the GUI.

GUIDE primarily is a set of layout tools and generates an M-file that contains code to handle the initialization and launching of the GUI. This M-file provides a framework for the implementation of the callbacks; the functions that execute when users activate components in the GUI. GUIDE stores GUIs in two files, which are generated the first time we save or run the GUI:

- ✓ **FIG-file:** A file with extension .fig that contains a complete description of the GUI figure layout and the components of the GUI: push buttons, menus, axes, etc. Changes to the GUI layout in the Layout Editor, our changes are saved in the FIG-file.

- ✓ **M-file:** A file with extension .m that contains the code that controls the GUI, including the callbacks for its components. This file is referred to as the GUI M-file. When we first run or save a GUI from the Layout Editor, GUIDE generates the GUI M-file with blank stubs for each of the callbacks. We can program the callbacks using the M-file editor.

**Figure 8.1** illustrates the parts of GUI implementation:

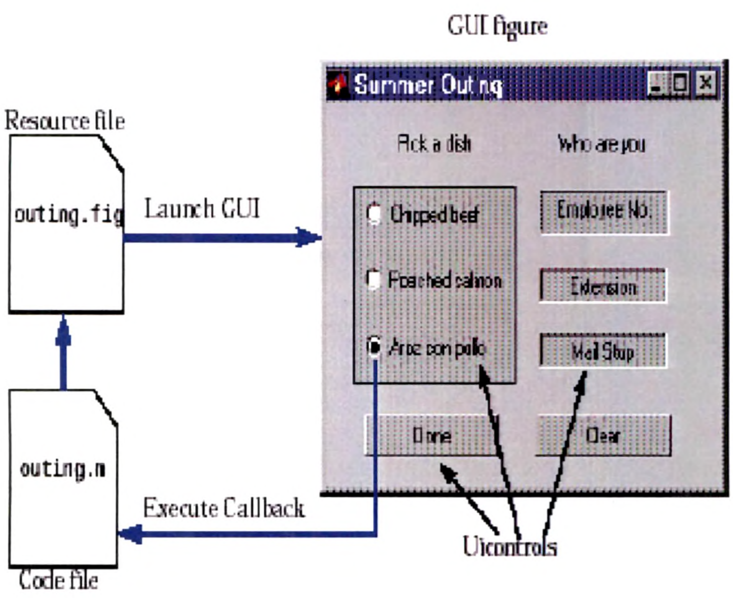



**Figure 8.1 Parts of GUI implementation**

### 8.3 GUIDE

It provides a set of tools for creating GUIs. These tools greatly simplify the process of laying out and programming a GUI.

When we open a GUI in GUIDE, it is displayed in the Layout Editor, which is the control panel for all of the GUIDE tools. The Layout Editor enables us to layout a GUI quickly and easily by dragging components, such as push buttons, pop-up menus, or axes, from the component palette into the layout area. The following picture shows the Layout Editor.

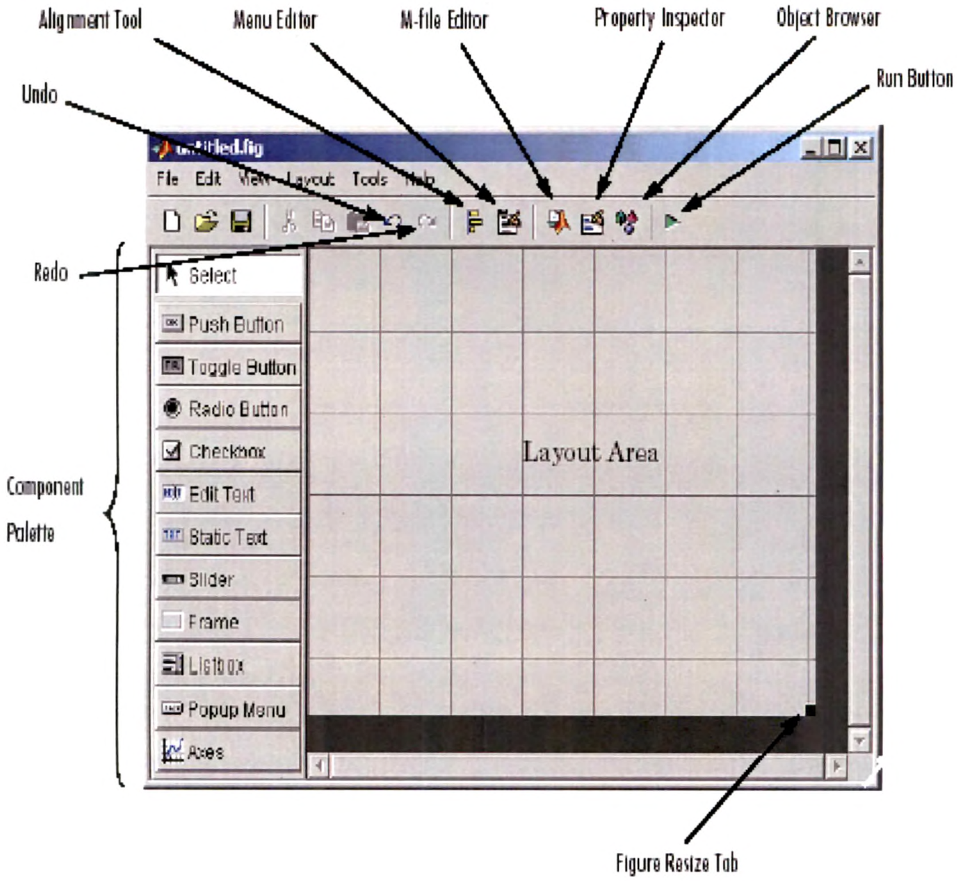



**Figure 8.2 Layout Editor**

Once GUI is layout and each component's properties are set, using the tools in the Layout Editor, we can program the GUI with the M-file Editor. Finally, when we press the Run button on the toolbar, the functioning GUI appears outside the Layout Editor window. The features of GUIDE are:

- ✓ **Laying Out GUIs:** Using The Layout Editor we can add and arrange objects in the figure window.

- ✓ **Aligning Components in the Layout Editor:** We can align objects vertically or horizontally and can distribute them in group of components with respect to each other.

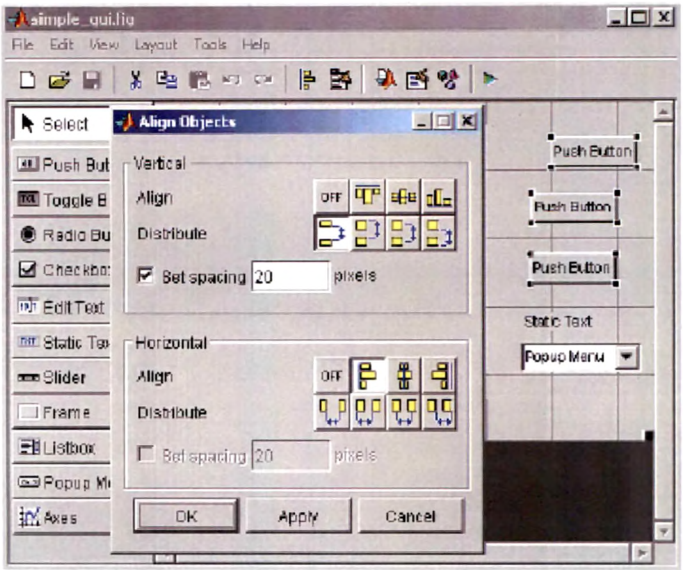



**Figure 8.3 Align objects**

✓ **Setting of Component Properties:** Using The Property Inspector we can inspect and set property values. Property Inspector provides a list of all settable properties and displays the current value. Each property in the list is associated with an editing device that is appropriate for the values accepted by the particular property.

Properties those can be set

Current values of properties.

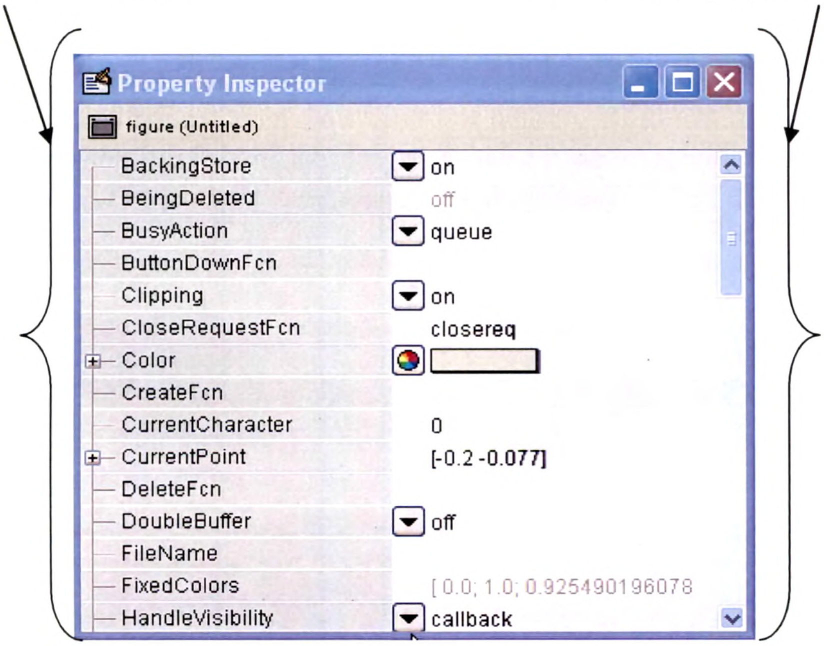



**Figure 8.4 Property inspector**

✓ **Viewing the Object Hierarchy:** Using The Object Browser we can observe a hierarchical list of the Handle Graphics objects in the current MATLAB session.

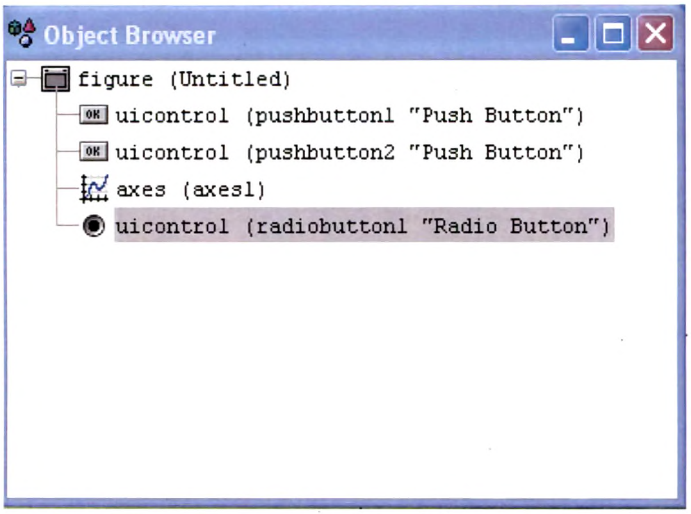



**Figure 8.5 Object Browser**

✓ **Creating Menus:** Using The Menu Editor we can create a menu bar; menus displayed on the figure menu bar; or a context menu; menus that pop up when users' right-click on graphics objects; for any component in the layout.

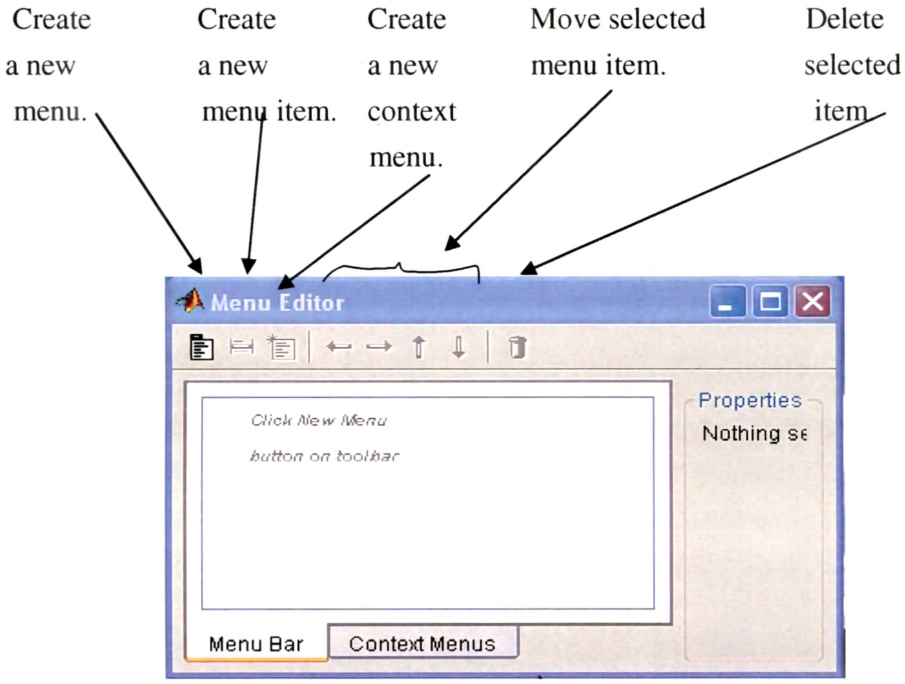



**Figure 8.6 Menu Editor**

✓ **Setting the Tab Order:** The Tab Order Editor displays the components of the GUI in their current tab order. Using The Tab Order Editor we can change the order in which components are selected by tabbing.

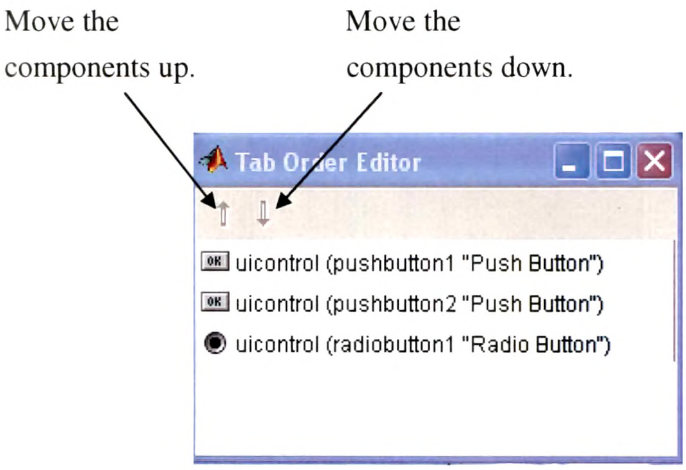



**Figure 8.7 Tab Order Editor**

### 8.4 Programming GUI Components

The section describes the features for programming the GUI components.

#### 8.4.1 GUI M-file

The M-file generated by GUIDE controls the GUI and determines how it responds to a user's actions, such as pressing a push button or selecting a menu item.
**Figure 8.8** depicts execution path of m-file.

The M-file contains all the code needed to run the GUI, including the callbacks for the GUI components. While GUIDE generates the framework for this M-file, we must program the callbacks to perform the desired functions. When GUI is run, the M-file creates a handles structure that contains all the data for GUI objects, such as controls, menus, and axes. The handles structure is passed as an input to each callback. These handles structure can be used to Share data between callbacks and access the GUI data. A code can also be added to:

1. Opening function; executes before the GUI becomes visible to the user
2. Output function; outputs data to the command line
3. Callbacks; so that it is executed each time the user activates corresponding component of the GUI.

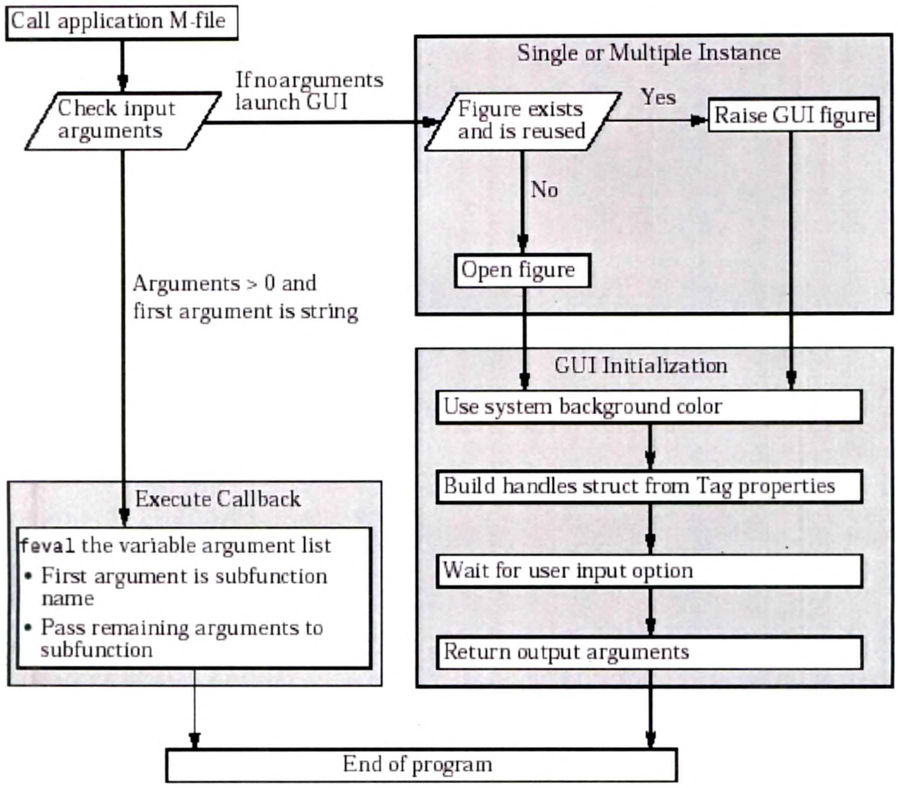



**Figure 8.8 M-file Execution path**

### 8.4.2 GUI Data Management

GUIDE provides a mechanism, called **the handles structure**, for storing and retrieving shared data using the same structure that contains the GUI component handles. The handles structure, which contains the handles of all the components in the GUI, is passed to each callback in the GUI M-file. Therefore, this structure is useful for saving any shared data.

- ✓ **Designing for Cross-Platform Compatibility:** We can use specific property settings to create a GUI that behaves more consistently when run on different platforms. e.g. use different the default font or the default background color or figure character units for different platforms.
- ✓ **Types of Callbacks:** We can define callbacks in addition to that defined by the uicontrol Callback property.
- ✓ **Interrupting Executing Callbacks:** By default, MATLAB allows an executing callback to be interrupted by subsequently invoked callbacks. The GUI programmer can control whether user actions can interrupt executing callbacks or not.

- ✓ **Controlling Figure Window Behavior:** When designing a GUI we need to consider how we want the figure window to behave. The user can appropriately control the behavior for a particular GUI depending on intended use.

After writing the callbacks, GUI can be executed by:

1. Selecting Run from the Tools menu or
2. Clicking the Run button on the GUIDE toolbar.

If we have not saved the GUI recently, GUIDE displays the dialog box shown in **Figure 8.9**.

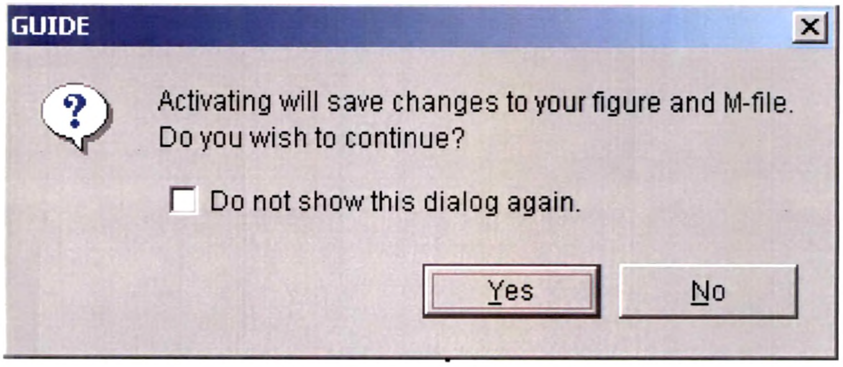



**Figure 8.9 Dialog box to save changes**

Click Yes to save the GUI files. If the directory where we save the GUI is not on the MATLAB path, GUIDE opens the following dialog, giving us the option of changing the current working directory to the directory containing the GUI files, or adding that directory to the MATLAB path as shown in **Figure 8.10**. Click **OK** to change the current working directory, and then GUIDE opens the GUI.

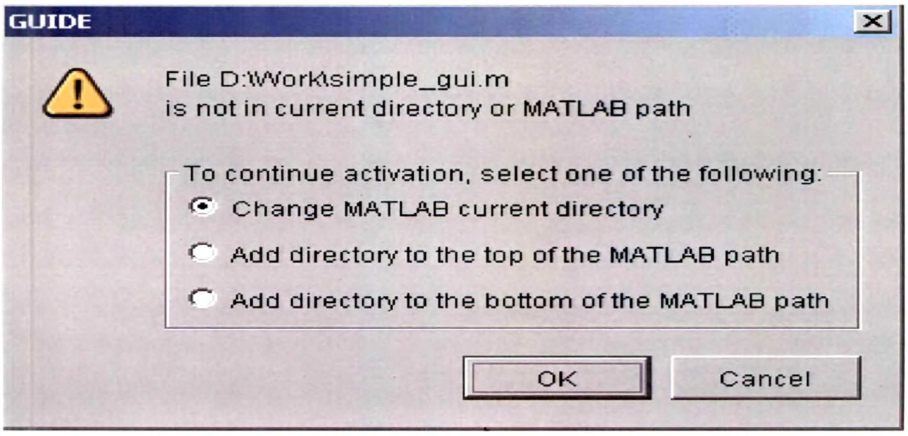



**Figure 8.10 Dialog box to change the current directory during execution**

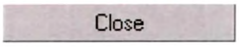
## 8.5 Design of GUI for Signal Processing

The proposed models and training algorithms described in chapters 5, 6 and 7 are implemented in MATLAB 6.5.

**Table 8.1 Detailed Summary of various files used**

| Operation | Network Configuration | Training Algorithm | Link File Name |
|---|---|---|---|
| Filtering | Hopfield (LS) | LS | hopfield_filt.m<br>hopfield_filt.fig |
| | Hopfield (RLS) | RLS | hopfield_filt_rls.m<br>hopfield_filt_rls.fig |
| | MLP ANN | Back Propagation | filtering_bkp.m<br>filtering_bkp.fig |
| | RBF ANN | RBF Training | filtering_rbfnn.m<br>filtering_rbfnn.fig |
| Compression | MLP ANN | Back Propagation | signal_comp_bkp.m<br>signal_comp_bkp.fig |
| | VQ ANN (Sample Difference) | Competitive Learning | compression_vqnn_diff.m<br>compression_vqnn_diff.fig |
| | VQ ANN (Absolute Samples) | Competitive Learning | compression_vqnn_no_diff.m<br>compression_vqnn_no_diff.fig |
| | MLP ANN | Extended Back Propagation | signal_comp_ext_bkp.m<br>signal_comp_ext_bkp.fig |
| Detection | VQ ANN | Competitive Learning | detection_vqnn.m<br>detection_vqnn.fig |

A figure and script file ( selection.fig / selection.m ) is developed. On execution of script file ( selection.m ) or a .fig file ( selection.fig ) through GUIDE will generate main screen shown in **Figure 8.11** which allows user to select the signal processing operation, network architecture as well as training algorithm. When any button is pressed corresponding link file is activated and executed.

A facility is provided to display input used for training or testing. 

button is provided for exiting from the signal processing environment.

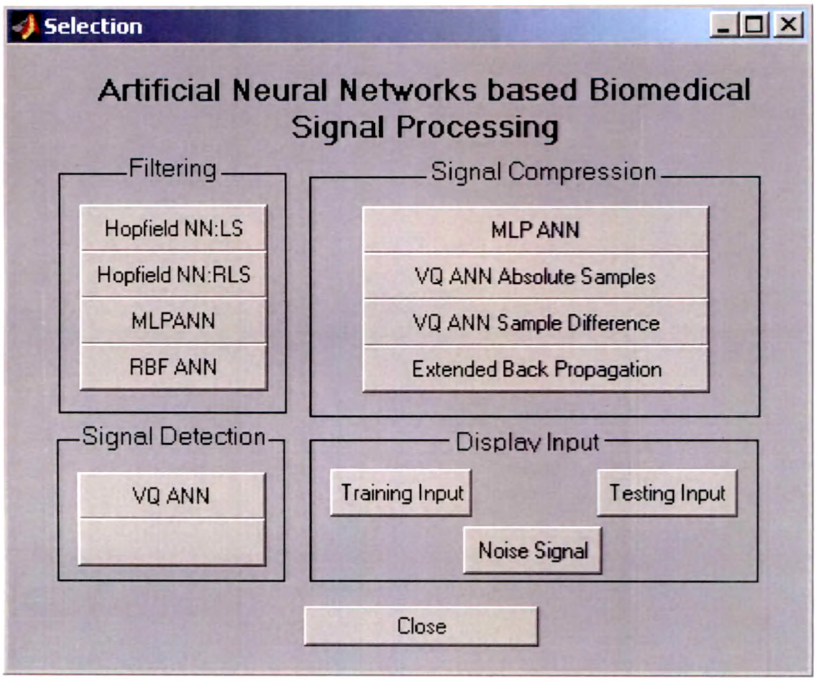



**Figure 8.11 Graphical User Interface: Main Screen for selection**

### 8.5.1 Filtering

Three network models are proposed for filtering, viz. Hopfield ANN, MLP ANN, RBF ANN. Pressing Hopfield NN:LS or Hopfield NN:RLS button will generate GUI of **Figure 8.12** and **Figure 8.13** respectively.

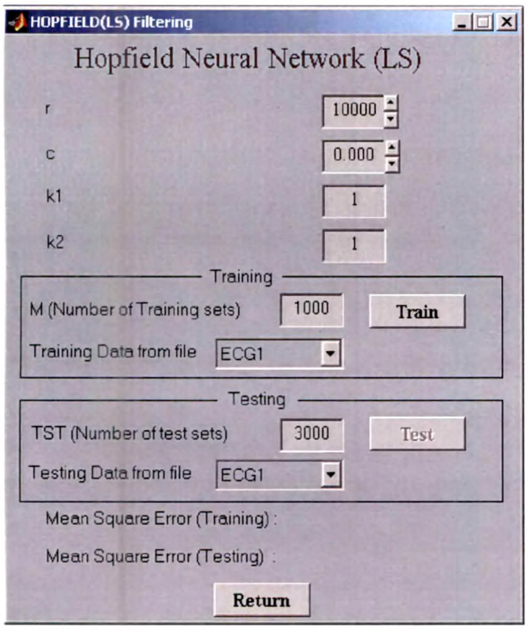



**Figure 8.12 GUI for Filtering using Hopfield NN(LS Algorithm)**

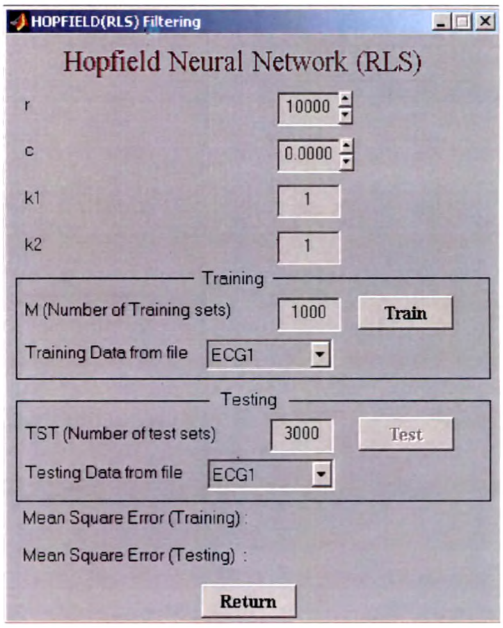



**Figure 8.13 GUI for Filtering using Hopfield NN(RLS Algorithm)**

Both the GUIs are same except that they will link to the training algorithm specified by the user. The default network parameters: Resistance (r), Capacitance (c), Multipliers (k1, k2), Number of Training Sets, Number of Test Sets are specified. User my change their value if he wishes, by entering new data in the space provided in the GUI.

Two default ECG data files named as ECG1 and ECG2 are provided for training as well as testing. Once the user selects the file name, **Train** button is enabled. On pressing **Train**, network will be trained using selected algorithm. On completion of training, Test button will be activated as **Test**. Pressing **Test** button testing will be done (**Figure 8.14** and **Figure 8.15**). The waveforms during training as well as testing described in **chapter 5** will be generated. The performance of the operation is measured in terms of Mean Square Error (MSE). The value of MSE during training and testing will be displayed on GUI screen. On pressing **Return**, the GUI is closed and main GUI will appear.

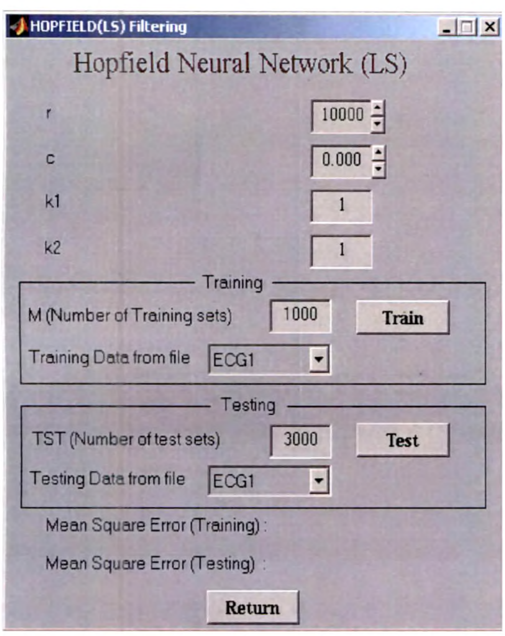



**Figure 8.14 GUI for Filtering using Hopfield NN(LS Algorithm) (with Test activated)**

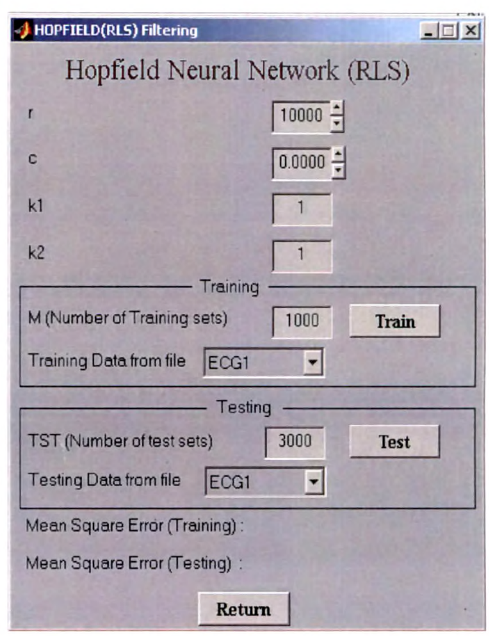



**Figure 8.15 GUI for Filtering using Hopfield NN(RLS Algorithm) (with Test activated)**

On pressing MLPANN, GUI of **Figure 8.16** will appear. The default values of the parameters are provided. However, user may change these values. Number of input layer nodes is decided by phase difference between noise signal and noise signal mixed up with information signal.

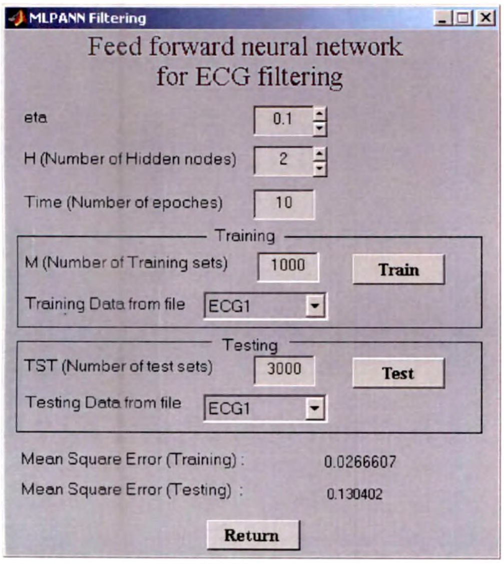



**Figure 8.16 GUI for Filtering using MLP ANN (Values of MSE displayed)**

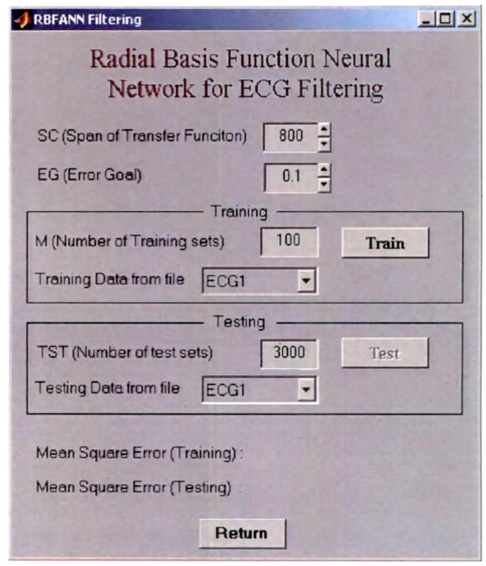



**Figure 8.17 GUI for Filtering using RBF ANN**

On pressing RBF ANN in main GUI, **Figure 8.17** will appear. This also allows setting parameters, select filename for training / testing. It displays results as well as plots during training / testing. It is possible for the user to use ECG files other than default. He can select 'Other' from the items of the popup menu. On selecting 'Other', **Help** will be activated (**Figure 8.18**) On pressing **Help** user may enter his own file name (**Figure 8.19)** which can be used for training or testing as per the selection through the same help window.

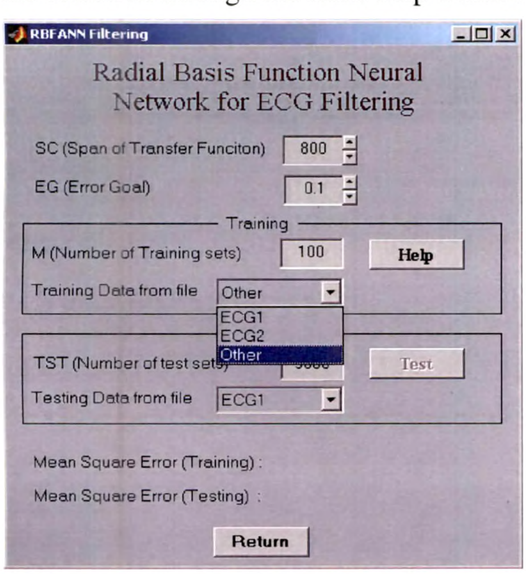



**Figure 8.18 GUI for Filtering using RBF ANN ( 'Other' Selection )**

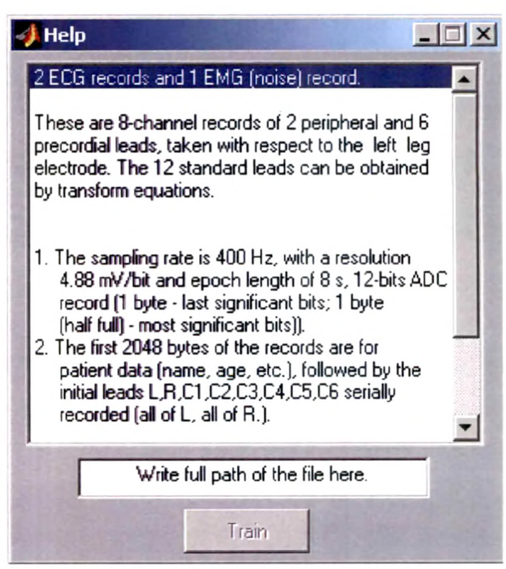



**Figure 8.19 GUI for Help to select other file and to carry out training**

### 8.5.2 Compression

The network models are provided with environment viz. MLP ANN and Vector Quantization ANN (VQ ANN). Pressing any one button on the screen will select the model and training algorithm and corresponding GUI will appear.

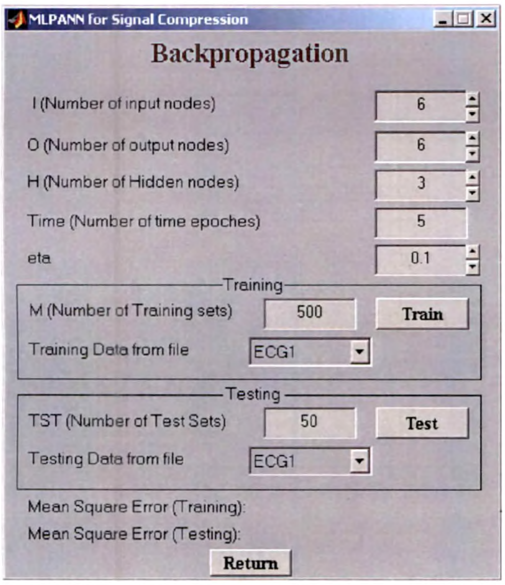



Figure 8.20 GUI for Compression using MLP ANN (Back Propagation)

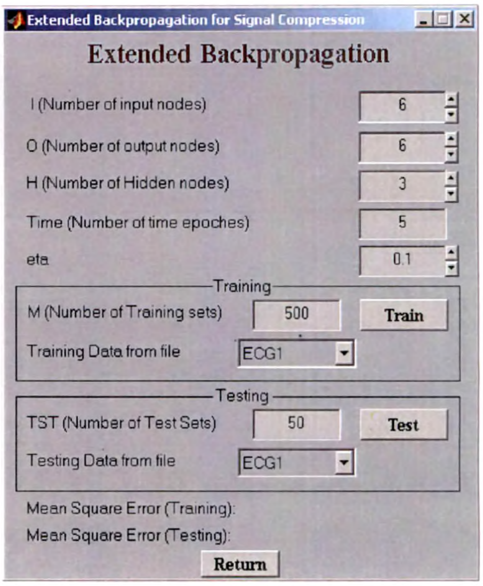



Figure 8.21 GUI for Compression using MLP ANN (Extended Back Propagation)

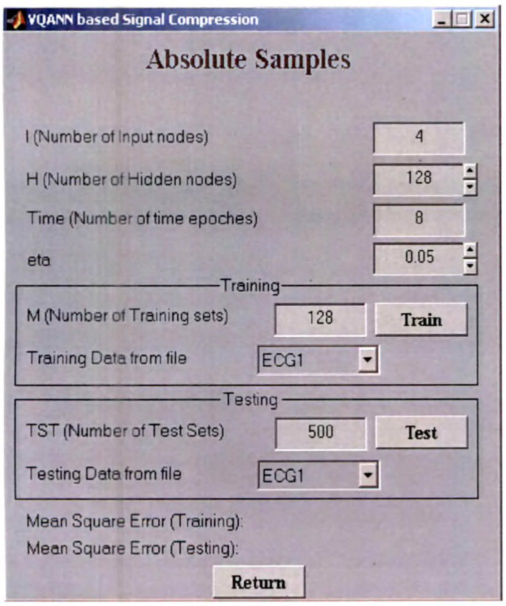



Figure 8.22 GUI for Compression using VQ ANN ( Absolute Sample )

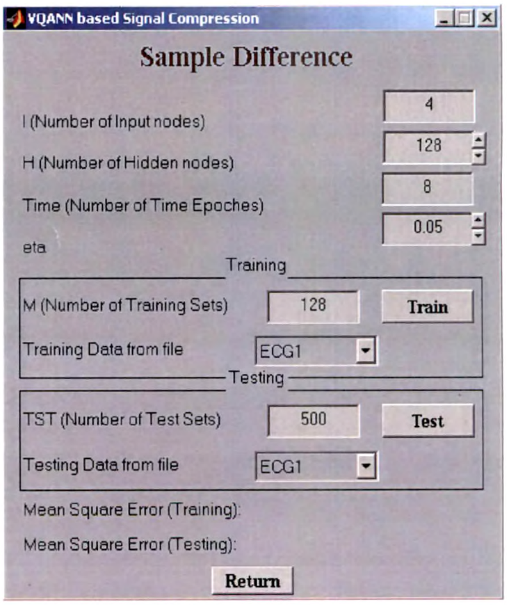



Figure 8.23 GUI for Compression using VQ ANN ( Sample Difference )

Like previous GUIs, the default parameters and files are provided. User may modify the values.

### 8.5.3 Detection

For VQ ANN based detection, parameters like number of input nodes, time epochs and eta are selectable (Figure **8.24**). Here number of competing nodes is not kept selectable because generally number of competing nodes are kept same as number of training sets.

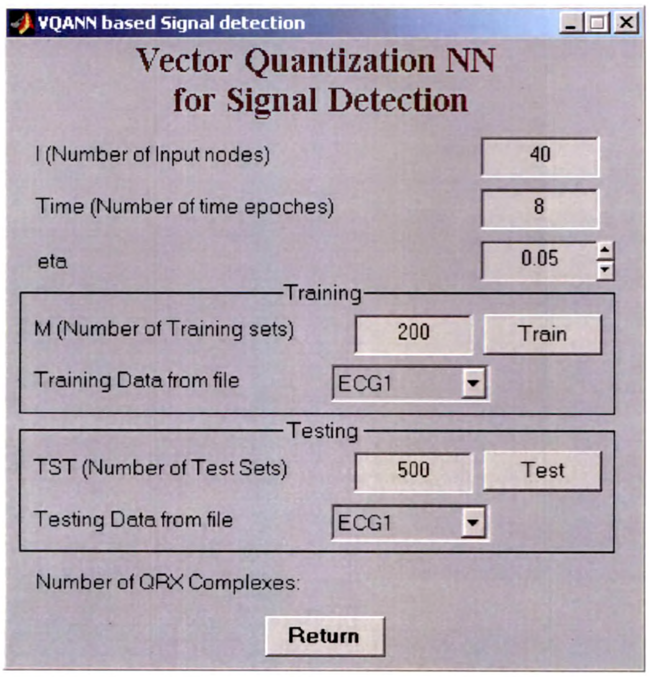



**Figure 8.24 GUI for Detection using VQ ANN**