Fuzzy Logic Control Using Matlab
Part II

Naveed Ahmed, FAST-NUCES, Lahore.

This 2-day workshop was held under the auspices of the Khwarzimic Science Society at the University of Lahore in April 2003. Full details and copies of this document can be obtained from http://www.khwarzimic.org.

Building Systems with the Fuzzy Logic Toolbox

Although it's possible to use the Fuzzy Logic Toolbox by working strictly from the command line, it is easier to build the system using graphical user interface. There are five primary GUI tools for building, editing, and observing fuzzy inference systems in the Fuzzy Logic Toolbox: the Fuzzy Inference System or FIS Editor, the Membership Function Editor, the Rule Editor, the Rule Viewer, and the Surface Viewer.

The FIS Editor handles the high level issues for the system: How many input and output variables? What are their names? The Membership Function Editor is used to define the shapes of all the membership functions associated with each variable. The Rule Editor is for editing the list of rules that defines the behavior of the system. The last two GUIs are used for looking at, as opposed to editing, the FIS. The Rule Viewer is a MATLAB-based display of the fuzzy inference diagram. Used as a diagnostic, it can show which rules are active, or how individual membership function shapes are influencing the results. Surface Viewer can display how one of the outputs depends on any one or two of the inputs- that is, it generates and plots an output surface map of the system.

The five principal GUI editors all exchange information, if appropriate. Any one of them can read and write both to the workspace and to the disk. For any fuzzy inference system, any or all of these five editors may be open. If more than one of these editors is open for a single system, the various GUI windows are aware of the existence of the others, and will, if necessary, update related windows.

Getting Started

The Basic Tipping Problem. *Given a number between 0 and 10 that represents the quality of service at a restaurant (where 10 is excellent), what should the tip be?*

The starting point is to write down the three rules for tipping, based on our personal experience.

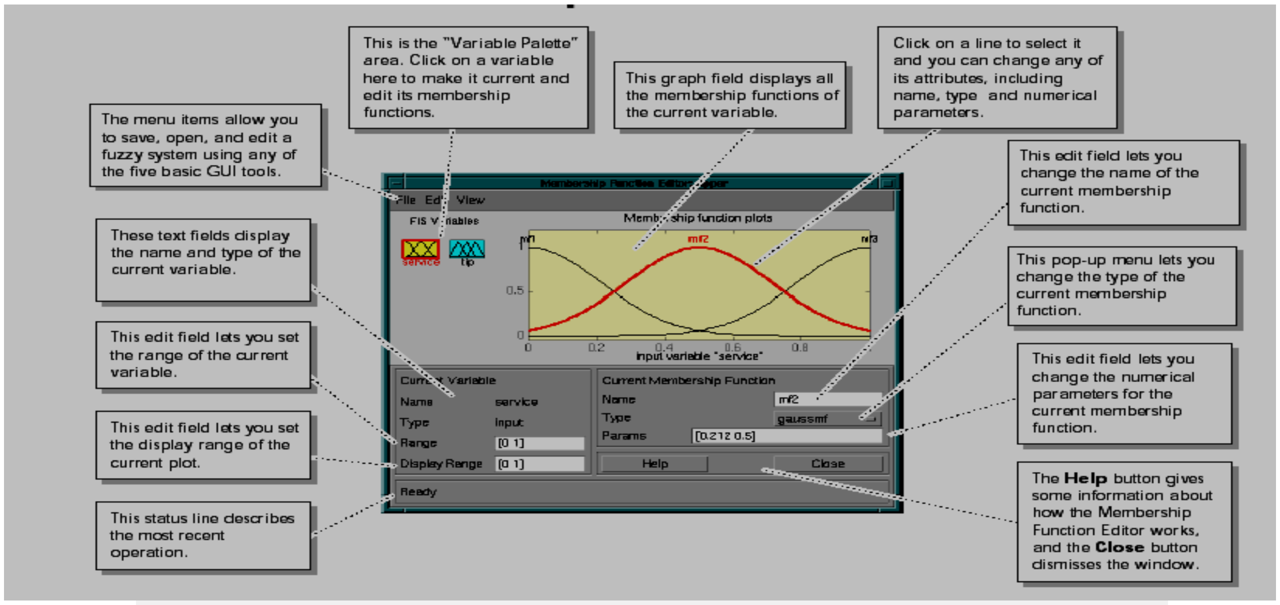
1. *If the service is poor then tip is cheap*
2. *If the service is good then tip is average*
3. *If the service is excellent then tip is generous*

Now that we have got the rules, we can start working with the GUI tools.

The FIS Editor

When creating a new fuzzy inference system for scratch, the place to start is the FIS Editor. To do that type *fuzzy*, this will call up a window that acts as the high-level view of a FIS. The system that is displayed is a default "start-up" system, since we didn't specify any particular system. You can either build a new system from scratch or you can load the already built system by typing *fuzzy tipper1*.

The FIS Editor displays general information about a fuzzy inference system. Default system already has one input and one output. We will change their names according to our tipping problem.


The menu items allow you to save, open, and edit a fuzzy system using any of the five basic GUI tools.

This is the "Variable Palette" area. Click on a variable here to make it current and edit its membership functions.

This graph field displays all the membership functions of the current variable.

Click on a line to select it and you can change any of its attributes, including name, type and numerical parameters.

This edit field lets you change the name of the current membership function.

These text fields display the name and type of the current variable.

This pop-up menu lets you change the type of the current membership function.

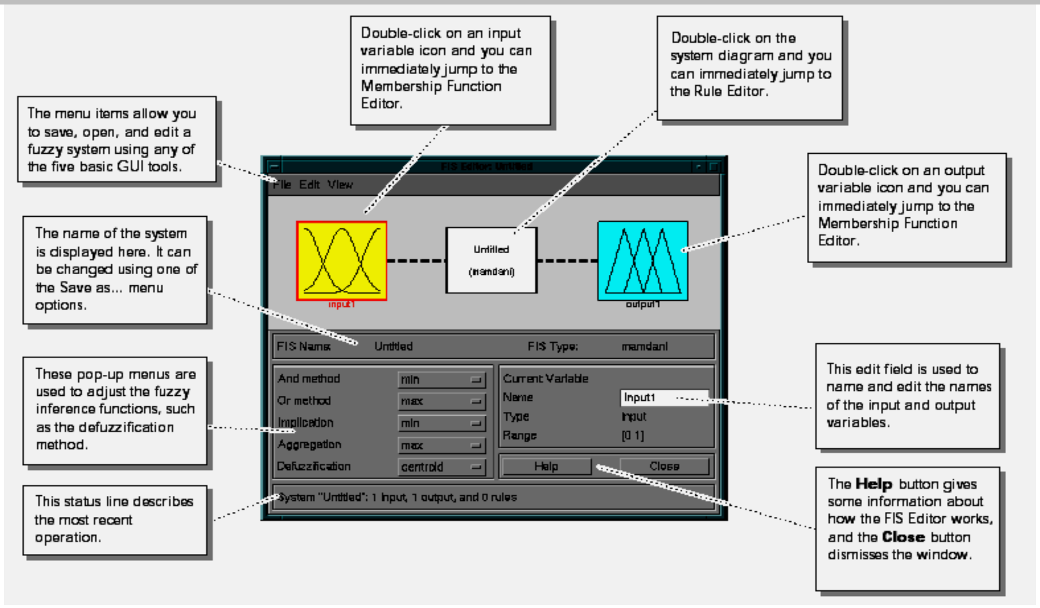
This edit field lets you set the range of the current variable.

This edit field lets you change the numerical parameters for the current membership function.

This edit field lets you set the display range of the current plot.

This status line describes the most recent operation.

The **Help** button gives some information about how the Membership Function Editor works, and the **Close** button dismisses the window.


Double-click on an input variable icon and you can immediately jump to the Membership Function Editor.

Double-click on the system diagram and you can immediately jump to the Rule Editor.

The menu items allow you to save, open, and edit a fuzzy system using any of the five basic GUI tools.

Double-click on an output variable icon and you can immediately jump to the Membership Function Editor.

The name of the system is displayed here. It can be changed using one of the Save as... menu options.

These pop-up menus are used to adjust the fuzzy inference functions, such as the defuzzification method.

This edit field is used to name and edit the names of the input and output variables.

This status line describes the most recent operation.

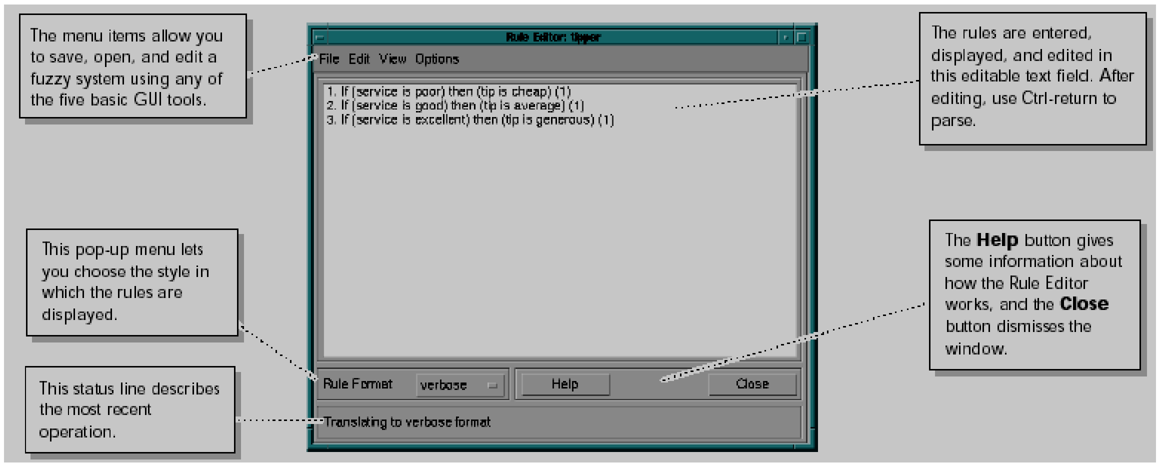
The **Help** button gives some information about how the FIS Editor works, and the **Close** button dismisses the window.

The next thing to do is to define the membership functions associated with each of the variables. To do this, we need to open up the Membership Function Editor by pulling down the **View** menu item and selecting **Edit Membership Functions….**

The Membership Function Editor

The Membership Function Editor shares some features with the FIS Editor. In fact, all of the five basic GUI tools have similar menu options, status lines, and **Help** and **Close** buttons. The Membership Function Editor is the tool that lets you display and edit all of the membership functions for the entire fuzzy inference system, including both input and output variables.

The Rule Editor



The menu items allow you to save, open, and edit a fuzzy system using any of the five basic GUI tools.

The rules are entered, displayed, and edited in this editable text field. After editing, use Ctrl-return to parse.

1. If (service is poor) then (tip is cheap) (1)
2. If (service is good) then (tip is average) (1)
3. If (service is excellent) then (tip is generous) (1)

This pop-up menu lets you choose the style in which the rules are displayed.

The **Help** button gives some information about how the Rule Editor works, and the **Close** button dismisses the window.

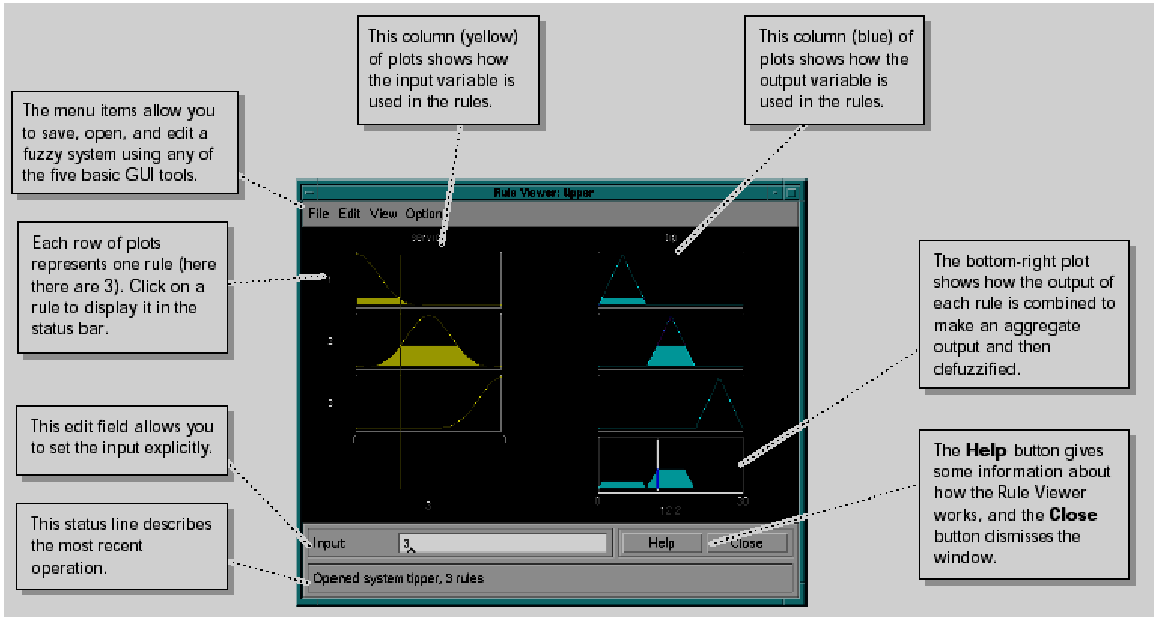
This status line describes the most recent operation.

Translating to verbose format

The Rule Editor contains a large editable text field for displaying and editing rules. You can use Add Rule Button to add new rules to the fuzzy inference system.

After the system has been completely defined: you've got the variables, membership functions, and rules necessary to calculate tips. It would be nice, at this point, to look at a fuzzy inference diagram like one at the last page of previous lecture notes and verify that every thing is behaving the way it should. This is exactly the purpose of Rule Viewer. From the **View** menu, select **View Rules…**.

The Rule Viewer



This column (yellow) of plots shows how the input variable is used in the rules.

This column (blue) of plots shows how the output variable is used in the rules.

The menu items allow you to save, open, and edit a fuzzy system using any of the five basic GUI tools.

Each row of plots represents one rule (here there are 3). Click on a rule to display it in the status bar.

The bottom-right plot shows how the output of each rule is combined to make an aggregate output and then defuzzified.

This edit field allows you to set the input explicitly.

The **Help** button gives some information about how the Rule Viewer works, and the **Close** button dismisses the window.

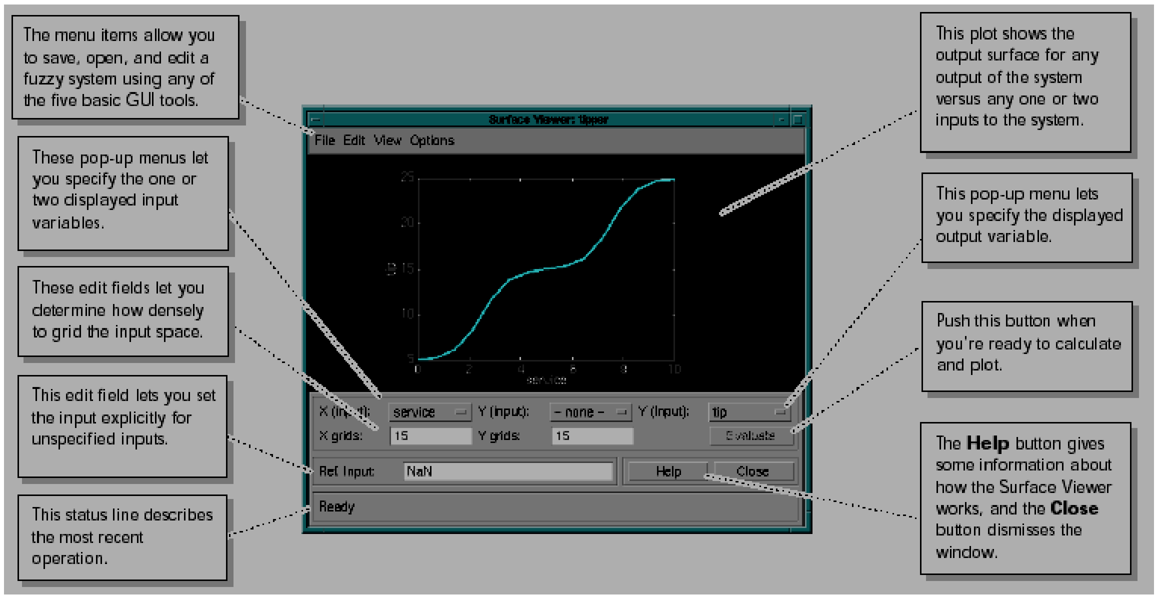
This status line describes the most recent operation.

Opened system tipper, 3 rules

The Rule Viewer displays a road map of the whole fuzzy inference process. Each rule is a row of plots, and each column is a variable. So the first column of plots (the three yellow plots) shows the membership functions references by the antecedent, or if-part, of each rule. The second column of plots (the three blue plots) shows the membership functions references by the consequent, or then-part of each rule. There is a yellow index line across the input variable plots that you can move left and right by clic king and dragging with the mouse. This changes the input value. When you release the line, a new calculation is performed, and you can see the whole fuzzy inference process take place before your eyes.

Finally the aggregation occurs down the second column , and the resultant aggregate plot is shown in the single plot to be found in the lower right corner of the plot field. The thick line passing through the aggregate fuzzy set shows the defuzzified output value. With Rule Viewer you can take in the whole fuzzy inference process in one sweeping view. The Rule Viewer is very good, for example, at showing how the shape of certain membership functions is influencing the overall results. The Rule Viewer shows one calculation at a time and in great detail. If you want to see the entire output surface of your system, that is the entire span of the output set based upon the entire span of input set, you need to open up surface viewer. You can open it by selecting **View Surface**…. from the **View** menu.

The Surface Viewer


The menu items allow you to save, open, and edit a fuzzy system using any of the five basic GUI tools.

These pop-up menus let you specify the one or two displayed input variables.

These edit fields let you determine how densely to grid the input space.

This edit field lets you set the input explicitly for unspecified inputs.

This status line describes the most recent operation.

This plot shows the output surface for any output of the system versus any one or two inputs to the system.

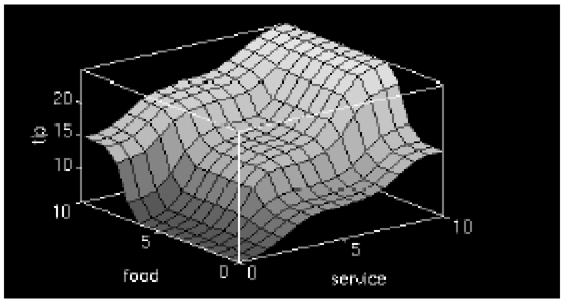
This pop-up menu lets you specify the displayed output variable.

Push this button when you're ready to calculate and plot.

The **Help** button gives some information about how the Surface Viewer works, and the **Close** button dismisses the window.

X (input): service Y (input): - none - Y (input): tip
X grids: 15 Y grids: 15
Ref. Input: NaN
Ready

Upon opening the Surface Viewer, we are presented with a two dimensional curve that represents the mapping from service quality to tip amount. Since this is a one-input one-output case, we can see the entire mapping in one plot. Two input one-output systems also work well, as they generate three-dimensional plots that MATLAB can adeptly manage. But when we move beyond three dimensions overall, we start to encounter trouble displaying the results. Accordingly, the Surface Viewer is equipped with pop-up menus that let you select any two inputs and any one output for plotting. Pushing the **Evaluate** button initiates the calculation, and the plot comes up soon after the calculation is complete.
If you have got two inputs and one output the output surface would be like the figure on right.



FIS Evaluation in Matlab
To evaluate the output of a fuzzy system for a given input, use the function evalfis. Write the following code upto evalfis and pres enter, to get answer.

```
a = readfis('tipper')
evalfis([1 2], a)
ans = 5.5586
```

This function can also be used for multiple collections of inputs, so each row of the input matrix is a different input vector. By doing multiple evaluations at once, you get a tremendous boost in speed.

```
evalfis([3 5; 2 7], a)
ans =12.2184
```

```
    7.7885
```

FIS Evaluation in C

In the fuzzy/fuzzy directory of the toolbox, you can find two C files, fismain.c and fis.c, which are provided as the source codes for a stand-alone fuzzy inference engine. The stand-alone fuzzy inference engine can read a FIS file and an input data file to perform fuzzy inference directly, or it can be embedded in other external applications.