FLORIDA INTERNATIONAL UNIVERSITY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

EEL-6681 FUZZY SYSTEMS

A Practical Guide to Model Fuzzy Inference Systems using MATLAB and Simulink

By Pablo Gomez

Miami, Summer 2004

OBJECTIVE

The purpose of this document is to provide a guide to using MathLab®, the Fuzzy Logic Toolbox® and Simulink® software to model and simulate Fuzzy Inference Systems.

This is a practical guide that shows how to implement a hypothetical Fuzzy Control System but makes no effort in explaining the theory behind it. For Fuzzy Logic theory the reader is referred to the handouts given in class or any good book on the subject.

THE MATLAB ENVIRONMENT

The following figure shows the MATLAB environment for modeling and simulating a Fuzzy Inference System. There are 3 software modules that exchange information:

- MATLAB® and its Workspace variables and functions
- The Fuzzy Logic Toolbox® to model the Fuzzy Inference System (fis)
- Simulink® to simulate a Fuzzy Control System

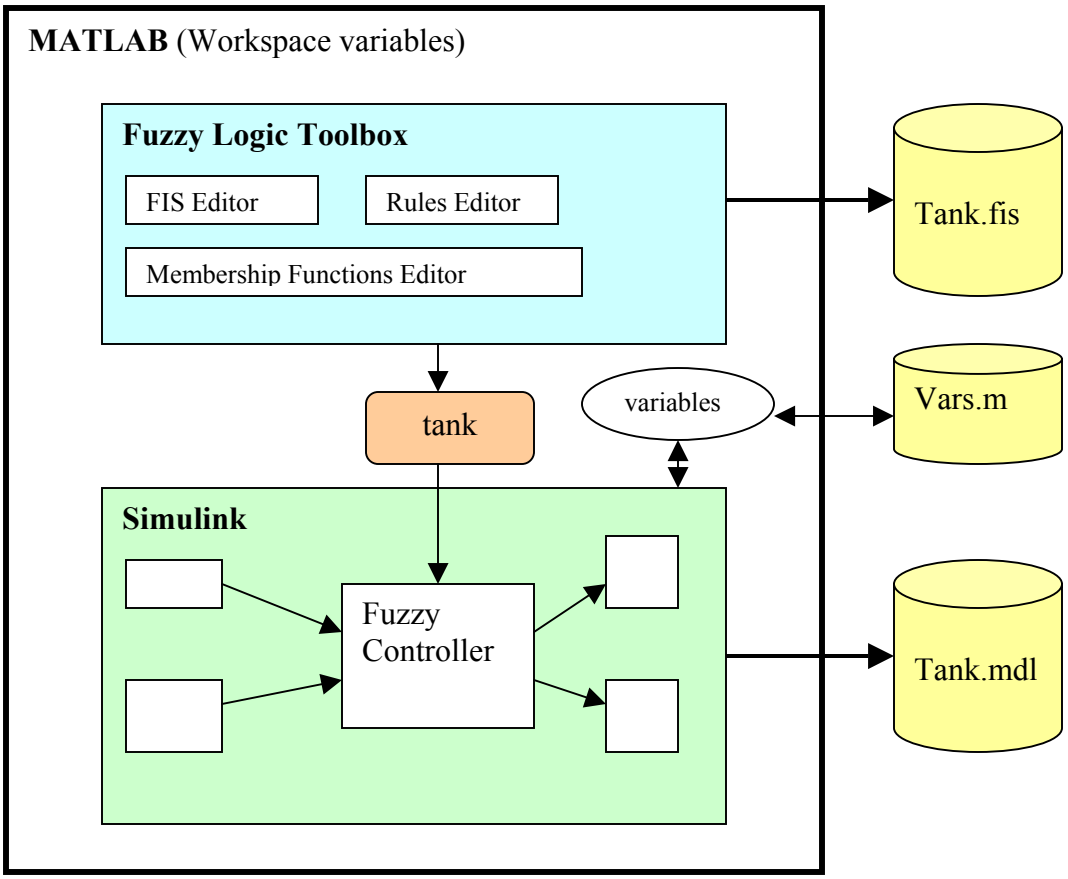


Figure 1. The MATLAB Environment

The general steps to create a Fuzzy System and simulate it are:

- Run MATLAB

- Invoke the Fuzzy Logic Toolbox by typing the command **FUZZY** from the MATLAB prompt
    - Use the FIS Editor to create the Fuzzy model: name (in this case *tank*), model type (Mamdani or Sugeno), Input Variables and Output Variables
    - Use the Membership Functions Editor to define the MF's of each input or output variable
    - Use the Rules Editor to define the Fuzzy System Rules
    - Save the model to a file (*tank.fis* in the example)
    - Use the Rules Viewer to verify the model
    - Export the model to a Workspace variable (*tank* in this example)

- Invoke Simulink by typing the command **SIMULINK** from the MATLAB prompt
    - Create a new Simulink Model
    - Add a Fuzzy Controller Block to the model and set its parameter to the FIS model exported to the workspace (*tank* in this example)
    - Connect the Fuzzy Controller inputs and outputs to other Simulink blocks accordingly
    - Set the simulation parameters
    - Run the simulation
    - Save the model to a file (*tank.mdl* in the example)
    - Optionally save simulation variables to the workspace and/or to an external file

THE SAMPLE FUZZY CONTROLLER

The hypothetical system that will be modeled and simulated is a tank of water where 2 chemicals A and B are mixed to obtain a product. The amount of chemical is measured in grams, and is a quantity from 0 to 1000 for each component. The Fuzzy Controller has to set the water level (in centimeters from 10 to 100) and its temperature (from 15 ºC to 100 ºC). This is a 2-input 2-output fuzzy controller.

MODELLING WITH THE FUZZY LOGIC TOOLBOX

To create the Fuzzy System we use the various editors that come with the Fuzzy Logic Toolbox. These are the specific steps:

From the MATLAB prompt type in the command **FUZZY**. The following figure shows the initial screen of the FIS editor with an empty model.

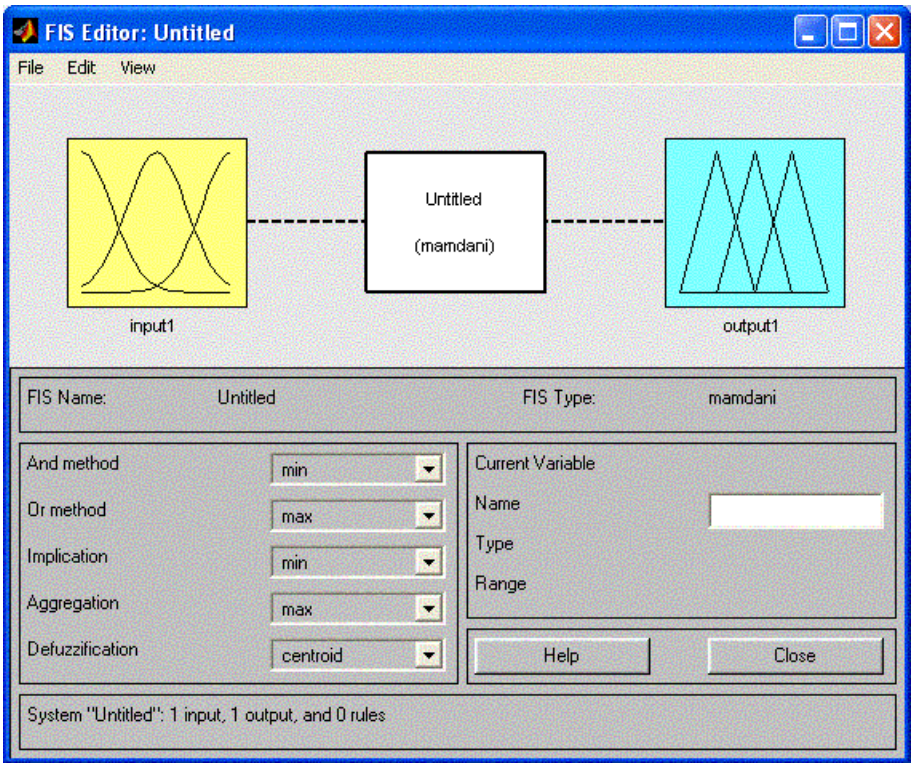


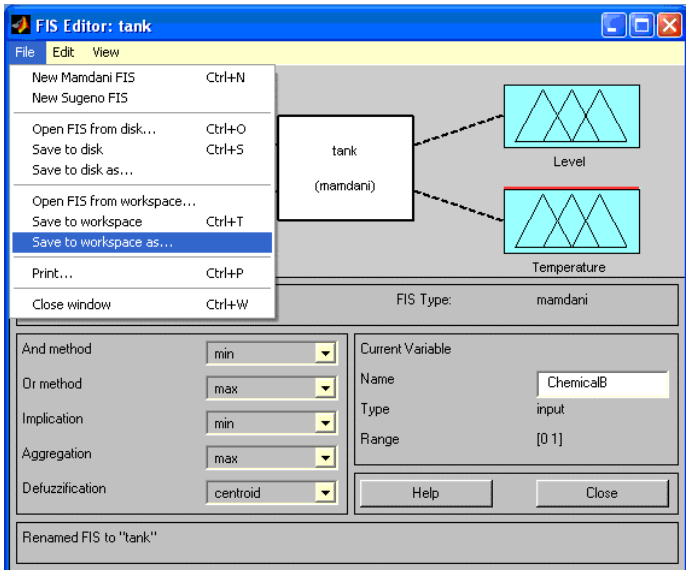
Figure 2. Initial FIS Editor Window

The FIS Editor

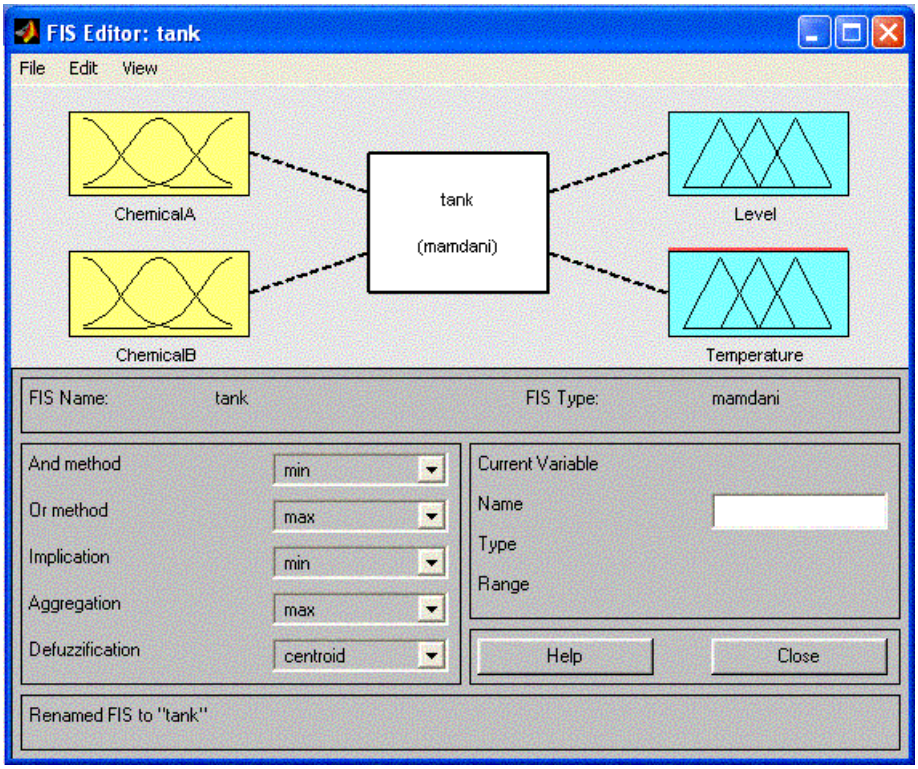
The generic untitled FIS Editor opens, with one input, labeled input1, and one output, labeled output1. For this example, we will construct a two-input, two-output system, so go to the **Edit** menu and select **Add input**. A second yellow box labeled input2 will appear. The two inputs we will have in our example are **ChemicalA** and **ChemicalB**. Our outputs are Level and Temperature. We'd like to change the variable names to reflect that, though:

Click once on the left-hand (yellow) box marked input1 (the box will be highlighted in red). In the white edit field on the right, change input1 to ChemicalA and press Enter. Repeat the same procedure to change input1 to ChemicalB.
Click once on the right-hand (blue) box marked output1. In the white edit field on the right, change output1 to Level. Repeat this step to change output2 to Temperature.

From the File menu select “Save to workspace as...” as shown below



and in the pop up window type in **tank** and click on the OK button. The model should look like this at this time:



Leave the inference options in the lower left in their default positions for now. You've entered all the information you need for this particular GUI. Next define the membership

functions associated with each of the variables. To do this, open the Membership Function Editor. You can open the Membership Function Editor in one of three ways:

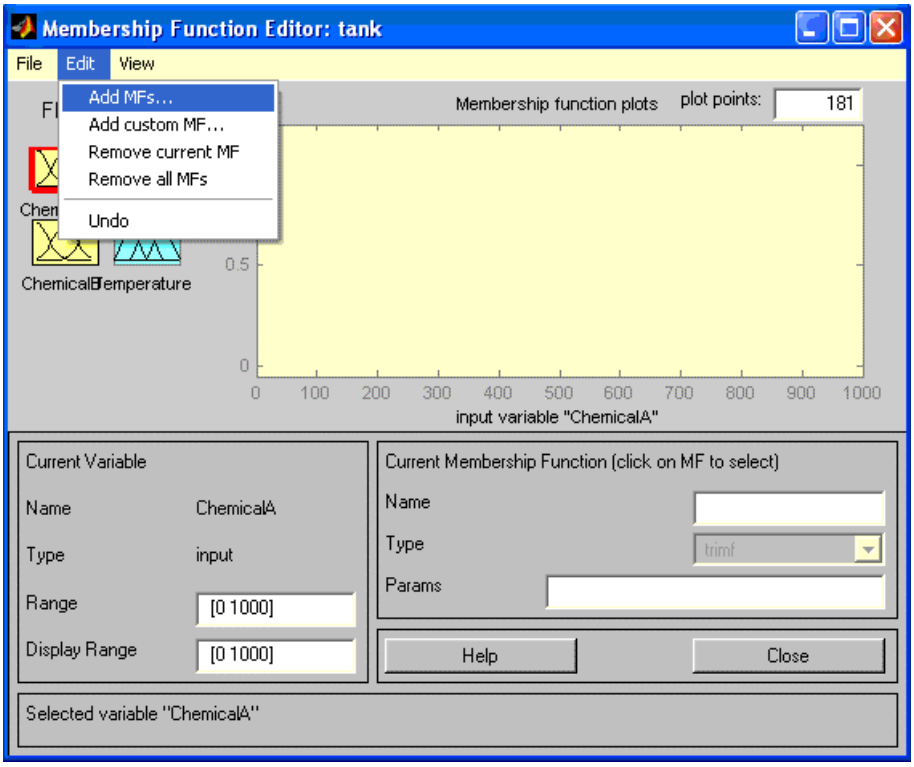
- Pull down the View menu item and select Edit Membership Functions....
- Double-click on the icon for one of the input or output variables
- Type mfedit at the command line.

The Membership Function Editor

When you open the Membership Function Editor to work on a fuzzy inference system that does not already exist in the workspace, there are not yet any membership functions associated with the variables that you have just defined with the FIS Editor.

On the upper left side of the graph area in the Membership Function Editor is a "Variable Palette" that lets you set the membership functions for a given variable. To set up your membership functions associated with an input or an output variable for the FIS, select an FIS variable in this region by clicking on it.

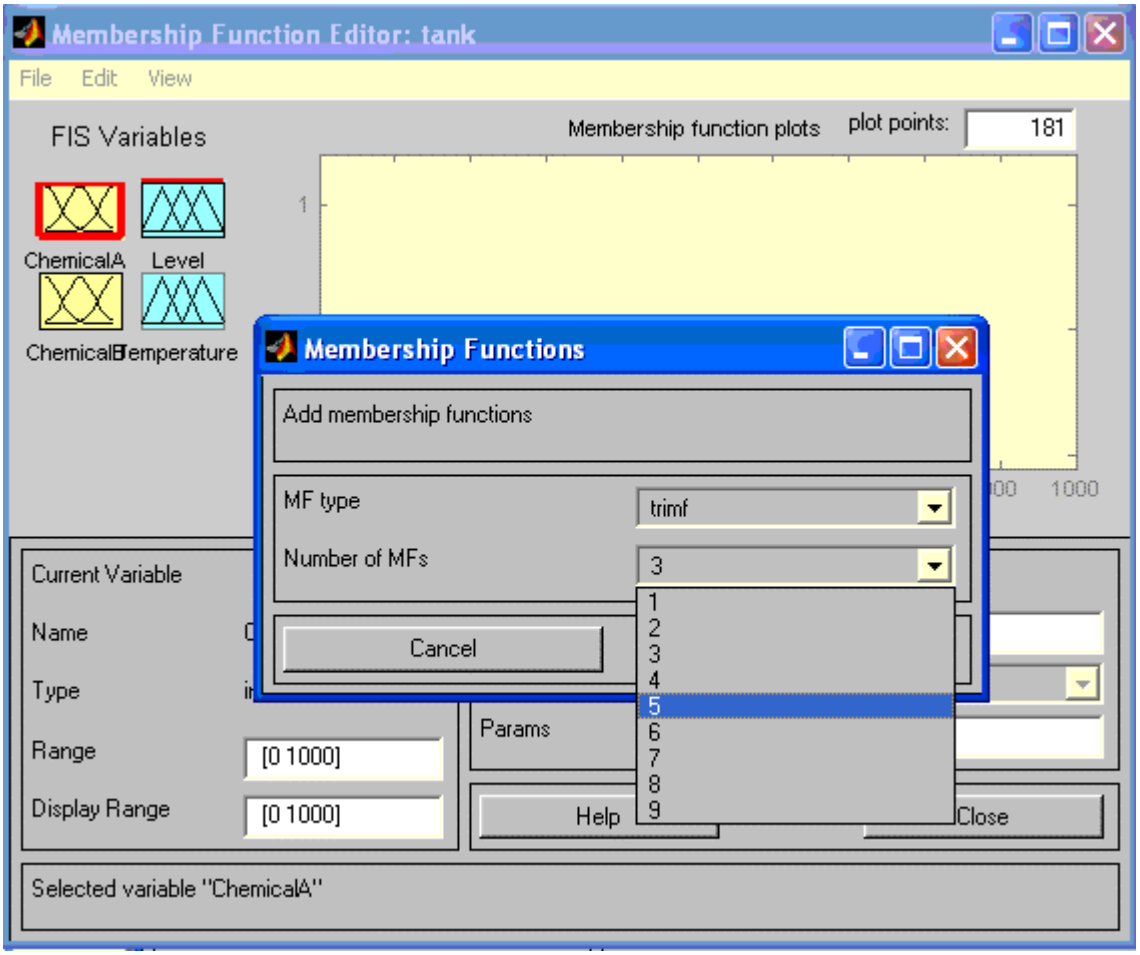
Next select the **Edit** pull-down menu, and choose **Add MFs....** A new window will appear, which allows you to select both the membership function type and the number of membership functions associated with the selected variable.



The process of specifying the input membership functions for this two input fuzzy problem is as follows:

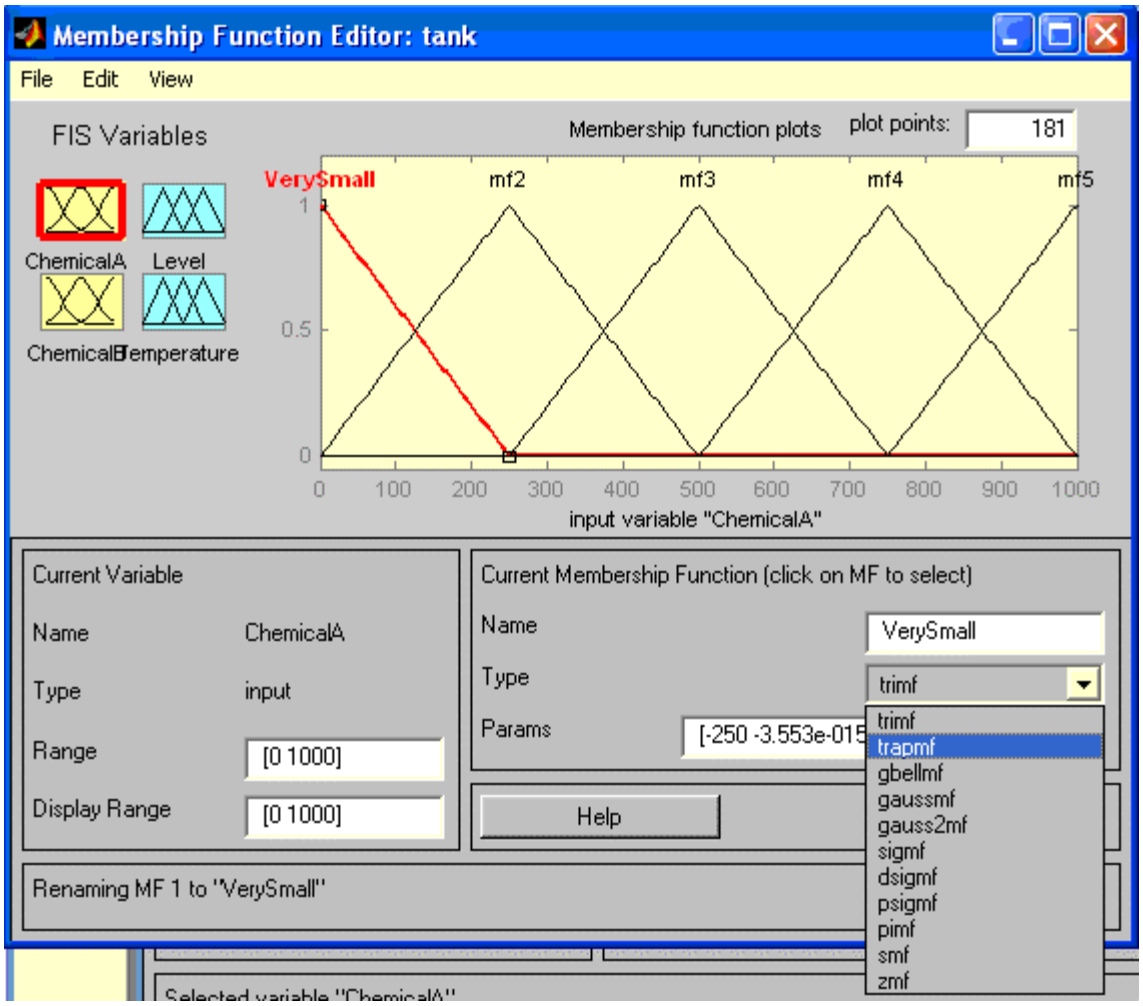
1. Select the input variable, ChemicalA, by double-clicking on it. Set both the Range and the Display Range to the vector [0 1000].

2. Select **Add MFs...** from the **Edit** menu. The window below pops open



3. Use the pull-down tab to choose **trimf** for **MF Type** and 5 for **Number of MFs**. This adds three triangular curves to the input variable ChemicalA.

The membership Editor for ChemicalA will be displayed as follows:

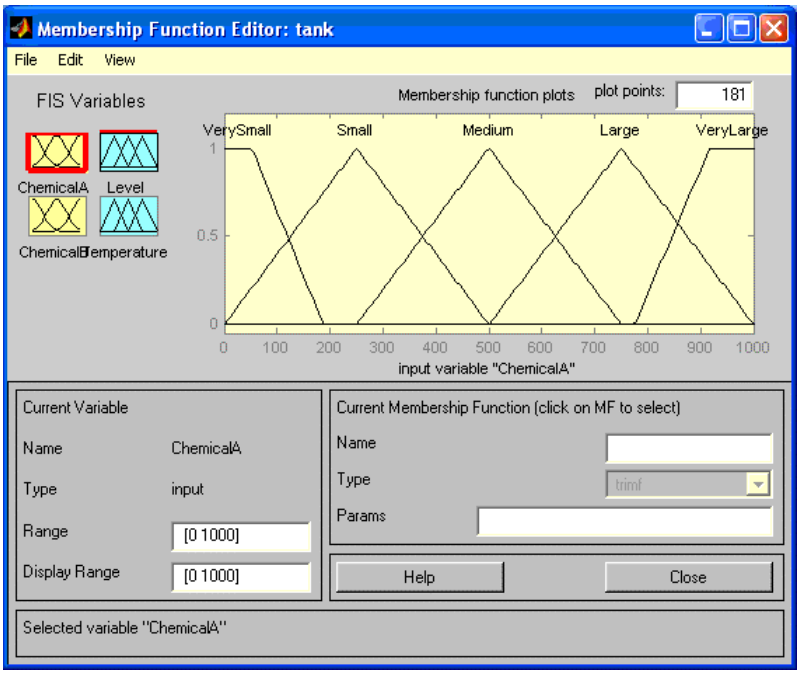


The GUI initially shows 5 triangular MF labeled MF1 through MF1. To change the names and attributes of the MF, click on the line describing the MF (it turns red) and use one of these two methods:

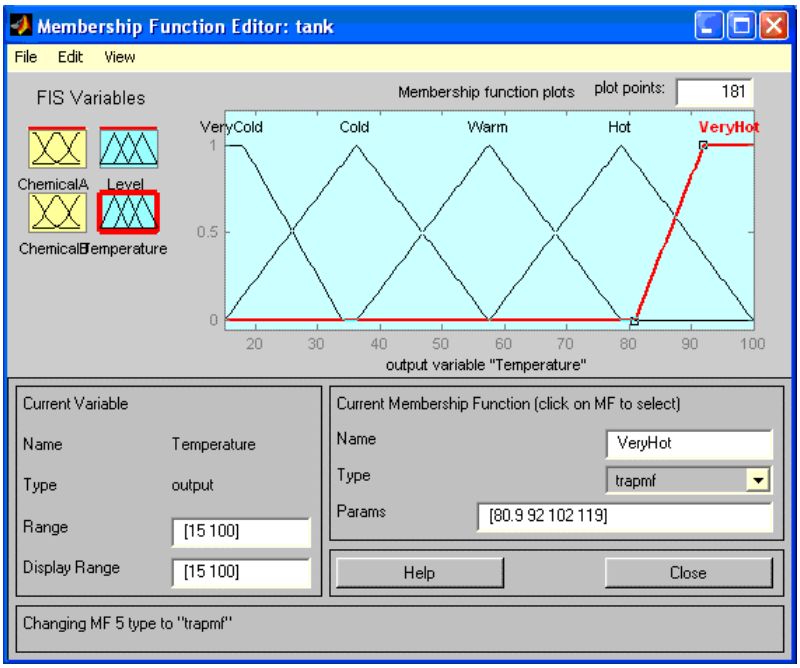
- The selected membership function can also be tagged for dilation or contraction by clicking on the small square drag points on the membership function, and then dragging the function with the mouse toward the outside, for dilation, or toward the inside, for contraction. This will change the parameters associated with that membership function.
- Enter the exact numeric points on the **Params** text box.

For ChemicalA, we need to change MF1 and MF5 to trapezoidal, by clicking on the Type list box, as shown on the previous figure.

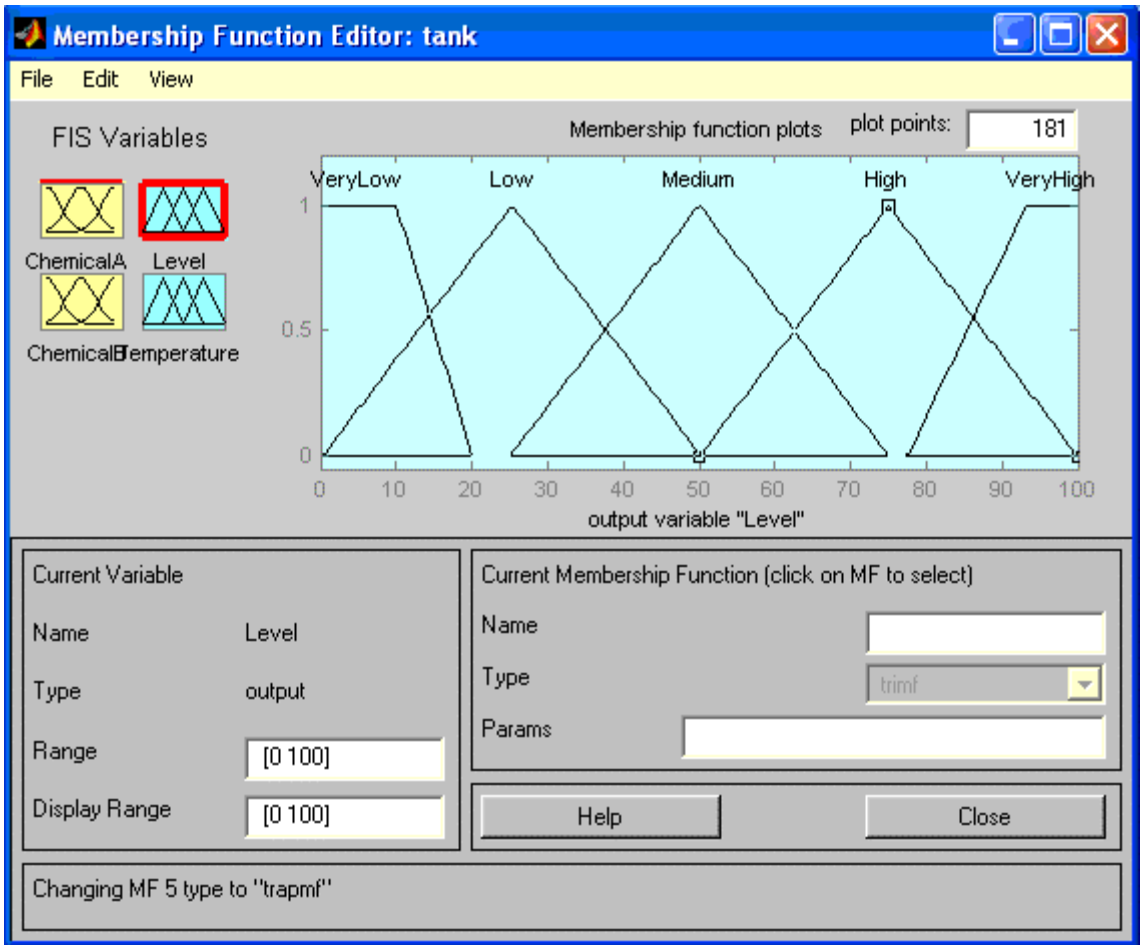
Use the above procedure to change the definitions of all 5 MFs until the ChemicalA variable looks as shown in the following figure.



Repeat the same procedure for input variable **ChemicalB** and for output variables **Temperature** and Water **Level**. The resulting membership functions are shown in the next two figures.



Membership Functions for output variable Temperature



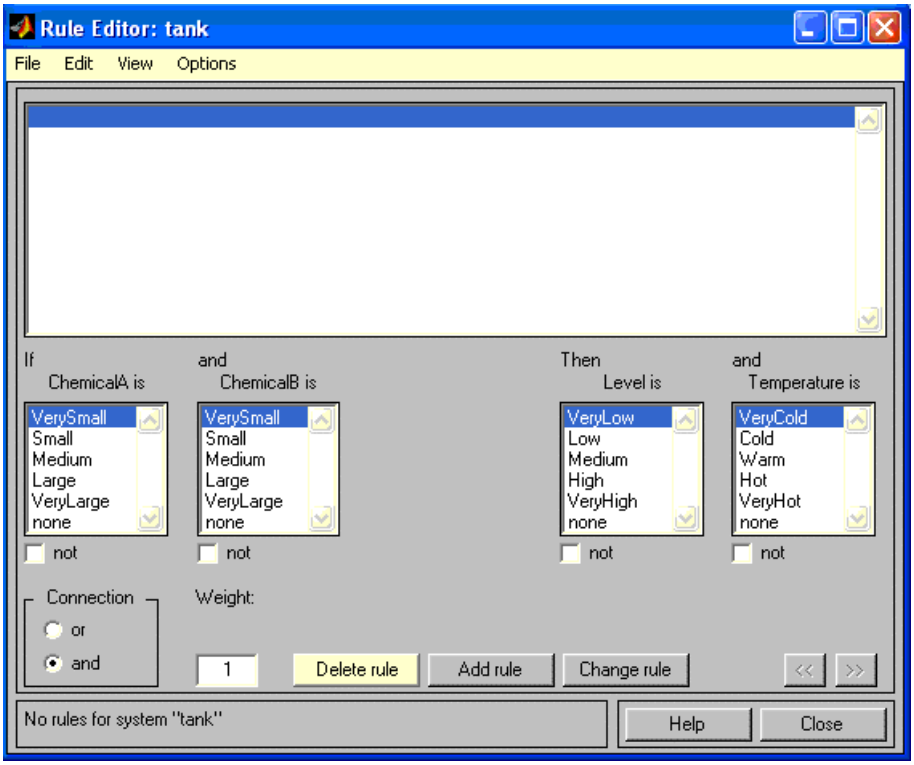
Membership Functions for output variable Level

The Rule Editor

Now that the variables have been named, and the membership functions have appropriate shapes and names, we are ready to write down the rules. To call up the Rule Editor, go to the **View** menu and select **Edit rules...,** or type **ruleedit** at the MATLAB command line.

Another way is to double-click on the main icon on the FIS Editor (on the *tank* rectangle).

The following figure shows the Rule Editor when invoked for the first time, that is, with no rules at all.

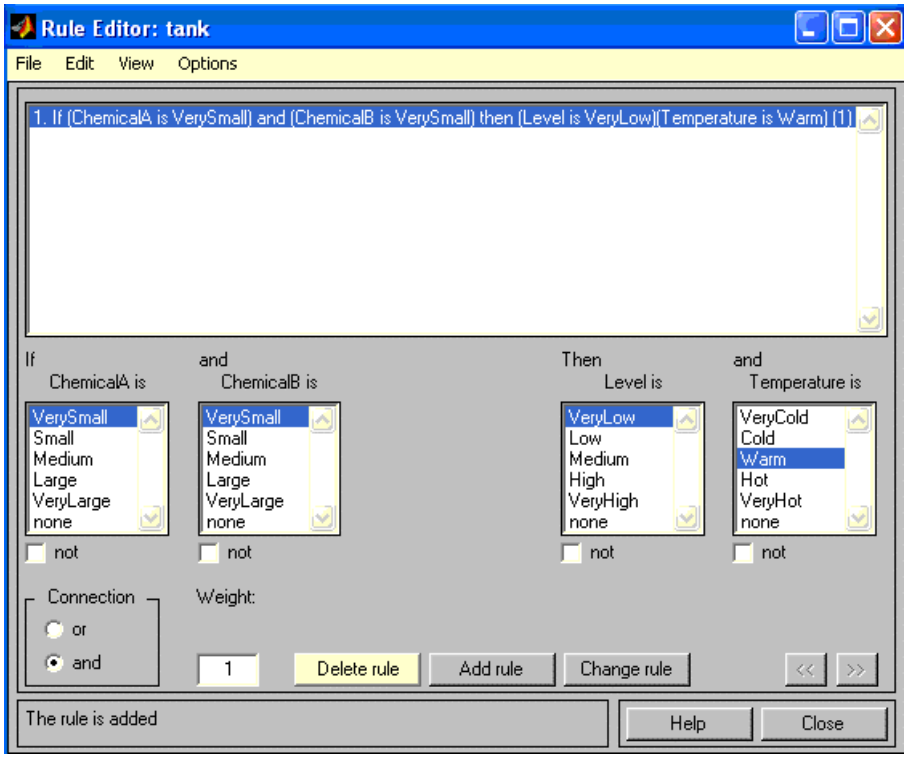


On top of this screen we will have the FIS rules. The rules are defined by combining conditions on the input variables with or, and, not and establishing the resulting values of the output variables. Three buttons allow us to *add*, *change* or *delete* rules. By default all rules have the same weight (1) but it can be changed on the *weight* box.

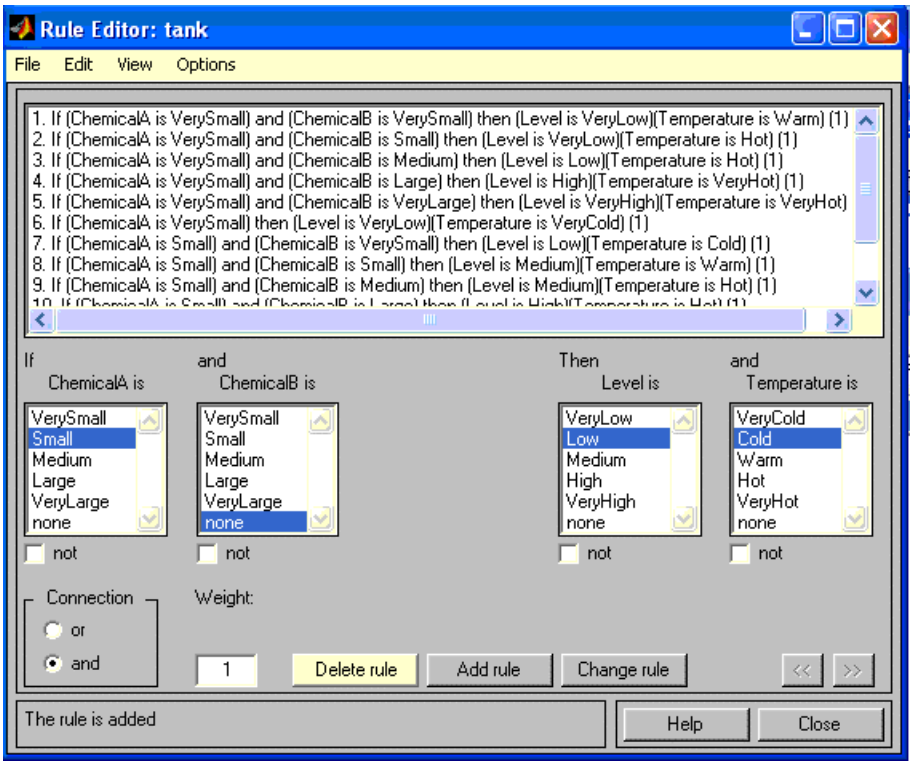
To add the first rule:

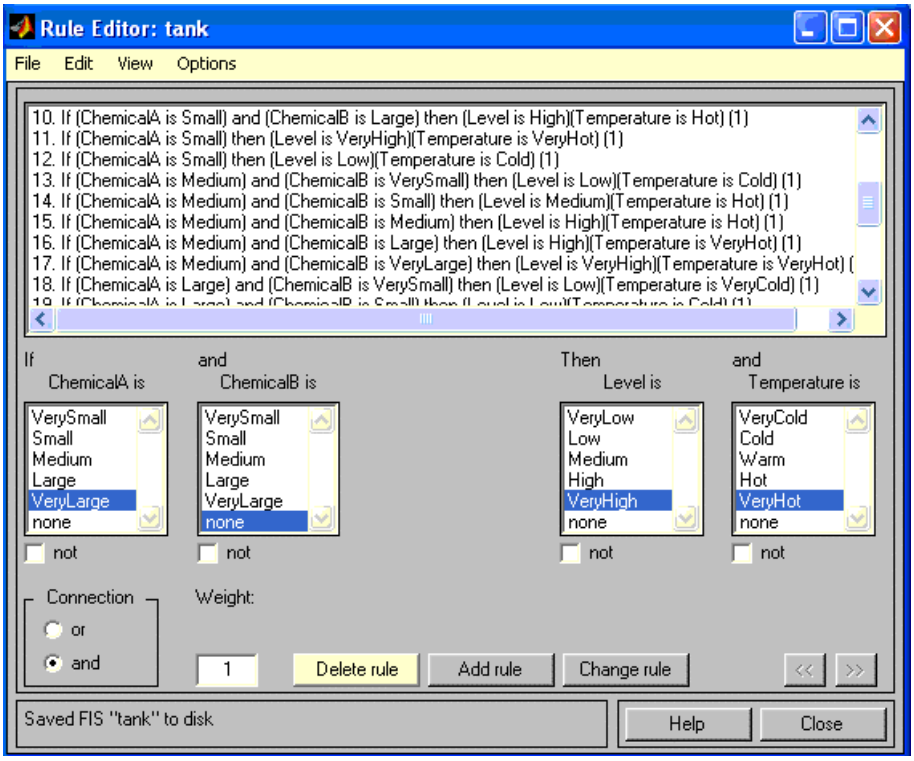
- Select *VerySmall* for ChemicalA
- Select *VerySmall* for ChemicalB
- Leave the ***and*** radio button checked
- Select *VeryLow* for Level
- Select *Warm* for Temperature
- Click on the *add rule* button

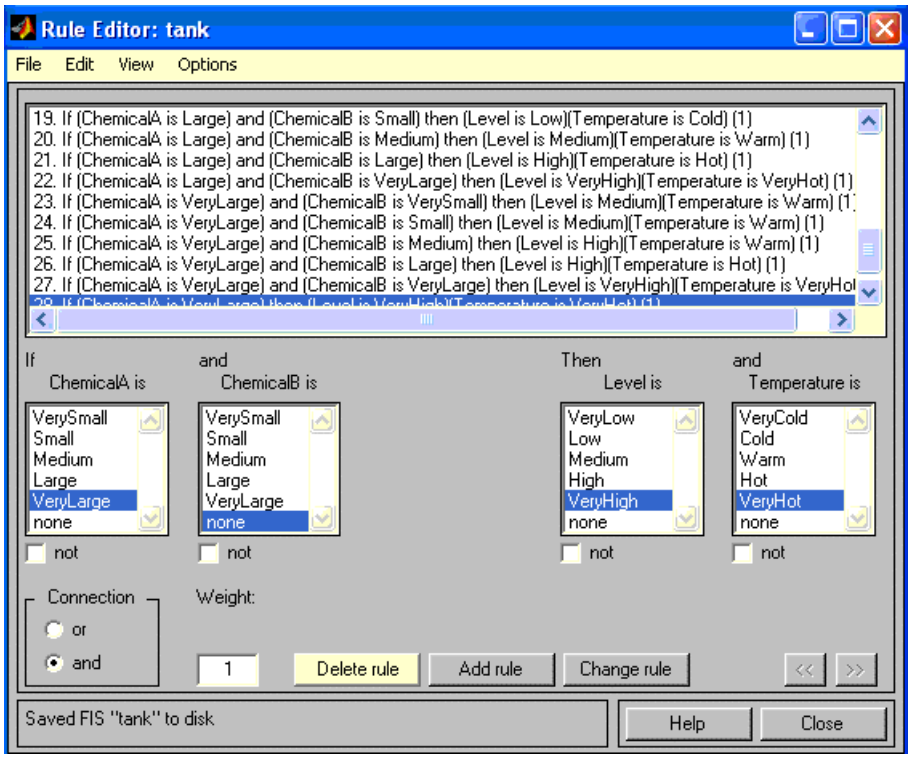
The following screen shows the first rule added to the FIS system.



Repeat the same procedure above until all the rules have been defined. The next three figures show the FIS Rules.

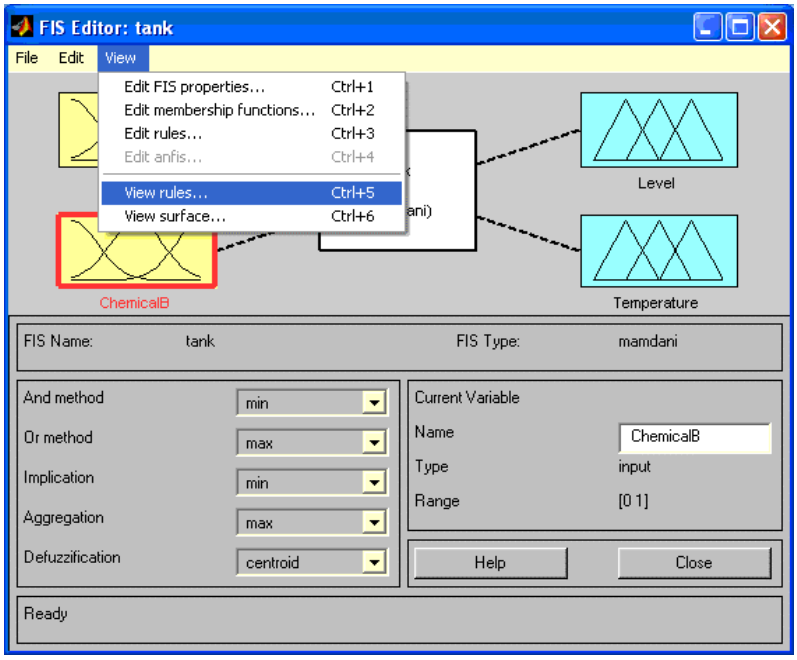



Rule Editor: tank
File Edit View Options
10. If (ChemicalA is Small) and (ChemicalB is Large) then (Level is High)(Temperature is Hot) (1)
11. If (ChemicalA is Small) then (Level is VeryHigh)(Temperature is VeryHot) (1)
12. If (ChemicalA is Small) then (Level is Low)(Temperature is Cold) (1)
13. If (ChemicalA is Medium) and (ChemicalB is VerySmall) then (Level is Low)(Temperature is Cold) (1)
14. If (ChemicalA is Medium) and (ChemicalB is Small) then (Level is Medium)(Temperature is Hot) (1)
15. If (ChemicalA is Medium) and (ChemicalB is Medium) then (Level is High)(Temperature is Hot) (1)
16. If (ChemicalA is Medium) and (ChemicalB is Large) then (Level is High)(Temperature is VeryHot) (1)
17. If (ChemicalA is Medium) and (ChemicalB is VeryLarge) then (Level is VeryHigh)(Temperature is VeryHot) (
18. If (ChemicalA is Large) and (ChemicalB is VerySmall) then (Level is Low)(Temperature is VeryCold) (1)
If
ChemicalA is
VerySmall
Small
Medium
Large
VeryLarge
none
not
and
ChemicalB is
VerySmall
Small
Medium
Large
VeryLarge
none
not
Then
Level is
VeryLow
Low
Medium
High
VeryHigh
none
not
and
Temperature is
VeryCold
Cold
Warm
Hot
VeryHot
none
not
Connection
or
and
Weight:
1
Delete rule
Add rule
Change rule
<<
>>
Saved FIS "tank" to disk
Help
Close

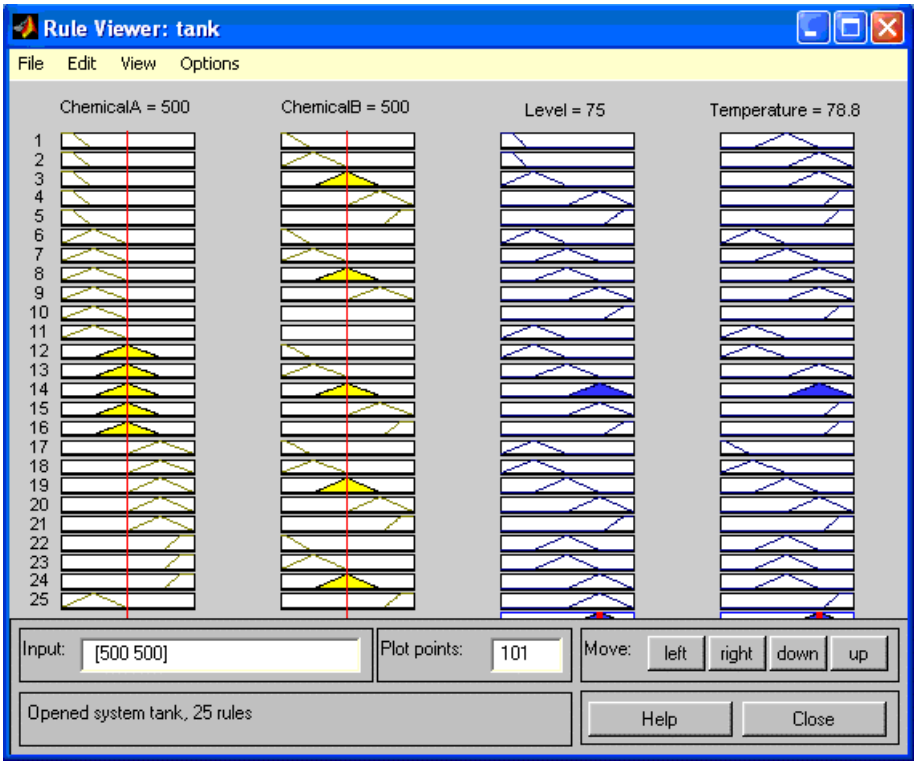

Rule Editor: tank
File Edit View Options
19. If (ChemicalA is Large) and (ChemicalB is Small) then (Level is Low)(Temperature is Cold) (1)
20. If (ChemicalA is Large) and (ChemicalB is Medium) then (Level is Medium)(Temperature is Warm) (1)
21. If (ChemicalA is Large) and (ChemicalB is Large) then (Level is High)(Temperature is Hot) (1)
22. If (ChemicalA is Large) and (ChemicalB is VeryLarge) then (Level is VeryHigh)(Temperature is VeryHot) (1)
23. If (ChemicalA is VeryLarge) and (ChemicalB is VerySmall) then (Level is Medium)(Temperature is Warm) (1)
24. If (ChemicalA is VeryLarge) and (ChemicalB is Small) then (Level is Medium)(Temperature is Warm) (1)
25. If (ChemicalA is VeryLarge) and (ChemicalB is Medium) then (Level is High)(Temperature is Warm) (1)
26. If (ChemicalA is VeryLarge) and (ChemicalB is Large) then (Level is High)(Temperature is Hot) (1)
27. If (ChemicalA is VeryLarge) and (ChemicalB is VeryLarge) then (Level is VeryHigh)(Temperature is VeryHot
If
ChemicalA is
VerySmall
Small
Medium
Large
VeryLarge
none
not
and
ChemicalB is
VerySmall
Small
Medium
Large
VeryLarge
none
not
Then
Level is
VeryLow
Low
Medium
High
VeryHigh
none
not
and
Temperature is
VeryCold
Cold
Warm
Hot
VeryHot
none
not
Connection
or
and
Weight:
1
Delete rule
Add rule
Change rule
<<
>>
Saved FIS "tank" to disk
Help
Close

The Rule Viewer

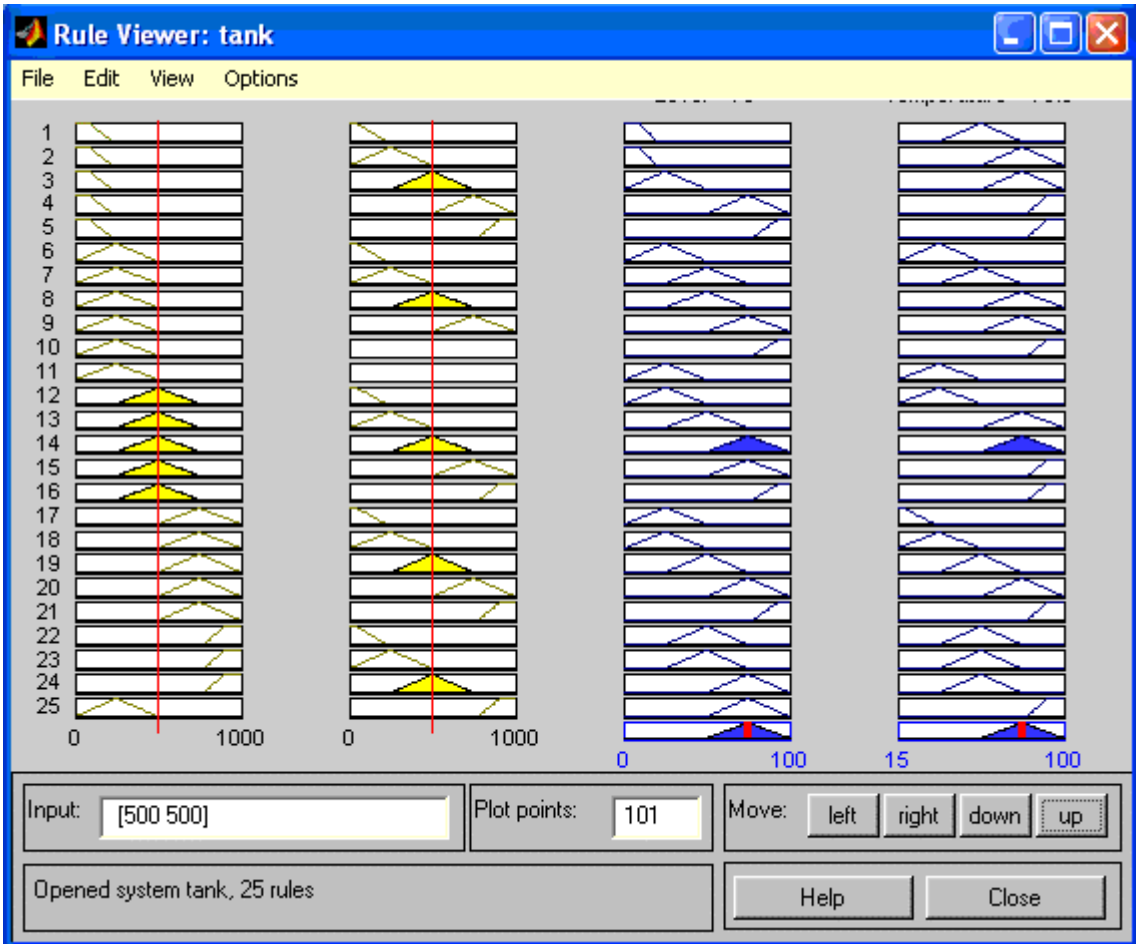
The first way to verify the FIS model is using the Rule Viewer. From the FIS Editor, pull down the **View** menu, select **View rules…** as shown below.



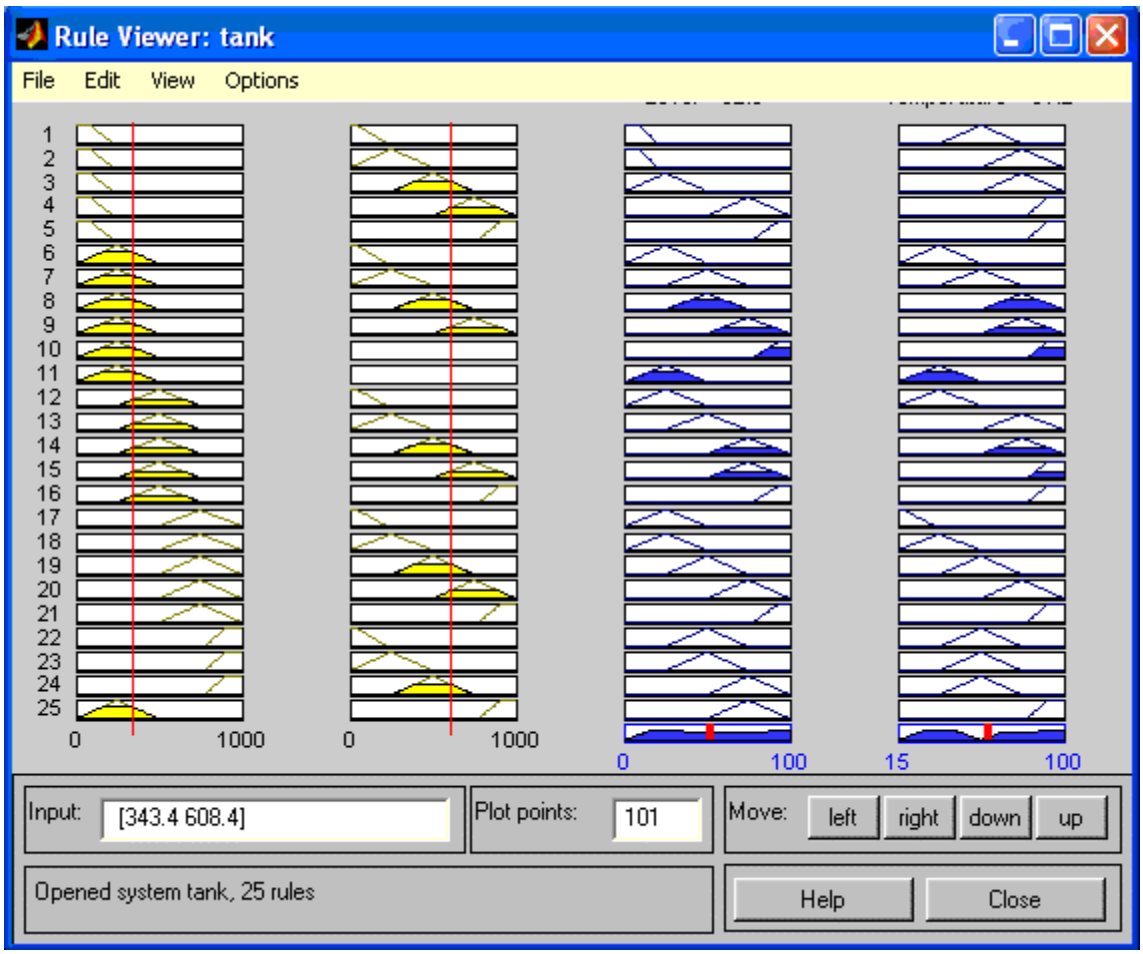
and the following window will be displayed



The default values for the Input variables are displayed on top, as well as the resulting values of the output variables after defuzzification. To see the defuzzification curves click on the Up button on the right lower side of the window. The following window will be displayed.

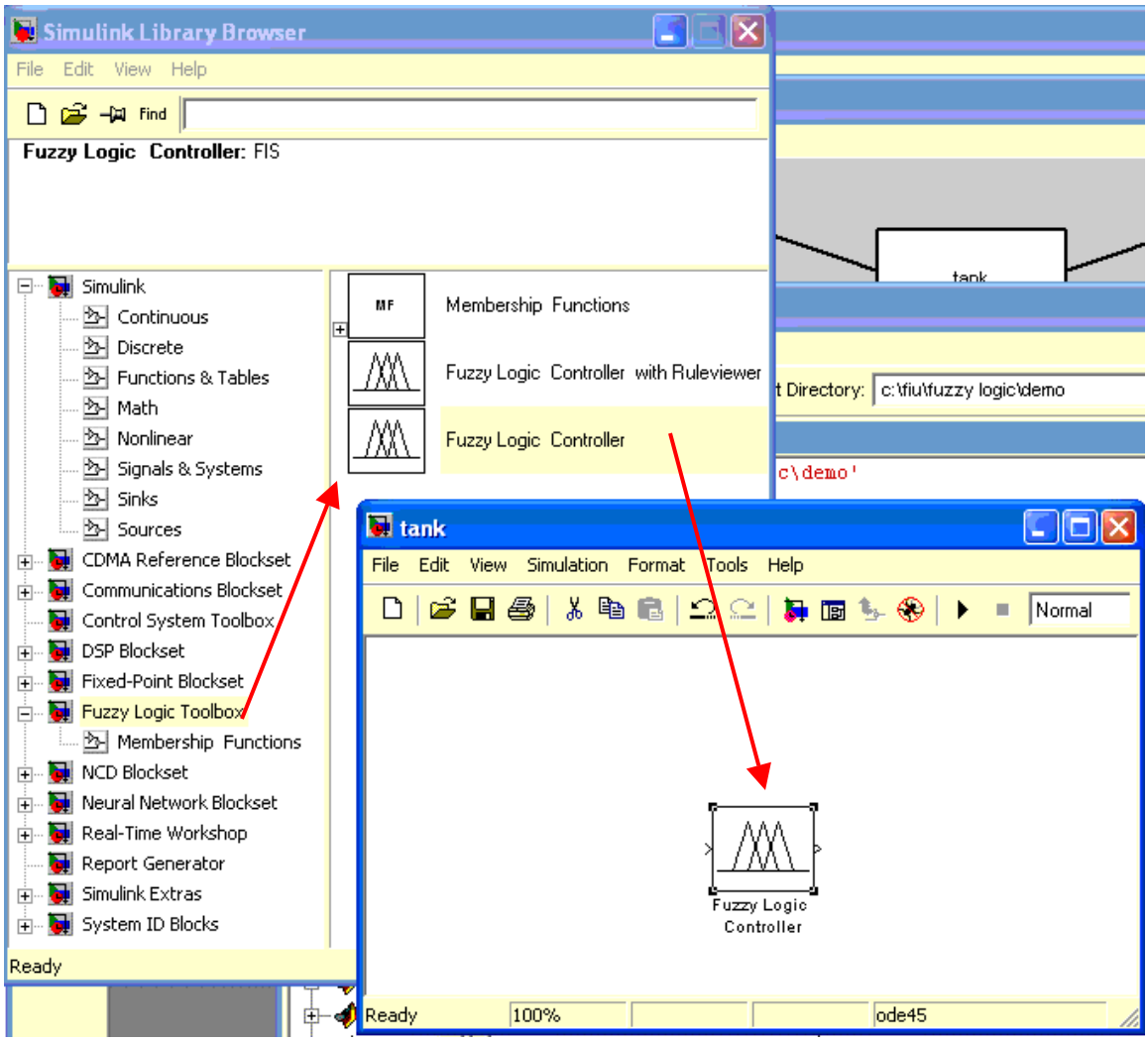


You can change the input values by dragging the red vertical lines corresponding to the input variables. The output variables change dynamically as the inputs are changed. The next figure shows an example.



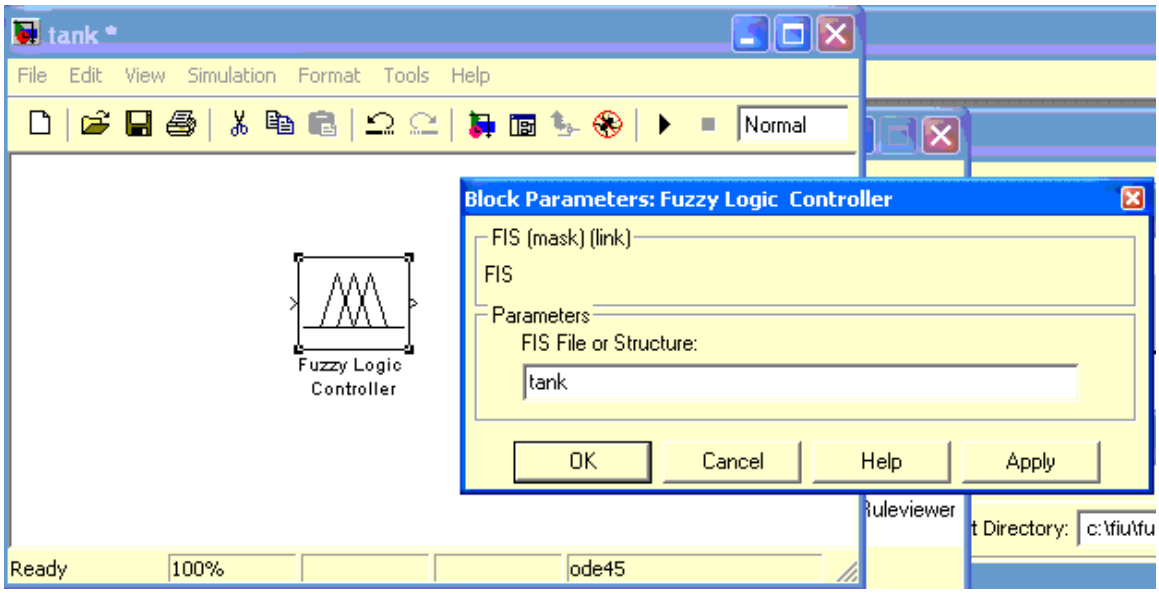
Simulink

To simulate the fuzzy system once it has been verified with the Rule Viewer, we need to create a Fuzzy Controller using Simulink. From the MATLAB prompt type in **SIMULINK**. The following window will be shown.

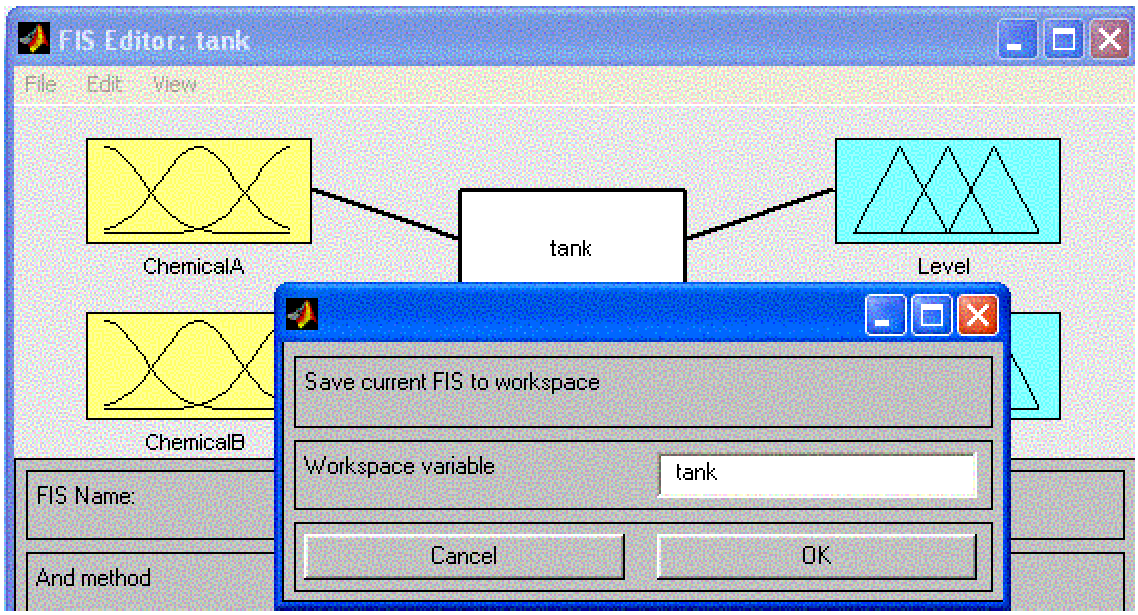


The Simulink menu shows the different Blocksets installed in your system. Using the **File** menu, select **New** to create a model (shown on the right lower side of the above figure). Locate the Fuzzy Logic Toolbox on the Simulink menu and expand it (click on the plus sign as shown). Find the Fuzzy Logic Controller block and drag it onto the new model as shown.

Double click on the Fuzzy Logic Controller to set its parameter. The following pop up window will be displayed.



Type in *tank* on the Parameters box. In order for this to work, a variable *tank* must exist in the MATLAB workspace. To obtain this variable, from the FIS Editor explained in the previous section, pull down the **File** menu and select **Save to Workspace**. On the pop up window type in *tank* and click on the OK button as shown below.



To verify that *tank* is already on the workspace, from the MATLAB prompt type in the command

>> whos

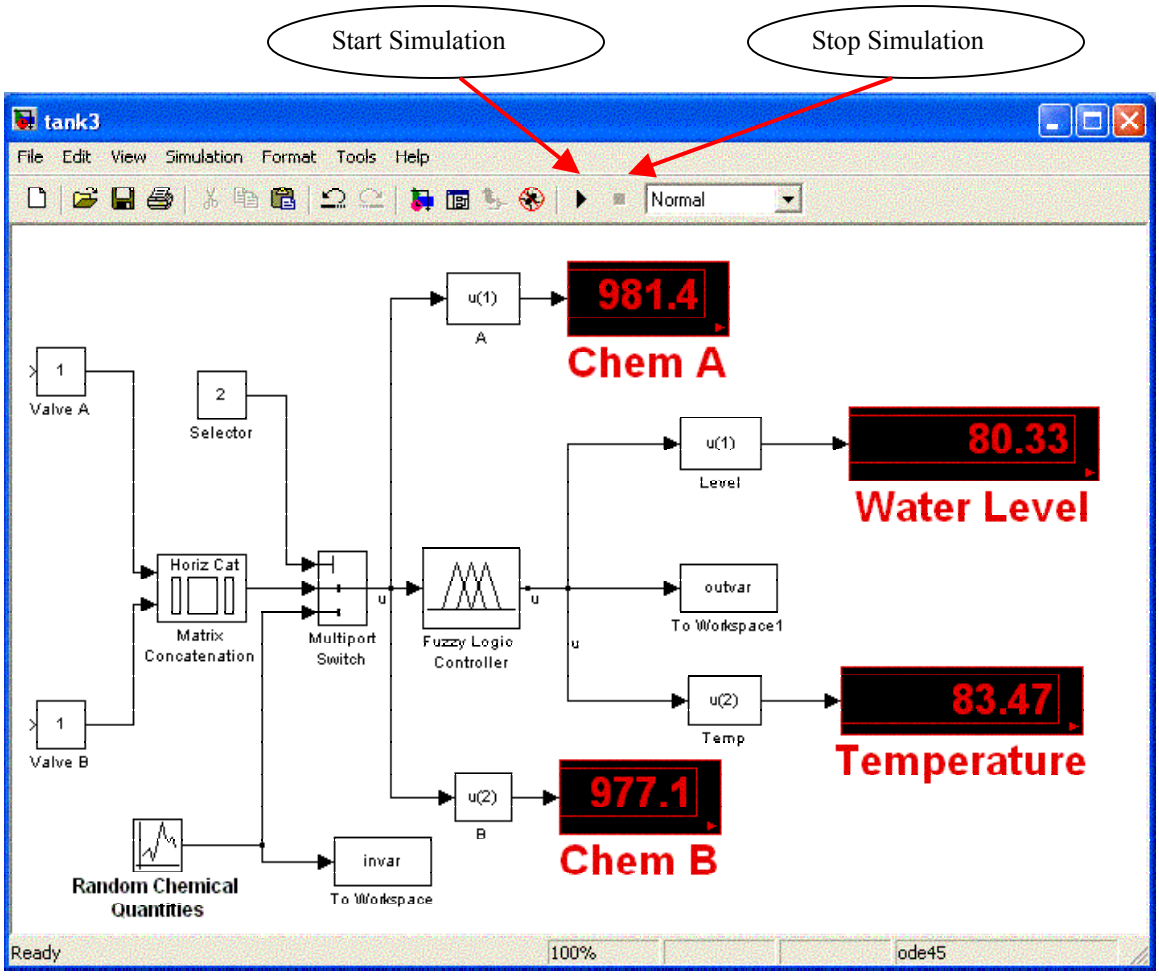
MATLAB responds with something like this

Name	Size	Bytes	Class
tank	1x1	21414	struct array

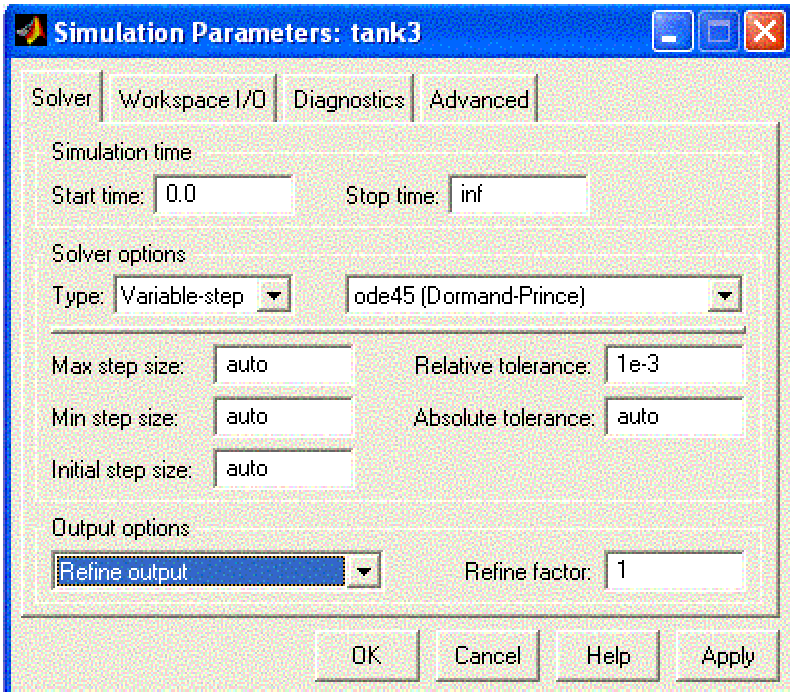
Grand total is 733 elements using 21414 bytes

The Fuzzy Logic Controller block in Simulink has one input and one output. Since our FIS has two inputs and two outputs, the block expects a 2-dimension input vector and a 2-dimension output vector. These are **column** vectors in MATLAB.

The following figure shows the finalized Tank Fuzzy Logic Controller with all the sources and sinks connected to it.



To Start or Stop the simulation click on the corresponding buttons on the toolbar as shown on the figure above. To set the simulation parameters, from the **Simulation** Menu, select Simulation Parameters, and the following window will be displayed.



Leave all parameters with their default values, except the **Stop Time**. Enter **inf** (infinity constant in MATLAB) to specify a continuous run simulation (does not end until the stop button is clicked). Or enter a number of time units, i.e. 1000, for one thousand samples. Simulink does not use a real time clock for simulation. The run time is set depending on the component with the fastest sampling rate (in this case is the random number generator with sampling time = 1, explained later).

Other important Simulink Menu items are:

- **File** Menu, to **Save** the model
- **Simulation** Menu, to change the simulation parameters (run time and others)
- **Edit** menu**,** to change signal names
- **Format**, to change the way controls are displayed

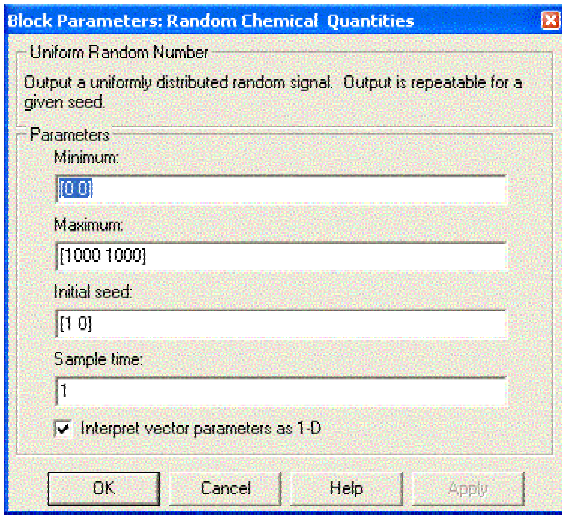
The Fuzzy Controller system is made out of the following components (Simulink Blocksets):

- The Fuzzy Logic Controller explained before
- A Uniform Random Generator labeled *Random Chemical Quantities* to provide random inputs to the fuzzy controller
- Two manual Sliders labeled *Valve A* and *Valve B*, to provide a manual input to the fuzzy controller
- A Matrix Concatenation block, to concatenate the 2 Slider outputs and provide a single input (2-column vector) to the Fuzzy controller
- A Multiport Switch, to select the input to the fuzzy controller: manual valves or random generator

- A constant value block labeled *Selector*, to control the multiport switch. When its value is 1, the input comes from the sliders (manual valves); when its value is 2, the input comes from the random number generator.
- Two To-Workspace block, to save the input and output variables to MATLAB variables *invar* and *outvar*
- Function blocks, labeled *A, B, Level* and *Temp*. These blocks split the 2-column matrices into 1-column vectors for individual displaying
- Four Displays to show the input and outputs of the fuzzy controller

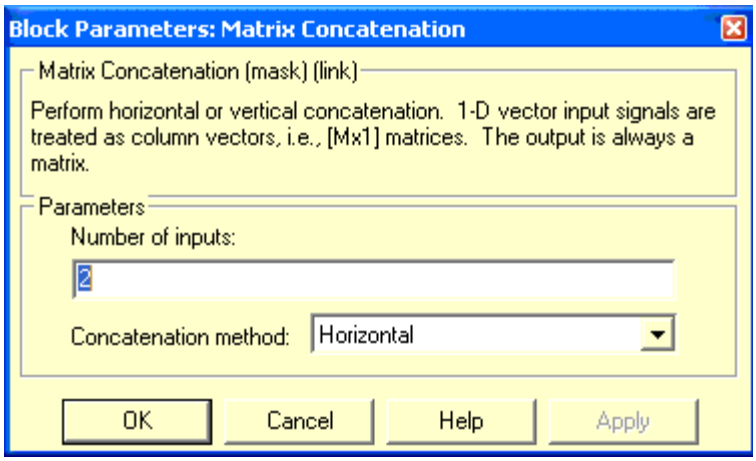
The following pages will describe how to setup the parameters of each block.

1) The Uniform Random Number generator, can be found on the Simulink – Sources Library. These are the parameters for a 2-column matrix random number generator. The Sample time = 1 (one time unit).

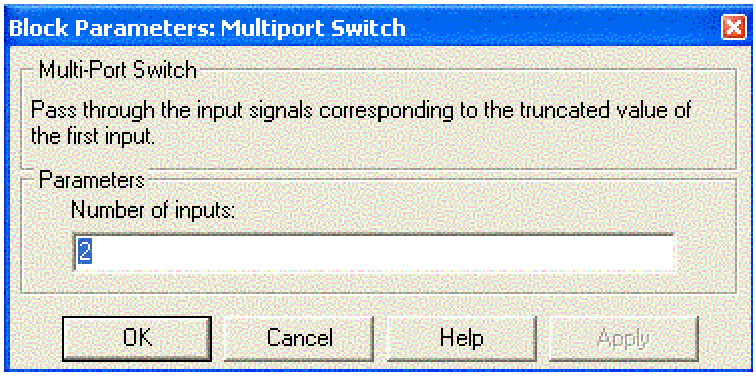


2) The Slider Gain blocks are found under Simulink – Mah Library. Set the Lower value to 0 and Maximum value to 1000 (for 0-1000 grams) for both sliders

3) The Matrix Concatenation Block is found under Simulink – Signals & Systems Library. These are the parameters:

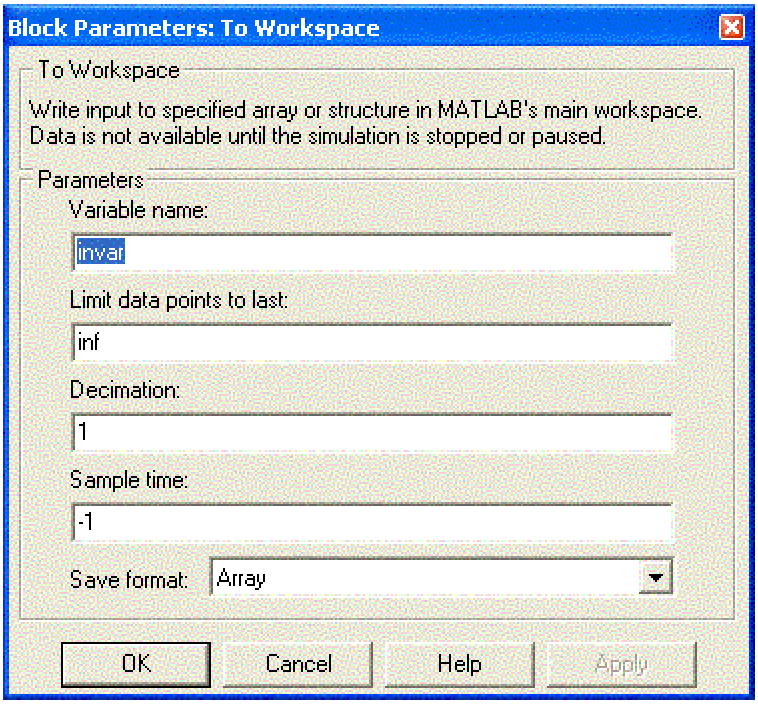


4) The Multiport Switch is found under Simulink – Non LinearLibrary. These are the parameters:

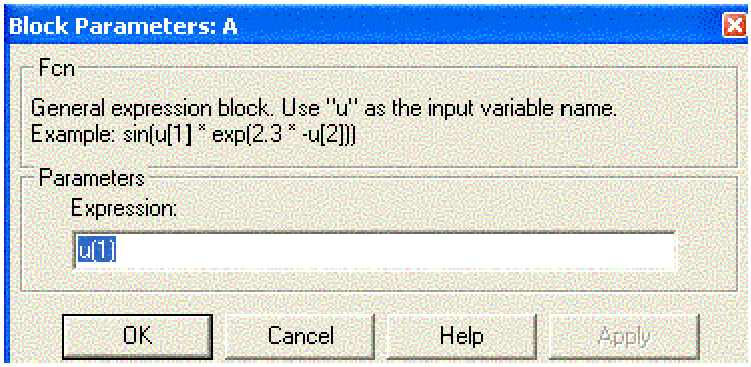


5) The Constant Value block for the *Selector*, can be found under Simulink – Sources Library. Double click on the block and enter the desired value.

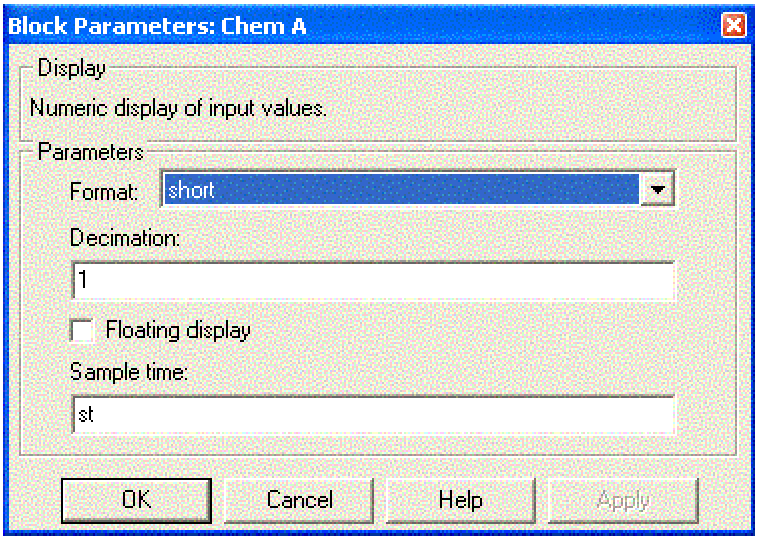
6) The To Workspace Block can be found under Simulink – Sinks Library. Double click on the block and enter the name of the MATLAB variable. Make sure to use Save Format = **Array** as shown below.



7) The Function block can be found under Simulink – Functions & Tables Library. It is mandatory that the input to the block is named *u*. To select column 1, double click on the block and enter *u*(1); for column 2 enter *u*(2). To name the input to the block, click on the line, select the **Edit** menu, Signal Properties, and in the Signal Name box enter *u*.



8) The four displays can be found under Simulink – Sinks Library. The default display has been changed by right-click the mouse on the display. The Background Color was changed to *black*, the Foreground Color to *red* and the Font size was increased to 26. The resulting display is shown below.



Notice the Sample Time = **st**. This is usually a number, multiple of the fastest sampling rate in the system (in this case 1, the random number generator). However, depending on the computer you are using, the display may change too fast or too slow. Since we have 4 displays, instead of changing the value on the four displays until the rate of change is acceptable, we use a MATLAB variable named **st**. This way, we only need to change the value of **st** and all displays will change the sample time. From the MATLAB prompt, type in:

```
>> st=50;
```

After running and stopping the simulation, using the Random Number Generator, the resulting input and output variables are stored on MATLAB variables. To verify this, enter the following command on the MATLAB prompt:

```
>> whos
```

MATLAB responds with something like this:

```
  Name        Size        Bytes  Class

  ans         1x1             8  double array
  invar    1352x2         21632  double array
  outvar   1352x2         21632  double array
  st          1x1             8  double array
  tank        1x1         21414  struct array
  tout     1000x1          8000  double array
```

As you can see we have a 2-column matrix for the input variables (chemical quantities), *invar*, and a 2-column matrix for the output variables (Water Level and Temperature), *outvar*.

To show a simple example of how to split the matrices into individual vectors and plot them, enter the following code at the MATLAB prompt.

```
chemA=invar(:,1);
chemB=invar(:,2);
Level=outvar(:,1);
Temp=outvar(:,2);

%plot the first 20 points; Level and temperature are scaled (10 times)
plot(chemA(1:20),'r');hold on;
plot(chemB(1:20),'b');hold on;
plot(10*Level(1:20),'g');hold on;
plot(10*Temp(1:20),'c');hold on;
grid on;
```

The resulting plot with the 4 variables will show something like this.