Working with Simulink

The Fuzzy Logic Toolbox is designed to work seamlessly with Simulink, the simulation software available from The MathWorks. Once you’ve created your fuzzy system using the GUI tools or some other method, you’re ready to embed your system directly into a simulation.

An Example: Water Level Control

Picture a tank with a pipe flowing in and a pipe flowing out. You can change the valve controlling the water that flows in, but the outflow rate depends on the diameter of the outflow pipe (which is constant) and the pressure in the tank (which varies with the water level). The system has some very nonlinear characteristics.

A controller for the water level in the tank needs to know the current water level and it needs to be able to set the valve. Our controller’s input will be the water level error (desired water level minus actual water level) and its output will be the rate at which the valve is opening or closing. A first pass at writing a fuzzy controller for this system might be the following.

1. If (level is okay) then (valve is no_change) (1)
2. If (level is low) then (valve is open_fast) (1)
3. If (level is high) then (valve is close_fast) (1)

One of the great advantages of the Fuzzy Logic Toolbox is the ability to take fuzzy systems directly into Simulink and test them out in a simulation environment. A Simulink block diagram for this system is shown below.
It contains a Simulink block called the Fuzzy Logic Controller block. The Simulink block diagram for this system is sltank. Typing

```
sltank
```

at the command line, causes the system to appear. At the same time, the file tank.fis is loaded into the FIS structure tank.

Some experimentation shows that three rules are not sufficient, since the water level tends to oscillate around the desired level. This is seen from the following plot.
We need to add another input, the water level’s rate of change, to slow down the valve movement when we get close to the right level.

4. If (level is good) and (rate is negative), then (valve is close_slow) (1)
5. If (level is good) and (rate is positive), then (valve is open_slow) (1)

The demo, sltank is built with these five rules. With all five rules in operations, you can examine the step response by simulating this system. This is done by clicking **Start** from the pull-down menu under **Simulate**, and clicking the Comparison block. The result looks like this.
One interesting feature of the water tank system is that the tank empties much more slowly than it fills up because of the specific value of the outflow diameter pipe. We can deal with this by setting the close_slow valve membership function to be slightly different from the open_slow setting. A PID controller does not have this capability. The valve command versus the water level change rate (depicted as water) and the relative water level change (depicted as level) surface looks like this. If you look closely, you can see a slight asymmetry to the plot.
Because the MATLAB technical computing environment supports so many tools (like the Control System Toolbox, the Neural Network Toolbox, and so on), you can, for example, easily make a comparison of a fuzzy controller versus a linear controller or a neural network controller.

For a demonstration of how the Rule Viewer can be used to interact with a Fuzzy Logic Controller block in a Simulink model, type

```
sltankrule
```

This demo contains a block called the Fuzzy Controller With Rule Viewer block.

In this demo, the Rule Viewer opens when you start the Simulink simulation. This Rule Viewer provides an animation of how the rules are fired during the water tank simulation. The windows that open when you simulate the sltankrule demo are depicted as follows.
The Rule Viewer that opens during the simulation can be used to access the Membership Function Editor, the Rule Editor, or any of the other GUIs, (see “The Membership Function Editor” on page 2-38, or “The Rule Editor” on page 2-42, for more information).

For example, you may want to open the Rule Editor to change one of your rules. To do so, select Rules under the Edit menu of the open Rule Viewer. Now you can view or edit the rules for this Simulink model.
It is best if you stop the simulation prior to selecting any of these editors to change your FIS. Remember to save any changes you make to your FIS to the workspace before you restart the simulation.
Building Your Own Fuzzy Simulink Models

To build your own Simulink systems that use fuzzy logic, simply copy the Fuzzy Logic Controller block out of s1tank (or any of the other Simulink demo systems available with the toolbox) and place it in your own block diagram. You can also find the Fuzzy Logic Controller block in the Fuzzy Logic Toolbox library, which you can open either by selecting Fuzzy Logic Toolbox in the Simulink Library Browser, or by typing

```
fuzblock
```

at the MATLAB prompt.

The following library appears.

![Library](fuzblock)

The Fuzzy Logic Toolbox library contains the Fuzzy Logic Controller and Fuzzy Logic Controller with Rule Viewer blocks. It also includes a Membership Functions sublibrary that contains Simulink blocks for the built-in membership functions.

The Fuzzy Logic Controller with Rule Viewer block is an extension of the Fuzzy Logic Controller block. It allows you to visualize how rules are fired during simulation. Right-click on the Fuzzy Controller With Rule Viewer block and select Look Under Mask, and the following appears.
To initialize the Fuzzy Logic Controller blocks (with or without the Rule Viewer), double-click on the block and enter the name of the structure variable describing your FIS. This variable must be located in the MATLAB workspace.

**About the Fuzzy Logic Controller Block**

For most fuzzy inference systems, the Fuzzy Logic Controller block automatically generates a hierarchical block diagram representation of your FIS. This automatic model generation ability is called the *Fuzzy Wizard*. The block diagram representation only uses built-in Simulink blocks and therefore allows for efficient code generation. For more information about the Fuzzy Logic Controller block, see the `fuzblock` reference page.

The Fuzzy Wizard cannot handle FIS with custom membership functions or with AND, OR, IMP, and AGG functions outside of the following list:

- `orMethod`: `max`
- `andMethod`: `min, prod`
- `impMethod`: `min, prod`
- `aggMethod`: `max`

In these cases, the Fuzzy Logic Controller block uses the S-function `sffis` to simulate the FIS. For more information, see the `sffis` reference page.