

Fuzzy Continuous Petri Nets (FCPN)

Fei Liu
Wujie Sun

December 4, 2018

Contents

1	Introduction	3
1.1	Continuous Petri nets and ODEs	3
1.2	Fuzzy logic	3
1.2.1	Fuzzy Inference	4
1.2.2	Fuzzification	5
1.2.3	Fuzzy logic operators	5
1.2.4	Defuzzification	5
1.3	Features - overview	5
1.3.1	Features for modeling	6
1.3.2	Features for simulation	6
2	Modeling	6
2.1	General modeling procedure	6
2.1.1	Draw the continuous Petri net	6
2.1.2	Define place	7
2.1.3	Define transition	8
2.1.4	Define arc	8
2.1.5	Use FIS	9
3	Simulation	18
3.1	Run simulation	18
3.2	Show simulation results	18
3.3	Export simulation results	20
3.4	Rich functions	20
4	Examples	22
4.1	1D Diffusion Reaction	22
4.1.1	Introduction	22
4.1.2	Model	22
4.1.3	Simulation result	27
4.2	Enzyme	30
4.2.1	Introduction	30
4.2.2	Model	30
4.2.3	Simulation result	31
4.3	RKIP	33
4.3.1	Introduction	33
4.3.2	Model	33
4.3.3	Simulation result	33
5	References	36

1 Introduction

Petri nets provide a formal and clear representation of systems based on their firm mathematical foundation for the analysis of system properties. This software provides fuzzy continuous Petri nets modeling and simulation functions for researchers in the field of systems biology.

The purpose of this system is to provide users with services to complete the simulation and fuzzy inference by constructing Petri nets. This software includes three main functions: continuous Petri nets modeling, fuzzy modeling, and hybrid simulation. Among them, the simulation is based on the transition and fire of the Petri model input by the user, and the simulation results are given.

In order to improve the user experience, we simplify the modeling page to give users a more intuitive sense of use without affecting the function during the design process. At the same time, the simulation and fuzzification are integrated in the system, so that the users' needs are satisfied as much as possible.

Table 1: Hardware environment

CPU model	Core number	Main frequency	Memory model	Memory size
Intel core i3	4	2.8GHz	1600MHz	2GB or more

Operating system: Windows

1.1 Continuous Petri nets and ODEs

A Petri net, also known as a place/transition (PT) net, is one of several mathematical modeling languages for the description of distributed systems. It is a class of discrete event dynamic system. A Petri net is a directed bipartite graph, in which the nodes represent transitions (i.e. events that may occur, represented by bars) and places (i.e. conditions, represented by circles). The directed arcs describe which places are pre- and/or postconditions for which transitions (signified by arrows). [1]

A continuous Petri net is a variant of Petri nets. In this model, the values identified in each place are non-negative real values (no longer required to be non-negative integer values), and the conditions and results of the transitions are also changed accordingly: the occurrences are positive real values.

ODE is the abbreviation for ordinary differential equations. In FCPN, we convert the model to ODEs to calculate the next step value of each input according to its current step value.

1.2 Fuzzy logic

Fuzzy logic is a form of many-valued logic in which the truth values of variables may be any real number between 0 and 1. It is employed to handle the concept of partial truth, where the truth value may range between completely true and completely false. By contrast, in Boolean logic, the truth values of variables may only be the integer values 0 or 1. It is based on the observation that people make decisions based on imprecise and non-numerical information,

fuzzy models or sets are mathematical means of representing vagueness and imprecise information, hence the term fuzzy. These models have the capability of recognising, representing, manipulating, interpreting, and utilising data and information that are vague and lack certainty. [2]

The process of fuzzy logic can divide into three steps: fuzzification, fuzzy logic operators, and defuzzification.

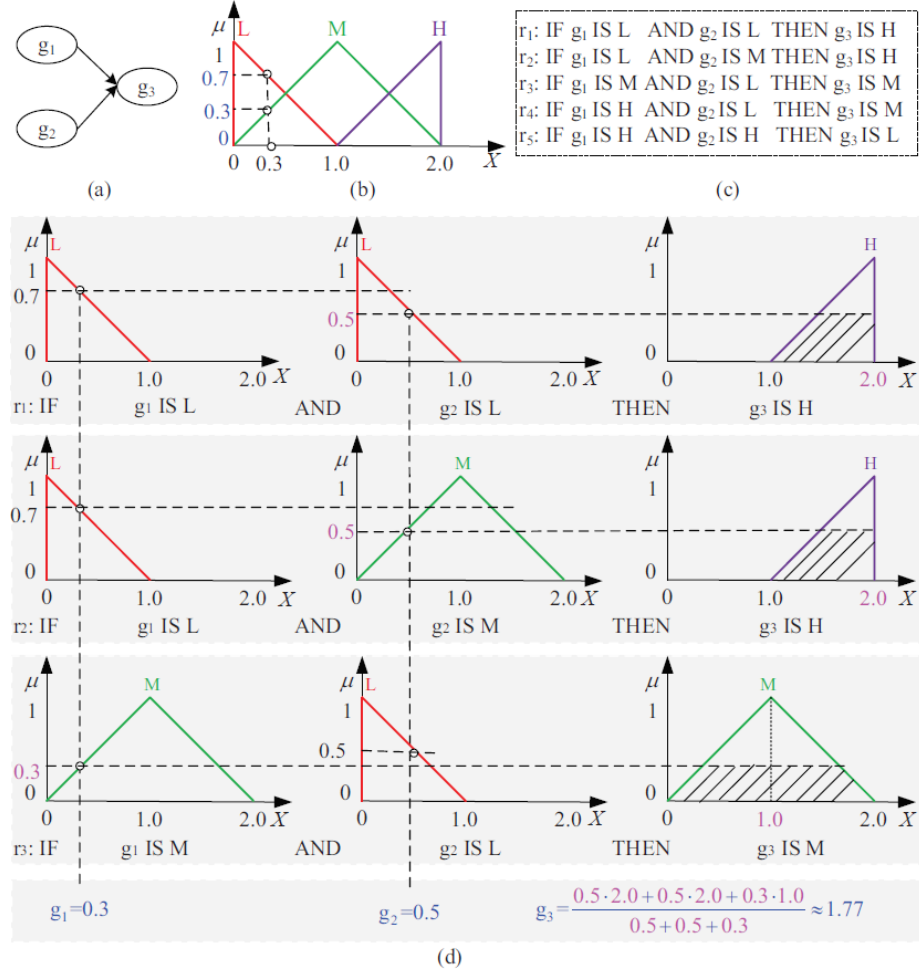


Figure 1: An example of fuzzy logic.

1.2.1 Fuzzy Inference

First, let me introduce the fuzzy inference. Fuzzy inference is the process of formulating the mapping from a given input to an output using fuzzy logic. The mapping then provides a basis from which decisions can be made, or patterns discerned. The process of fuzzy inference involves all the pieces that are described in Membership Functions, Logical Operations, and If-Then Rules. [3] In our system, we provide two kinds of fuzzy inference - Mamdani and T-S.

- Mamdani fuzzy inference is the most commonly seen fuzzy methodology and was among the first control systems built using fuzzy set theory. It expects the output membership functions to be fuzzy sets. After the aggregation process, there is a fuzzy set for each output variable that needs defuzzification. [4]
- T-S can be called as Sugeno or Takagi-Sugeno-Kang. This method is similar to the Mamdani method in many respects. The first two parts of the fuzzy inference process, fuzzifying the inputs and applying the fuzzy operator, are the same. The main difference between Mamdani and Sugeno is that the Sugeno output membership functions are either linear or constant. [5]

1.2.2 Fuzzification

Fuzzification is about to fuzzify all input values into fuzzy membership functions.

For example, in Figure 1 (a), g_1 and g_2 are inputs while g_3 is the output. And Low, Medium, and High are three fuzzy sets. We assume that the minimum value and the maximum value of g_1 , g_2 , and g_3 are both 0 and 2, respectively. After fuzzification, the fuzzy membership functions are shown as Figure 1 (b) where X indicates the value of the input and μ indicates the membership grade of the fuzzy set. And we have three fuzzy membership functions L (Low), M (Medium), and H (High) in each input. When X is equal to 0.3, the membership grades of each membership set is 0.7, 0.3, and 0, respectively.

1.2.3 Fuzzy logic operators

In fuzzy logic operators, we execute all applicable rules in the rulebase to compute the fuzzy output functions.

As we can see from Figure 1 (c), there are 5 rules. We should calculate the value of g_3 of each rule. For rule 1, g_1 's membership grade of Low is 0.7 while g_2 's membership of Low is 0.5. We should choose the minimum value, that is 0.5. And we can see that the point where g_3 's membership grade of High is 1 is 2.0. For each rule, we calculate the v_1 (such as 0.5) and v_2 (such as 2.0).

1.2.4 Defuzzification

Defuzzification is about to de-fuzzify the fuzzy output functions to get "crisp" output values.

For for each rule's v_1 and v_2 , we multiply v_1 and v_2 to get v_3 (for rule 1, it's $v_1 * v_2 = 0.5 * 2.0 = 1.0$), and add each rule's v_3 together to get the numerator. Denominator is the result of adding each v_1 together. Therefore the defuzzification result is about 1.77 shown as Figure 1 (d). The v_1 of rule 4 and 5 are both 0, so there is no need to add them in the calculation.

1.3 Features - overview

Before exploring all features in detail in the following sections, we will give a brief overview for the expected features here.

1.3.1 Features for modeling

- Concise and efficient interface design.
- Drawing of the Petri net graph as usual.
- Flexible user-defined functions.
- Multiple fuzzy logic choices.
- Simple and fast fuzzy logic settings.
- Rich shortcut settings, such as undo, redo, save, print, etc.

1.3.2 Features for simulation

- Highly automated simulation process.
- Diversified export of simulation results.
- Custom simulation result drawing.

2 Modeling

In this section, we will first demonstrate how to construct a continuous Petri net and consider several key modeling problems afterwards.

2.1 General modeling procedure

This section will present a general step-by-step procedure of how to construct a continuous Petri net. A simple example will be used for the illustration of the procedure.

2.1.1 Draw the continuous Petri net

After opening the application, what we should do is to draw the continuous Petri net. Fig. 1. shows the main interface of the software. We can use it to build our own continuous Petri Net.

The menu bar above and the tool bar on the left helps us to create the net faster and more conveniently. The tools from top to bottom is new, open, save, save as, print, undo, redo, delete, normal cursor, place, transition, arc, and simulation.

The top window is used to draw the net and the bottom window is used to output the log. The slider on the lower right can be used to zoom in or zoom out. Now, let's start to draw with the following steps:

- The drawing should be in the top window.
- Select the place (the circle on the tool bar).
- Put three places on the place where you want by clicking the left mouse.
- Select the transition (the square on the tool bar).
- Put one transition on the place where you want by clicking the left mouse.



Figure 2: Main interface of the software.

- Select the arc (below the transition).
- When the mouse is on a place, press the left mouse and hold, move the mouse to the transition, then release, in this way you can draw an arrow from place to transition.
- Draw an arrow from another place to transition.
- Draw an arrow from transition to the leaving place.
- After these steps, the continuous Petri net should be like Figure 3.

2.1.2 Define place

In the next step, we define each place.

- Double click the place in the top window.
- A window named Place Attributes will show.
- Change the Name and Marking.
- After editing, click OK button.
- Change the name of p0, p1, p2 to H2, O2, H2O, respectively.
- Change the marking of p0, p1, p2 to 30, 20, 0, respectively.
- Please note that the name of each place should be different and the marking should be nonnegative real number.

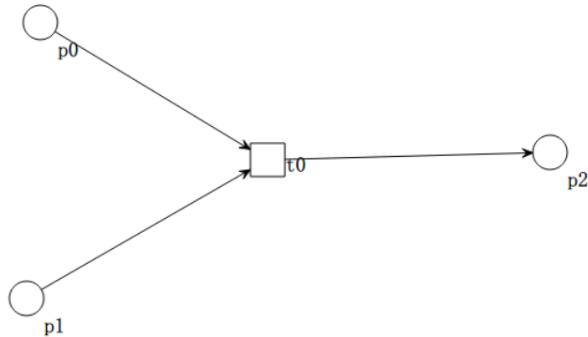


Figure 3: Draw the net.

2.1.3 Define transition

In the next step, we define the transition.

- Double click the transition in the top window.
- A window named Transition Attributes will show.
- Change the Name and Function.
- After editing, click OK button.
- In this example, we didn't change the name.
- Change the function from `MassAction(1)` to `MassAction(0.1)`.
- Please note that the name of each transition should be different and the function should be like "1", "H2", "MassAction(0)", "MassAction(O2)", "MassAction(H2+O2)". Remember not to change the "MassAction(1)" to "massaction(1)" or something like that. And do not leave the function empty. Otherwise compile error might occur.

2.1.4 Define arc

In the next step, we define the arc.

- Double click the arc in the top window.
- A window named Arc Attributes will show.
- Change the Expression.
- After editing, click OK button.

Place Attributes

General **Graphic**

ID : 0

Name : p0

Marking : 0

Comment : ☐ show

Cancel OK

Figure 4: Define place.

- Change the expression of arc from H2 to t0 to 2. For the expression of arc from t0 to H2O, see next step.
- Please note that the expression should be like “1”, “H2” if you don’t want use fuzzy logic, and do not leave the expression empty. Otherwise compile error might occur.

2.1.5 Use FIS

FIS is the abbreviation for fuzzy inference system. Now double click the arc from t0 to H2O and click the FIS editor button. We provide two kinds of FIS type: Mamdani and T-S. The steps are different between Mamdani and T-S. So the first step is to choose the FIS Type. Then enter the time step and select the

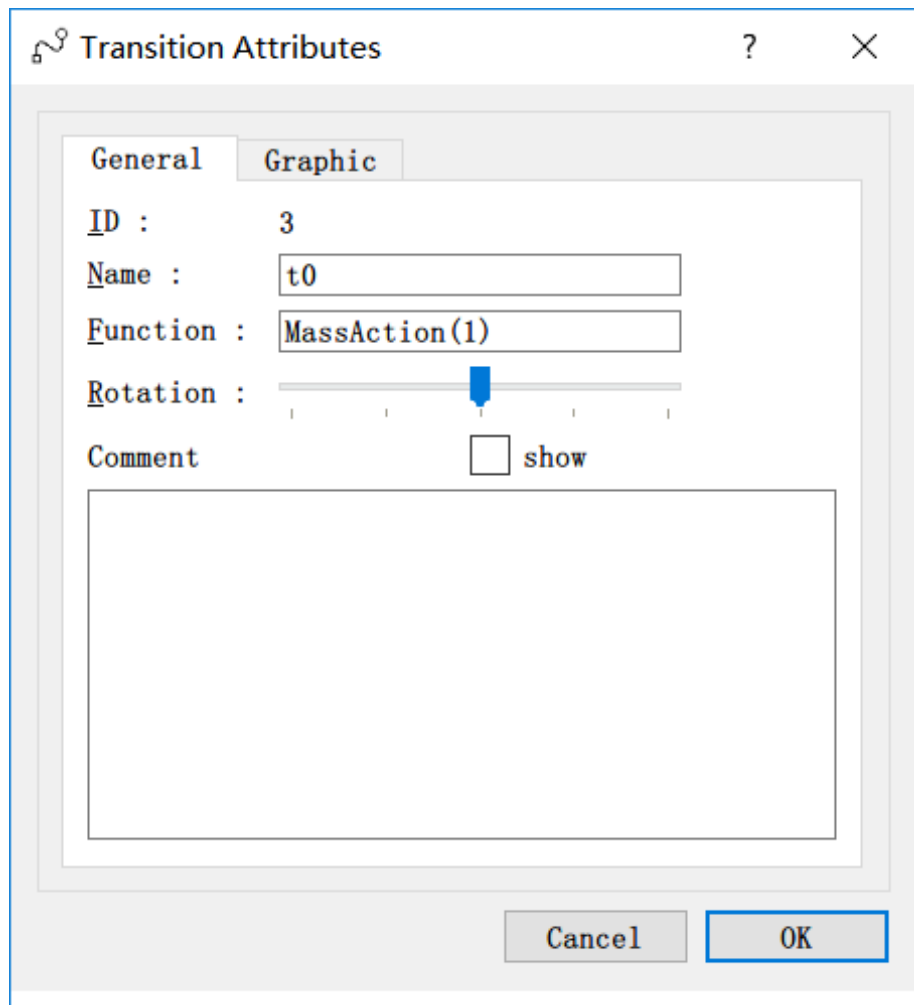


Figure 5: Define transition.

number of input variables. After that, click the Apply button. Here are steps for using Mamdani:

- Choose each variable and its type in the table.
- Make sure there is only one variable whose type is PN_OUTPUT, and it should be the variable connected by the arc you choosed. In this example, that's H2O.
- Enter the minimum change and maximum change in one step, which are corresponding to the Range-Min and Range-Max of the table.
- The values are shown in Figure 8.

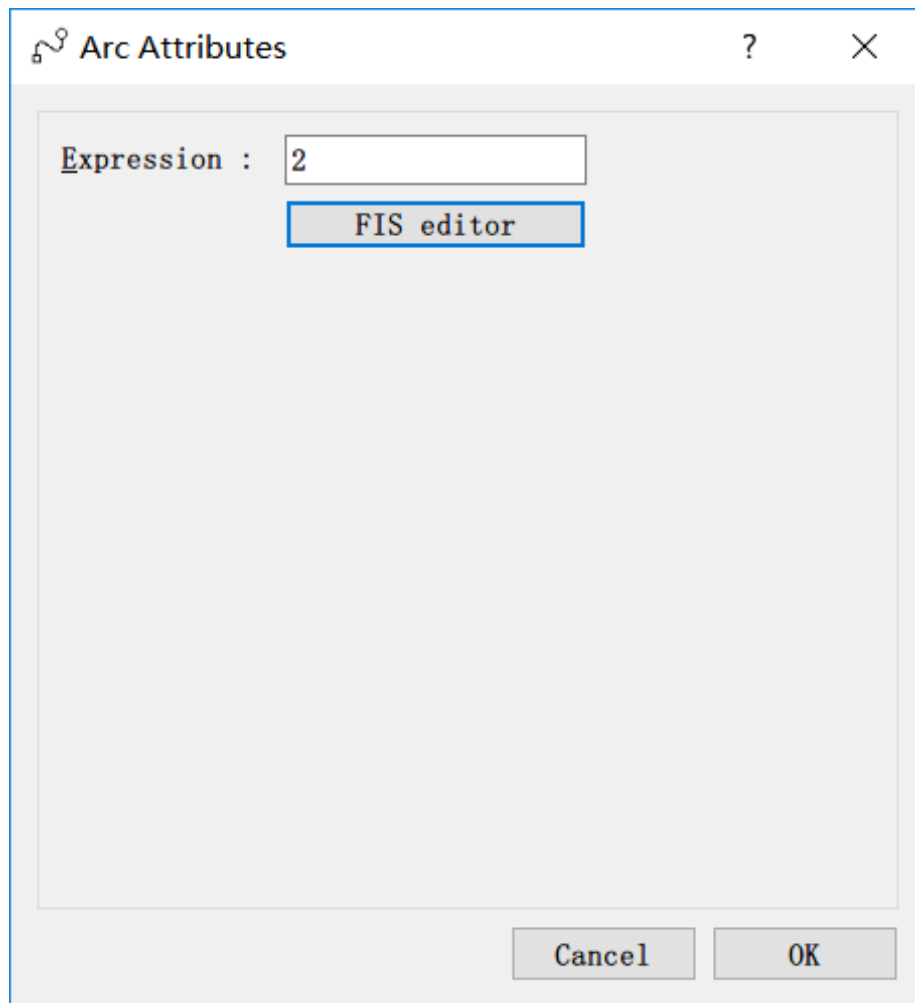



Figure 6: Define arc.

 Edit FIS properties
 ?
×

FIS Name

Time Step

0.1

FIS Type

Mamdani

Number of input variables

1

Apply

Defuzzification

centroid

List of variables

Edit	Variable	Type	Range-Min	Range-Max

Edit rules

OK

Cancel

Figure 7: FIS editor.

Edit FIS properties

FIS Name: FIS (H2, O2;H2O) Time Step: 0.005

FIS Type: Mamdani Number of input variables: 2 **Apply**

Defuzzification: centroid

List of variables

edit	Variable	Type	Range-Min	Range-Max
Edit	H2	PN_INPUT	0	30
Edit	O2	PN_INPUT	5	20
Edit	H2O	PN_OUTPUT	0	18

Edit rules **OK** **Cancel**

Figure 8: Edit variables.

- Click the Edit button on the first column to edit each variable's membership functions.
- Choose the number of linguistic variables, we provide 2, 3, 5, 7 for the user to choose.
- Click the Apply button and the according label will show.
- For each function, enter the value of left, center and right which means the x values when y is equal to 0, 1, 0, respectively.
- Click the Show Plot button to see whether it is the graph you want.
- Click OK button to save the change and close the window.
- The values are shown in Figure 9, 10, 11.

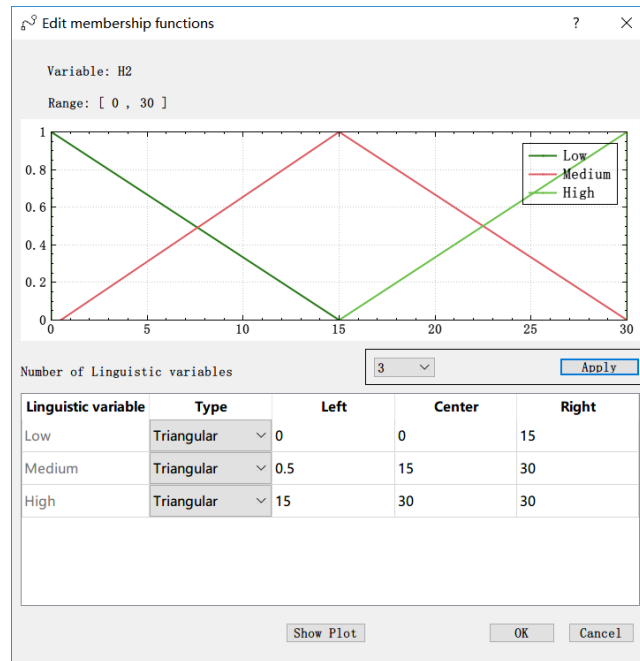


Figure 9: Membership functions of H2.

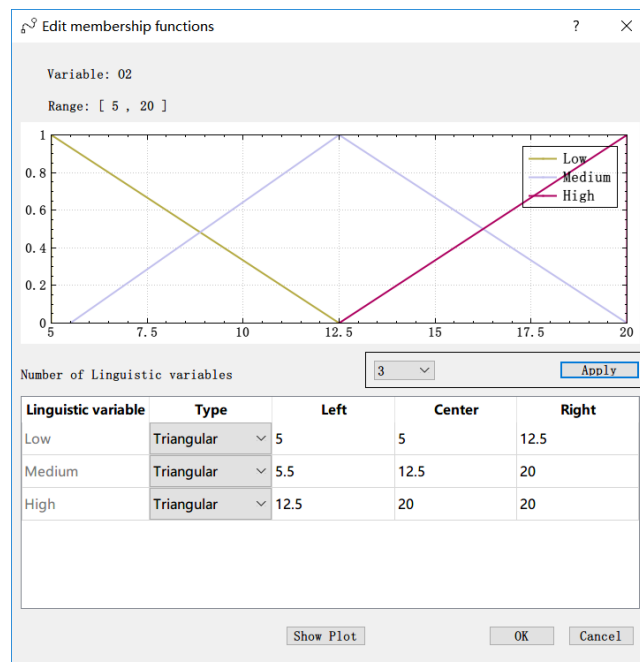


Figure 10: Membership functions of O2.

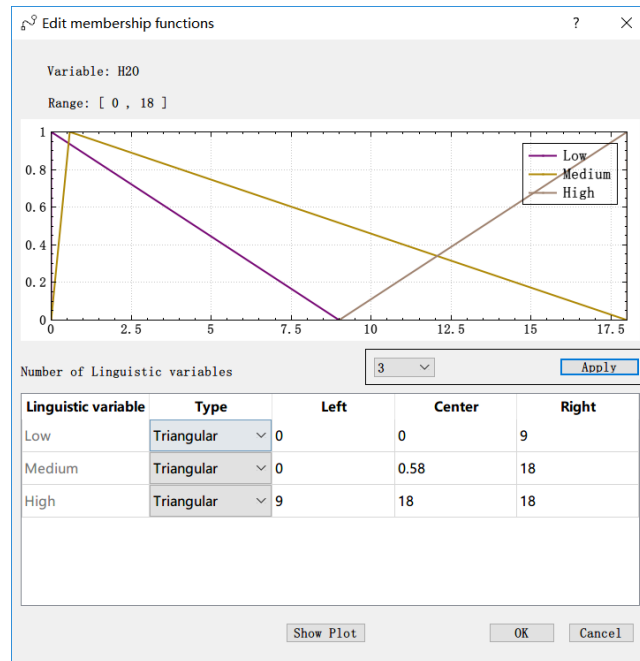


Figure 11: Membership functions of H2O.

Now we should edit rules.

- Click the Edit rules button on the center of bottom.
- Select the items of a rule, and click Add rule button to add it.
- We can select a row and click Change rule button to change it to the rule you selected in the below. Choosing a row and clicking the Delete rule button can be used to delete the rule.
- Make sure that the meanings of input's linguistic variables and output's linguistic variables are different. For example, as is shown in Figure 12, one of the rules is - IF H2 IS Low AND O2 IS Low Then H2O IS Low. This means that the low values of H2 and O2 will cause low value change of H2O, not meaning that the value of H2O will be low.
- Click the OK button to save the changes and close the window.
- The rules are shown in Figure 12.

Now click the OK button of FIS editor and click the OK button of arc to return to the main interface. The net should be like Figure 13.

Edit rules
?
X

	IF	Variable	IS	Term	AND	Variable	IS	Term	Then	Variable	IS	Term
1	IF	H2	IS	Low	AND	O2	IS	Low	Then	H2O	IS	Low
2	IF	H2	IS	Medium	AND	O2	IS	Low	Then	H2O	IS	Low
3	IF	H2	IS	High	AND	O2	IS	Low	Then	H2O	IS	Low
4	IF	H2	IS	Low	AND	O2	IS	Medium	Then	H2O	IS	Low
5	IF	H2	IS	Low	AND	O2	IS	High	Then	H2O	IS	Low
6	IF	H2	IS	Medium	AND	O2	IS	Medium	Then	H2O	IS	Medium
7	IF	H2	IS	High	AND	O2	IS	Medium	Then	H2O	IS	Medium
8	IF	H2	IS	Medium	AND	O2	IS	High	Then	H2O	IS	Medium
9	IF	H2	IS	High	AND	O2	IS	High	Then	H2O	IS	High

If H2 is
and O2 is
Then H2O is

Low
Low
Low

Add rule
Change rule
Delete rule
OK
Cancel

Figure 12: Edit rules.

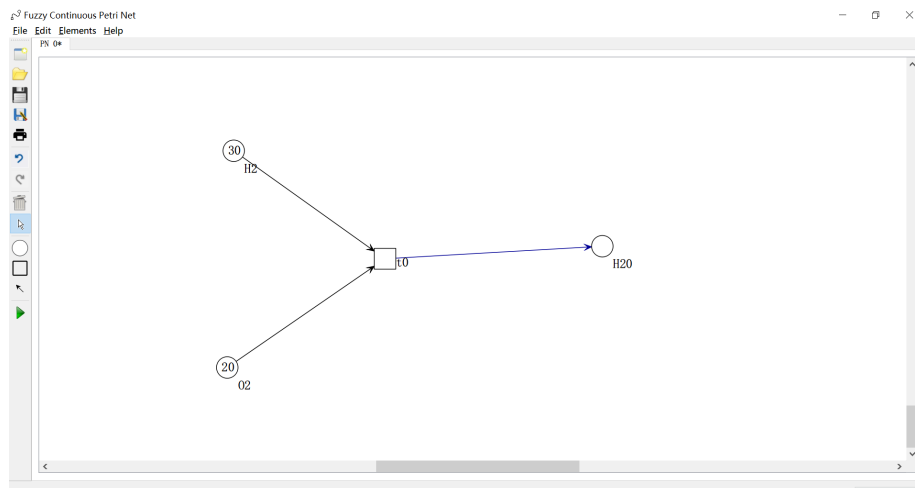


Figure 13: After FIS.

There are some differences between Mamdani and T-S. First, in T-S, when click the Edit button of variable whose type is PN_OUTPUT, there is no need to edit its membership functions. After editing variables' membership functions, click Edit rules button to edit rules.

Dialog box: Edit FIS properties

FIS Name: FIS (H2, O2; H2O) Time Step: 0.005

FIS Type: T-S Number of input variables: 2 [Apply]

Defuzzification: centroid

List of variables

edit	Variable	Type	Range-Min	Range-Max
Edit	H2	PN_INPUT	0	30
Edit	O2	PN_INPUT	5	20
Edit	H2O	PN_OUTPUT	NO NEED	NO NEED

[Edit rules] [OK] [Cancel]

Figure 14: T-S FIS.

Second, when editing rules, the last column should be edited by the user as is shown in Figure 15. Add rules first and then change the content of last column by double clicking the content of last column of each row and entering the formulas. Click OK button to save changes. Other steps are similar to Mamdani.

Dialog box: Edit rules

	IF	Variable	IS	Term	AND	Variable	IS	Term	Then	Variable	IS	Term
1	IF	H2	IS	Low	AND	O2	IS	Low	Then	H2O	IS	0.001*H2^2*O2^0.5
2	IF	H2	IS	Low	AND	O2	IS	High	Then	H2O	IS	0.001*H2^2*O2^0.5
3	IF	H2	IS	High	AND	O2	IS	High	Then	H2O	IS	0.001*H2^2*O2^0.91
4	IF	H2	IS	High	AND	O2	IS	Low	Then	H2O	IS	0.001*H2^2*O2^0.5

If H2 is Low and O2 is High Then H2O is 0.001*H2^2*O2^0.5

[Add rule] [Change rule] [Delete rule] [OK] [Cancel]

Figure 15: Enter rules when using T-S.

3 Simulation

After modeling, click the simulation button on the tool bar to use the simulation function.

3.1 Run simulation

In the simulation shown as Figure 16, the user can first set simulation parameters, then click the Start Simulation button to start simulation. The settings include:

- Setting interval start.
- Setting interval start.
- Choosing ODE solver.
- Setting time step.
- Choosing to check negative value or not.
- Please make sure the time step is smaller than FISs' step size. Otherwise the result might be incorrect.

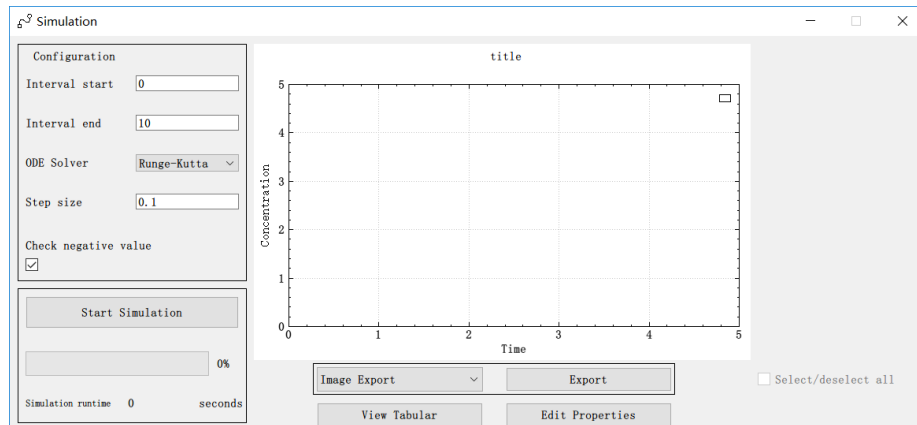


Figure 16: Simulation.

3.2 Show simulation results

During simulation, the progress bar shows the progress. And the time is updated. After simulation, the graph will show. The user can zoom in or zoom out to see the details and check or uncheck the boxes on the right to show or hide the according variables.

To avoid misoperation, after entering the simulation, most of the functions in main interface such like adding items, can not be used. So if you want to be back to main interface, remember to click normal cursor on the left tool bar. Otherwise you will find that most of the function can't use.

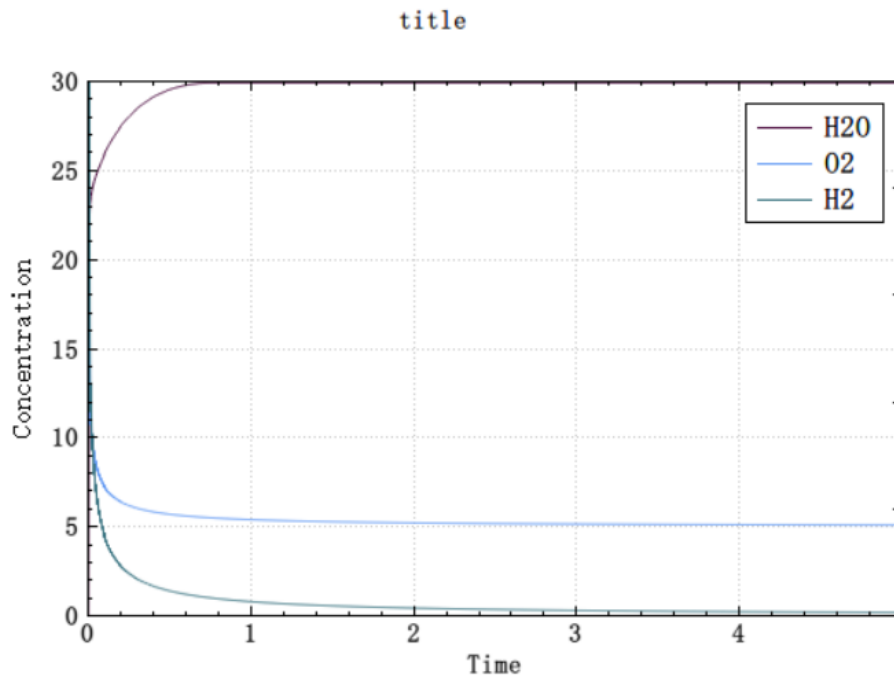


Figure 17: Simulation result (Mamdani).

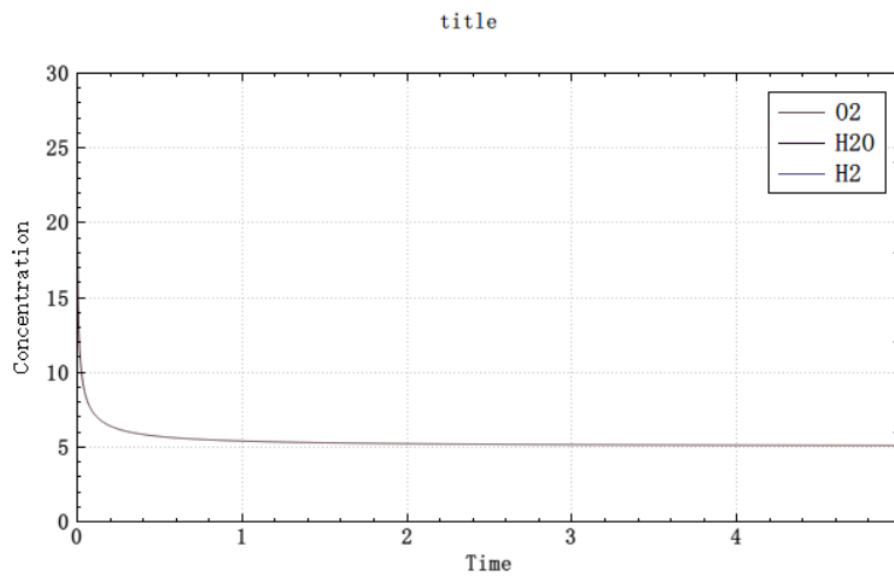


Figure 18: Only show the result of O₂.

Now close the simulation and click normal cursor button to back to normal. Delete the FIS arc and add a new arc whose expression is 2. Start simulation

again to see the result. Actually, the user can choose to use no FIS, Mamdani, T-S, hybrid FIS to run the simulation. The result can be a little bit different, but still shows the general trend.

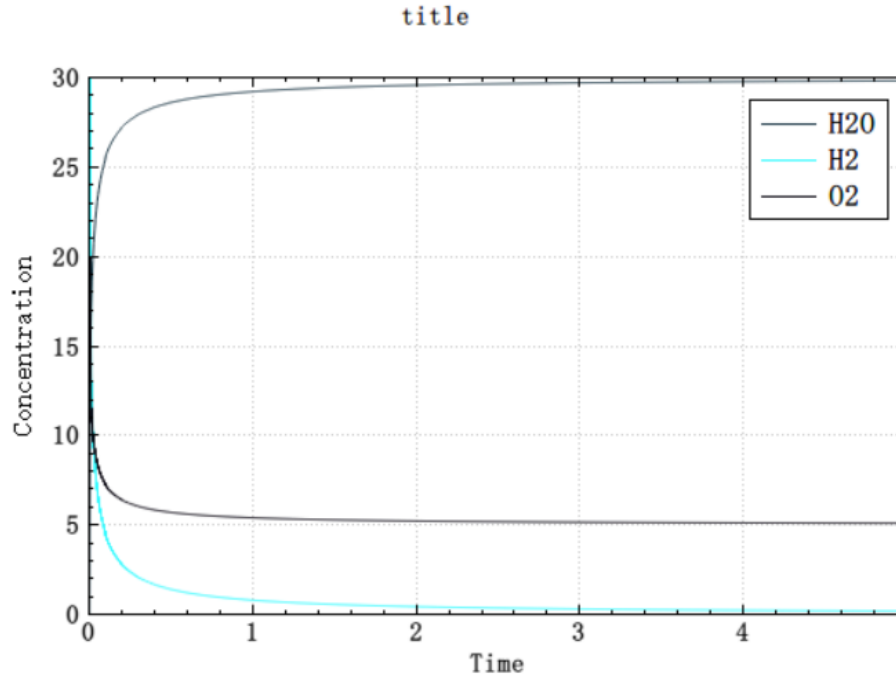


Figure 19: Simulation result (No FIS).

3.3 Export simulation results

The user can choose the export method and click the Export button to export.

3.4 Rich functions

Different tools are provided where located at the center of bottom. The user can view the Tabular to see the values of variables by clicking the View Tabular button or edit properties by clicking the Edit Properties.

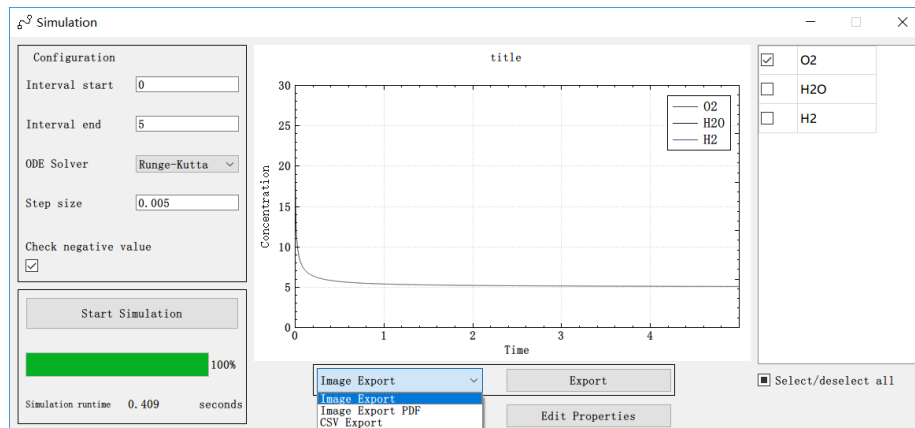


Figure 20: Export result.

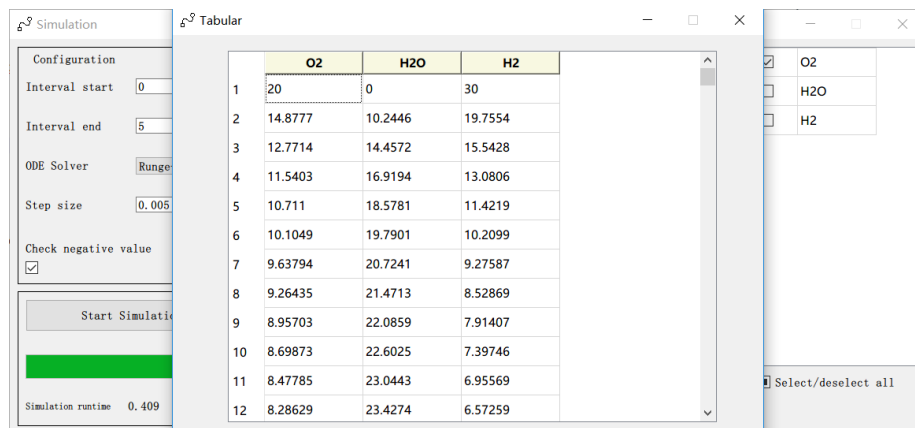


Figure 21: View Tabular.

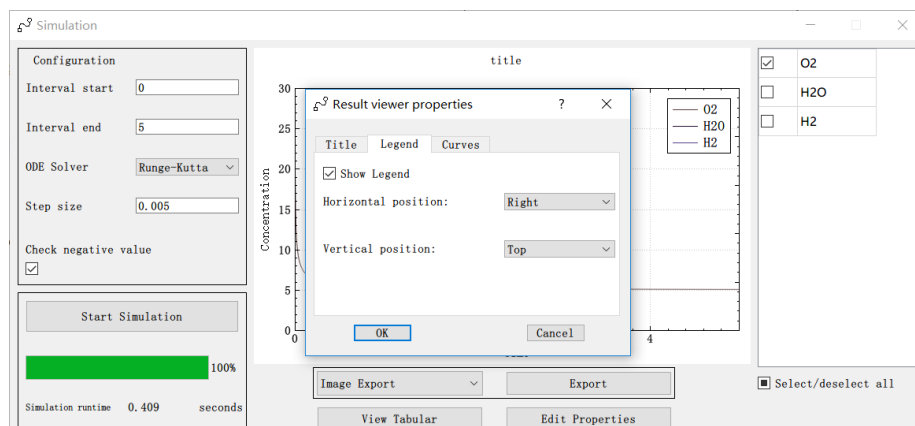


Figure 22: Edit properties.

4 Examples

In this section, we will show three examples. And we provide the file of the examples named A.B.C.D where A is the name of the example, and B is the FIS type we used, and C is the interval end and D is the step size. The default interval start is 0. Open the application and click the Open button on the tool bar or use Ctrl+O to open the file. Then you can run the simulation.

4.1 1D Diffusion Reaction

4.1.1 Introduction

The first example is 1D diffusion reaction.

4.1.2 Model

The model is shown as Figure 23.

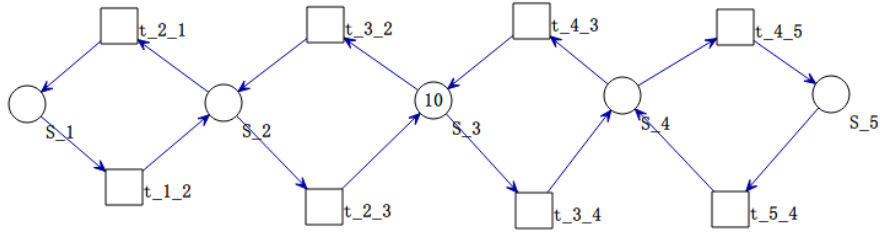


Figure 23: The model of 1D diffusion reaction.

Table 2: Transition functions of 1D Diffusion Reaction.

Transition	Function
t_1_2	MassAction(1)
t_2_1	MassAction(1)
t_2_3	MassAction(1)
t_3_2	MassAction(1)
t_3_4	MassAction(1)
t_4_3	MassAction(1)
t_4_5	MassAction(1)
t_5_4	MassAction(1)

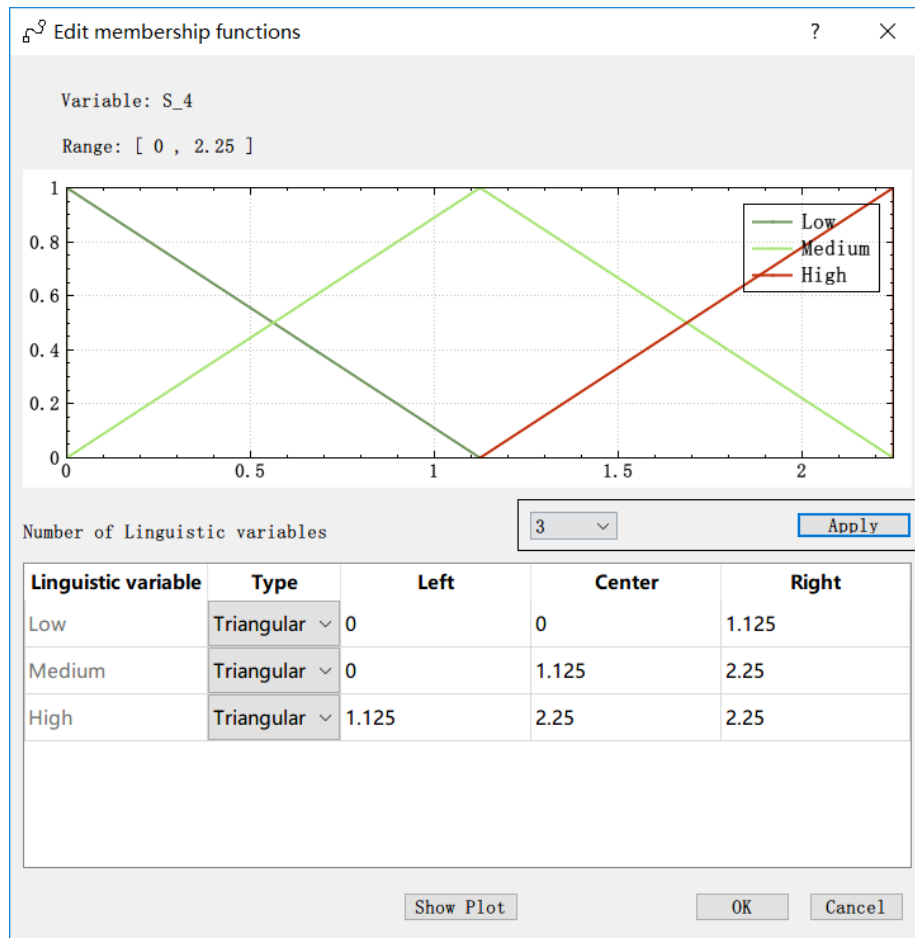


Figure 26: Membership functions of 1D Diffusion Reaction using T-S.

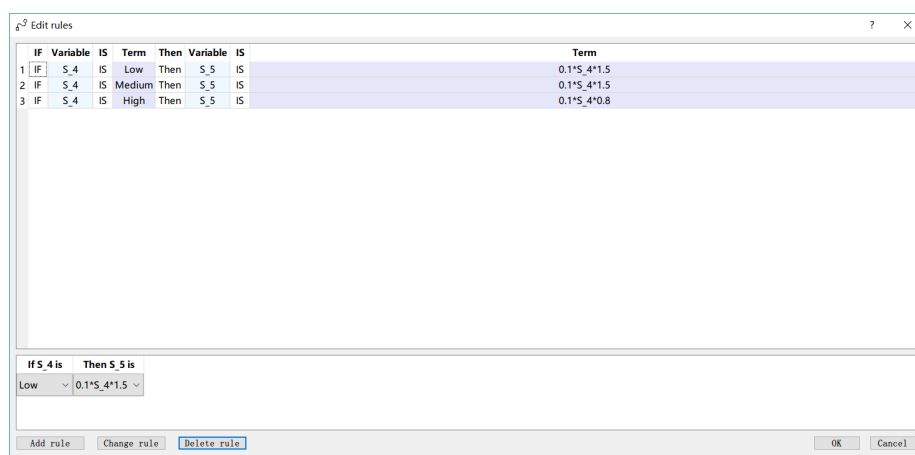


Figure 27: Rules of 1D Diffusion Reaction using T-S.

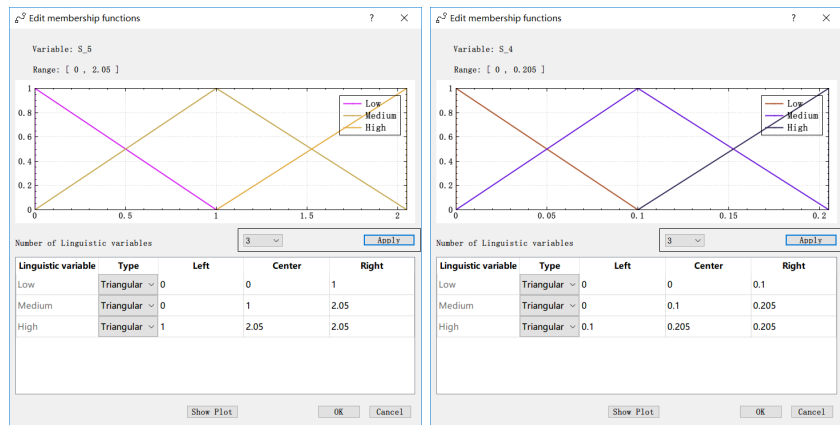


Figure 28: Membership functions of 1D Diffusion Reaction using hybrid FIS (Mamdani).

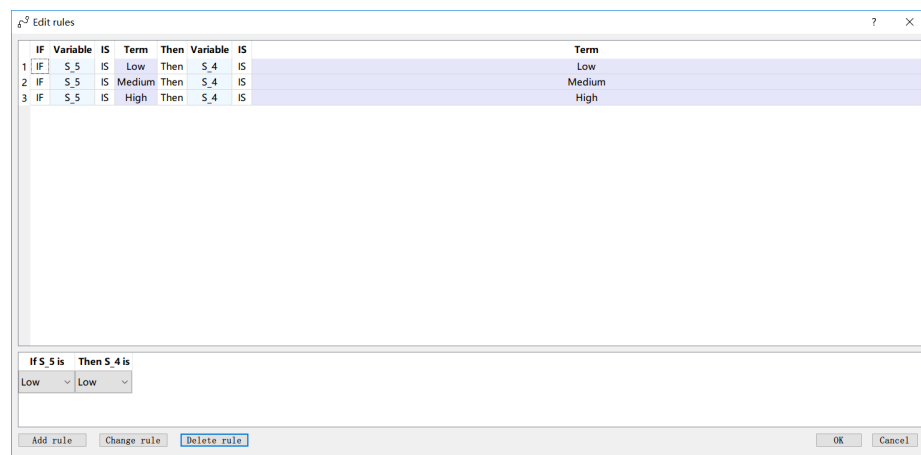


Figure 29: Rules of 1D Diffusion Reaction using hybrid FIS (Mamdani).

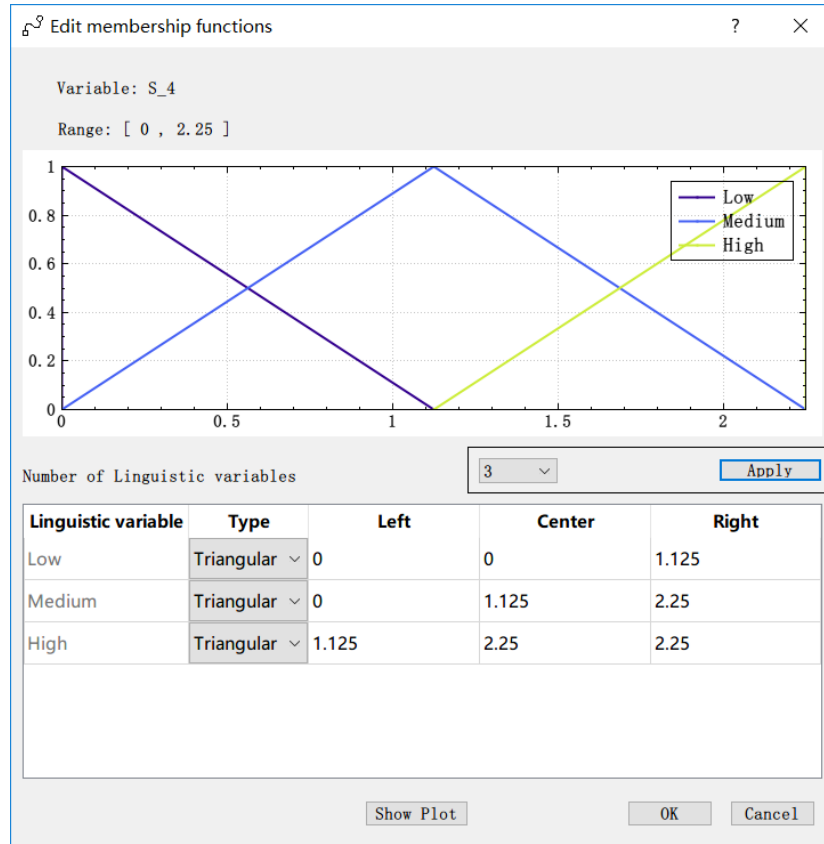


Figure 30: Membership functions of 1D Diffusion Reaction using hybrid FIS (T-S).

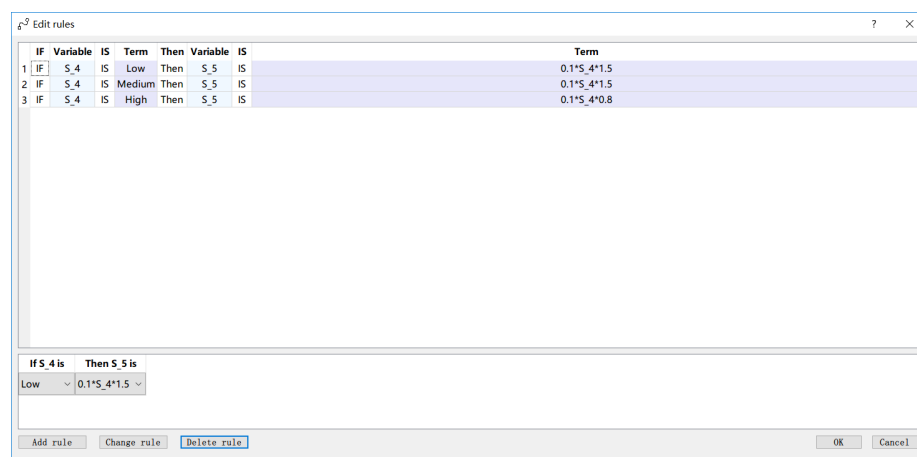


Figure 31: Rules of 1D Diffusion Reaction using hybrid FIS (T-S).

4.1.3 Simulation result

In this example, we use only ODEs, Mamdani, T-S, and hybrid FIS. When using Mamdani, we choose to change the arc from t_{5_4} to S_4 to FIS. When using T-S, we choose to change the arc from t_{4_5} to S_5 to FIS. When using hybrid FIS, we choose to change the arc from t_{5_4} to S_4 to Mamdani and change the arc from t_{4_5} to S_5 to T-S. In method of only ODEs, the value of all variables eventually becomes 2 while in other three methods, the final values of the variables are not 2, but are all very close to 2. And the trends of curves are consistent.

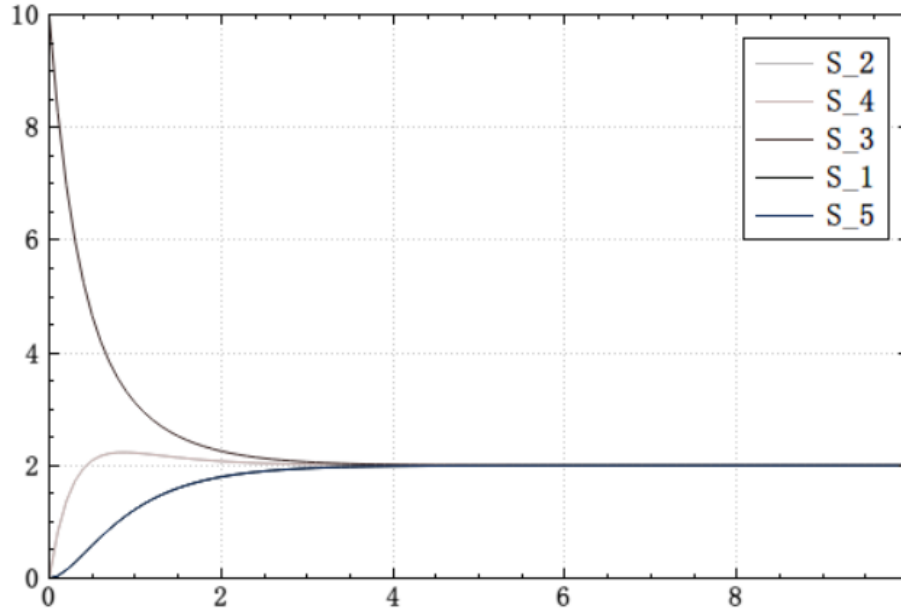


Figure 32: Simulation result of 1D diffusion reaction using only ODEs.

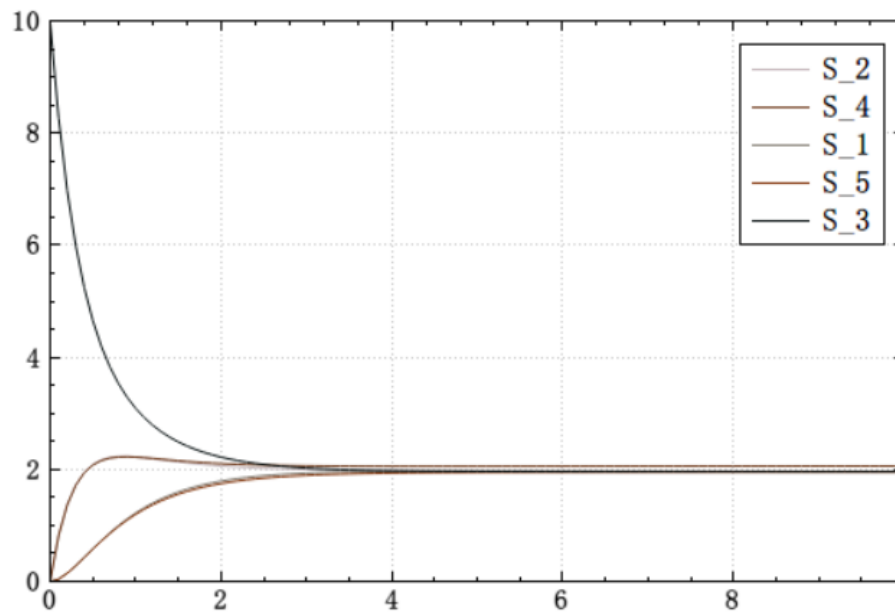


Figure 33: Simulation result of 1D diffusion reaction using Mamdani.

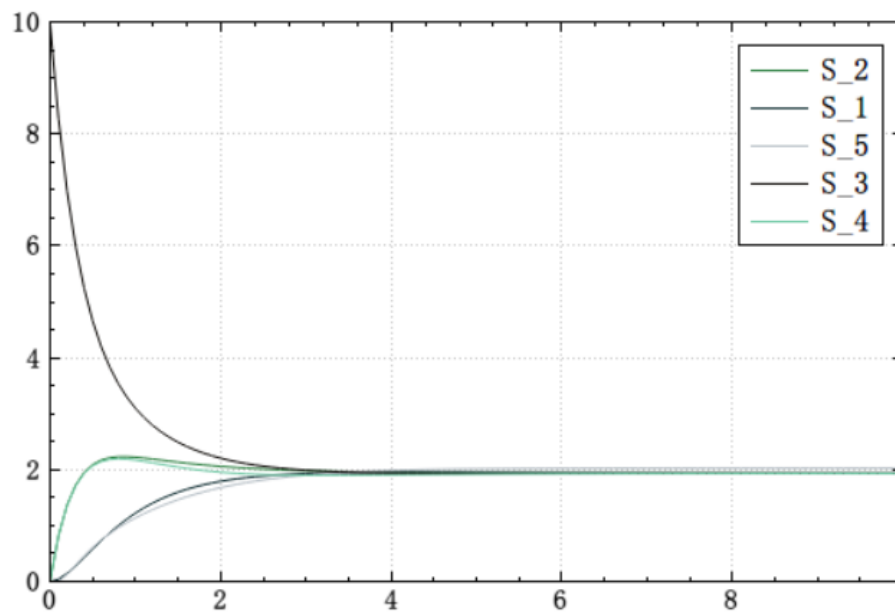


Figure 34: Simulation result of 1D diffusion reaction using T-S.

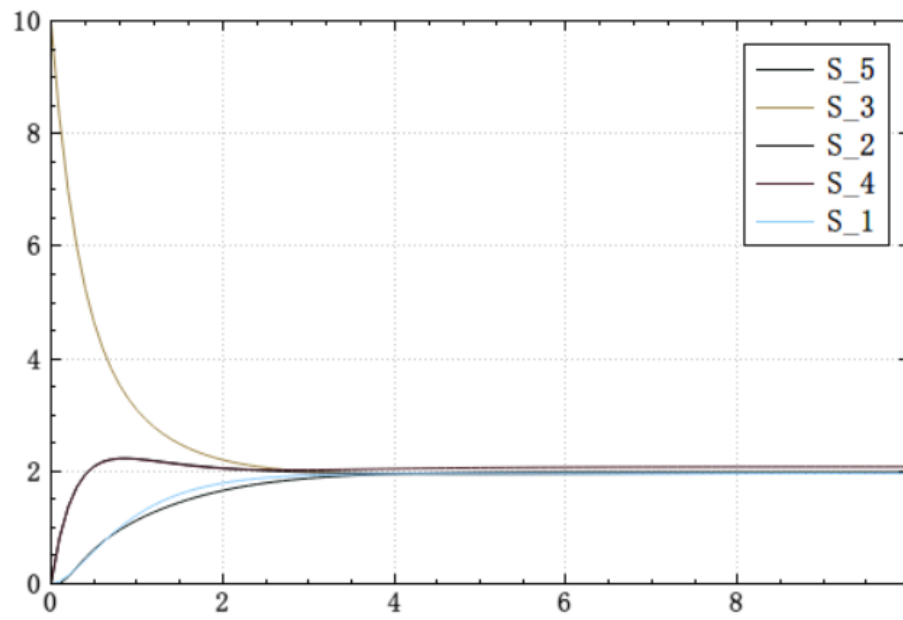


Figure 35: Simulation result of 1D diffusion reaction using hybrid FIS.

4.2 Enzyme

4.2.1 Introduction

The second example is enzyme. Enzymes are macromolecular biological catalysts. Enzymes accelerate chemical reactions. The molecules upon which enzymes may act are called substrates and the enzyme converts the substrates into different molecules known as products. Almost all metabolic processes in the cell need enzyme catalysis in order to occur at rates fast enough to sustain life. Metabolic pathways depend upon enzymes to catalyze individual steps. [6]

4.2.2 Model

The model is shown as Figure 36.

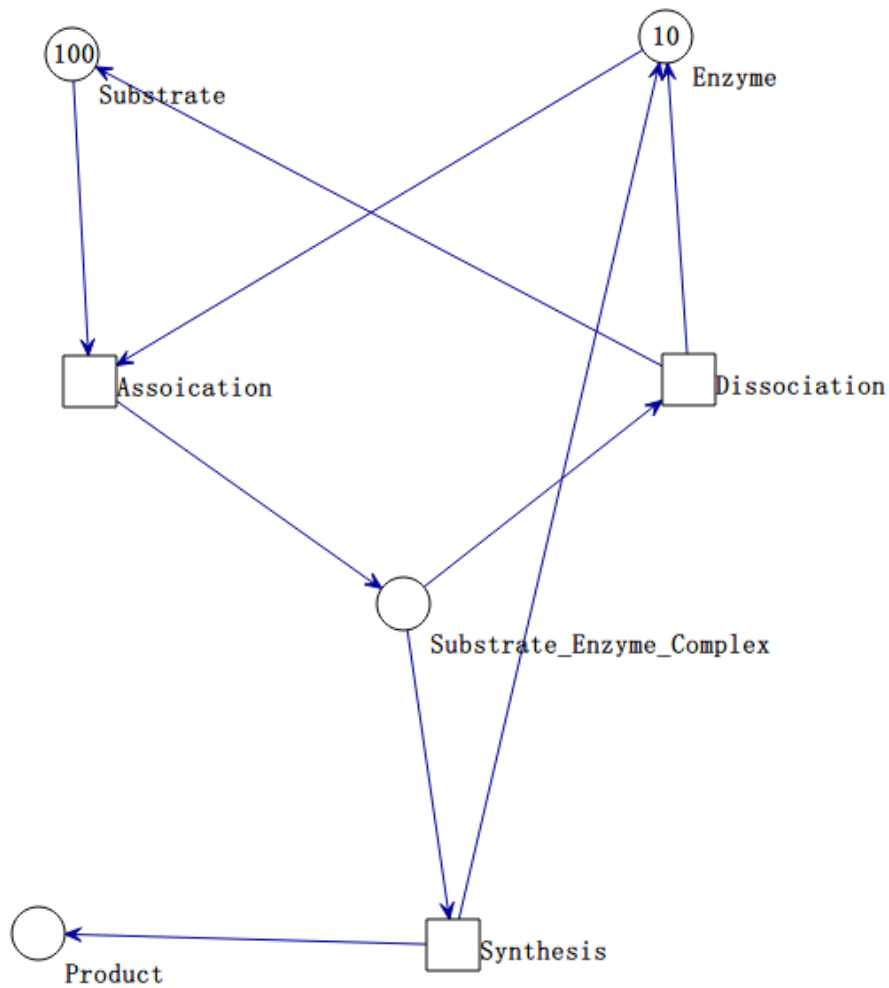


Figure 36: The model of enzyme.

Table 3: Transition functions of enzyme.

Transition	Function
Assoication	MassAction(0.1)
Dissociation	MassAction(0.1)
Synthesis	MassAction(0.1)

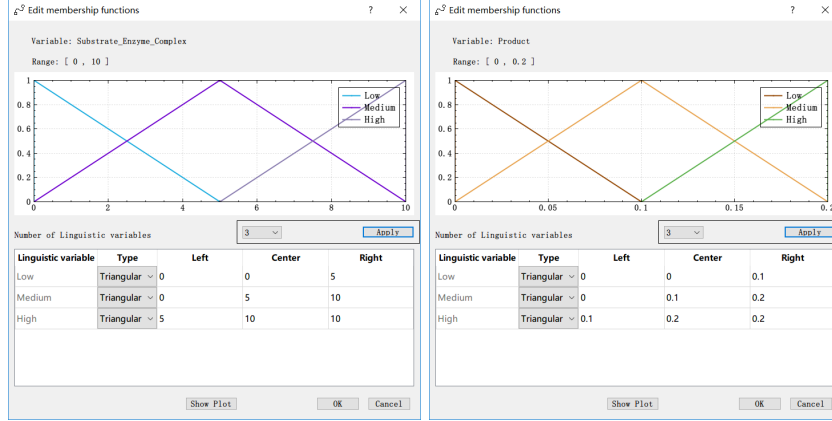


Figure 37: Membership functions of enzyme using Mamdani.

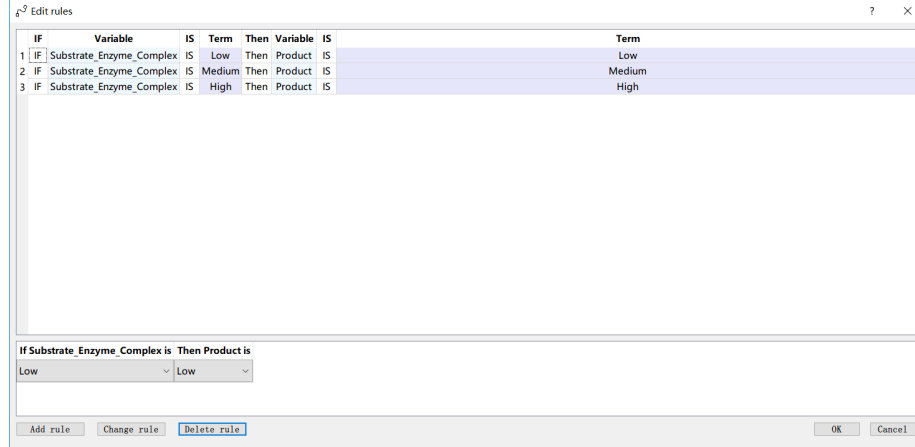


Figure 38: Rules of enzyme using Mamdani.

4.2.3 Simulation result

In this example, we use only ODEs and Mamdani. When using Mamdani, we choose to change the arc from Synthesis to Product to FIS. As we can see, the results are almost exactly the same.

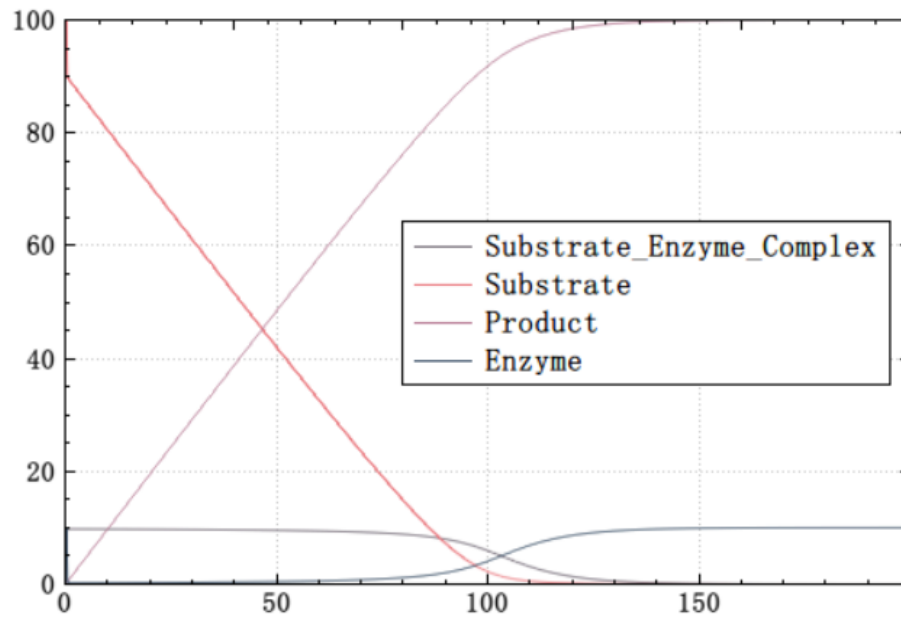


Figure 39: Simulation result of enzyme using only ODEs.

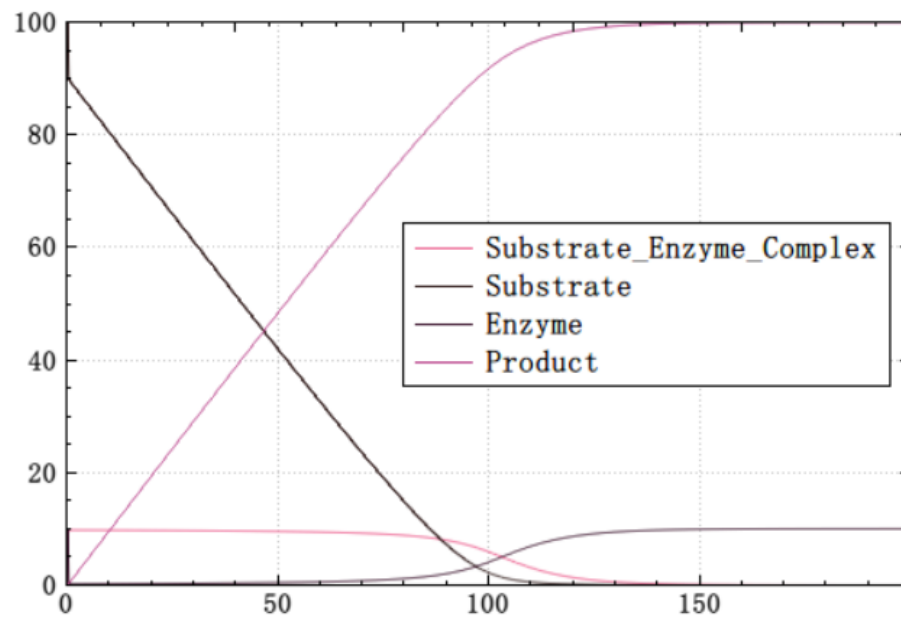


Figure 40: Simulation result of enzyme using Mamdani.

4.3 RKIP

4.3.1 Introduction

The last example is RKIP. The Raf kinase inhibitor protein (RKIP) is a kinase inhibitor protein, that regulates many signaling pathways within the cell. RKIP is a member of the phosphatidylethanolamine-binding protein family and has displayed disruptive regulation on the Raf-1-MEK1/2, ERK1/2 and NF-kappaB signalling pathways, by interaction with the Raf-1 kinase. [7]

4.3.2 Model

The model is shown as Figure 41.

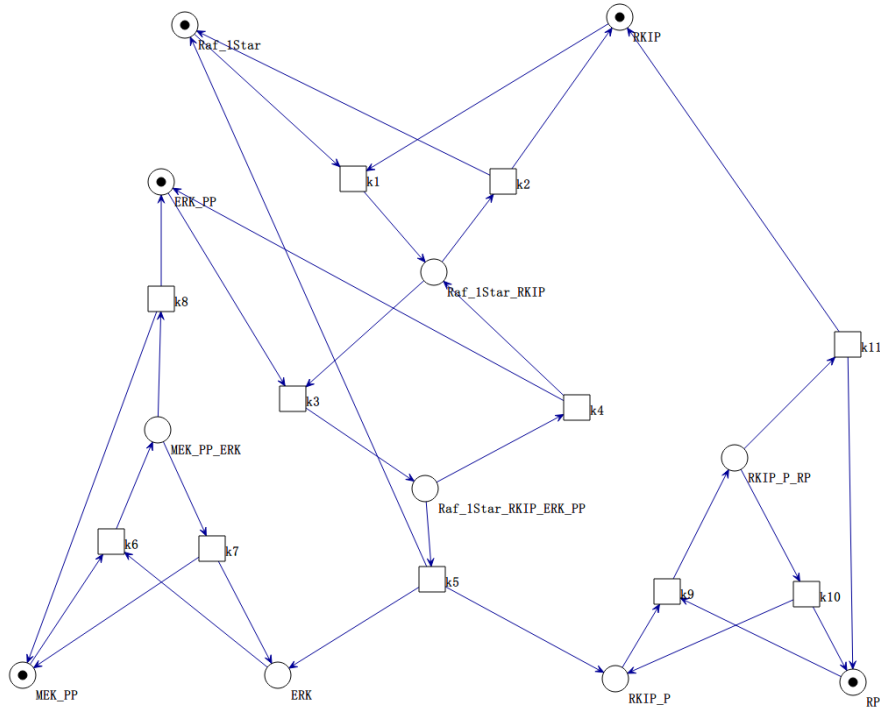


Figure 41: The model of RKIP.

4.3.3 Simulation result

In this example, we use only ODEs and T-S. When using T-S, we choose to change the arc from k3 to Raf_1Star_RKIP_ERK_PP to FIS. As we can see, the results are almost consistent. Pay attention to the bold blue curves of Figure 44 and Figure 45. The final result is a little different. But we can adjust the rules to get a better result.

Table 4: Transition functions of RKIP.

Transition	Function
k1	MassAction(0.53)
k2	MassAction(0.0072)
k3	MassAction(0.625)
k4	MassAction(0.00245)
k5	MassAction(0.0315)
k6	MassAction(0.6)
k7	MassAction(0.0075)
k8	MassAction(0.071)
k9	MassAction(0.92)
k10	MassAction(0.00122)
k11	MassAction(0.87)

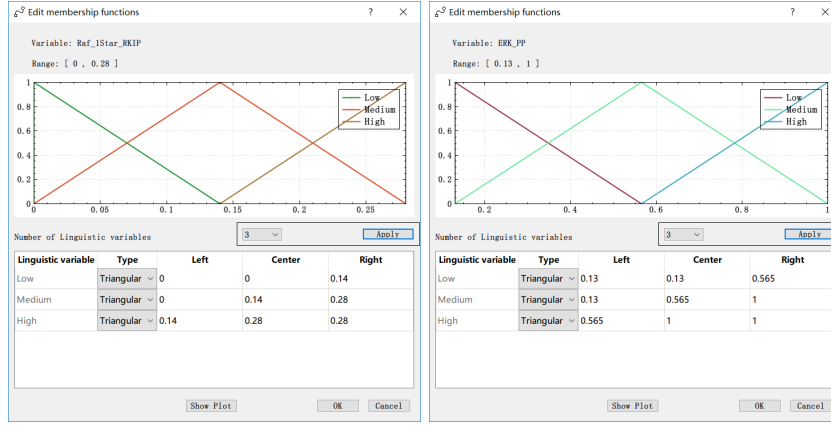


Figure 42: Membership functions of RKIP using T-S.

Figure 43 shows a screenshot of the 'Edit rules' dialog box for T-S fuzzy inference. It displays a table of rules with columns for IF, Variable, IS, Term, AND, Variable, IS, Term, Then, Variable, IS, and Term. The rules are numbered 1 to 9. Below the table, there is a summary of the rules and buttons for 'Add rule', 'Change rule', and 'Delete rule'.

IF	Variable	IS	Term	AND	Variable	IS	Term	Then	Variable	IS	Term
1	IF Raf_1Star_RKIP	IS Low	AND ERK_PP	IS Low	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP				
2	IF Raf_1Star_RKIP	IS Medium	AND ERK_PP	IS Low	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP				
3	IF Raf_1Star_RKIP	IS High	AND ERK_PP	IS Low	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP				
4	IF Raf_1Star_RKIP	IS High	AND ERK_PP	IS Medium	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*1.2				
5	IF Raf_1Star_RKIP	IS High	AND ERK_PP	IS High	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*1.2				
6	IF Raf_1Star_RKIP	IS Medium	AND ERK_PP	IS Medium	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.9				
7	IF Raf_1Star_RKIP	IS Medium	AND ERK_PP	IS High	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.9				
8	IF Raf_1Star_RKIP	IS Low	AND ERK_PP	IS High	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.8				
9	IF Raf_1Star_RKIP	IS Low	AND ERK_PP	IS Medium	Then Raf_1Star_RKIP.ERK_PP	IS	0.0625*Raf_1Star_RKIP*ERK_PP*0.8				

IF Raf_1Star_RKIP is and ERK_PP is Then Raf_1Star_RKIP.ERK_PP is

Low Low 0.0625*Raf_1Star_RKIP*ERK_PP

Add rule Change rule Delete rule OK Cancel

Figure 43: Rules of RKIP using T-S.

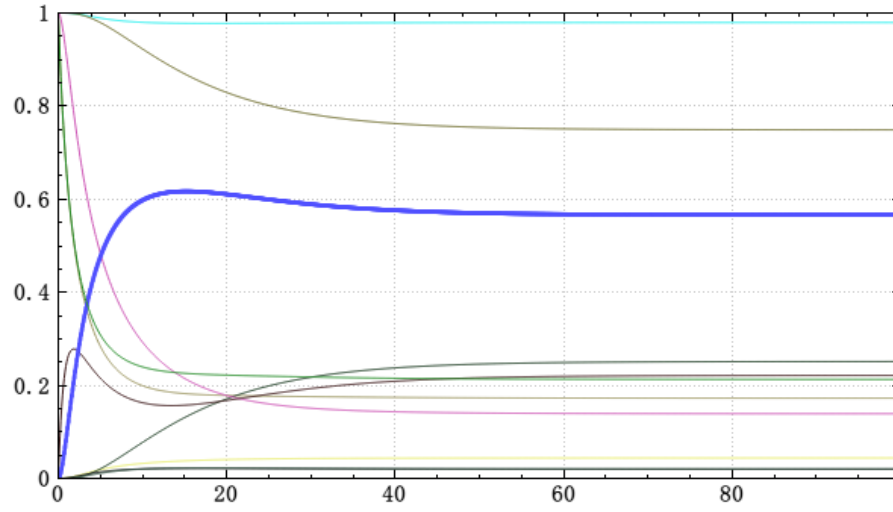


Figure 44: Simulation result of RKIP using only ODEs.

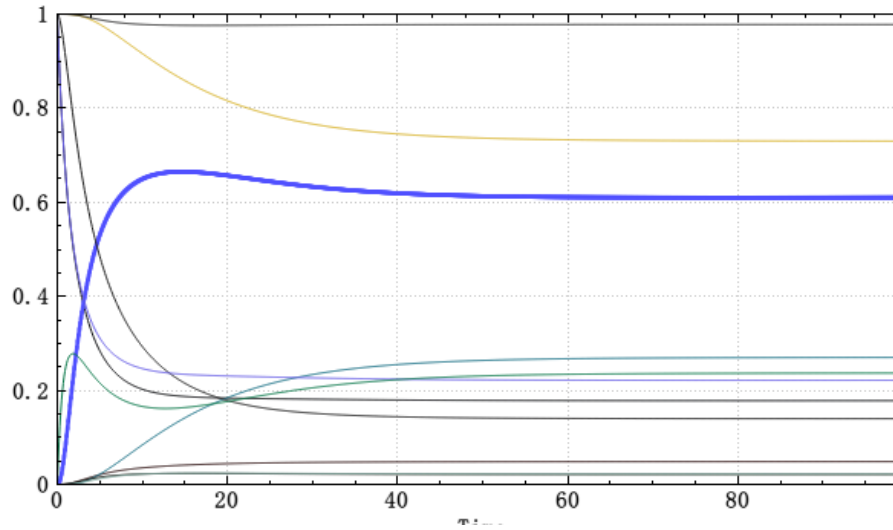


Figure 45: Simulation result of RKIP using T-S.

5 References

- [1] Petri net: https://en.wikipedia.org/wiki/Petri_net
- [2] Fuzzy logic: https://en.wikipedia.org/wiki/Fuzzy_logic
- [3] Fuzzy Inference Process: <https://www.mathworks.com/help/fuzzy/fuzzy-inference-process.html>
- [4] Mamdani-Type Fuzzy Inference: <https://www.mathworks.com/help/fuzzy/what-is-mamdani-type-fuzzy-inference.html>
- [5] Sugeno-Type Fuzzy Inference: <https://www.mathworks.com/help/fuzzy/what-is-sugeno-type-fuzzy-inference.html>
- [6] Enzyme: <https://en.wikipedia.org/wiki/Enzyme>
- [7] Raf kinase inhibitor protein: https://en.wikipedia.org/wiki/Raf_kinase_inhibitor_protein