FisPro : An open source portable software for fuzzy inference systems

fispro@supagro.inra.fr

Version 3.5

January, 2013

FisPro (*Fuzzy Inference System Professional*) allows to create fuzzy inference systems and to use them for reasoning purposes, especially for simulating a physical or biological system. Fuzzy inference systems are briefly described in the fuzzy logic glossary given in the user documentation. They are based on fuzzy rules, which have a good capability for managing progressive phenomenons.

First of all, the FisPro implementation allows to design fuzzy systems from the expert knowledge available in a given field, for instance in winemaking. This approach is illustrated by an example given in the user guide *Quickstart with FisPro*.

FisPro also allows the complete design of a fuzzy inference system from the numerical data related to the problem under study. Many automatic learning methods unfortunately lead to "black box" systems. In FisPro, constraints are imposed to the algorithms to make the reasoning rules easy to interpret([16]), so that the user understands how the fuzzy system operates. This novel approach is one of the originalities of the software. Some examples are given in the user guide *Induction with FisPro*.

Both approaches, expert rule design and automatic induction, can be combined to create more complete and better performing systems. FisPro offers educational tools that illustrate the reasoning mechanism, and other tools to measure the system performance on datasets.

This software is made of two distinct parts: a C++ function library, which can be used independently, and a graphical Java interface, which implements most functionalities if the C++ library. It is portable, and can run on most existing platforms.

Authors

- Design and C++ implementation:
  Serge GUILLAUME, Irstea, UMR ITAP
  (`http://ser.gui.free.fr/homepage`),
  Brigitte CHARNOMORDIC, INRA, UMR MISTEA (`http://www.inra.fr`)
- Java interface : Jean-Luc LABLEE, Irstea, UMR ITAP
- Contributions:
  - C++
    - François OLIVIER, optimization module, from the work by Pierre-Yves GLORENNEC, *Algorithmes d'apprentissage pour systèmes d'inférence floue*, Edns. Hermes, 1999
    - Sébastien DESTERCKE, orthogonal least squares rule induction (ols)
    - Vincent THERRY, with the support of the Envilys company, cascading systems (superfis)
    - Russel STANDISH, optimized and parallelisable functions (OPENMP standard)
    - Hazaël JONES, design of the *implicative systems* part
    - Lydie DESPERBEN, C++ implementation of the *implicative systems* part
  - Java
    - Pierre-Marie BOYER, JNI interface (Java-C++)
    - Mathieu GRELIER, data visualization
    - Anne TIREAU, java interface of the *implicative systems* part.
  - Communication
    - Jean-Michel FATOU, FisPro icons and web site design
    - Moacir Jr PEDROSO, Portuguese translation of the interface
    - Juan Luis CORTI, Spanish translation of the *Quickstart with FisPro* manual

Acknowledgments

The development of the first version of FisPro was sustained by public funds from the French government and from Languedoc-Roussillon regional funds, during a research project, COST 2000-012, coordinated by the TRANSFERTS LR association
`http://www.transferts-lr.org` with the industrial partnership of the *Cave Cooperative "La Malepère", Arzens, Aude*.

Contents

Part I
A powerful user-friendly environment

When the program is started, the main window appears. Several menus are available which are detailed in this section. Configuration information: language, working directory, are recorded in the file located in the FisPro installation directory.

The choice of the conjunction operator, which is used in calculating the matching degree of each rule premise, is done in the main window. Three possibilities are given: product, minimum and the Lukasiewicz operator, $max(0, a + b - 1)$.

In a multiple output system, the aggregation and defuzzification operators (see 1) can change with each output. They are not defined in the main window, but in each output window.

Context menus: Context menus are available in the Input, Output windows of the *FIS menu*, and in the Table window of the *Data* menu. They are called through a right mouse click.

Printing and exporting graphics : In the Input, Output windows of the *FIS menu*, in the *Rule* window, in the 2D and 3Dpartbiblifonc graphic visualization submenus of *FIS* and *Data*, two options *Print* and *Export* are available. They allow to print the displayed graphics, or to export it in one of the usual formats (eps, jpeg, gif, png, pdf).

1 Fis Menu

The option *New* allows the step by step interactive definition of a Fuzzy Inference System (FIS). This includes input and output definition, fuzzy partitioning specification and fuzzy rule definition.

Note: The FIS can be designed by hand or self-generated if a data file is open (see the *Generate a Fis without rules, Generate rules and Generate conclusions* options).

Save and Load options are available and allow to backup and reload a given FIS configuration. All files are in text format, easy to read and edit in a text editor.

Each FIS component: input, output or rule can be declared active or inactive, at any time. This functionality allows to extensively explore the FIS behaviour

without modifying the data file.

1.1 Input Variable Window

Fuzzy partitioning can be defined using a regular or irregular grid with a given number of triangular fuzzy sets. This grid has the properties of a standardized fuzzy partition

Partitioning can also be built by hand.

In any case, membership functions properties are editable, and each parameter -type, break points- can be modified at any time. Available types include triangular, trapezoidal, lower and upper semi trapezoidal, sinus, lower and upper semi sinus, Gaussian, discrete or rectangular (*door*) membership functions. The last type corresponds to crisp values. **Note:** for consistency reasons, if a data file is open, its number of columns must be at least equal to the number of inputs declared in the FIS.

Note: New input - input removal

Following the previous note, a new input cannot be added to a FIS if a data file is open. The data file must be closed first.

1.2 Output Variable Window

An output variable has the same parameters as an input variable, and some extra ones:

- fuzzy or crisp output
- implicative/not implicative output
- classification option
- disjunction operator for rule aggregation
- defuzzification operator

For conjunctive outputs, whatever the output nature, crisp or fuzzy, two aggregation operators, max and sum, are available (see part II, section 1

For conjunctive fuzzy outputs, several defuzzification operators are available: weighted area, mean-max and the Sugeno operator. The weighted area defuzzification is analogous to centroid defuzzification, the only difference being that areas shared by two fuzzy sets are counted twice. The mean-max operator returns the middle point of the segment corresponding to the alpha cut which has the greatest

matching degree. In case of ambiguous maxima, a warning message is displayed. The Sugeno operator returns as the rule conclusion the middle point of the corresponding fuzzy set kernel.

For crisp outputs, two defuzzification operators can be used: MaxCrisp, a mean of maximum adaptation for crisps outputs, and the Sugeno operator. In this case, the output is limited to a constant value, to ensure interpretability. The Sugeno operator returns an output equal to the weighted sum of the rule conclusions, the weight being the matching degree.

Note on defuzzification of fuzzy outputs

For all fuzzy output defuzzification operators, a correction is applied to the upper bound (resp. lower) of the upper semi-trapezoidal MF (resp. lower), located at the fuzzy partition edge. This allows to infer extreme values (output range boundaries). This correction is automatically applied when building a whole partition: regular or irregular grid, or when generating a FIS without rules.
If these bounds are edited by hand, the correction is launched by a change in the choice of the defuzzification operator.
This correction can also be applied when loading a previously saved FIS. To do so, one needs to check the checkbox labelled *Ensure the output bounds to be inferred*.

Alarm threshold parameter

The *threshold* parameter is a tolerance threshold associated to a warning message at the defuzzification stage (see the alarm paragraph in section 2).

Classification option

If the classification option is chosen:

- crisp output: the inferred output will be rounded to the closest class label. Class labels are built using the corresponding data file column, if a data file is open and if this column is available. If not, the rule conclusions are used as class labels.
- fuzzy output: membership degrees of the inferred value to each fuzzy set defined for the output variable will appear in the inference results.

Note: Discrete MFs

Discrete MFs in a fuzzy output are not compatible with a weighted area defuzzification.

New output - output removal

As for inputs, a new output cannot be added to a FIS if a data file is currently opened. The data file must be closed first.

Implicative rules

The choice between implicative rule systems and conjunctive ones is done through the output paramers. Indeed the input variable partitioning, as well as the computation of the rule matching degree for given input data, are independent of the rule nature.

Implicative rules are available using an *Impli* checkbox in the output window. If this option is chosen, the only available defuzzification operator is called *Impli*, and the choice of the implication operator is done by selecting the rule agregation method (*Disjunction* field). Three operators are available: *Resher-Gaines*, *Goguen* et *Gödel*. The possibility distributions inferred using one of these operators all have the same kernel, and only differ by their supports.

The agregation method specific characteristic of implicative rules imposes a particular output partition if one wants to avoid having too many conflicting rules and empty inferred outputs. We advise to use the *quasi-standard partitions* (QSF) derived from the *standard fuzzy partitions* (SFP). They are automatically built when using the menu option *Regular Grid*.

The choice of implicative rules forbids the use of a *Crisp* output. The *Regular Grid* option builds a QSP partition, which is adapted to implicative rules.
When transforming a conjunctive output into an implicative one, by checking the *impli* checkbox, the partition modification from SFP to QSP is systematically proposed.

Conversely, when transforming an implicative output into a conjunctive one by unchecking the *Impli* checkbox, the current QSP partition will be proposed to be transformed into a SFP one.

The *Classification* option has no effect on implicative rules.

1.3 Rule Window

The *Rule Window* is or fuzzy rule base specification. Rules are built using input and output labels which must have been previously defined. An empty string can be used in a rule. It means that the variable is absent from the rule. It will not be taken into account in the inference process.

The *Active* column allows to activate/inactivate rules.

Warning: Only active rules are stored when saving the FIS configuration file.

Rules are displayed as a table (1 column per input/output). They can be sorted by column.

An option in the *Display* menu allows to display or not display inactive variables. This option is useful to deal with highly dimensional systems.

Another option in the *Display* menu allows to give a weight to rules. The weights must be positive and are stored in the FIS configuration file, with 3 decimal digits, as soon as one weight is given a value different from default, which is

equal to 1. Expert weights change the inference result. All rule matching degrees are multiplied by the corresponding expert weights at the rule aggregation functions (sum or max). Note: for fuzzy outputs, the effect of weights greater than 1 may may saturate quickly. For fuzzy output defuzzification operators, except *SugenoFuzzy*, the weight sets the alpha-cut level of the relevant output MF. It is therefore lower than 1.

1.4 Inference Window

There are two ways to infer using FisPro: inference using a data file, which is available from the *Data* menu, and will be described later, and graphical inference, which we now detail. Graphical inference is displayed as an array. Each line corresponds to an active rule, and each column to one fuzzy set in the rule premise or conclusion.

The user can change the value of an input variable by moving a cursor, or entering the numerical value, and see the effect onto the fuzzy inference system output. The matching degree of each rule is displayed, as well as the result of rule aggregation.

The inference window is for educational purposes mainly and is limited to the use of small size systems, in terms of variable and rule number.

This window is implemented as a table structure, this allows rule sorting using a click in any column header.

Any change in other windows takes effect immediately. The inference being done dynamically, the results displayed correspond to the current configuration.

Implicative rules

Two inference mechanisms are implemented in Fispro. The mechanism used by conjunctive rules is called FITA (First Infer Then Aggregate) and allows to separately infer a conclusion value for each rule before aggregating these values. The same mechanism is also used for implicative rules when input data are precise values. If this is not the case, when at least one input value is imprecise, a different mechanism is requiered, called FATI (First Aggregate Then Infer).

For a detailed comparison of the two types of rules, the inference mechanisms and the FATI implementation chosen in Fispro, see [13].

In the interface, when the FATI algorithm is used, the rule conclusion corresponding to the input data is not displayed as it cannot be calculated.

Fuzzy data

The inference window menu includes an option called *Fuzzy Data*, with is again divided into two options:

- *Template*: used to transform each precise input into a fuzzy number centered on that value. Two parameters are required: the kernel and support width. The support width is constrained to be bigger than or equal to the kernel one. A different template can be defined for each input, it can be saved into a file with *.tpl* extension, and reloaded from such a file.

  In the inference itself, the cursor allows to move the central value, and the fuzzy number will be automatically built using this central value and the given template. Manual inference with fuzzy data is available for all types of rules. It is limited to 2 inputs and 1 output if the system has at least one implicative output.

  This option is not available yet for the inference from a data file.

- *Alpha-cut*: allows to specify the α cut number used to approximate the imprecise input value in the FATI inference mechanism. This number is common to all system inputs.

1.5 System behaviour

This submenu displays the variation of a FIS output in function of the variations of one or two input variables, the other input variables being set to a fixed value.

Remark: the graphics displayed here do not take into account the current data file. They are generated only according to the FIS characteristics: partitions and rules. The curves or surfaces are built using an interpolation grid which covers the chosen data range and respects the given resolution.

The **Visualize** option of the *Data* menu displays a 2D or 3D graphics based on the actual points of a data file. No interpolation is done.

1. **section**

   2D (X,Y) graphics that displays an output Y on the y axis,versus an input variable X on the x axis.

   Range setting

   For the X variable, the display range can be set (min and max), as well as the number of points used for interpolating.

   When the FIS has more than one active input, it is necessary to set the values for all other inputs.

   The *Breakpoints* checkbox imposes the choice of the MF breakpoints as possible values. To set a value for a variable, incrementation or decrementation arrows located on the right of the variable name can be used. The left

and right arrows of the numerical pad can also be used, once the cursor has been placed in the text area. One can also type the numerical value.

2D graphics

The figure is displayed in the lower part of the window. It can be exported in a standard graphical format (*file* menu).

The graphics reacts to various commands. For instance, by pressing down for a period of time one of the incrementing/decrementing arrows, the curve is updated in real time, which is useful for simulating continuous variations.

2. **surface**

   3D (X,Y,Z) graphics that displays a response surface of a given FIS output on the Z axis, versus 2 input variables, X and Y.

   Range setting

   The upper part of the window allows to choose the settings for the variable ranges and the resolution for each of the two X and Y variables. See the paragraph above.

   3D graphics

   The figure is displayed in the lower part of the window. It can be exported in a standard graphical format (Menu *file*).

   The *Export* option of the *file* menu proposes an exportation of the 3D figure. The *Record* option records a series of graphics in function of a range of parameters for calculating the surface. The recording is stored as a sequence of indexed jpeg pictures, and can be transformed into a movie.

   Here is an example of a (GNU)Linux command that can be used to do so:

   mencoder mf://*.jpg -mf w=800:h=600:fps=25:type=jpg -ovc lavc -lavcopts vcodec=mpeg4 -oac copy -o output.avi

   The *avi* movie can be viewed on any platform.

   Meaning of surface colors

   The surface color changes from red (minimum) and purple (maximum), according to the Z value. The changes follow the rainbow order

   Note : the colors are meaningful only within a given figure. The red color corresponds to the minimum inferred value for that figure.

   Mouse actions

The graphics reacts to various commands. For instance, by pressing down for a period of time one of the incrementing/decrementing arrows, the surface is updated in real time, which is useful for simulating continuous variations.

If some points do not activate any rule, the z value is the default FIS output value.

A left click in the graphical area displays the (x, y, z) coordinates, a right click shows the activated rules for a given point. The default activation threshold is 0.001, and it can be changed in the *Options* menu. A double click opens the *Inference* window (see section 1.4) for a given point (the values are not editable within the *Inference* window).

Viewpoint

Three actions allow to change the viewpoint:

- Press left button + move mouse: rotation,
- Press right button + move mouse: translation ;
- Press wheel Molette centrale + move mouse: zoom.

The *Reset view* restores the initial 3D settings.

Section

The *Section* menu displays a section of the response curve onto the X or the Y axis. A popup window allows to set the input value for the section, and to make dynamic changes. The section plane can be displayed on the 3D graphics, and follows the dynamic changes of parameters.

1.6 Generate a FIS without rules

To use this option, there must be a current data file.
It is used as reference to generate the fuzzy partitions.

This option allows to generate the FIS inputs and outputs. The hierarchy type can be chosen (see section 3.1.2 - Generate vertices), as well as the number of fuzzy sets per variable, the tolerance on unique values or the number of groups for a k-means hierarchy.

The generated FIS has as many variables as the data file has columns.

It has a single output, which corresponds to the last column in the data file.

The conjunction operator is parameterized, the default output is a fuzzy output. It can be later edited to be transformed into a crisp output.

The FIS without rules can be used as an entry point to any of the following rule induction procedures: option *Generate rules* and *Generate conclusions*, fuzzy decision trees or the Wang & Mendel algorithm.

The current FIS is not overwritten, a new FIS window pops up.

1.7 Generate Rules

There must be a current FIS, with at least one active input variable. This option generates all possible combinations of rules for the fuzzy sets which have been defined for each input variable.

The rules are sorted by lexicographical order.

Moreover, if there is a current data file, the rule generation procedure only keeps the rules with a maximum matching degree (on the whole data set) greater than a user given threshold. In this case the rules can optionally be sorted by decreasing order of cumulated weight.

Warning : the rule conclusions are initialized with a default value equal to one (crisp output), or to the first MF label (fuzzy output) . They can be reset by a separate procedure, described below.

The current FIS is not overwritten, a new FIS window pops up.

1.8 Generate Conclusions

To use this option, there must be a current data file.

The rule conclusions can be generated, using a method inspired from the fpa (Fast Prototype) algorithm, described in [6, 7], or a least squares minimization procedure.

fpa is a simple efficient technique that initializes or updates rule conclusions using data. The *ols* least squares minimization procedure minimizes the sum of squares of errors (error=difference between inferred output and observed output).

For both options, the output vocabulary can be reduced.

FPA

The rule conclusions are calculated using the observed values of a subset of examples, chosen among the whole data set. This subset, called E_r for the r_{th} rule, is selected for each rule, in function of several criteria that will be explained at the end of this section. First we detail how the conclusion values are calculated, depending on the output type.

In the classification case (class output), the rule conclusion is simply taken as the majority class in E_r.

In the regression case (continuous output), each rule conclusion initialization is done by summing the observed outputs E_r, and weighting them with their corresponding matching degree, as follows. Let us call $\mu_r(x_i)$ the matching degree of the i_{th} example for the r_{th} rule, and y_i the observed output for the i_{th} example.

$$C_r = \frac{\sum\limits_{i \in E_r} \mu_r(x_i) * y_i}{\sum\limits_{i \in E_r} \mu_r(x_i)} \tag{1}$$

Fuzzy output initialization is done in two steps.

1. calculate a crisp value as for a continuous output (equation 1),
2. set the rule conclusion as the fuzzy set for which the membership degree of the above crisp value is the greatest.

Choice of the $\mathbf{E_r}$ subset

Its elements are selected in function of the item matching degree for the rule. The selection can be done in two different ways, depending of the chosen strategy.

The first strategy is called *decrease*. It retains the examples which most activate the rule. The user can specify two parameters : the cardinality threshold *cardmin*, and the matching degree *matchmin* threshold. If the number of examples matching the rule to a degree $\geq matchmin$ is lower than *cardmin*, the required matching degree is decreased according to a given step (set by the STEP_DEC constant, in the C++ library, whose default value is equal to *0.1*). The default matching degree is set to 0.7 by the START_DEC constant. The decrease procedure stops as soon as the required cardinality is reached, or when the required matching degree goes below a limit value *MuMin*.

This strategy privileges the rule prototypes (in a wide sense), whose definition is given in the glossary (section IV). It is assumed that the examples with a lower matching degree will be dealt with by means of interpolating during the inference procedure.

The other strategy is called *minimum*. it retains all the examples whose matching degree for the rule is $\geq matchmin$ threshold.

Independently of the applied strategy, a rule will be eliminated if $Card(E_r) < minmatch$, in reason of its insufficient representativity in the data set.

The parameters *minmatch* and *mincard* do not play the same role in the two different strategies. The decrease strategy is driven by the *mincard* parameter, the *minmatch* parameter being only used as a limit, whereas the minimum strategy is driven by the *minmatch* parameter, the *mincard* parameter being only used for checking.

Least square minimization
No parameters for this option.

Reducing the output vocabulary
In the case of a crisp regression output, the rule conclusion values are all different from each other. Reducing the output vocabulary improves the readability of the rule base.

Two choices are available. With the first one (default one), a clustering is performed using the rule conclusions, with the second one it is done by using the data file output values. The clustered values are chosen as the new rule conclusions.

The number of distinct conclusions can be set, or the tolerated loss of performance. Indeed reducing the voabulary usually goes with a loss of accuracy.

Once the vocabulary has been reduced, the output can be fuzzified. This option is available in the *Output* window, by checking the *Fuzzy output* box. A standardized fuzzy partition is then built using the rule conclusions as MF centers.

2 Data Menu

In many cases, we wish to use the FIS to infer output values for a whole data set, stored in an external file. To be usable as a data file in FisPro, the file must be in text format, with one line per example, and as many fields as the total number of input variables (active plus inactive) declared in the FIS. The data file may also include as many supplementary fields as there are output variables.

The field separator is automatically detected. Spaces or tabulations are not valid separators.

Learning menu options are available only if a data file has been opened. All of them need at least one output field to be present in the data file.

2.1 Sample generation

This option creates sample files by random sampling from a data file. Two possibilities: to generate learning and test pairs, or K sample files. Let N be the number of rows.

In the first case, each pair includes a sample file and its complement. One can choose the number of pairs, the relative sample file size, and the random sampling seed. The procedure creates as many file pairs as asked for, they have the data file name, followed by textitlrn.sample.n for one file, and by *tst.sample.n* for the other one, where n varies from 0 (first pair) to N-1 (Nth pair).

In the second case, the procedure splits the data file into K blocks, each of them of size floor(N/K) if the *constant size* option is checked, or else of size floor(N/K) for the first K-1 files, and N-K*floor(N/K) for the last one.

Other options (valid in both cases):

A choice of zero (0) for the seed means a new sampling, another value (1 for instance) sets the seed to a fixed value, allowing to repeat a given sampling.

The *classif.* checkbox imposes sampling to respect the class proportions in a data file column, by default the last one. In that case, a tolerance value (default=0.01) can be set, and will be used to determine classes from data.

2.2 View

The View option offers three possibilities. Figures can be exported in various formats (EPS, JPEG, etc.).

1- **Histogram**

Distribution of values, displayed together with the fuzzy partition, if there is a current FIS. Two indices are displayed below each partition: the partition coefficient, PC,to be maximized, and the entropy coefficient, PE, to be minimized.

In the formulae below, c is the number of MFs, n the number of rows in the dataset and u_{ik} is the membership of example k in the ith MF.

$$PC = \frac{\sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^2}{n}$$

$$PE = -\frac{1}{n}\left\{\sum_{k=1}^{n}\sum_{i=1}^{c}\left[u_{ik}\log_a(u_{ik})\right]\right\}.$$

2 - **X-Y graph**

On the 2 variable plot, options allow to display the regression line, or the axis bisector.

The left click on a data point activates/deactivates it, with a dynamical link to the data table presented below. In case of a graphical difficulty to identify the point, it may be easier to activate/deactivate it from the data table, which has checkboxes for that purpose.

An info balloon shows the row number of the data point near the mouse location.

3 - **X-Y-Z graph**

The points corresponding to the chosen variables are displayed on a 3D graphics.

The same functionalities than for the 2D plot are available, plus some other mouse actions:

- Press left button + move: rotation
- Press right button + move: translation
- Press wheel + move: zoom.

The *Reset view* option of the *File* menu resets the 3D graph to its initial values.

2.3 Table

This option displays data in a table, allowing to sort them by column, and to deactivate examples. Deactivated examples are not taken into account during inference. The active/inactive state is saved when closing the data file and is reassigned when reopening it. The popup window for *Opening a file* has an option allowing to reset all examples to an active state.

A double click on a table row opens the inference window corresponding to these data row values.

2.4 Infer

This option implements the *Performance* function. See section 2 for a complete description of arguments and results. The *Links between rules and data* option calculates the fired rules for each data file row,beyond the chosen blanck threshold. That procedure can be slow for a big data file. It is the default option for data files with less than 200 rows.

The results of the fuzzy inference performed on a whole data set are saved in a text file, readable in a spreadsheet. The file format, in particular the number of columns, depends on the type of defuzzification operator chosen for each output and on the classification flag. It is detailed in section 2, part II.

For instance, in the case of a crisp output, and with the classification option selected, each inferred value is recorded, with the class that it was assigned to.

In case of multiple output, one of them must be selected in the inference window. When the observed output is present in the data file, it is compared to the inferred output. A performance index is given. In the classification case, it is the number of misclassified examples. Otherwise, three numerical indices are given, *PI*, *RMSE* et *MAE* as defined in the glossary section (voir section IV. A performance summary file is written, see 2 that describes its format.

Note: all learning procedures are guided and characterized by *RMSE*.

The same viewing options are available for graphical representation as in the Data menu: histograms, X-Y plots with or without a regression line, or confusion matrix for classification data.

If the *Links between rules and data* option was chosen in the inference window, additional information is available in the X-Y plot by moving the mouse over a data point: line number in the data file, and activated rule numbers (beyond the activation threshold).

For a classification system, the result is a *confusion matrix*. The example given in the figure 1 shows that the 50 examples of class 1 are well classified, 49 examples of class 2 are well classified and 1 is misclassified (inferred class=3), 47 examples of class 3 are well classified and 3 are misclassified (inferred class=2).

Note: classification corresponds to a crisp output, with the classification option checked.

Implicative outputs

For implicative outputs, the X-Y graph plots the inferred value as an interval centered on the defuzzified value, and bounded by the minimum and maximum kernel values of the inferred possibility distribution.

A new menu option is available for viewing results of implicative rules. It gives a cumulated matching degree of inferred outputs with observed ones. This option is inspired from the classification *confusion matrix*, but it takes into account the specificity of inferring an output possibility distribution and not a single defuzzified value, as this is the case for conjunctive rules.

The matching degree varies between 0 and 1 for each item. It is computed using the intersection of the inferred possibility distribution with output fuzzy sets, and also considering the distribution width. The result is cumulated over all data file items, and displayed as a table, with one row per output fuzzy set, a *Matching degree* column and an *Item* one.

Warning: Whatever the type of rules, implicative or conjunctive, the performance index only takes into account the examples that activate the rules beyond the chosen threshold (default value is 0.1).

		observed		
		class 1	class 2	class 3
infered	class 1	50	0	0
	class 2	0	49	3
	class 3	0	1	47
not classified		0	0	0

Figure 1: Example of a confusion matrix in a classification case

2.5 Links Menu

This utility allows to view the links between rules and sample items, and also the links between the various rules. It creates several work files, with a default name prefix equal to the data filename.

- rules.items

  This file has as many lines as there are rules in the system, plus 2.

  * First line: number of rules
  * Second line : maximal number of items that activate a rule
  * Third line : description of the first rule, i.e. the rule number (starting from 1), cumulated weight, number of items that activate a rule (beyond a threshold parameter with default value equal to 1e-6) followed by their number.

- items.rules

  This file has as many lines as there are items in the data sample.

  For each line, the item number (starting from 1), followed by the rule numbers that are activated by it.

- rules.links

  The notion of link between rules is useful to appreciate the consistency of a rule base. If two rules are strongly linked, and have different conclusions, then the inconsistency can come from an insufficiently specific input range rule coverage. This situation can also correspond to an exception in the data sample. The linkage level of the i_{th} rule with the j_{th} rule is calculated as follows:

$$L_{i,j} = \frac{N_{i,j}}{N_i}$$

N_i is the cardinal of the subset E_i of the items that activate the i_{th} rule, $N_{i,j}$ is the cardinal of $E_i \cap E_j$.

This file is formatted as a square matrix, whose size is equal to the number of rules. The i, j cell gives the corresponding linkage level. Note : the matrix is not symmetric.

- rules.sorted

  if the sort option is selected, rules are sorted by cumulated weight (which represents their influence in the data sample).

Note: links do not depend on the FIS outputs.

2.6 Distance

Fuzzy partitions may be useful to generate a distance function applicable to real values, called FP-distance. This allows the introduction of expert knowledge in distance calculations. The program generates a (n x n) matrix of all distances between the (n) rows of the current data file. By default, the distance is normalized in each dimension between 0 and 1. FP-distances and Euclidean distances can be combined in the multi-variate case. Their aggregation is done using a Minkowski combination, with the p exponent as parameter. The matrix can be exported to be introduced in clustering algorithms, for instance in R.

A color image of the one-dimensional distorsion of the Euclidean distance by the FP-based distance is available.

The FP-based distance is presented in detail in:

Serge Guillaume, Brigitte Charnomordic, and Patrice Loisel.

Fuzzy partitions: a way to integrate expert knowledge into distance calculations. International Journal of Information Sciences, page In Press, 2012.

3 Learning Menu

The *Learning Menu* includes supervised learning procedures, simplification procedures and optimization procedures.

Warning: all learning procedures are compatible with implicative rule systems, but, in this case, results must be interpreted with the semantic specific to implicative rules.

Learning allows to design a FIS from a data set, called learning set. The main learning methods have been described in [8]. To ensure interpretability, we chose to separate the partition design stage from the rule induction one, even when both are mixed together in the original publications.

3.1 Partitions

3.1.1 Generate a FIS without rules

The *Generate a FIS without rules* option can be called prior to the rule induction methods. It is described in 1.6.

3.1.2 HFP MF and HFP FIS

The HFP method, which means Hierarchical Fuzzy Partitioning is described in detail in [9, 10].
It includes a fuzzy partitioning stage and a rule selection one.

We only present here the four steps proposed in the menu. A current data file must be available.

1. Generate a HFP configuration file
   The maximum partition size must be specified for each active variable. The most important parameter is the type of hierarchy. Three choices are available:

   (a) regular grid

   (b) k-means

   The k-means option uses the well known clustering method of the same name. There is no a priori relationship, for a given variable, between the partition centers for partitions of different sizes.

   (c) hfp

   This last option corresponds to an ascending procedure. At each step, for each given variable, two fuzzy sets are merged. The computational time can be high, and essentially depends on the initial number of fuzzy sets. There are two ways to build the initial partition. The first one consists of considering the values in the learning set as identical if they differ by less than the threshold (given as a percentage of the input range). Each group of values, the number of groups depending on that threshold, then corresponds to one fuzzy set. The second way sets the number of groups, and calculates the group centers by applying a k-means clustering.

2. Generate vertices

   This option creates a file that records the vertex coordinates for all proposed partitions. The partition size goes from two to the maximum number (default is 7) set at step 1.

3. View vertices

   This option is for viewing the vertex location, together with the data distribution (histogram). Two indices are displayed below each partition: the partition coefficient, PC,to be maximized, and the entropy coefficient, PE, to be minimized.

   In the formulae below, c is the number of MFs, n the number of rows in the dataset and u_{ik} is the membership of example k in the ith MF.

$$PC = \frac{\sum_{k=1}^{n}\sum_{i=1}^{c} u_{ik}^2}{n}$$

$$PE = -\frac{1}{n}\left\{\sum_{k=1}^{n}\sum_{i=1}^{c}\left[u_{ik}\log_a(u_{ik})\right]\right\}.$$

4. Select the fuzzy partitions

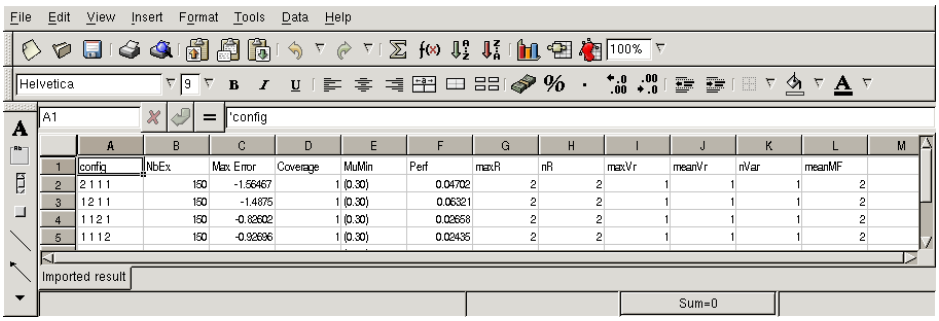
   The one-dimensional hierarchies calculated at step 1 are used to build fuzzy inference systems. It is an iterative procedure which starts from a 1 or 2 fuzzy set partition for each input dimension, builds the fuzzy inference system including all possible rules and assigns it performance and coverage indices. At each iteration, a single input variable fuzzy partition is refined by adding one fuzzy set, and the corresponding fuzzy inference system is built. The best fuzzy inference system will be kept considering the performance criterion. The procedure uses a HFP configuration file and a vertex file. The first five parameters are identical to those described in the Generate Conclusions Menu (fpa algorithm). The next one corresponds to the initial number of fuzzy sets in the partition. It can be set to 1 or 2. The default value is 1, and corresponds to introducing variables one by one.

   The last two parameters allow to set a maximum number of iterations, and to specify a validation file for evaluating the FIS performance. The default validation file is the current data file. Other files can be used, in particular the active or inactive data file created by the use of the Data-table menu option.

   Output files:

The results of each iteration are stored as a line in both files "result" and "result.min". The "result" file records all attempts (adding a fuzzy set on each input variable), and the "result.min" file holds the configuration retained after each iteration.

Format: columns are delimited by the & character, which allows easy import into a spreadsheet, or immediate introduction as an array in a LaTeX document.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1	config	NbEx	Max Error	Coverage	MuMin	Perf	maxR	nR	maxVr	meanVr	nVar	meanMF	
2	2 1 1 1	150	-1.56467	1	(0.30)	0.04702	2	2	1	1	1	2	
3	1 2 1 1	150	-1.4875	1	(0.30)	0.06321	2	2	1	1	1	2	
4	1 1 2 1	150	-0.82602	1	(0.30)	0.02658	2	2	1	1	1	2	
5	1 1 1 2	150	-0.92696	1	(0.30)	0.02436	2	2	1	1	1	2	

Figure 2: Example of an hfp result file

first column: p (equal to the number of input variables) integers, separated by a white space. They indicate the current number of fuzzy sets for each variable,

second column: number of items in the validation set

third column: Max error

fourth column: coverage index

fifth column: threshold value used to compute the coverage index

sixth column: performance index

following columns : rule base characteristics

The performance and coverage indices are described in part II,section 2, while the rule base characteristics are introduced in section 3,part II.

Output file analysis allows the user to select one or more FIS to generate. Elements to be considered are the performance (column 6) and the coverage level (column 4) but also the complexity of each FIS, which is measured by the number of rules (column 3),and other rule base characteristics provided by the *InfoRB* structure.

The performance index, which is an error measure, should be as small as possible (minimum 0), and the coverage level as high as possible (maximum 1), while the number of rules and of fuzzy sets must be kept reasonably small. The compromise between complexity and accuracy is under user control.

Note: This option does not generate a FIS.

5. Generate a FIS

   This option, available in the Rule Induction-HFP FIS option, creates a FIS configuration file for a given combination.

   The HFP configuration file and the vertex file are given as parameters. The other parameters are used to initialize the rule conclusions using the FPA algorithm. The generated FIS can be used as such, or as an input to the simplification menu.

3.2 Rule induction

Several rule induction methods are available. Most of them use a FIS, the "FPA" algorithm, the "Wang & Mendel" algorithm and "fuzzy decision trees". In that case, existing rules, if there are any, are ignored, and MFs are not modified. The HFP method is a particular case. It is a partition refinement algorithm, described in the *select a partition* option (see section 3.1.2), which only uses data from a data file. If there is a current FIS, it is ignored by the method. The OLS method can use a FIS configuration, which means that the input fuzzy partitions are kept as such, it can also work from a data file only. In that case, one step is needed to generate the input fuzzy partitions.

Simplification can be applied to any FIS, independently of the induction method used to build it. It does not modify MFs, but only changes the rules.

Optimization can be applied to any FIS, independently of the induction method used to build it. It can modify all FIS elements including MFs.

We will specify whether there must be a current data file, and/or a current FIS, prior to the learning procedure. If these conditions are not fulfilled, the corresponding learning option will be grayed.

Note: most learning methods use a tolerance threshold EPSILON=10^{-6}. Before using these methods, it is better to normalize the data between 0 and 1, if their range is very high or very small.

Note: for all learning procedures and numerical outputs, the accuracy index is the *RMSE* performance index.

3.2.1 FPA Menu

It needs both a current FIS configuration file and a current data file. The *Generate a FIS without rules* option generates the partitions from data (see 1.6).

The *FPA* option allows to generate the rules and set their conclusions in one pass (see 1.7 and 1.8 for details).

3.2.2 Wang & Mendel

Contrary to the original method [15], this procedure needs predefined fuzzy partitions. They can be automatically generated from data using for instance the *Generate a FIS without rules* option of the *FIS* menu.

The prodedure handles all outputs, and ignores existing rules. It needs both a current FIS configuration file and a current data file.

It starts by generating one rule for each data pair of the training set.

The i^{th} pair one is written :

$IF\ x_1\ is\ A_1^i\ AND\ x_2\ is\ A_2^i\ \ldots AND\ x_p\ is\ A_p^i\ THEN\ y\ is\ C^i$.

The fuzzy sets A_j^i are those for which the degree of match of x_j^i is maximum for each input variable j from pair i. The fuzzy set C_i is the one for which the degree of match of the observed output, y_i, is maximum.

For a crisp output, an internal conversion creates MFs centered on the output values, then reassigns crisp numbers equal to MF centers to the rule conclusions.

A degree is assigned to each rule. For a given rule it is equal to the rule fire strength for the considered pair. In case of identical premises for two rules, only the one with the higher degree is kept.

This procedure is very easy to use as it does not not require any parameter.

Limitation: this procedure is not well suited to crisp outputs representing continuous numerical values. In that case it can only be used if the observed output has 15 different values or less.

3.2.3 OLS Menu

Different options are available if a FIS file is available or not. The OLS algorithm [2, 3, 11] transforms each data file example into a fuzzy rule, and selects the most important rules with the least squares criterion, by linear regression and Gram-Schmidt orthogonalization. After the selection step, a second passage is done to optimize the selected rule conclusions.

As for all FisPro learning procedures, two steps must be distinguished: generating the partitions if necessary, and inducing rules.

Our implementation is motivated and described in detail in [5].

Generate partitions

If there is no current FIS, the first step is the partition generation from data. Two options are available. We advise to use standard fuzzy partitions (as in the **FIS** menu *Generate a FIS without rules*, section 1.6).

We also propose the original OLS algorithm, which consists of generating a Gaussian MF for each data point variable value. These MFs are then clustered to limit their number to MAX_MF (999) . Note that, in that case, the partitions are not standard fuzzy partitions.

If there is a current FIS file, the input fuzzy partitions are kept.

Generate rules

The second step consists of two stage: the rule selection étape and the rule conclusion optimization. The algorithm uses one output (default=first output, or last column in data file). Two criteria are used together to select the rules in the first stage:

- Part of output inertia not explained by the rules (Default=0.1).
- Number of rules (Default=100).

The prodedure stops as soon as one criterion is satisfied. With the default values, the inertia one is usually the first to be fulfilled.

The stage 2 uses a least squares optimization to set the selected rule conclusion values. If there is a current FIS, an option allows to keep the existing rules, which means that only the rule conclusions are changed.

The optimization, which does not change the partitions nor the rule premise, but only the rule conclusions, is also available in the *Generate conclusions* option of the *FIS* menu (section 1.8).

Reduce the output vocabulary

The rule conclusions generated by OLS are all distinct from each other, so it can be interesting to reduce the number of distinct conclusions.

This is done by a clustering operation, for which two options are available: start from the rule conclusions, or start from the output data values in the data file.

The number of distinct conclusions, or the tolerated performance loss, can be set. Indeed the voabulary reduction usually goes together with an accuracy loss

With that option, the output can be fuzzified, meaning that a standard output fuzzy partition is built using the new conclusions as MF centers. “fuzzifier” la sortie, c’est à dire construire une partition floue

The *Reduce the output vocabulary* option is also available in the *FIS* menu “Generate Conclusions” option.

3.2.4 Tree Menu

Fuzzy decision trees are an extension of classical decision trees [1, 14]. They are formed of one root node, which is the tree top or starting point, and a series of other nodes. Terminal nodes are called leaf nodes, or leaves. Each node corresponds to a split on the values of one input variable. This variable is chosen in order to reach a maximum of homogeneity amongst the examples that belong to the node, relatively to the output variable (response variable). This is equivalent to minimizing the entropy. The paths from the root node towards the leaf nodes are easy to interpret as decision rules, strict of fuzzy depending on the nature of the tree.

The tree can be pruned, by transforming a node into a leaf node, if the performance loss is low. This procedure facilitates the tree interpretation. Pruning is based on the performance of the tree equivalent FIS.

Note: the pruned tree may have a better performance than the full tree. as tree building is based on a purity criterion and not directly on a performance criterion.

Fuzzy decision trees proposed in FisPro are based on a fuzzy implementation of the ID3 algorithm [12].

Generation

The fuzzy decision tree generation procedure in FisPro needs both a current FIS configuration file and a current data file. The tree building is based on one output only, even if the FIS has several outputs. This output is user chosen.

Output Type

Four cases are possible:

- fuzzy output, with the classification option
- fuzzy output, without the classification option
- crisp output, with the classification option
- crisp output, without the classification option (regression tree).

The first three cases use the fuzzy entropy criterion to build the tree, the last one uses a deviance criterion, based on the output value dispersion at each node and is better suited to a regression case.

For a crisp output with the classification option, classes are determined from data, and discrete MFs corresponding to the class labels are assigned to the output. The maximum number of classes is 100.

With the classification option, the majority class is assigned to each node, without it, the average of the node attracted example output is assigned to it.

In the fuzzy output case, the tree summary gives the fuzzy proportions of the node attracted example output for each MF.

Rules

If rules are present in the FIS configuration file, they are ignored.

Options

The *Generate tree option* opens a pop up windows allowing to choose:

- the tree file name. The tree file format is a description suited for viewing the tree (extension *.tree*),
- the maximum tree depth, default is the number of variables.
- the minimum matching degree μ for an item to be considered as belonging to the node (for entropy calculation),
- The minimum number of samples attracted by a node, with a node membership greater than μ, required for further splitting the node,
- the tolerance on the matching degree to the node majority class,
- the minimum entropy/deviance gain required (relative value) to build a tree branch
- for classification only, the entropy criterion for building the full tree: absolute (default) or relative entropy gain.
- for regression only, it is possible to optimize rule conclusions in a second pass, using the least square optimization (OLS).

  The *relative entropy* favors the variables with an unequal distribution of examples (not classes) between the MFs. It also favors the variables with a small number of MFs.

- the pruning option:

  Pruning consists in a recursive node removal, from tree bottom to top. The removal is done if the equivalent FIS performance does not decrease, or decreases only a little.

  The relative loss of performance (compared to the performance of the full tree) has a default value of 0.1. It is user editable. The default validation file used for computing the performance is the current data file. Other files can be used, in particular the active or inactive data file created by the use of the Data-table menu option.

  Pruning can be done by removing a full split, or by removing node after node.

- an intermediate output display.

Output

The procedure creates one tree or two (if pruning was selected). It opens two windows for each tree:

- a viewing window.

  The user can select the informations to display for all nodes: number of items attracted by the node, node entropy, majority class and sample number class distribution (classification case), or else mean and standard deviation (regression case). The part of the tree displayed in the window can be exported or printed.

  The initial tree location is done automatically but the user can also select tree branches and maually move them to improve the location. The scale and font are editable.

- a FIS window. It displays a FIS equivalent to the generated fuzzy tree, which can then be used as such and saved.

- a result file, called result.fistree (figure 3):

  Each line in the *result.fistree* file describes a fuzzy tree (full or pruned tree):

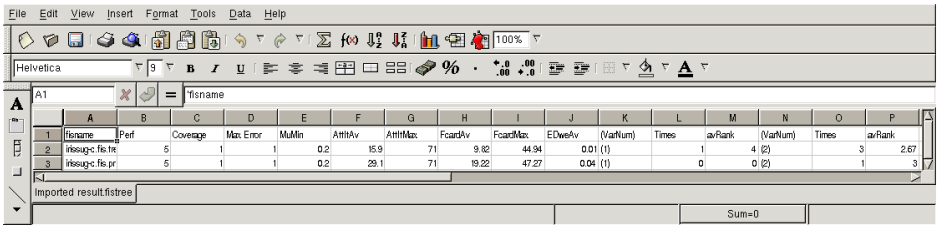


Figure 3: The result.fistree file

The first column gives the FIS equivalent name (full tree or prunded tree). Then we have the indices described in section 2:

2nd column: performance index

3rd column: coverage index

4th column: max error

5th column: membership threshold for computing the coverage index

Then follow some indices specific to fuzzy trees:

6th column: average number of examples attracted by a leaf

7th column: maximum number of examples attracted by a leaf

8th column: average fuzzy cardinality per leaf

9th column: maximum fuzzy cardinality for the leaves

10th column: average entropy or deviance per leaf, weighted by the fuzzy cardinality

groups of 3 columns (number of groups equal to number of active FIS input variables)

- variable number
- number of times when it was chosen as the split variable
- average appearing rank

following columns: rule base characteristics (see section 3).

View tree
This submenu allows to view an existing tree. Two possibilities are offered.

Graph: the viewing window described above.

Table: non graphical viewing, displaying the same information in a table.

It opens the viewing window described above.

3.2.5 HFP FIS

This option is detailed in Section 3.1.2.

3.3 Simplification

Simplification principles are presented in [9]. It aims to eliminates the less useful variables from the rules of a fuzzy inference system. The procedure needs a current FIS configuration file and a current data file. It takes the same parameters as the FPA algorithm (used for generating conclusions), and some specific ones: the tolerated loss of performance (as a percentage of the initial performance index), and the homogeneity threshold. This last parameter is used for checking that the widening of the rule input space coverage resulting from the simplification procedure does not lead to abusive simplifications. It sets the dispersion maximum value for the observed outputs of the items attracted by a rule. If the homogeneity threshold value is two low, only very specific rules will be kept, which will

make interpolating difficult in the fuzzy inference system. We advise to start the procedure with a value of *0.5*.

The *KeepLast* option is used by the procedure to not remove the last rule whose conclusion is a given class label, case of a crisp output with the classification flag, or a membership function in the case of a fuzzy output.

A validation file can be specified. In that case, it is used at each step to decide whether to keep not to keep the simplification resulting from the use of the learning file.

The procedure output is summarized in the file *result.simple*. Each line is a summary of one attempt to simplify the system. The first column gives the name of the FIS configuration file created during the run. The name is formed by appending to the initial FIS configuration file name the structure $.jb.x$, x being an index number, the indices given in section 2 and finally the rule base characteristics, see section 3.

1st column: initial Fis filename

2nd column: performance index

3th column: coverage index

4th column: max error

5th column: activation threshold used for computing the coverage index

following columns: the rule base characteristics

At the end of the simplification procedure, a window appears that corresponds to the simplest FIS found.

3.4 Optimization Menu

Two options are proposed in this menu. The first one, called *OLS (Least Squares)* allows to optimize the rule conclusions. It is detailed in the ols menu, section 3.2.3.

The second option allows the optimization of the FIS elements: MF bound location for fuzzy inputs and outputs, rule conclusions. The Solis and Wetts method is available. It is a mono agent evolutionist strategy. The reference [7] gives the detailed algorithms. It is available in two forms:

- *Solis Wets* This is a ready to use standard version. It allows an easy selection of the components to optimize: first inputs in the user given order, then output and finally rules. The optimization is an iterative procedure, with a default number of steps equal to 1.

  A minimum distance constraint between two neighboring MFs is proposed. Its default value is 1% of each variable range.

  A cross-validation option is available. A chosen number (10 by default) of random learning-test pairs is generated. For each pair, an optimized FIS is designed from the learning sample, and its accuracy is measured on the corresponding test sample. The optimized FIS are then aggregated into a unique final FIS. Each of the final FIS parameters is the median of the corresponding parameters for the optimized FIS. The final system is usually more accurate (in average) on the test samples than each of the optimized systems.

  A detailed presentation of the optimization module can be found in:

  Serge Guillaume and Brigitte Charnomordic. Parameter optimization of a fuzzy inference system using the fispro open source software. In IEEE Catalog Number: CFP12FUZ-USB, editor, IEEE International Conference on Fuzzy Systems, pages 402-409, Brisbane, Australia, June 2012. IEEE.

- *Custom Solis Wets* It gives access to the whole set of parameters but is more difficult to use.

  In any case the optimization procedure does not find the absolute best solution, but one among the solutions corresponding to the given criteria.

Possible constraints

Whatever the FIS element to optimize, the optimization is based on the FIS performance improvement.

The algorithm can be sumitted to some user defined constraints. The solutions found will onlu=y be retained if they satisfy the constraints.

Parameters:

- *choosing the output*, if the datafile has several outputs. The performance will be based on one output only.

- *maximum number of iterations*: default value=100.

  A higher value will increase the success rate of the procedure.

- *blank threshold*: fuzzy inference parameter for the performance calculation (see the corresponding explanation in section 2).

- *Advanced parameters:*

  Opens a popup window allowing to set some parameters to control the algorithm.

  - *seed value*
    Initialize the random generator.
  - Solis & Wetts constants
  - *Gaussian noise standard deviation*: internal parameter, that must be between 0.0 and 1.0. If too high, convergence is slow, default value: 0.005.
  - *max number of constraints*: default value=1000. Number of random draws allowed to consider a configuration as a candidate before the iteration number is incremented.
    Raise this number if needed to increase the chances for the search algorithm to find a valid solution.

  The solutions found by the algorithm are kept only if the constraints are respected.

- **FIS elements to optimize:**

  Check the checkboxes to choose the FIS elements to optimize.

- *Select all inputs* :

  This checkbox preselects/unselects all MFs for all inputs.

- *Select all rules* : This checkbox preselects/unselects all rules for rule conclusion optimization.

- *Standard*

  If this checkbox is checked, the partitions are standardized fuzzy partitions, which guarantees the respect of the semantic [4] and the rule interpretability.

- *optimize fuzzy sets*

  For each input, each MF is listed with a checkbox to select/unselect it.

  For the output:

  For a fuzzy output, check/uncheck Standard (for Standard Fuzzy Partition) and select the MFs to optimize, as for inputs.

- *Rule r*

  For each rule, checkbox to select/unselect it. The selected rules will have their conclusion optimized.

  If the output is crisp, the *Limited vocabulary in FIS* option restraints the conclusion values to a permutation of the initial values given in the FIS configuration. Otherwise, the conclusions can take any value.

- *Display the key*

  Displays the key for a copy/paste operation, to reuse it as an argument of the *fisopt* program.

Result

In case of success, the procedure creates a new optimized FIS, which is opened in a new window. Otherwise, a warning is displayed.

A perf.res file is written, that includes the accuracy of the initial FIS, of each optimized FIS and of the medain FIS, calculated for the initial dataset, each test sample and each learning sample.

Advice for users

The optimization procedure does not find the absolute best solution, but one solution corresponding to the given criteria.

It is advisable to proceed with successive steps, rather than to optimize everything at once. It is always possible to iterate the optimization procedure, by reusing the FIS created at the previous optimization step.

4 Options Menu

FisPro is a localized software. The menus, buttons and error messages are available in several languages (currently French, English and Spanish).

The Options Menu allows the user to select the working language, and to modify the windows look and feel. Other languages can easily be added, by translating the message files in the subdirectory resources of the Java class directory. The environment variable setting corresponding to the language locale (platform dependent) will then make the new language files available in FisPro.

Part II
C++ Function Library

The class hierarchy is given in figure 4.

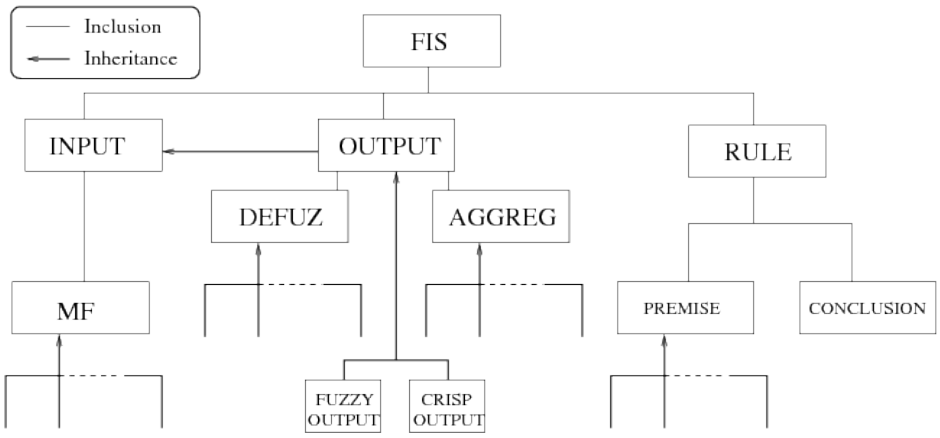


Figure 4: Basic class hierarchy

Each class is generally implemented as two files, the declaration file with the .h extension, and the definition file with the .cpp extension.

1 FIS structure

The FIS handled by the function library are *MIMO* type FIS, Multiple Input Multiple Output. The FIS is built by using the information of the configuration file. The basic module contains two main functions, defined within the *FIS* class: *Infer* and *Performance*.

The *Infer* function infers a value for each active output from the input values. Fuzzy inference obeys the following steps:

1. Fuzzification: done within the *INPUT* class. For a given input value, the *GetDegs* function fills the *Mfdeg* array with the membership values to each input fuzzy set. This array is public.

2. Inference: Each rule conclusion is weighted with the item matching degree for the rule. The rules have the following structure (for a two input one output FIS):

   IF Input 1 is MF 2 AND Input 2 is MF 1 THEN Output 1 is Value 1

   (crisp output)

   IF Input 1 is MF 1 AND Input 2 is MF 1 THEN Output 1 is MF 2

   (fuzzy output)

The *ExecRule* function of the *RULE* class calculates the matching degree of the item to the rule, also called rule weight. The calculation is done within the *PREMISE* class. It uses a conjunction operator between the fuzzy sets in the rule premises for the input variables (*Input 1* and *Input 2*). These fuzzy sets are described by the *number 2* and *number 1* membership functions for the first rule, *1* and *1* membership functions for the second one). The *PREMISE* class includes a pointer on the input variable array, which gives it access to the *Mfdeg* field for each input. The rule matching degree is stored in the public variable *Weight* within the *RULE* class. When expert weigts are set (see section *FIS menu- Rule window*), the rule matching degree is multiplied by the expert weight.

3. Inferred value computation: This computation is handled by *FISOUT* class. It consists of two main steps.

   - **Rule aggregation:** Once all the rules have been fired, two options are available to aggregate the rule conclusions: *max* or *sum*.

     The rule conclusions are numerical values for a crisp output, or MF labels for a fuzzy output. They constitute a set of possible values.

     The resulting levels are stored into the *MuInfer* array, the *RuleInfer* one contains the number of the rule corresponding to the maximum, in case of *max* aggregation, or the last rule number whose degree has been added, in the *sum* aggregation case. Both operators can be used with crisp or fuzzy output.

     Note :

     r the number of fuzzy rules, $w^r(x)$ the rth rule matching degree for the multidimensional vector x.

     m the number of output MFs (fuzzy output) or the number of distinct rule conclusion values (crisp output).

     C^r the conclusion of the rth rule.

     The activation levels after aggregation are the following:

     - max: $\forall j = 1, \ldots, m$
       $W^j = \left\{ \underset{r}{max} \left(w^r(x) \right) \mid C^r = j \right\}$
     - sum:
       * crisp output $\forall j = 1, \ldots, m$
         $W^j = \sum_r \left(w^r(x) \right) \mid C^r = j$
       * fuzzy output $\forall j = 1, \ldots, m$
         $W^j = min \left(1, \left\{ \sum_r \left(w^r(x) \right) \mid C^r = j \right\} \right)$

- **Defuzzification:** This operation uses aggregation results. The inferred value, $\widehat{y}_i$ for the i_{th} example, depends on the output nature and its defuzzification operators.
  - crisp output
    (a) **sugeno** operator

$$\widehat{y}_i = \frac{\sum_{j=1}^{m} W^j C^j}{\sum_{j=1}^{m} W^j} \tag{2}$$

    (b) **max crisp** operator

$$\widehat{y}_i = \left\{ j = argmax\left(W^j\right) \ | j = 1 \dots m \right\}$$

  - fuzzy output
    Figure 5 illustrates the defuzzification possibilities for a fuzzy output.

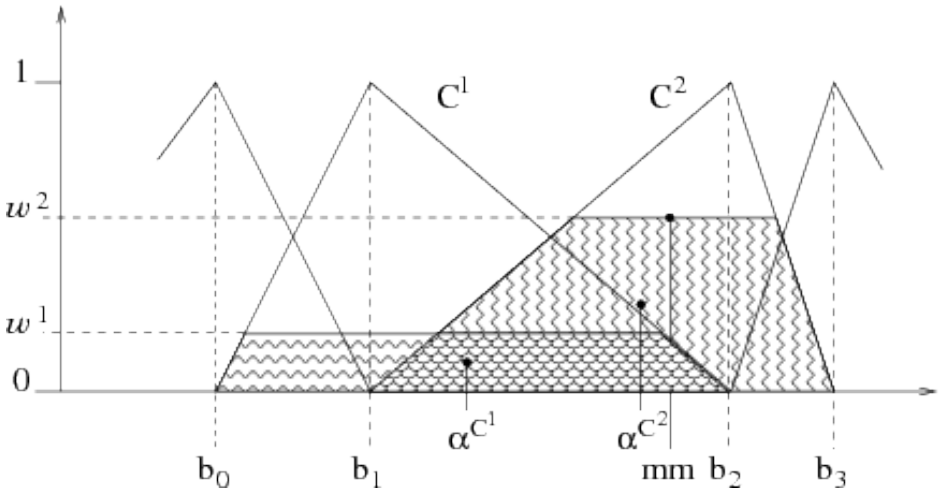


Figure 5: Defuzzification

(a) **weighted area** This operator favors the interpolation between linguistic labels. The inferred output is:

$$\widehat{y}_i = \frac{\sum_{j=1}^{m} \alpha^{C^j} \ area(C_\alpha^j)}{\sum_{j=1}^{m} area(C_\alpha^j)} \tag{3}$$

where m is the number of MFs in the partition, $\alpha = W^j$ is the activation level for the j_{th} MF, α^{C^j} is the centroid abscissa of C^j_α, and C^j_α a new MF, defined from C^j as follows:

$$\mu^{C^j_\alpha}(x_i) = \begin{cases} \mu(x_i) & if\ \mu^{C^j}(x_i) \leq \alpha \\ \alpha & sinon \end{cases}$$

(b) **average of maxima** The output is $\widehat{y}_i = mm$ (figure 5). Only the segment corresponding to the maximum activation level is considered, which means the operator mainly operates within one linguistic label.

(c) **sugeno** It is the same formula as for a crisp output (equation 2), but C^j now represents the middle of the j_{th} MF kernel.

Implication operators
For implicative outputs, three implication operators are available :

- Resher-Gaines : $a \rightarrow b = \begin{cases} 1 & si\ a \leq b \\ 0 & sinon \end{cases}$
- Goguen : $a \rightarrow b = \begin{cases} 1 & si\ a = 0 \\ min(1, \frac{b}{a}) & sinon \end{cases}$
- Gödel : $a \rightarrow b = \begin{cases} 1 & si\ a \leq b \\ b & sinon \end{cases}$

The inference result is a possibility distribution, which may be defuzzified. In the current implementation, no choice is available for the defuzzification operator. The default one is the *Mean of maxima*, which corresponds to the middle of the distribution kernel.

2 The Performance Function

This function is called for comparing the observed and inferred outputs for a data sample. It has two main arguments: the output number to be tested, and the data set reference, which can be either a file reference or a pointer on an array. The performance depends on the output classification flag.

The function writes a result file containing an array, with as many lines as there are items in the data set. The number of columns depends on the data file and on the output nature (defuzzification type and classification option).

If the data file has an output column, the FIS performance ith respect to a given data set is estimated by three complementary indices: maximum error, performance index and coverage index (which depends on the blank threshold).

- Max error: It is computed as the maximum absolute value, over the whole set of examples, between the observed output, y_i for the ith example, and the the one inferred by the system, $\widehat{y_i}$.

- Coverage index: Some data items may match rules with a low degree. A data item is called blank if its cumulated matching degree on the rule base is lower than a given threshold. This threshold is a user parameter, with a default value of *0.2*. Blank items do not appear in the performance results. Their total number is used to define a coverage index, equal to $CI = 1 - \frac{I}{n}$. *I* is the number of blank items and *n* the total number of items in the data file.

  Note: The matching degree highly depends on the rule premise conjunction operaror (min or product). The coverage index can decrease very fast if the number of variables is high, and the blank threshold must be chosen accordingly.

- Performance index: It depends upon the output type. For a crisp output using the classification flag, it corresponds to the number of misclassified items. Otherwise it is parameterized using the SetErrorIndex function, and can be one of *PI*, *RMSE* ou *MAE* (see glossary in section 1. This performance index is returned by the function.

Example:

Figure 6 displays the result file that corresponds to a FIS having conjunctive rules, a 3 MF fuzzy output, and the classification option.
In that case, the file has 9 columns for each example: observed value, inferred value after defuzzification, alarm flag (0 if everything is OK, 3 columns MF1, MF2, MF3 with the example output membership degree to each of them, current error, Bl flag (0 for a non blank example, 1 for a blank example), and cumulated error.

General Format

The first line of the result file is a column header. The possible labels are defined in the file fis.h:

- "OBS": The observed output, part of the data set

- "INF": The inferred output

- "Al": Alarm risen while inferring (see below)

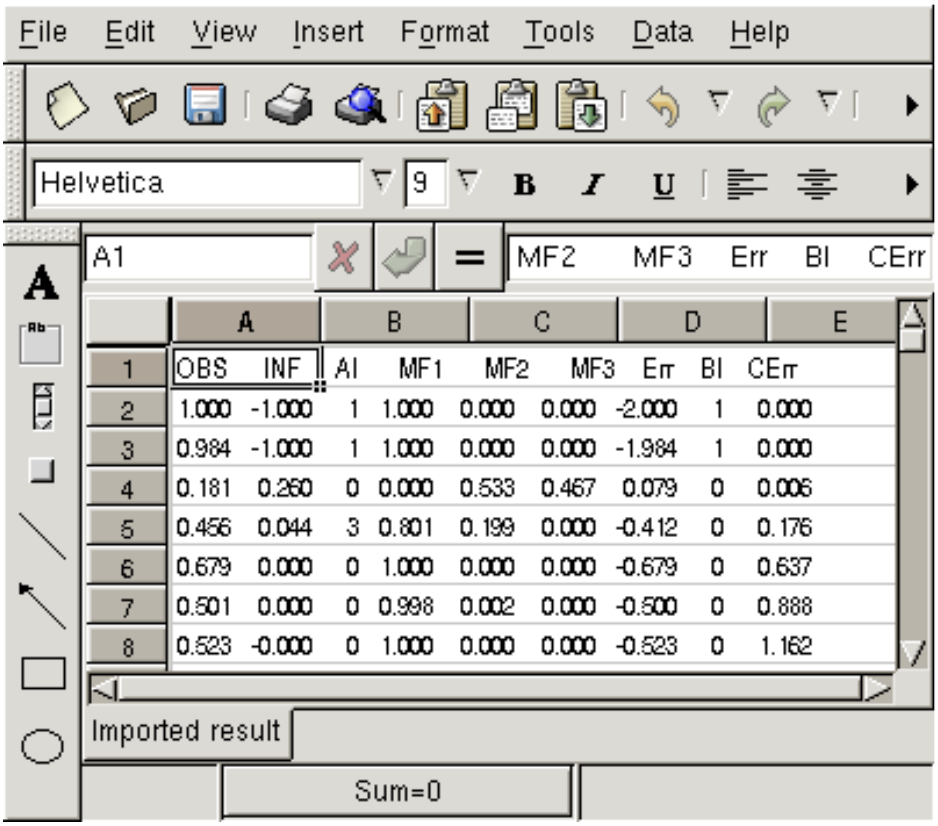


Figure 6: Inference result file - 3 MF fuzzy output, classification option

- "ClINF": Inferred class label, for crisp output and classification flag
- "CLAl": Alarm risen while inferring (see below)
- "Err": Difference between inferred and observed output
- "Bl": Blank flag, 1 if this item is considered as blank, o otherwise
- "CErr2": square cumulated error over the previous examples.

WARNING: The cumulated error does not take into account the blank examples.

The first column holds the observed output, if it is available in the data file, as inference can also be done without an observed output.
The second one gives, when the observed output value is available, the difference

between observed and inferred values.
A variable number of columns follows, depending on the output nature.

Output	Classif	Defuzzification	Field #	Available fields			
crisp	yes	sugeno	4	infered v.	Alarm	infered class	Alarm
crisp	no	sugeno	2	inferred v.	Alarm		
crisp		MaxCrisp	2	inferred v.	Alarm		
fuzzy	yes	sugeno/area	n+2	infered v.	Alarm	$\mu_1, \mu_2 \ldots \mu_n$	
fuzzy	no	sugeno/area	2	infered v.	Alarm		
fuzzy	yes	MeanMax	n+2	infered v.	Alarm	$\mu_1, \mu_2 \ldots \mu_n$	
fuzzy	no	MeanMax	2	infered v.	Alarm		

Note: $\mu_1, \mu_2 \ldots \mu_n$ are available if the example activates at least one rule. For a fuzzy output defuzzified with *sugeno* ou *area* operators, they are equal to the inferred output value membership degrees to MF $1, 2 \ldots n$. For a fuzzy output defuzzified with a *MeanMax* operator, or for a crisp output obtained through a *MaxCrisp* operator, they represent the matching degree of all possible outputs: MF values for a fuzzy output, or real values for a crisp output.

The column *Bl* is 1 if the item is inactive, 0 otherwise. Finally, the last column gives the current value of the performance index.

Available alarm values:
Integer values which depend on the defuzzification operator:

- NOTHING (Value 0): All types. Everything is normal.
- NO_ACTIVE_RULE (Value 1): All types. The example does not fire any rule of the rule base.
- AMBIGUITY (Value 2): SugenoClassif (Crisp output) and MaxCrisp. The difference between the two main classes is less than a threshold (default value AMBIGU_THRES = 0.1).
- NON_CONNEX_AREA (Value 3): WeArea - Set when the area defined by the fired fuzzy sets (threshold set to MIN_THRES = 0.1 by default) is non connex.
- NON_CONNEX_SEGMENT (Value 4): MeanMax - Set when the max corresponds to two fuzzy sets (with a tolerance threshold set by default to EQUALITY_THRES = 0.1) and the resulting segment is non connex.

The performance function also computes the coverage index, which depends on the minimum matching degree given as an argument.

General file format for implicative rules

If the output on which the performance is calculated is implicative, new columns are introduced in the perf file, in the following order.

- k columns $MF_1, \ldots MF_k$ right after column *OBS*. k is the number of output fuzzy sets. The value written in each column is the membership degree of the observed output to the corresponding fuzzy set.
- k columns $MF_1, \ldots MF_k$ right after column *Al.* k is the number of output fuzzy sets. The value written in each column is the membership degree of the inferred output to the corresponding fuzzy set.
- *MINK*, which contains the inferred possibility distribution minimum kernel value.
- *MAXK*, which contains the inferred possibility distribution maximum kernel value.
- *MINS*, which contains the inferred possibility distribution minimum support value.
- *MAXS*, which contains the inferred possibility distribution maximum support value.
- *MATCH*, which gives the matching degree between the observed output and the inferred possibility distribution.

3 Rule base characteristics

The main characteristics of a rule base are summarized within a dedicated structure called *InfoRB*, the rule base information structure.

It is made up of the following attributes:

- maxR : maximal number of rules according to the FIS structure
- nR : number of rules
- maxVr : maximum number of variables in a rule
- meanVr: mean number of variables per rule
- nVar : number of distinct variables used in rules
- meanMF: mean number of MF per used variable

- nClass: number of classes (0 if not classif)
- nRc : number of rules per class or MF

The structure also includes two methods allowing attribute handling: *Print* and *WriteHeader* to print the column labels. These print functions use as field separator the '&' character (Well known to latex users!)

The *AnalyzeRB* function, from class FIS, is used to fill a *InfoRB* structure with respect to a given output of a fuzzy inference system.

Part III
Detailed class structure

A programmer's documentation of the C++ and Java classes is available in html format on the FisPro home page.

1 Known problems

- Most learning methods use a tolerance threshold EPSILON=10^{-6}. Before using these methods, it is better to normalize the data between 0 and 1, if their range is very high or very small.
- The HFP induction method is based on a data file only. It considers that there is a single output which corresponds to the data file last column.
- Once memory is allocated, some Java virtual machines (JVM) do not free it while the Java process is active, even if the memory is no longer used. Large arrays can be allocated in FisPro when reading big data files. The only solution to free memory is then to save current work, exit FisPro and relaunch it.

Part IV
Fuzzy Logic Elementary Glossary

1 Linguistic variable and fuzzy inference system

- fuzzy set : A fuzzy set is defined by its membership function. A point in the universe, x, belongs to a fuzzy set, A with a membership degree, $0 \leq \mu_A(x) \leq 1$.

  Figure 7 shows a triangle membership function.

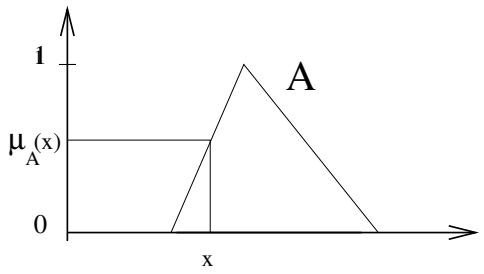


Figure 7: A triangle membership function

- Fuzzy set support: the set of real values for which the membership degree is greater then zero, $S_A = \{x|\mu_A(x) > 0\}$

- Fuzzy set kernel: the set of real values for which the membership degree is equal to 1, $K_A = \{x|\mu_A(x) = 1\}$

- Fuzzy set prototype: a point is a fuzzy set prototype if its membership degree is equal to 1.

- Partitioning: Partitioning defines the fuzzy sets for a given variable range. These sets are denoted A_1, A_2, ...

- Standardized fuzzy partition: a fuzzy partition of the X_i variable is called a standardized fuzzy partition if $\forall x \in X_i, \sum_j \mu_{A_j^i}(x) = 1$.

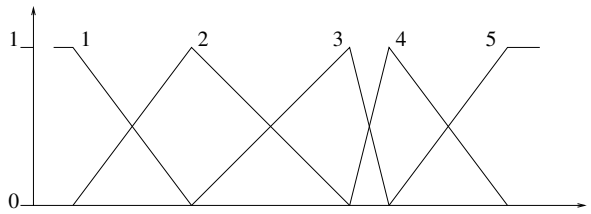


Figure 8: Example of a standardized fuzzy partition

- Linguistic variable: In a standard fuzzy partition, each fuzzy set corresponds to a linguistic concept, for instance *Very low*, *Low*, *Average*, *High*, *Very High*. During reasoning the variables are referred to by the linguistic terms so defined, and the fuzzy sets determine the correspondence with the numerical values.

- Fuzzy rule: A fuzzy rule is written as ***If*** *situation* ***Then*** *conclusion*. The situation, called rule premise or antecedent, is defined as a combination of relations such as $x\ is\ A$ for each component of the input vector. The conclusion part is called consequence or conclusion.

- Operators:
  - IS : the relation $x\ is\ A$ is quantified by the membership degree of x to the fuzzy set A.
  - AND : conjunction operator, denoted $\wedge$, the most common operators are minimum and product.
  - OR : disjunction operator, the most common are maximum and sum.

- Incomplete rule: A fuzzy rule is said to be incomplete if its premise is defined by a subset of the input variables. For instance, the rule

$$IF\ x_2\ is\ A_2^1\ THEN\ y\ is\ C_2$$

  is an incomplete rule, as the variable x_1 does not appear in its premise. Expert rules are generally incomplete rules. Formally an incomplete rule can be rewritten as an implicit combination of logical connectors AND and OR operating on all the variables. If the universe of the variable x_1 is split into three fuzzy sets, the above rule can also be written as:

$$IF\ \ (x_1\ is\ A_1^1\ OR\ x_1\ is\ A_1^2\ OR\ x_1\ is\ A_1^3)\ \ AND\ x_2\ is\ A_2^1\ THEN\ y\ is\ C_2.$$

- Item : an item or individual is composed of a p-dimensional input vector x, and eventually of a q-dimensional output vector.

- Matching degree: For a given data item and a given rule, the rule matching degree, or weight, is denoted w. It is obtained by the conjunction of the premise elements: $w = \mu_{A_1^i}(x_1) \wedge \mu_{A_2^i}(x_2) \wedge \ldots \wedge \mu_{A_p^i}(x_p)$, where $\mu_{A_j^i}(x_j)$ is the membership degree of the x_j value to the fuzzy set A_j^i.

- Activation: An item is said to activate a rule, if the rule matching degree for the item is greater than zero.

- Blank threshold: An example is said to be blank or inactive if its maximum matching degree over the rule base is smaller than a threshold value, called blank threshold.

- Rule prototype: an item is a rule prototype if the rule matching degree for this item is equal to 1, i.e. if example values are proptypes of the premise MFs.

- Fuzzy inference system (FIS): A fuzzy inference system is composed of three blocks, as shown in Figure 9. The first block is the fuzzification block. It transforms numerical values into membership degrees to the different fuzzy sets of the partition. The second block is the inference engine, with the rule base. The third one implements the defuzzification stage if necessary. It yields a crisp value from the rule aggregation result.

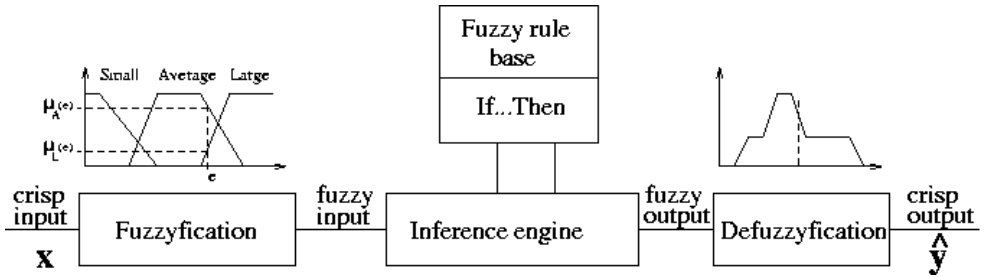


Figure 9: A fuzzy inference system

- System inferred output: denoted $\widehat{y_i}$ for the i_{th} item. The inferred value, for a given input, depends both on the rule aggregation and the defuzzification operators.

- The rules *If x is A then y is C* are of two types:

    1. Conjunctive rules: In this case the rule represents positive knowledge, inputs (*A*) and output (*C*) represent pairs of conjunctly possible values. The rule semantic is: *If Input is of type A Then a possible output value is C*. The rule relationship between inputs and outputs is modelled by a conjunction (t-norm). These rules are inspired from the data base paradigm *it is possible because it has been observed*, and implement reasoning through similarities. The inferred output is a guaranteed possibility distribution.
    2. Implicative rules: The rule represents negative knowledge and its semantic becomes: *If Input is of type A Then Output must be in C*. This more formal expression is issued from the Logic research branch of Artificial Intelligence and Computer Science. Instead of cumulating possible values, it works by successive elimination, and removes all values that do not satisfy the rule constraints. The rule relationship between inputs and outputs is modelled by an implication, which may or may not be fuzzy. The inferred output is a (usual) potential possibility distribution. For an expert, the passage from positive to negative knowledge requires a modelling step.

    For a detailed comparison between conjunctive and implicative rules, please refer to [13].

- Inference mechanisms: two different ones are necessary:

    1. FITA : First Infer Then Aggregate. It allows inferring a conclusion for each rule separately before aggregating these conclusions. It is used for conjunctive rules, and for implicative ones with precise input data.
    2. FATI : First Aggregate Then Infer. In this case all rules (constraints) are aggregated before inferring the output possibility distribution. Much more difficult to implement, this method is required for implicative rules, as soon as at least one input value is imprecise.

2 Conjonctive rules

There are two main families of conjunctive fuzzy rules:

1. Mamdani type. The rule conclusion is a fuzzy set.
   The rule is written as:

$$IF\ x_1\ is\ A_1^i\ AND\ x_2\ is\ A_2^i\ \ldots\ AND\ x_p\ is\ A_p^i$$
$$THEN\ y_1\ is\ C_1^i\ \ldots\ AND\ y_q\ is\ C_q^i$$

   where $A_j^i\ and\ C_j^i$ are fuzzy sets defining the input and output space partitioning.

2. Takagi-Sugeno type. The rule conclusion is a crisp value.
   The conclusion of the i_{th} rule for the j_{th} output is calculated as a linear function of the input values: $y_j^i = b_{jo}^i + b_{j1}^i x_1 + b_{j2}^i x_2 + \cdots + b_{jp}^i x_p$, also denoted $y_j^i = f_j^i(x)$.

   In FisPro, for interpretability reasons, the conclusion is limited to a constant b_{jo}^i.

- **Rule aggregation** is done in a disjunctive way for conjunctive rules, meaning that each rule opens a possible range for the output. Two operators are available to aggregate the rule conclusions: *max* or *sum*.

  The rule conclusions are numerical values for a crisp output, or MF labels for a fuzzy output. They constitute a set of possible values.

  The resulting levels are stored into the *MuInfer* array, the *RuleInfer* one contains the rule number corresponding to the maximum, in case of *max* aggregation, or the last rule number whose degree has been added, in the *sum* aggregation case. Both operators can be used with crisp or fuzzy output.

  Note :

  $C\ r$ the conclusion of the rth rule.

  m the number of output MFs (fuzzy output) or the number of distinct rule conclusion values (crisp output).

  The activation levels after aggregation are the following:

  - max: $\forall j = 1, \ldots, m$
    $W^j = \left\{ \max_r \left(w^r(x) \right) \mid C^r = j \right\}$

– sum:

* crisp output $\forall j = 1, \ldots, m$
  $W^j = \sum_r (w^r(x)) \mid C^r = j$
* fuzzy output $\forall j = 1, \ldots, m$
  $W^j = min\left(1, \left\{\sum_r (w^r(x)) \mid C^r = j\right\}\right)$

- **Defuzzification**: This operation is required for conjunctive rules and uses aggregation results.

  The inferred value, $\widehat{y}_i$ for the i_{th} example, depends on the output nature and its defuzzification operators.

  – crisp output

  1. **sugeno** operator

$$\widehat{y}_i = \frac{\sum_{j=1}^{m} W^j C^j}{\sum_{j=1}^{m} W^j} \tag{4}$$

  2. **MaxCrisp** operator

$$\widehat{y}_i = \left\{j = argmax W^j \left(C^j\right) \mid j = 1 \ldots m\right\}$$

  – fuzzy output

  Figure 10 illustrates the defuzzification possibilities for a fuzzy output.

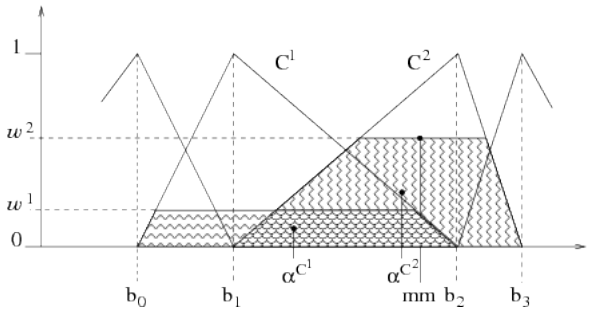


Figure 10: Defuzzification

  1. **weighted area** This operator favors the interpolation between linguistic labels. The inferred output is:

$$\widehat{y_i} = \frac{\sum_{j=1}^{m} \alpha^{C^j}\ area(C_\alpha^j)}{\sum_{j=1}^{m} area(C_\alpha^j)} \tag{5}$$

where m is the number of MFs in the partition, $\alpha = W^j$ is the activation level for the j_{th} MF, α^{C^j} is the centroid abscissa of C_α^j, and C_α^j a new MF, defined from C^j as follows:

$$\mu^{C_\alpha^j}(x_i) = \begin{cases} \mu(x_i) & if\ \mu^{C^j}(x_i) \leq \alpha \\ \alpha & sinon \end{cases}$$

2. **average of maxima** The output is $\widehat{y_i} = mm$ (figure 10). Only the segment corresponding to the maximum activation level is considered, which means the operator mainly operates within one linguistic label.
3. **sugeno** It is the same formula as for a crisp output (equation 4), but C^j now represents the middle of the j_{th} MF kernel.

3 Implicative rules

Implicative rule aggregation is conjunctive, because each rule imposes a constraint on the output. The handling of this set of constraints is done using an intersection operator (t-norm). If no rule is active for a given input vector, the output is the universal fuzzy set, meaning that, in case of total ignorance, all output values are equally possible.

Three implication operators are available:

- Resher-Gaines : $a \rightarrow b = \begin{cases} 1 & si\ a \leq b \\ 0 & sinon \end{cases}$
- Goguen : $a \rightarrow b = \begin{cases} 1 & si\ a = 0 \\ min(1, \frac{b}{a}) & sinon \end{cases}$
- Gödel : $a \rightarrow b = \begin{cases} 1 & si\ a \leq b \\ b & sinon \end{cases}$

The *Resher-Gaines* operator is not fuzzy: the possibility distribution corresponds to the kernel of the other two, *Goguen* and *Gödel*.

The output partitions must be adapted to the aggregation mode specific to implicative rules. We propose the concept of Quasi standard partition (QSP) as a compromise between interpretability and coherence. Indeed the QSP is derived from a SFP partition, which is well known for its advantages in terms of interpretability. The extra fuzzy sets allow to have a non empty intersection when several rules are simultaneously activated. Figure 11 displays a SFP partition and the equivalent SFP one.

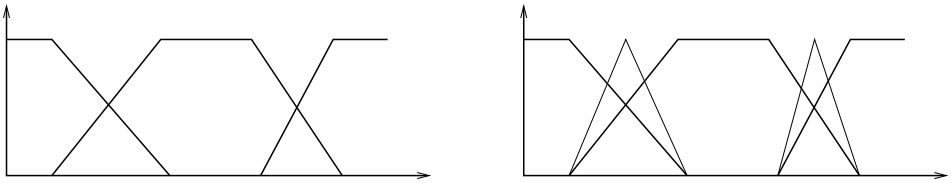


Figure 11: A SFP partition and the QSP equivalent one

Implicative rules offer several advantages compared to conjunctive ones:

- Modus Ponens: The classical version, $A \wedge (A \rightarrow C) \models C$, is generalized using fuzzy sets and a fuzzy implication: $A' \wedge (A \rightarrow C) \models C'$.
- Inferential independence: In the rule base $\{A_j \rightarrow C_j, j = 1, \ldots, n\}$, The inferred output for A_i is equal to C_i, if the rule $A_i \rightarrow C_i$ exists, whatever the other rules are.
- Respect imprecision: Because rule accumulation corresponds to eliminating possible output values, the imprecision of the output possibility distribution is meaningful and may be analysed.

  It is also possible to extract a precise value by defuzzification. The current implementation does not allow to parameterize the defuzzification operato. The default one is the *Mean of maxima*, which corresponds to the middle of the distribution kernel.
- Detect conflicts: An empty output possibility distribution means there is an inconsistency in the rule base, the contraints are not compatible.

4 Learning

- Supervised learning: It induces an input-output mapping from a data set, called learning data set. It is usually limited to a MISO (multiple input

single output) system. The learning data set includes n items.

- Coverage index: Some items may activate the rules with a low matching degree. An item will be considered as inactive if its maximum matching degree over all rules is lower than a user chosen threshold (default value= *0.1*. Inactive items are not managed by the FIS. The number of inactive items is used to define a coverage index, calculated as follows: $CI = fracAn$. *A* is the number of active items and *n* the number of rows in the data file.

- Performance indices, calculated from the active examples only.

  For a classification case:

  $$PI = \sum_{i=1}^{A} mc(i)$$

  where $mc(i)$ equa;s 1 for a misclassified item, 0 otherwise.

  For a regression case, three indices are available:

  - $PI = \frac{1}{A}\sqrt{\sum_{i=1}^{A} \|\widehat{y_i} - y_i\|^2}$
  - $RMSE = \sqrt{\frac{1}{A}\sum_{i=1}^{A} \|\widehat{y_i} - y_i\|^2}$
  - $MAE = \frac{1}{A}\sum_{i=1}^{A} |\widehat{y_i} - y_i|$

  PI is the historical accuracy index used by previous versions of *FisPro*, *RMSE* is the Root Mean Squared Error, et *MAE* is the Mean Absolute Error.

5 Possibility distribution

Let U a set of elementary events, u. One calls possibility measure denoted Π a function defined on the set of all parts $P(U)$ of U, taking values in $[0, 1]$ such as:

- $\Pi(\emptyset) = 0$
- $\Pi(U) = 1$

- $\forall i, A_i \in P(U), \Pi(\bigcup A_i) = sup\Pi(A_i)$

A possibility degree $\Pi(A) = 1$ means that the event A is completely possible, inversely $\Pi(A) = 0$ means that A is impossible.

A possibility distribution assigns to each element u of U a possibility $\pi(u) \in [0, 1]$. The distribution is normalized: $\sup_{u \in U} \pi(u) = 1$.

Guaranteed possibility

A guaranteed possibility measure Δ is a function defined on the set of all parts of U, $[0, 1]$ taking values in $[0, 1]$, such as:

- $\Delta(\emptyset) = 1$
- $\Delta(A_1 \cup A_2) = min(\Delta(A_1), \Delta(A_2))$

Relation between possibility degree and guaranteed possibility degree:

$$\forall A \subseteq U, \Delta(A) = inf_{u \in A} \pi(u)$$

.

Thus, if $\Delta(A) = \alpha$, all elementary events $u \in A$ are guaranteed possible at level α.

Guaranteed possibility distributions are denoted δ.

Interpretation of possibility degree and guaranteed possibility degree

- $\pi_X(u) = 1$: nothing keeps x from being equal to u, u is a completely possible value.
- $\pi_X(u) = 0$: u is an impossible value for x.
- $\delta_X(u) = 1$: u is a possible value for x, for instance because it was observed.
- $\delta_X(u) = 0$: it does not mean that u is an impossible value for x, but it only indicates that nothing guarantees it.

References

[1] L. Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees*. Wadsworth International Group, Belmont CA, 1984.

[2] S. Chen, S. A. Billings, and W. Luo. Orthogonal least squares methods and their application to non-linear system identification. *Int. J. Control*, 50:1873–1896, 1989.

[3] S. Chen, C. F. N. Cowan, and P. M. Grant. Orthogonal least squares learning algorithm for radial basis function networks. *IEEE Transactions on Neural Networks*, 2 (No 2):302–309, March 1991.

[4] J. Valente de Oliveira. Semantic constraints for membership functions optimization. *IEEE Transactions on Systems, Man and Cybernetics*, 29:128–138, 1999.

[5] Sébastien Destercke, Serge Guillaume, and Brigitte Charnomordic. Building an interpretable fuzzy rule base from data using orthogonal least squares-application to a depollution problem. *Fuzzy Sets and Systems*, 158:2078–2094, 2007.

[6] Pierre-Yves Glorennec. Quelques aspects analytiques des systèmes d'inférence floue. *Journal Européen des Systèmes automatisés*, 30 (2-3):231–254, 1996.

[7] Pierre-Yves Glorennec. *Algorithmes d'apprentissage pour systèmes d'inférence floue*. Editions Hermès, Paris, 1999.

[8] Serge Guillaume. Designing fuzzy inference systems from data: an interpretability-oriented review. *IEEE Transactions on Fuzzy Systems*, 9 (3):426–443, June 2001.

[9] Serge Guillaume and Brigitte Charnomordic. *A new method for inducing a set of interpretable fuzzy partitions and fuzzy inference systems from data*, volume 128 of Studies in Fuzziness and Soft Computing, pages 148–175. Springer, 2003.

[10] Serge Guillaume and Brigitte Charnomordic. Generating an interpretable family of fuzzy partitions. *IEEE Transactions on Fuzzy Systems*, 12 (3):324–335, June 2004.

[11] J. Hohensohn and J. M. Mendel. Two pass orthogonal least-squares algorithm to train and reduce fuzzy logic systems. In *Proc. IEE Conf. Fuzzy Syst.*, pages 696–700, Orlando, Florida, June 1994.

[12] H. Ichihashi, T. Shirai, K. Nagasaka, and T. Miyoshi. Neuro-fuzzy id3: a method of inducing fuzzy decision trees with linear programming for maximizing entropy and an algebraic method for incremental learning. *Fuzzy Sets and Systems*, 81:157–167, 1996.

[13] Hazael Jones, Brigitte Charnomordic, Didier Dubois, and Serge Guillaume. Practical inference with systems of gradual implicative rules. *IEEE Transactions on Fuzzy Systems*, 17 (1):61–78, 2009.

[14] J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81–106, August 1986.

[15] Li-Xin Wang and Jerry M. Mendel. Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man and Cybernetics*, 22 (6):1414–1427, November/December 1992.

[16] L. A. Zadeh. Is there a need for fuzzy logic? *Information Sciences*, 178 (13):2751–2856, 2008.