2014 IEEE International Conference on Fuzzy Systems (FUZZ-IEEE)
July 6-11, 2014, Beijing, China

Specialized Software for Fuzzy Natural Logic and Fuzzy Transform Applications

Vilém Novák, Viktor Pavliska and Radek Valášek

***Abstract*—In this paper, three special soft computing software systems are presented. The systems are based on the original results in two areas: fuzzy natural logic and fuzzy transform.**

The first software is LFL Controller that is a universal SW system that can be used in fuzzy or linguistic control, and in decision-making. The system implements results of fuzzy natural logic, namely the theory of evaluative linguistic expressions and perception-based logical deduction. This means the control or decision strategy are defined directly in natural language. Additionally, it can also realize classical fuzzy control on the basis of relational interpretation of fuzzy IF-THEN rules. This system has wide range of applications and was already applied in control of real plants.

The second system is LFL Forecaster that is a specialized SW for analysis and forecasting of time series. The analysis is realized using F-transform and forecasting using results of fuzzy natural logic.

The third system is FT-Studio that is a specialized SW for computation of fuzzy transform of functions that can be defined either using a formula, or given by data.

I. Introduction

In this paper we describe three special software systems: *LFL-Controller, LFL-Forecaster* and *FT-studio* that implement original results of the theoretical research in *fuzzy natural logic* and *fuzzy transform* (F-transform). The first two systems are the main representatives of a class of *Linguistic Fuzzy Logic* software systems developed in our institute. The name, shortened to LFL, means that results of the mathematical theory of fuzzy natural logic are implemented in the software. These systems have many kinds of applications. The LFL-Controller has applications in control and decision-making, and the LFL-Forecaster is devoted to the analysis and forecasting of time series.

The FT-studio is a specialized software whose aim is to enable the user to learn the method and technique of fuzzy transform and to make various kinds of experiments with it. The system works with several possibilities how the input function can be specified.

Below, we will briefly outline the main principles of the theories implemented in the software and then describe functioning of the considered software systems. Demo versions of all three systems can be downloaded from our WEB page http://irafm.osu.cz/ .

Vilém Novák, Viktor Pavliska and Radek Valášek are with the University of Ostrava, Institute for Research and Applications of Fuzzy Modeling, Center of Excellence IT4Innovations, 30. dubna 22, 701 03 Ostrava 1, Czech Republic (email: {Vilem.Novak,Viktor.Pavliska,Radek.Valasek}@osu.cz).

The paper has been supported by the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

II. Elements of fuzzy natural logic

The fuzzy natural logic[*)] is a formal mathematical theory that consists of:

(a) A formal theory of evaluative linguistic expressions explained in detail in [1] (see also [2]).
(b) A formal theory of fuzzy IF-THEN rules and approximate reasoning presented in [3], [4], [5], [6], [7].
(c) A formal theory of intermediate and generalized fuzzy quantifiers, presented in [8], [9], [10].

So far, the theories (a) and (b) are implemented in the LFL software systems. The central role is there played by the theory of *evaluative linguistic expressions*. These are expressions with the general form

$$\langle\text{linguistic modifier}\rangle\,\langle\text{TE-adjective}\rangle \qquad (1)$$

where ⟨TE-adjective⟩ is one of the adjectives "small, medium, big" (and possibly other, so called gradable, specific adjectives), or "zero" as well as arbitrary symmetric fuzzy number. The ⟨linguistic modifier⟩ is an intensifying adverb such as "very, roughly, approximately, significantly", etc. Since these expressions characterize values in an ordered scale, they may have also a sign ("positive–negative").

Simple evaluative expressions of the form (1) can be combined using logical connectives (usually "and" and "or") to obtain *compound* ones. At the same time, a limited usage of the particle "not" is also possible. Recall that there are limitations in natural language when using such connectives. It should be noted that compound evaluative expressions *do not* form a boolean algebra.

The linguistic modifiers in (1) belong to a wider linguistic phenomenon called *hedging* that is represented by a class of linguistic expressions specifying more closely the topic of utterance. In (1) are considered mostly special adverbs.

The modifiers can have narrowing and extending effect. *Narrowing* modifiers are "extremely, significantly, very, typically" and *widening* ones are "more or less, roughly, quite roughly, very roughly". Note that narrowing modifiers make the meaning of the whole expression more precise while widening ones do the opposite. Thus, "very small" is more precise than "small", which, on the other hand, is more precise (more specific) than "roughly small".

It should be noted that the case when ⟨linguistic hedge⟩ is not present (expressions such as "weak, large", etc.) is dealt

[*)] In older author's publications, this logic is called *fuzzy logic in broader sense*.

978-1-4799-2072-3/14/$31.00 ©2014 IEEE

with as a presence of *empty linguistic hedge*. Thus, all the simple evaluative expressions have the same form (1).

We distinguish *evaluative expressions* and *evaluative predications*. The latter are expressions of natural language of the form 'X is $\mathcal{A}$'. The $\mathcal{A}$ is an evaluative expression and X is a variable which stands for objects, for example "degrees of temperature, height, length, speed", etc. Examples are "temperature is high", "speed is extremely low", "quality is very high", etc. In general, the variable X represents certain features of objects such as "size, volume, force, strength," etc. and so, its values are usually real numbers.

Important notion is that of *linguistic context*. In our theory this is a triple of numbers $\langle v_L, v_S, v_R\rangle$ where v_L is the leftmost typically small value, v_S is typically medium value and v_R is the rightmost typically big value. The concept of context turned out to be very powerful in applications.

The evaluative linguistic predications are basic constituents of fuzzy/linguistic IF-THEN rules that are special conditional clauses of natural language. A set of such rules is called *linguistic description*, that is, a finite set of fuzzy/linguistic IF-THEN rules

$$\begin{aligned} \mathcal{R}_1 &= \mathsf{IF}\ X \text{ is } \mathcal{A}_1 \ \mathsf{THEN}\ Y \text{ is } \mathcal{B}_1,\\ \mathcal{R}_2 &= \mathsf{IF}\ X \text{ is } \mathcal{A}_2 \ \mathsf{THEN}\ Y \text{ is } \mathcal{B}_2,\\ &\ldots\ldots\ldots\ldots\ldots\ldots\ldots\\ \mathcal{R}_m &= \mathsf{IF}\ X \text{ is } \mathcal{A}_m \ \mathsf{THEN}\ Y \text{ is } \mathcal{B}_m \end{aligned} \tag{2}$$

where "X is $\mathcal{A}_j$", "$\mathcal{B}_j$ is Y", $j = 1,\ldots,m$ are evaluative linguistic predications. The linguistic description can be understood as a specific kind of a (structured) text which can be used for description of various situations or processes, or for effective description of a *control or decision strategy*.

If the linguistic description is understood as a special text then it requires a special inference method. Such a method is the *Perception-based Logical Deduction* (PbLD) that is a specific inference method working with genuine meaning of evaluative linguistic expressions and based on formal properties of mathematical fuzzy logic. It was described in several papers [2], [5], [7], [11]. It is specific for this inference that it is based on local properties of the linguistic description. Consequently, we can distinguish between the rules but at the same time deal with them as vague expressions of natural language. To obtain the specific conclusion, we also must use a defuzzification method. The PbLD method uses special methods: DEE (Defuzzification of Evaluative Expressions) and its smooth modification.

To demonstrate our method, let us consider the following linguistic description:

$$\begin{aligned} \mathcal{R}_1 &= \mathsf{IF}\ X \text{ is } \mathit{small} \ \mathsf{THEN}\ Y \text{ is } \mathit{small},\\ \mathcal{R}_2 &= \mathsf{IF}\ X \text{ is } \mathit{medium} \ \mathsf{THEN}\ Y \text{ is } \mathit{big},\\ \mathcal{R}_3 &= \mathsf{IF}\ X \text{ is } \mathit{big} \ \mathsf{THEN}\ Y \text{ is } \mathit{small} \end{aligned} \tag{3}$$

This description characterizes linguistically a function that has small functional values on the left and right side of the graphs and big ones in the middle. The result using PbLD method is depicted in part (a) of Fig. 1. In part (b) are extensions of the used evaluative expressions in the context $\langle 0, 0.4, 1\rangle$.

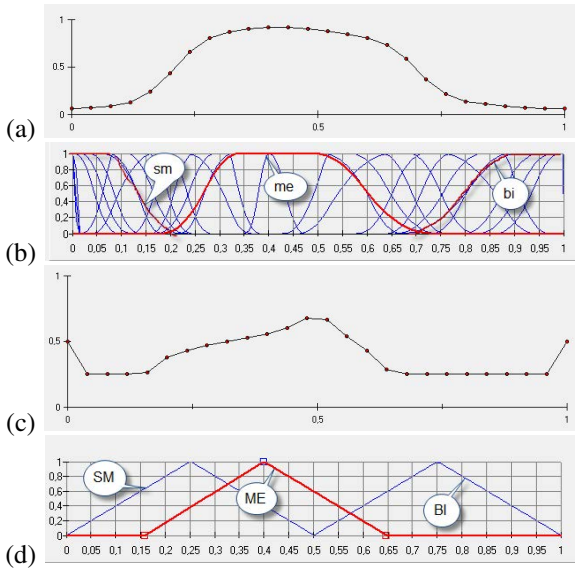

Fig. 1. (a) A function obtained from the simple linguistic description (3) using the PbLD method with smooth DEE defuzzification. (b) Extensions of the evaluative expressions "small–medium–big" in the context $\langle 0, 0.4, 1\rangle$ (c) A function obtained using Mamdani's-COG method from linguistic description of the form (3) interpreted as fuzzy relation constructed using triangles. (d) the triangular membership functions of the expressions SM, ME, BI used for obtaining of the function above.

III. Relational interpretation of linguistic description

An alternative interpretation of the linguistic description is the widely known relational one. The linguistic description consisting of m rules of the form (2) is in this case assigned one of two special fuzzy relations that are called *normal forms* (see [12], [13], [14], [15]).

(a) *Disjunctive normal form*

$$R_{DNF}(u,v) = \bigvee_{j=1}^{m} (A(u) \otimes B(v)), \qquad u \in U, v \in \tag{4}$$

where the operation $\otimes$ is a either of the minimum, product, or Łukasiewcz t-norm.

(b) *Conjunctive normal form*

$$R_{CNF}(u,v) = \bigwedge_{j=1}^{m} (A(u) \to B(v)), \qquad u \in U, v \in V \tag{5}$$

where $\to$ is an implication function (most often the Łukasiewicz one $a \to b = 1 \wedge (1 - a + b), \quad a, b \in [0,1]$).

In this case, the conclusion is formed by an image under a fuzzy relation. This procedure can also be explained as a composition of fuzzy relations.

Let an observation "X is $\mathcal{A}'$" interpreted by a fuzzy set $A' : U \longrightarrow L$ be given. Then the conclusion is a fuzzy set

$B' : V \longrightarrow L$ given by the membership function

$$B'(v) = \bigvee_{u \in U} (A'(u) \otimes R(u,v)), \qquad v \in V \tag{6}$$

where $R : U \times V \longrightarrow L$ is a fuzzy relation interpreting the linguistic description (it can be obtained, e.g., either as a disjunctive normal form from (4) or as a conjunctive normal form from (5)). The $\otimes$ is a special t-norm (quite often the minimum). Recall that the disjunctive normal form-based method with COG defuzzification is usually called "Mamdani's inference".

Let us emphasize that formula (6) provides a good approximation of a given function (see the mathematical justification in [12, Chapter 6]). One must be, however, more careful about shapes of the used membership functions of the fuzzy sets A, B in (4) or (5). For example, if we simply replace the evaluative expressions in the linguistic description (3) by triangular fuzzy sets SM, ME, BI depicted in Fig. 1(d) then using the Mamdani-COG method we obtain the function in Fig. 1(c).

Let us remark that in the literature, one can find also another concept of linguistic description (see, e.g., [16]). The authors, however, do not analyze syntax of the considered linguistic expressions — they use the unspecified term "linguistic labels". Moreover, they consider only simple extensions taken as their semantics.

IV. The Principle of F-transform

The fundamental idea of the theory of fuzzy transform (F-transform) is to map a continuous function $f : [a,b] \longrightarrow \mathbb{R}$ to a finite vector of numbers (*direct F-transform*) and then to transform it back (*inverse F-transform*). The result is a function $\hat{f}$ that approximates the original function f. The parameters of the F-transform can be set in such a way that the approximating function $\hat{f}$ has desired properties.

The first step of the F-transform procedure is to form a *fuzzy partition* of the domain $[a,b]$. It consists of a finite set of fuzzy sets $\mathcal{A} = \{A_0, \ldots, A_n\}$, $n \geq 2$, defined over nodes $a = c_0, \ldots, c_n = b$. Properties of the fuzzy sets from $\mathcal{A}$ are specified by five axioms, namely: *normality, locality, continuity, unimodality, and orthogonality*. The orthogonality is formally defined by

$$\sum_{i=0}^{n} A_i(x) = 1, \qquad x \in [a,b] \tag{7}$$

and it is often called the *Ruspini condition*.

A fuzzy partition $\mathcal{A}$ is called *h-uniform* if the nodes $c_0, \ldots, c_n$ are *h-equidistant*, i.e., for all $k = 0, \ldots, n-1$, $c_{k+1} = c_k + h$, where $h = (b-a)/n$ and the fuzzy sets $A_1, \ldots, A_{n-1}$ are shifted copies of a *generating function* $A : [-1,1] \longrightarrow [0,1]$ such that for all $k = 1, \ldots, n-1$

$$A_k(x) = A\left(\frac{x - x_k}{h}\right), \qquad x \in [c_{k-1}, c_{k+1}]$$

(for $k = 0$ and $k = n$ we consider only half of the function A, i.e. restricted to the interval $[0,1]$ and $[-1,0]$, respectively).

The membership functions $A_0, \ldots, A_n$ of fuzzy sets forming the fuzzy partition $\mathcal{A}$ are often called *basic functions*. Once the basic functions $A_0, \ldots, A_n \in \mathcal{A}$ are selected, we define a *direct F-transform* of a continuous function f as a vector $\mathbf{F}[f] = (F_0[f], \ldots, F_n[f])$, where each k-th *component* $F_k[f]$ is equal to

$$F_k[f] = \frac{\int_a^b f(x) A_k(x)\,dx}{\int_a^b A_k(x)\,dx}, \qquad k = 0, \ldots, n.$$

The meaning of $F_k[f]$ component is a *weighted average* of the functional values $f(x)$ where weights are the membership degrees $A_k(x)$. The *inverse F-transform* of f with respect to $\mathbf{F}[f]$ is a continuous function[†)] $\hat{f} : [a,b] \longrightarrow \mathbb{R}$ such that

$$\hat{f}(x) = \sum_{k=0}^{n} F_k[f] \cdot A_k(x), \qquad x \in [a,b].$$

It is proved that the function $\hat{f}$ differs from f (unless f is a constant function) but, under certain conditions, the sequence $\{\hat{f}_n\}$ uniformly converges to f for $n \to \infty$. All the details and full proofs can be found in [17], [18].

The F-transform introduced above is F^0-transform (i.e., zero-degree F-transform). Its components are real numbers. If we replace them by polynomials of arbitrary degree $m \geq 0$, we arrive at the higher degree F^m transform. This generalization has been in detail described in [18]. Let us remark that the F^1 transform enables to estimate also derivatives of the given function f as average values over wider area.

V. LFL Controller

The LFL Controller is a general software that makes it possible to work with linguistic descriptions, modify them and test their behavior. It implements more possibilities how they can be interpreted and more possibilities for derivation of the conclusion. The leading interpretation is linguistic in combination with PbLD method but relational interpretation is also available.

[†)]By abuse of language, we call by direct as well as inverse F-transform both the procedure as well as its respective results $\mathbf{F}[f] = (F_0[f], \ldots, F_n[f])$ and $\hat{f}$.

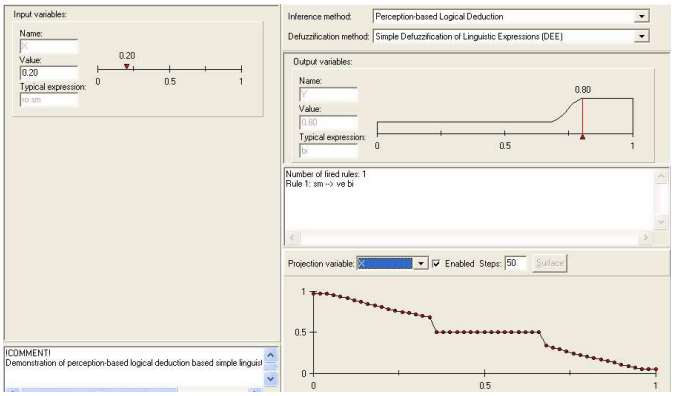

Fig. 2. The main screen of LFL Controller.

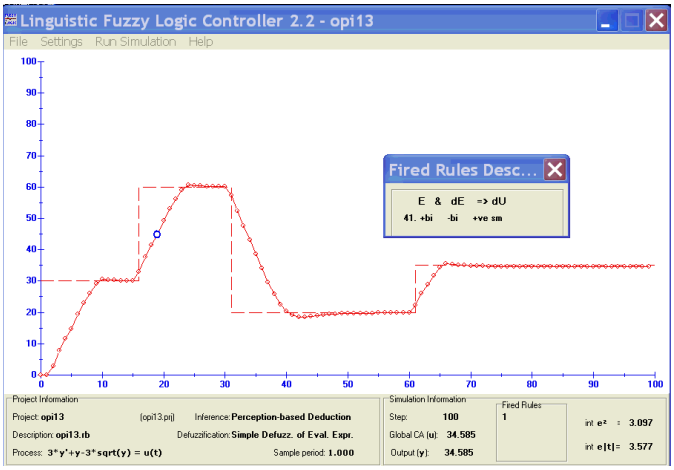

Fig. 3. The main screen of simulation of linguistic control in a closed feedback loop using LFL Controller.

The LFL Controller consists of the following parts:

(i) GUI using which we can design and test behaviour of the linguistic descriptions. The main screen is in Fig. 2. It is divided into 5 parts: the upper left contains 1–5 gauges using which values of the antecedent (input) variables are set. The lower left is comment, the upper right contains resulting fuzzy set with marked defuzzified value, the center right is the list of fired rules and the lower right contains the course of the consequent values (output) for all possible values of antecedents.

(ii) GUI (LFLCSim) using which we can test fuzzy control in a closed feedback loop of simple processes characterized using differential equations with constant coefficients extended possibly by few non-linearities. A typical screen with simulation of linguistic control of a simple, slightly non-linear third-order process is depicted in Fig. 3. The figure displays also one of the fired rules.

(iii) MATLAB / Simulink library which brings fuzzy modeling entities into the MATLAB environment in a graphical modeling manner (see example in Fig. 5). Practical problems are usually too complex and difficult to be modeled only by one technique, therefore a combination of several approaches is used to design system for solving the task.

(iv) General interface for other software systems using Microsoft COM (Component Object Model) technology – `RBaseCOM`. COM is an object framework which is used by developers to create re-usable software components, link components together to build applications, and take advantage of Windows services. COM objects can be used inside various programming languages.

The main objective of the LFL Controller is to enable design of linguistic descriptions and to realize inference on the basis of them. The leading method is PbLD. Hence, LFL Controller makes it possible to realize fully the original idea of fuzzy control — to apply genuine linguistic description of a control strategy in control of processes. Therefore, we speak about *linguistic control* which has the following properties

(i) Mathematical theory of the meaning of special expressions of natural language is applied. The computer behaves as if "understanding" them.

(ii) Application of the perception-based logical deduction makes it possible to derive a conclusion on the basis of (vague) linguistic description formulated directly in natural language.

LFL Controller can be applied both for linguistic control and for decision-making. In the former case, the control engineer can focus only on the control strategy which is described in natural language and needs not care about shapes of fuzzy sets; these are usually hidden to him. Instead, the control engineer modifies the used evaluative linguistic expressions.

The LFL Control shares nice properties of the classical fuzzy control (cf. [19]) but has several additional ones:

(a) The linguistic description is written in genuine linguistic form which is well understandable to people, even after years. Therefore, it is easy to modify the description any time without big effort, if necessary.

(b) The linguistic description characterizes a *general control strategy* which is often common to many kinds of processes. Therefore, the same description can be used for control of various kinds of processes.

(c) The control is very robust and does not require modifications even if the conditions are varying a lot and/or the control is subject to many random disturbances.

(d) The linguistic context of the input variables can be automatically learned. Moreover, it can also be continuously modified so that the resulting control is very precise.

LFL Controller also enables to view extensions of all evaluative expressions that can be used in the application. The extensions are in Fig. 4. It is also possible to modify the meaning of hedges and thus, to modify the extensions. However, this is not recommended because all of them were carefully tuned to fit linguistic feeling of people. The user is recommended to define linguistic expressions directly and the computer is expected to "understand" them and to act accordingly.

The LFLCSim (the testing GUI program of LFL Controller) enables us to simulate control using one of four fuzzy versions of the classical P, PD, PI, and PID controllers. Let us consider the variables *error* E_t, its *derivative/change* dE_t, its *second derivative/change* d^2E_t, *control action* U_t and its *derivative* dU_t. Then the following linguistic descriptions can be used for process control:

(i) *PD-fuzzy controller* with rules of the form

$$\text{IF } E_t \text{ is } \mathcal{A} \text{ AND } dE_t \text{ is } \mathcal{B} \text{ THEN } U_t \text{ is } \mathcal{C}.$$

(ii) *PI-fuzzy controller* with rules of the form

$$\text{IF } E_t \text{ is } \mathcal{A} \text{ AND } dE_t \text{ is } \mathcal{B} \text{ THEN } dU_t \text{ is } \mathcal{C}.$$

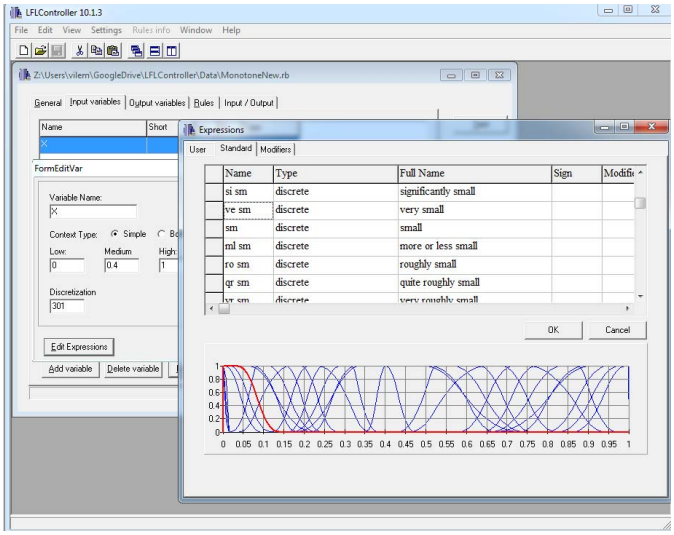

Fig. 4. Extensions of evaluative expressions in the given context $\langle 0, 0.4, 1\rangle$.

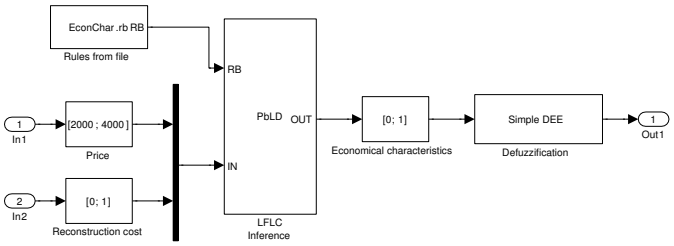

Fig. 5. Example of Simulink schema using FGML.

(iii) *PID-fuzzy controller* with rules of the form

$$\text{IF } E_t \text{ is } \mathcal{A} \text{ AND } dE_t \text{ is } \mathcal{B} \text{ AND } d^2E_t \text{ is } \mathcal{C} \text{ THEN } dU_t \text{ is } \mathcal{D}$$

where $\mathcal{A}, \mathcal{B}, \mathcal{C}, \mathcal{D}$ are specific evaluative expressions. Of course, any other kinds of variables can also be considered.

When LFL Controller is applied in decision-making then the decision problem must be decomposed into several subproblems, each of which can be described using some linguistic description — see Figure 5. The final decision is obtained as an output of the summarizing linguistic description and it can be expressed both numerically as well as linguistically. More details including a sophisticated example can be found in [20].

An important feature of LFL Controller is also possibility to *learn linguistic description* from data. The method is described in [21]. It is used for learning linguistic description on the basis of monitored hand-operated successful control and also in LFL Forecaster described below. Many practical tests demonstrate that the learned linguistic description can repeat the control as successfully as the original hand-operated control.

Besides the linguistic interpretation, the LFL Controller makes it possible also to work with the classical relational interpretation of the linguistic description, namely, interpretation of the linguistic predications are just fuzzy sets that can be explicitly specified by the user.

It should be stressed that the relational interpretation, in fact, is not based on expert knowledge expressed in natural language. The used fuzzy sets have in most cases triangular shape and they are further modified to obtain the best control. Consequently, the resulting rule base comprises of fuzzy sets which have minimal, if any, relation to the original meaning of the words used by experts when presenting their knowledge. The control engineers, in fact, use a system that provides efficient, mathematically well justified *approximation of a control function*. This is the reason why modification of shapes of fuzzy sets is necessary. LFL Controller enables to do the same. However, its main strength is in the above discussed linguistic control.

Besides the two GUI systems, LFL Controller includes also COM object which has following interface:

- `LoadFromFile(`**`string`** `FileName)`: loads rulebase from rb file
- **`int`** `NumInputVars()`: returns number of input variables
- **`double`** `LoBoundOfVar(`**`int`** `VarIndex)`: returns low bound of variable context
- **`double`** `HiBoundOfVar(`**`int`** `VarIndex)`: returns high bound of variable context
- **`string`** `VarName(`**`int`** `VarIndex)`: returns variable name
- **`int`** `NumFiredRules()`: returns number of fired rules of latest inference
- **`int`** `FiredRule(`**`int`** `i)`: returns i-th fired rule of latest inference
- **`double`** `Inference(`**`double`** `*Inputs)`: computes inference over given array of doubles as input
- `setLoBoundOfVar(`**`int`** `VarIndex,` **`double`** `NewLo)`: sets low bound of variable context
- `setHiBoundOfVar(`**`int`** `VarIndex,` **`double`** `NewHi)`: sets high bound of variable context
- `setInferenceMethod(`**`int`** `infMethodIndex)`: sets inference method used for following inference Possible values of `infMethodIndex` are:
  1) Perception-based Logical Deduction (PbLD).
  2) Fuzzy Approximation (CNF).
  3) Fuzzy Approximation (DNF) with Minimum t-norm.
  4) Fuzzy Approximation (DNF) with Product t-norm.
  5) Fuzzy Approximation (DNF) with Łukasiewicz t-norm.
- `setDefuzMethod(`**`int`** `defuzMethodIndex)`: sets defuzzification method used after inference. Possible values of `defuzMethodIndex` are:
  1) Simple Center of Gravity (COG).
  2) Modified Center of Gravity (MCOG).
  3) Simple Defuzzification of Linguistic Expressions (DEE).

4) Simple DEE using COG.
5) Defuzzification of Linguistic Expressions.
6) Mean of Maxima (MOM).

The defuzzification methods 3)–5) should be used with the PbLD inference method.

Linguistic control realized using LFL Controller can be applied for control of various kinds of processes. Besides many tens of simulations and control of physical models such as MATLAB helicopter or magnetic levitation, it was applied in control of real plants: control of a plaster kiln, system of hydraulic transition water–oil, control of massive 100t steam generator. The most successful application is control of 5 smelting furnaces TLP9 in Al Invest company in a small village Břidličná in the Czech Republic. The application has been in detail described in [22].

Let us mention that LFL Controller makes it possible to work also with the disjunctive and conjunctive normal forms in combination with several kinds of defuzzifications. The leading method, however, is the PbLD method. Application of the conjunctive normal form is not so frequent but gives also good results. A more detailed analysis of this possibility can be found in [15].

VI. LFL Forecaster

This is a specialized software system that provides analysis of time series, forecast of its future behavior and linguistic evaluation of trend in various parts of it (specified by the user). Its functioning is based on combination of the F-transform with fuzzy natural logic.

A time series is a stochastic process (see [23], [24]) $X : Q \times \Omega \longrightarrow \mathbb{R}$ where $Q = \{0, \ldots, p\} \subset \mathbb{N}$ is a finite set whose elements are interpreted as time moments. Our basic assumption is that the time series can be decomposed into three components:

$$X(t,\omega) = TC(t) + S(t) + R(t,\omega), \qquad t \in Q, \omega \in \Omega, \quad (8)$$

where $TC(t)$ is a *trend-cycle* and $S(t)$ is a seasonal component of the time series $X(t)$. The $TC(t)$ is assumed to be an ordinary real function. The seasonal component $S(t)$ is considered to be a mixture of complex periodic functions

$$S(t) = \sum_{j=1}^{r} P_j \, e^{i(\lambda_j t + \varphi_j)} \quad (9)$$

for some finite r where λ_j are frequencies, φ_j phase shifts and P_j are amplitudes. The $R(t,\omega)$ is a random noise, i.e. each $R(t)$ for $t \in Q$ is a random variable with the zero mean value finite variance.

The trend-cycle TC is estimated using the F-transform method described above. It was proved that this estimation is obtained with high fidelity (cf. [25]). By setting a proper fuzzy partition (determined on the basis of computation of periodogram — see [24]), we first compute the F-transform of $X(t)$

$$\mathbf{F}[X] = (F_1[X], \ldots, F_n[X]).$$

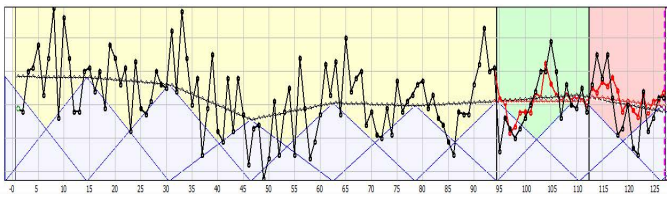

Fig. 6. Main screen of LFL Forecaster on which a time series together with estimation of its trend-cycle using F-transform, its forecast and forecast of the whole time series are depicted. The used fuzzy partition is also depicted.

Estimation of the trend-cycle is obtained using the inverse F-transform:

$$TC(t) \approx \hat{X}(t).$$

The LFL Forecaster system forecasts future components $(F_{n+1}[X], \ldots, F_{n+k}[X])$ and also the seasonal component $S(t)$. The forecast of the future components is obtained using the *perception-based logical deduction* on the basis of a linguistic description (2) learned from the past data. As antecedent variables, we consider the F-transform components of the given time series $F[X_i]$, $i = 1, \ldots, n-1$ as well as their first- and second-order differences:

$$\Delta F[X_i] = F[X_i] - F[X_{i-1}], \qquad i = 1, \ldots, n-1$$
$$\Delta^2 F[X_i] = \Delta F[X_i] - \Delta F[X_{i-1}], \qquad i = 2, \ldots, n-1$$

respectively. All possible combinations are formed and trained w.r.t. validation set, the best combination is chosen and used in the forecast. The automatically generated linguistic description is also provided to the user so that he/she can get better idea about behavior of the studied time series.

A special task is linguistic evaluation of the trend (tendency) of the time series. This is based on application of the mentioned first-degree F-transform that provides estimation of the average slope (tangent) in a specified area. Values of the slope are then linguistically evaluated using the theory of evaluative (linguistic) expressions. Typical examples of such evaluation are *fairly large decrease*, *huge increase*, *stagnating*, etc. These expressions characterize trend (tendency) of the time series in an area specified by the user. Note that for some time series, such as that depicted in Fig. 6, the tendency is not clear even when seeing its graph. Our solution provides objective estimation of it.

VII. FT-studio

The Fuzzy Transform Studio (FT-Studio) is a software system that computes and graphically depicts fuzzy transform of the degree 0–2 applied to functions with one variable. It is written in the Qt Framework.

The first step when defining the F-transform is to set fuzzy partition. The fundamental fuzzy partition is equidistant which fulfills the Ruspini condition (7). The user can define a number of nodes and the shape of fuzzy sets. The system makes it possible to select three kinds of fuzzy partitions that can be further tuned up as follows:

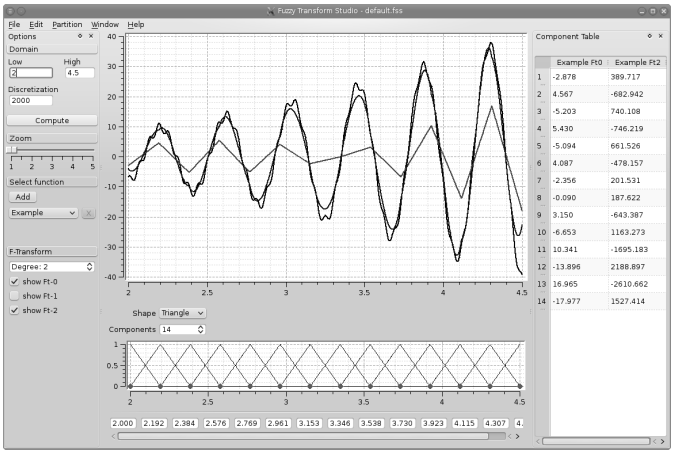

Fig. 7. Main screen of FT-studio on which approximation of a simple sin-function is depicted using F^0- and F^2-transform.

- shift any node but preserve the Ruspini condition,
- shift all nodes and keep them equidistant,
- shift any node and change the width of fuzzy sets (the Ruspini condition is then harmed).

Typical result of the F-transform of a given function f realized using FT-Studio is depicted in Fig. 7. The defined fuzzy partition is in lower part and the result of the inverse F-transform of a given function is graphically depicted in upper part. Values of the F-transform components are on the right. All the values can also be exported into Excel.

The original function f can be given either by mathematical formula or loaded in the form of data from a file. In Fig. 8, a Formula Builder is shown. Using it, the user can edit the formula and write it as a string. The parser can evaluate all standard trigonometric, logarithmic functions and binary operators. Moreover, special functions such as min, max, sgn and noise generators can also be used. It is possible to use

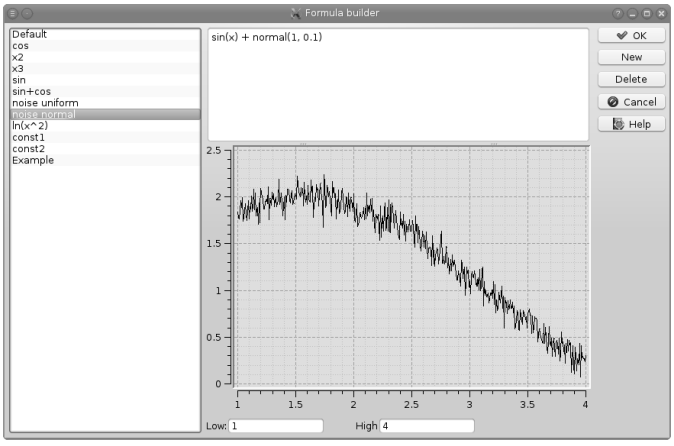

Fig. 8. FT-studio with depicted noise generated using build-in function.

also conditional expression in a form "(predicate) ? true : false" (the syntax is taken from C++). For example $(x < 0)$? normal(0, 1): $\sin(x)$ means that if x is less than zero then the function is generated as a noise with normal distribution and otherwise, it is generated as $\sin(x)$ for x greater or equal to zero. "normal(0, 1)" means normal distribution with zero mean value and variance equal to 1.

FT-Studio stores changes in a session file that can be later opened with all the fuzzy transform setting and a widget positions. Components of the fuzzy transform can be exported to the file and graph of all defined functions can be plotted.

Recall that F-transform has a lot of various applications. Besides time series mentioned above, many applications are in image processing (fusion, compression, edge detection, reduction, removing damage), numerical methods (solving differential equations), or data mining. More can be found in [17], [26], [27], [28].

VIII. Conclusions

In this paper, we presented three special soft computing software systems. The first one is LFL Controller that is a universal system for fuzzy or linguistic control and also for decision-making. The system has two GUI programs and provides also modules that can be included in user's application.

The second system is LFL Forecaster that is a specialized system for analysis and forecasting of time series. The analysis is realized using F-transform and forecasting using the methods of fuzzy natural logic.

The third system is FT-Studio that is a specialized system for computation of fuzzy transform of functions that can be defined either using a formula or given by data.

References

[1] V. Novák, "A comprehensive theory of trichotomous evaluative linguistic expressions," *Fuzzy Sets and Systems*, vol. 159, no. 22, pp. 2939–2969, 2008.

[2] ——, "Mathematical fuzzy logic in modeling of natural language semantics," in *Fuzzy Logic – A Spectrum of Theoretical & Practical Issues*, P. Wang, D. Ruan, and E. Kerre, Eds. Berlin: Elsevier, 2007, pp. 145–182.

[3] A. Dvořák and V. Novák, "Fuzzy logic deduction with crisp observations," *Soft Computing*, vol. 8, pp. 256–263, 2004.

[4] A. Dvořák and V. Novák, "Formal theories and linguistic descriptions," *Fuzzy Sets and Systems*, vol. 143, pp. 169–188, 2004.

[5] V. Novák, "Perception-based logical deduction," in *Computational Intelligence, Theory and Applications*, B. Reusch, Ed. Berlin: Springer, 2005, pp. 237–250.

[6] V. Novák and S. Lehmke, "Logical structure of fuzzy IF-THEN rules," *Fuzzy Sets and Systems*, vol. 157, pp. 2003–2029, 2006.

[7] V. Novák and I. Perfilieva, "On the semantics of perception-based fuzzy logic deduction," *International Journal of Intelligent Systems*, vol. 19, pp. 1007–1031, 2004.

[8] A. Dvořák and M. Holčapek, "L-fuzzy quantifiers of the type $\langle 1 \rangle$ determined by measures," *Fuzzy Sets and Systems*, vol. 160, pp. 3425–3452, 2009.

[9] V. Novák, "A formal theory of intermediate quantifiers," *Fuzzy Sets and Systems*, vol. 159, no. 10, pp. 1229–1246, 2008.

[10] P. Murinová and V. Novák, "A formal theory of generalized intermediate syllogisms," *Fuzzy Sets and Systems*, vol. 186, pp. 47–80, 2012.

[11] V. Novák, "Genuine linguistic fuzzy logic control: Powerful and successful control method," in *Computational Intelligence for Knowledge-Based Systems Design*, E. Hüllermeier, R. Kruse, and F. Hoffmann, Eds. Berlin: Springer, LNAI 6178, 2010, pp. 634–644.

[12] V. Novák, I. Perfilieva, and J. Močkoř, *Mathematical Principles of Fuzzy Logic*. Boston: Kluwer, 1999.

[13] I. Perfilieva, "Fuzzy function as an approximate solution to a system of fuzzy relation equations," *Fuzzy Sets and Systems*, vol. 147, pp. 363–383, 2004.
[14] ——, "Logical approximation," *Soft Computing*, vol. 7, no. 2, pp. 73–78, 2002.
[15] M. Štěpnička, U. Bodenhofer, M. Daňková, and V. Novák, "Continuity issues of the implicational interpretation of fuzzy rules," *Fuzzy Sets and Systems*, vol. 161, pp. 1959–1972, 2010.
[16] D. Sanchez-Valdes, A. Alvarez-Alvarez, and G. Trivino, "Linguistic description about circular structures of the mars' surface," *Applied Soft Computing*, vol. 13, p. 4738?4749, 2013.
[17] I. Perfilieva, "Fuzzy transforms: theory and applications," *Fuzzy Sets and Systems*, vol. 157, pp. 993–1023, 2006.
[18] I. Perfilieva, M. Daňková, and B. Bede, "Towards a higher degree F-transform," *Fuzzy Sets and Systems*, vol. 180, pp. 3–19, 2011.
[19] K. Michels, F. Klawonn, R. Kruse, and A. Nürnberger, *Fuzzy Control: Fundamentals, Stability and Design of Fuzzy Controllers*. Berlin: Springer, 2006.
[20] V. Novák, I. Perfilieva, and N. G. Jarushkina, "A general methodology for managerial decision making using intelligent techniques," in *Recent Advances in Fuzzy Decision-Making*, E. Rakus-Anderson, R. Yager, N. Ichalkaranje, and L. Jain, Eds. Heidelberg: Springer, 2009, pp. 103–120.
[21] R. Bělohlávek and V. Novák, "Learning rule base of the linguistic expert systems," *Soft Computing*, vol. 7, pp. 79–88, 2002.
[22] V. Novák and J. Kovář, "Linguistic IF-THEN rules in large scale application of fuzzy control," in *Fuzzy If-Then Rules in Computational Intelligence: Theory and Applications*, R. Da and E. Kerre, Eds. Boston: Kluwer Academic Publishers, 2000, pp. 223–241.
[23] J. Anděl, *Statistical Analysis of Time Series*. Praha: SNTL, 1976 (in Czech).
[24] J. Hamilton, *Time Series Analysis*. Princeton University Press: Princeton, 1994.
[25] V. Novák, I. Perfilieva, M. Holčapek, and V. Kreinovich, "Filtering out high frequencies in time series using F-transform," *Information Sciences*, (to appear).
[26] I. Perfilieva, V. Novák, and A. Dvořák, "Fuzzy transform in the analysis of data," *Int. Journal of Approximate Reasoning*, vol. 48, pp. 36–46, 2008.
[27] I. Perfilieva, "Fuzzy transforms: A challenge to conventional transforms," in *Advances in Images and Electron Physics, 147*, P. Hawkes, Ed. San Diego: Elsevier Academic Press, 2007, pp. 137–196.
[28] ——, "Fuzzfy transform: Application to reef growth problem," in *Fuzzy Logic in Geology*, R. B. Demicco and G. J. Klir, Eds. Amsterdam: Academic Press, 2003, pp. 275–300.