LE MATEMATICHE
Vol. LI (1996) – Fasc. I, pp. 59–76

IMPLEMENTING FUZZY POLYNOMIAL INTERPOLATION (FPI) AND FUZZY LINEAR REGRESSION (LFR)

MARIA CRISTINA FLORENO - GIOVANNI NOVELLI

This paper presents some preliminary results arising within a general framework concerning the development of software tools for fuzzy arithmetic. The program is in a preliminary stage. What has been already implemented consists of a set of routines for elementary operations, optimized functions evaluation, interpolation and regression. Some of these have been applied to real problems.
This paper describes a prototype of a library in C^{++} for polynomial interpolation of fuzzifying functions, a set of routines in FORTRAN for fuzzy linear regression and a program with graphical user interface allowing the use of such routines.

1. Introduction.

Fuzzy sets theory, as it is known, is a powerful mathematical tool for tackling real-world problems, particularly for its capability in treating uncertainty, ambiguity, complex systems, natural language and for the wide range of its applications.
As a matter of fact traditional mathematical theories consider data uncertainty and imprecision only as perturbative factors, and in such a way one can miss

Entrato in Redazione l'8 gennaio 1996.

Keywords: Fuzzy arithmetic; Fuzzy polynomial interpolation; Fuzzy linear regression.

a relevant part of the original information. Fuzzy sets theory on the contrary allows a schematization of real phenomena which globally preserves data uncertainty in the most transparent way for final users [20]. In particular, the use of fuzzy arithmetic as an extension and generalization of interval arithmetic allows to handle imprecise numerical quantities with the introduction of the concept of fuzzy number. We could then represent a fuzzy number as a set of confidence intervals, each of them providing the related numerical value at a given presumption level $\alpha \in [0, 1]$ (see [10] for a detailed definition).
In the following we are going to describe an extension of the effective and user friendly working tool for fuzzy arithmetic recently developed by our team [1] for the basic mathematical operations. In particular we implemented the fuzzy polynomial interpolation (FPI) and the fuzzy linear regression (FLR) as an extension and generalization of the respective non fuzzy theories.
We focused our attention on FPI and FLR as they could be very interesting for applications in several research fields such as:

- computer graphics;
- image analysis and processing;
- engineering design;
- statistical studies.

It is a matter of fact that a lot of interest is growing in these fields, even if these applications are not well known, due to the lack of practical tools. We hope this paper will fill this gap and extend the possible applications in this field. Moreover, in the future fuzzy arithmetic libraries could be implemented in Hw, reducing computational costs.

2. The fuzzy polynomial interpolation (FPI).

Our FPI implementation follows the rules of most fuzzy tools, that is to consider FPI as an extension of polynomial interpolation (PI): in such a way we look at its applications in a wider way.
In the last years interval extensions of PI have been utilized in CAD/CAM applications, especially in the reconstruction (by approximation) of curves and surfaces (with others of lower degree or simpler analytical form, or that require less data for their specification) [16]. The problem was to preserve detailed information about approximation error through interval analysis, because this information has a considerable weight in some applications like tolerance analysis of mechanical components.
Another example of CAD application could be found in a new technique, called

interproximation [2], recently developed, that allows to find curves that both interpolate all the crisp points and also approximate the uncertain points (passing through the regions that specify the uncertainty of these points). This technique however, during the approximation, loses information and has prohibitive computational costs due to the use of trial-error methods in the approximation process.
Also image processing, especially for what it may concern shape reconstruction from imprecise data collected by sensors, could be a practical field of application of FPI. A potentially important area of application is in the rendering of contours of natural objects (which, as a rule, do not have sharp boundaries).
In these contexts FPI is a good solution because it does not cause losses of information contents, it requires computational resources similar to those of PI, it allows a smooth passage from a certain point to an uncertain one, and it manages uncertainty with its elective tool, the fuzzy number.

2.1. Fuzzy polynomial interpolation theory.

The problem of interpolating "fuzzy values" was initially stated by L.A. Zadeh [21].
We search a fuzzyfying function $\Im : \mathbb{R} \to \Gamma(\mathbb{R})$ (where $\Gamma(\mathbb{R})$ is the set of all compact, convex and normal fuzzy sets, that is the set of fuzzy numbers on $\mathbb{R}$) which is continuous in the appropriate topology and satisfies interpolation conditions (see Lowen [11] for a rigourous definition).
We follow the formulation of Zadeh's problem made by Lowen [11]:

let now $x_0 < \ldots\ldots < x_n$ *be* $n+1$ *points in* $\mathbb{R}$ *and let* $(\mu_i)_{i=0,\ldots\ldots n}$, *be* $n+1$ *fuzzy sets in* $\Gamma(\mathbb{R})$.
Construct a function $\Im : \mathbb{R} \to \Gamma(\mathbb{R})$ *fulfilling the following properties:*

(i) *for all* $i = 0, \ldots, n : \Im(x_i) = \mu_i$;
(ii) $\Im$ *is continuous;*
(iii) *if for all* $i = 0, \ldots, n$, $\mu_i = 1_{y_i}$ *where* $y_i \in \mathbb{R}$ *and if* f *is the unique polynomial of degree* $\leq n$ *which passes through the points* $(x_i, y_i)_{i=0,\ldots,n}$, *i.e. such that* $(f(x_i) = y_i$ *for all* $i = 0, \ldots, n$, *then* $\Im$ *reduces to the function* $\Im(x) = 1_{f(x)}$ *for all* $x \in \mathbb{R}$
(and from this last definition it is clear that FPI is a generalization of PI)

Lowen then proved the existence and uniqueness of the interpolating function.
Concerning the computational methods used in FPI routines implementation we follow Kaleva's paper [9] about the representation of the interpolating polynomial in terms of α-level sets:

$$\Im^{\alpha}(x) = \{y \in \mathbb{R} \mid y = p_{do\ldots dn}(x),\ d_i \in \mu_i^{\alpha}\}$$

where $\Im^{\alpha}(x)$ is the α-level set of $\Im(x)$ and $p_{do\ldots dn}$ is the polynomial interpolating the data $(x_i, d_i)_{i=0,\ldots,n}$, and μ^{α} is the α-level set of μ.
Kaleva also showed a representation of the interpolating polynomial using Lagrange basic polynomials:

$$\Im^{\alpha}(x) = \sum_{i=0}^{n} L_i(x)\mu_i^{\alpha}\,, \quad \text{hence} \quad \Im(x) = \sum_{i=0}^{n} L_i(x)\mu_i, \tag{2.1.1}$$

where $L_i(x)$ is

$$L_i(x) = \prod_{j=0,\, j\neq i}^{n} \frac{(x - x_j)}{(x_i - x_j)}.$$

Likewise it is possible to express a generic fuzzifying spline fs using bases of real splines: let fs be an interpolating fuzzy spline of order l. If $fs^{\alpha}(x)$ is the α-level set of $fs(x)$ then

$$fs^{\alpha}(x) = \{y \in \mathbb{R} \mid y = s_{do\ldots dn}(x),\ d_i \in \mu_i^{\alpha}\}$$

where $S_{do\ldots dn} \in S_l$ (the family of splines of order l) is the spline which interpolates the data $(x_i, d_i)_{i=0,\ldots,n}$.
If $S_i \in S_l$ interpolates the data $(x_j, f_j)_{j=0,\ldots,n}$, where $f_j = 1$, if $j = i$ and zero otherwise, then

$$s_{do\ldots dn}(x) = \sum_{i=0}^{n} d_i s_i(x) \quad \text{and we have} \quad fs(x) = \sum_{i=0}^{n} s_i(x)\mu_i\,, \tag{2.1.2}$$

Although these representations are interesting for their formal elegance they are not convenient for effectively computing interpolating functions, but we utilized Kaleva's computational methods that requires less computational resources.
Therefore the methods for the Lagrange FPI(LFPI) and for fuzzy spline interpolation (FSI) computations are based on the same principle which we briefly explain in the following. Each interpolated point is computed in terms of α-level sets. Each level set is represented as a couple of end-points that are computed with a conventional interpolation of a proper vector of points. The elements of this vector are conveniently chosen between the end-points of the corresponding level sets of the points $(x_i, \mu_i)_{i=0,\ldots,n.}$. Such a choice is based on the control of the sign of basic polynomials.
With more details we may generalize the problem as follows.
Given the points $(x_i, \mu_i)_{i=0,\ldots,n}$ and a family of fuzzifying interpolants β on

(x_0, x_n), which is a linear space, such that (b_0, b_n) is the basis which corresponds to the subdivision $\Delta = \{x_0 < \ldots < x_n\}$ of $[x_0, x_n]$, the fuzzyfying interpolation for (x_i, μ_i) may be expressed as

$$\tilde{b}(x) = \sum_{i=0}^{n} b_i(x)\mu_i \, .$$

Let $x^* \in [x_0, x_n]$, then

$$\tilde{b}(x^*)^\alpha = \sum_{i=0}^{n} b_i(x^*)\mu_i^\alpha$$

with $\mu_i^\alpha = [a_{i1}^\alpha, a_{i2}^\alpha]$ an interval on $\mathbb{R}$ where a_{i1}^α is the lower point and a_{i2}^α the upper point. Hence $\tilde{b}(x^*)^\alpha = [a_1^\alpha, a_2^\alpha]$ is also an interval on $\mathbb{R}$, where a_1^α is a linear combination of conveniently chosen end points a_{i1}^α and a_{i2}^α, that is, for each $i = 1, \ldots, n$

$$a_1^\alpha \quad \text{is} \quad \begin{cases} a_{i2}^\alpha & \text{if } b_i(x^*) \geq 0 \\ a_{i2}^\alpha & \text{otherwise.} \end{cases}$$

Analogously

$$a_2^\alpha \quad \text{is} \quad \begin{cases} a_{i2}^\alpha & \text{if } b_i(x^*) < 0 \\ a_{i2}^\alpha & \text{otherwise.} \end{cases}$$

In other words each end point of $\tilde{b}(x^*)^\alpha$ (the α-level set of interpolation points) may be computed with a conventional interpolation of conveniently chosen end points of μ_i^α (the corresponding level set of fuzzy numbers μ_i). Hence the LFPI sign control is simple while for FSI's it may become more difficult because of the existence of one or more degrees of freedom for a spline of a particular subdivision.
Kaleva therefore considers a particular kind of splines, "not a knot" splines [4], whose sign may be kept under control.
In the next section we shall explain our implementation of the above outlined algorithms.

2.2. FPI implementation.

We have implemented our library as an extension of a working tool for fuzzy arithmetic [1] with reference also to an additional library for polynomial interpolation. It has been obtained with the object oriented programming language C^{++}. Thanks to the use of this development tool, the fuzzy arithmetic library provides users with elementary operations on fuzzy numbers in a simple way and hides the implementation of the data structure of the fuzzy number data

type. This allows an easier programming. Moreover, this type of implementation is sufficiently flexible to allow changes to be made without rewriting all the routines. The efficiency and the numerical precision of our library will be improved in the future with new versions of the fuzzy arithmetic and PI libraries. Therefore, concerning Fuzzy Interpolation, our tool provides Lagrange FPI, Fuzzy linear and cubic spline interpolation and Fuzzy B-splines.

2.2.1. Lagrange FPI.

We implemented two kinds of routines: the first are based on representation (2.1.1), that is on the definition of Fuzzy Lagrange interpolating polynomials as linear combination of basic Lagrange polynomials; the second are based on Kaleva's computational methods described in the preceding section (2.1.3), and just utilize, for the output, the same representation of fuzzy number already realized [1].
Therefore, concerning to the Lagrange FPI we implemented two kinds of routines:

- one for the linear combination of basic Lagrange polynomials with fuzzy data; this combination is easily obtained due to the presence of the fuzzy arithmetic elementary operators of sum and product defined in the fuzzy arithmetic library [1] with the C^{++} programming language;
- the other based on the computational methods mentioned before, that computes interpolated fuzzy data in terms of level sets; i.e. for each level set it computes the end points through conventional polynomial interpolation applied to properly selected data vectors. There is also a routine that carries out the same operations optimizing the evaluation order making use of Newton's divided difference method for the conventional interpolant polynomials computation (it allows the calculation of the level curves through precomputed divided difference values).

The coexistence of the two types of routines aims at making transparent the different amount of computational resources employed. For both two types we allowed computing interpolated fuzzy data in one or several points (not necessarily equidistant). Figure 1 provides a simple graphical example of the interpolated data obtained by our routines.
Let the input data of this example be triangular fuzzy numbers, expressed as $\mu_i = (l_i, m_i, r_i)$ where l_i and r_i are the lower and the upper end points of the support, while m_i represents the point where the membership function assumes the value 1. Suppose we have the following data (x_i, μ_i):

$$(1.0, (-2.0, 0.0, 1.0)),\ (1.5, (4.0, 5.0, 7.0)),\ (2.0, (1.0, 1.0, 4.0)),$$
$$(3.5, (0.0, 4.0, 7.0)),\ (3.9, (-3.0, 0.0, 2.0)),\ (4.0, (0.0, 1.0, 2.0)).$$

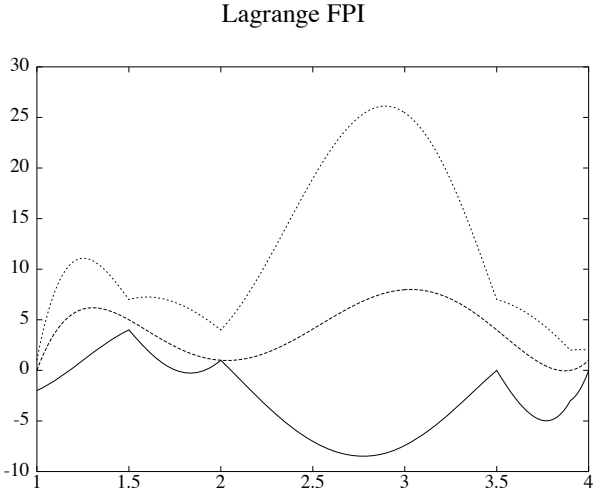

Figure 1

The central line in the figure represents the crisp interpolation (that is where the membership function assumes the value 1), while the upper and lower ones define the support of the interpolated fuzzy values (in other words the width between them gives an idea of fuzzy values uncertainty). The figure shows just two levels to have an immediate idea of Fuzzy interpolation; however it is possible to visualize the most useful levels for a specific application.

2.2.2. Linear spline interpolation.

In order to provide a better explanation about [our] fuzzy linear spline implementation, it could be useful a briefly review of conventional linear splines.
Given the data $(x_i, y_i)_{i=0,\ldots,n}$ where $K = \{x_0 < x_l < x_2 < \cdots < x_n\}$ is the knot sequence, we calculate the unique linear spline L that satisfies the interpolation conditions $L(x_j) = y_{j\ j=0,\ldots,n}$.
Then for a generic value x we may obtain the linear spline with the following steps:

- searching the value i such that $x \in [x_i, x_{i+1}]$;
- computing the affine map $A(y_i, y_{i+1}, t) = (1 - t)y_i + ty_{i+1}$ where $t = \dfrac{x - x_i}{x_{i+1} - x_i}$ $t \in [0, 1]$.

For fuzzy linear spline we use the following expression:

$$fs(x) = \frac{(x_{i+1} - x)}{(x_{i+1} - x_i)}\mu_i + \frac{(x - x_i)}{(x_{i+1} - x_i)}\,\mu_{i+1}$$

obtained by (2.1.2) with the linear splines basis functions; we can now rewrite the last expression as:

$$\widetilde{A}(\mu_i, \mu_{i+1}, t) = (1 - t)\mu_i + t\mu_{i+1}\,.$$

Therefore we implemented two routines for fuzzy linear spline interpolation, where the computation of $\widetilde{A}$ is carried out with the fuzzy arithmetic elementary operators already implemented in [1]. Moreover the computational resources involved are similar to that of conventional linear splines interpolation (grow linearly with the data dimension).

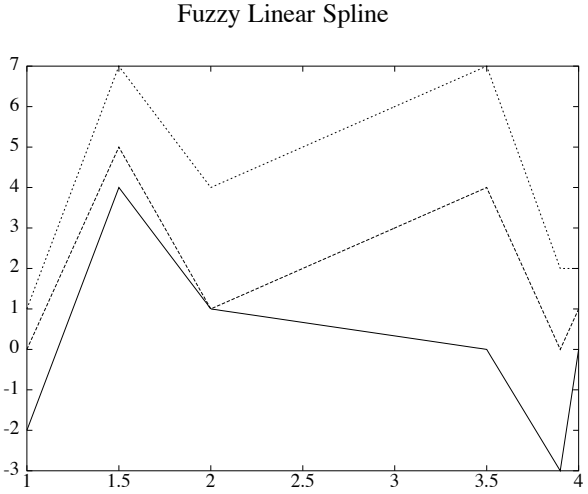

Figure 2

Figure 2 shows the output of these last routines applied to the same input data of Figure 1.

2.2.3. Cubic spline interpolation.

For what it may concern fuzzy cubic splines we carry out the computation with two different methods:

1) directly from the expression (2.1.2), as the linear combination of a basis of interpolant cubic splines with fuzzy data U_i;
2) using the Kaleva's computational method already described.

For both we use the following calculation process:
Given a segment of an interpolant cubic spline as:

$$sp_j(x) = \sum_{k=0}^{3} c_{kj}(x - x_j)^k$$

with $x \in [x_j, x_{j+1}]$ and $j = 0, \ldots, n-1$.
Then we represent the cubic splines coefficients c_{kj} $(k = 0, \ldots, 3,\ j = 0, \ldots, n-1)$ with the moments M_j $(i = 0, \ldots, n)$, which are the values of the second derivative at the knots $x_0, \ldots, x_n$, and we calculate M_j solving a tridiagonal system of $n-1$ equation; the solution process is carried out using a classical algorithm.
We compute two types of cubic splines interpolants:

$(*)$ those satisfying the natural condition;
$(**)$ those satisfying the "not and knot" condition [4] at knots x_1 and x_{n-1}.

Fuzzy Cubic Spline (Method 1)

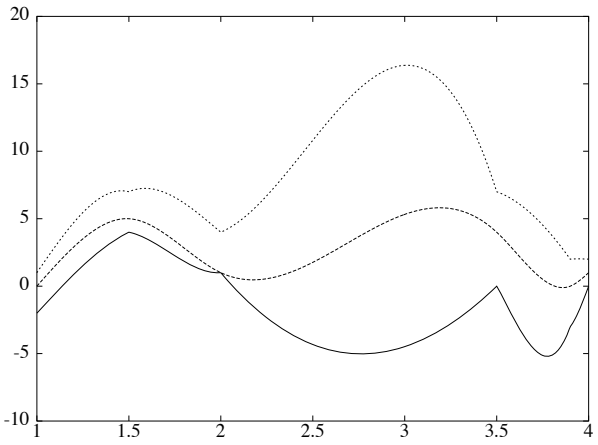

Figure 3

Method 1.

Concerning the first method of computation we use the cardinal basis of both two types of splines (*) and (**). In particular, in the routines implemented for the computation of the interpolant fuzzy value in several crisp points, the evaluation of basic cubic splines is carried out only once, with the consequent decreasing of time complexity.
An immediate visualisation of the output of these routines is provided by Figure 3 (with the same input data of Figure 1).

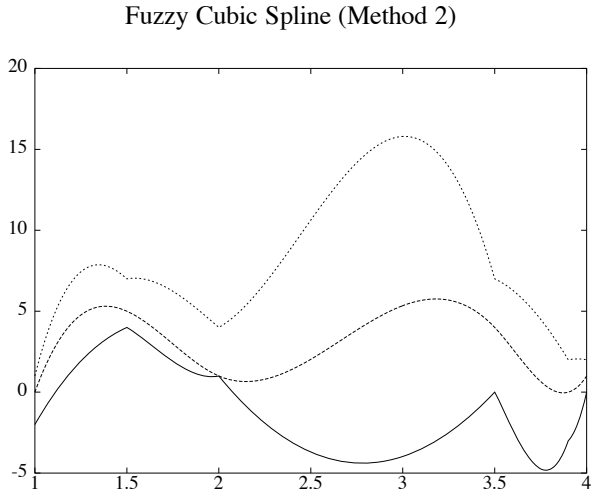

Figure 4

Method 2.

On the other hand, in the second method of computation that we implemented, we used only the basis of cubic splines satisfying the "not a knot" condition [4] at the knots x_1 and x_{n-1}.
Due to a theorem stated by Kaleva [9], it is simple to keep under control the sign of this type of splines. as we mentioned before, Kaleva's computation method is based on the use of these splines to evaluate the end-points of each α-level set of interpolant fuzzy values.
Concerning this second method of computation for fuzzy cubic splines we have implemented the following routines:

- the first computes the interpolant fuzzy value on one point and the computation process is carried out level by level; the end-points of each α-level set are evaluated with conventional cubic splines, properly interpolating selected vectors of data and satisfying the "not a knot" condition.
- the second computes interpolant fuzzy values on several points using the same computational method of the previous routine; moreover it optimizes the evaluation order by avoiding repeated evaluation of same values. Figure 4 then provides an easy approach to the output of these two last routines (with the same input data of Figure 1).

3. The fuzzy linear regression (FLR).

Ordinary regression (OR) is widely used in almost every field of science, such as economics, engineering and natural sciences. However, there are circumstances for which OR is not useful as it needs strict statistical assumptions, and it assumes that the data and the relationships between them are crisp; Fuzzy Regression (FR) methods have been proposed, in recent years, to be used in such circumstances.
In other words, while in OR the deviation between the observed values and the estimated values is taken to be a measurement error, in FR this gap is attributed to the vagueness and ambiguity of the modelled system. As a matter of fact in a general FR model the relationship between input and output data is expressed by a fuzzyfing function and a possibilistic distribution of the data is assumed.
In this section we will briefly recall basic facts about Fuzzy Linear Regression (FLR) theory and methods and explain our implementation.

3.1. Fuzzy linear regression theory.

FLR was originally introduced by Tanaka et al. [19] in 1982 and up to now several approaches [7], [17], [18], [15], [14], [13], to this theory have been formulated.
In FLR analysis we express the relationship between a dependent variable Y and the independent variables $x_1, \ldots .x_n$, with a linear fuzzy function:

$$Y = A_0 + A_1 x_1 + \ldots\ldots + A_n x_n = Ax$$

where x is the crisp vector of the independent variable and A is the fuzzy vector of the coefficients. But fuzzy coefficients A are fuzzy sets, so we can define them by membership functions. In literature these membership functions are expressed according to the definition of L-R fuzzy numbers given by Dubois and Prade [5].

Here we refer to the definition of fuzzy coefficients as symmetric L-R fuzzy numbers:
a fuzzy coefficient A_j is expressed as $A_j = (\alpha_j, c_j)$, where α_j is the centre and c_j the width of the fuzzy number A_j, and its membership function is defined as

$$\mu_A(x) = L((x - \alpha)/c) \qquad c > 0$$

where $L(x)$ is the reference function such that

$$\begin{array}{lll} 1) & L(x) = L(-x) & \\ 2) & L(0) = 1 & \\ 3) & L(x_1) > L(x_2) & \forall x_1, x_2 \in [0, \infty) : \ x_1 < x_2 \,. \end{array} \tag{3.1.1}$$

According to this formulation of FLR, Tanaka et al. [19] proved that the membership function for the output of a linear possibility system is

$$\mu_Y(y) = L((y - x^t\alpha)/c^t|x|)$$

where $|x| = (|x_1|, \ldots, |x_n|)^t$. In other words, for the possibility distribution Y the centre is $\sum \alpha_j x_j$ and the width is $\sum c_j|x_j|$. Since all coefficients are fuzzy sets also Y is a fuzzy set, and we can express it as a symmetric L-R fuzzy number $Y_i = (y_i, e_i)$, where y_i is the centre and e_i is the width.
In order to express the degree h to which the given data Y_i is included in an inferred (predicted) one Y_i^*, Tanaka et al. [17] employed h-level sets. Therefore if Y_i^h and Y_i^{*h} are h-level sets of Y_i and Y_i^*, the degree of fit of Y_i to Y_i^* is given by the maximum H, said $\bar{h}_i$, such that $Y_i^H \subset Y_i^{*H}$.
Then the degree of fitting of the fuzzy linear model to the data is defined by $H = \min \bar{h}_i$. (see Tanaka [19] for a rigourous definition).
According to the formulation stated above, in the last years several methods for the solution of a FLR model have been proposed. we will review some of these methods that we implemented in our applications, based on the following assumptions:
given the data (Y_i, x_i) the basic idea is to find two models expressed by the fuzzyfing linear functions $\widetilde{f}$ and $\bar{f}$ such that $\widetilde{f}(x_i) \underset{h}{\subset} Y_i \underset{h}{\subset} \bar{f}(x_i)$. The fuzziness of these functions is estimated by coefficients $\widetilde{A}$ and $\overline{A}$ through a functional $J : \mathbb{R}^{n+1} \to \mathbb{R}$ to be specified later. Therefore it is possible to find the parameters of the two models solving the following two problems:

a) Problem of minimum: $\min_{\overline{A}_j} J(\bar{c}) \qquad \bar{f}(x_i) \underset{h}{\supset} Y_i$

b) Problem of maximum: $\max_{\tilde{A}_j} J(\bar{c}) \quad \tilde{f}(x_i) \underset{h}{\subset} Y_i$.

Tanaka proves the equivalence of these two problems with two linear programming ones. In the following we will briefly show just three FLR methods with a different definition of the functional J.

1) A first method, the original fuzzy linear regression method, introduced by Tanaka et al. [19] defines J as the sum of the fuzzy widths, that is: $J(c) = \sum_{i=0}^{n} c_i$.
2) Afterwards Tanaka [17] proposed a second method where he defines J as a weighted sum of fuzzy widths, that is: $J(c) = \sum_{i=1}^{N} c^t |x_i|$.
3) Savic and Pedrycz [15] introduced then another formulation of fuzzy linear regression referring to their approach as "Fuzzy least squares linear regression", based on two phases:
   a) fitting the vector α with a crisp regression line by using the available information about the centre points of input data.
   b) using the fittest vector α^* to carry out method 2.

3.2. FLR implementation.

Our implementation of the methods above briefly explained has been carried out with the programming language FORTRAN. For each method we have implemented three routines: one to solve the minimization problem associated to fuzzy linear regression with standard data, the other two for the minimization and maximisation problems associated respectively to the FLR minimum and maximum problems with fuzzy data.
The sample data set (Y_i, x_i), $i = 1, \ldots, m$, is processed by our routines as input parameter under matrix form.
In order to represent the fuzzy parameters, which gives the solution of the model, we utilize two vectors of real numbers which contain, as in the theory above explained, respectively, the centres (α) and the widths (c) of fuzzy parameters. The solution is found carrying out three steps:

1) by translating the initial fuzzy linear regression problem into a linear programming one;
2) by solving the latter making use of the revised simplex method;
3) by translating the optimal feasible vector into the two vectors mentioned above.

As we have seen in the previous paragraph, the solution to the minimum problem is granted (which does not happen for the solution of the maximum

problem); for these reasons our routines generate a flag that points out the existence of the solution to the user. Moreover the output provides the value assumed by the functional J at the fuzzy parameters which have been found: in such a way we make use of J as a measure of the parameter fuzziness, that is a measure of the model uncertainty.
It is worth value to underline that our routines run with all symmetric L-R fuzzy numbers representations because it is not necessary to specify the reference function $L(x)$ used in the definition (3.1.1) in the previous paragraph, but only the value of its inverse function at the selected degree of fit H, that is $L^{-1}(H)$.
Concerning the implementation of the third method our routines implement the two phases described at the end of the 3.1 paragraph:

a) calling a routine for the solution of linear systems with least-squares method;
b) finding the solution of a linear programming problem in which the α vector is assumed to be constant.

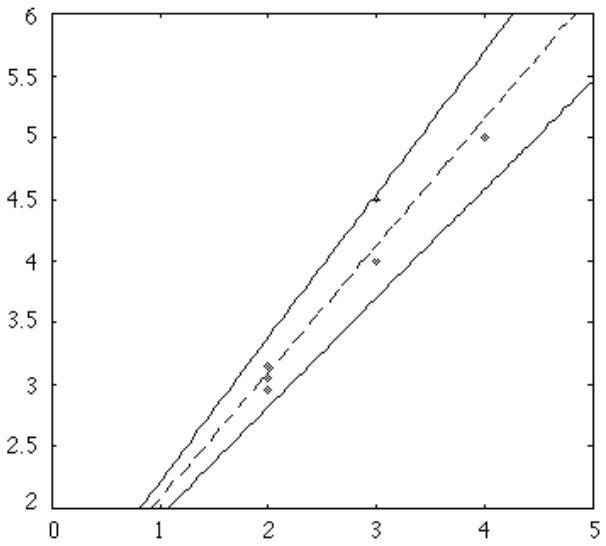

Figure 5

Figure 5 shows the fuzzy line obtained applying the third method for fuzzy

linear regression to the following data:

$$\begin{array}{l} y1 = 2.95 \;\; x1 = 1.00 \;\; x2 = 2.00 \\ y1 = 3.05 \;\; x1 = 1.00 \;\; x2 = 2.00 \\ y1 = 3.15 \;\; x1 = 1.00 \;\; x2 = 2.00 \\ y1 = 4.00 \;\; x1 = 1.00 \;\; x2 = 3.00 \\ y1 = 4.50 \;\; x1 = 1.00 \;\; x2 = 3.00 \\ y1 = 5.00 \;\; x1 = 1.00 \;\; x2 = 4.00 \end{array}$$

The dashed line represents the centre of the fuzzy line while the two solid lines respectively the lower and the upper end lines of its support.

4. Goodies and blames.

It could be said that, as in the real case, fuzzy cubic splines are more useful than the other forms of fuzzy polynomial interpolants. This is due to the fact that lower-degree polynomials do not provide a large flexibility in the shape controlling of the fuzzy curve, and higher-degree polynomials can introduce undesidered wiggles in level curves, causing a dilation of the support of interpolated data; this last characteristic is well shown not only in the figures of the previous paragraphes, but with more details in the Figure 6. Moreover fuzzy cubic splines require acceptable computational resources in comparison with higher-degree polynomials ones. In particular we observed that Fuzzy cubic spline Method 2, compared to Method 1, allows to decrease the computational costs.
However the computational burdain, which is definitely large that in the crisp case, could be alleviated by using specialized chips (e.g. Interval analysis chips [6]; see also the chips being developed at the University of Karlsruhe). Furthermore one could also enhance the computation speed by using parallel architectures [3].
Concerning the Regression, although FLR is more effective that OR, especially when we have an ambiguous or vague system, we would also outline the main possible weakness [14] of these methods:

- scale dependence and coefficient crispness [13];
- effect of outliers;
- redundance of constraints;
- infinite solutions;
- sensitivity of fuzzy width to the degree of fit, H, of the estimated fuzzy linear model to the entire data set [5].

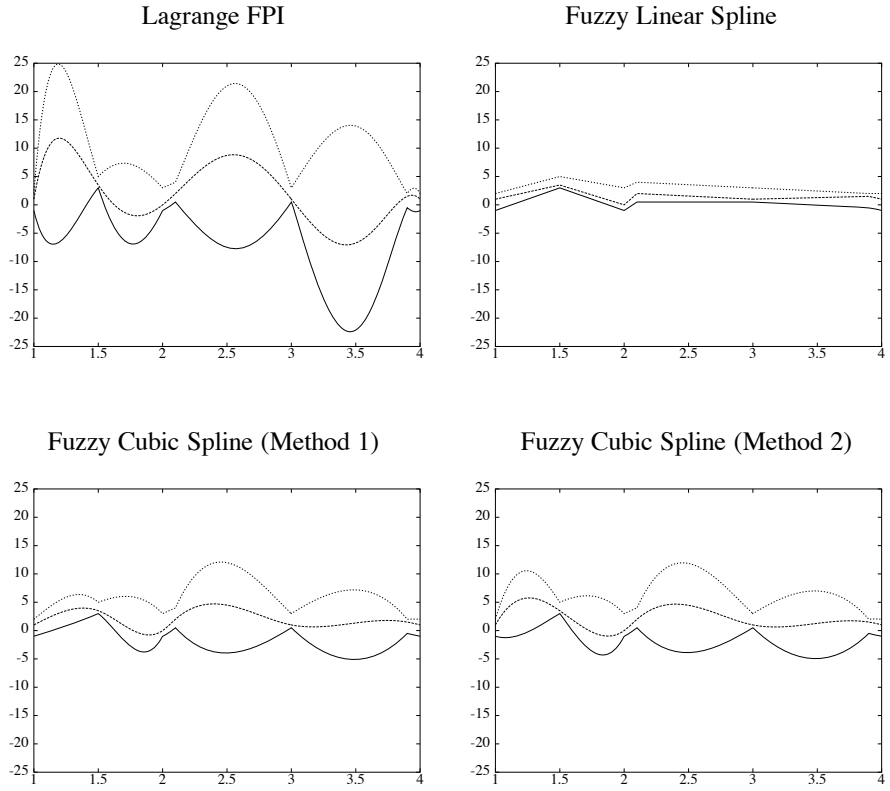

Figure 6

Figure 6 shows examples of FPI, with the same scale, on the following input data: (x_i, μ_i):

(1.0, (−1.0, 1.0, 2.0)), (1.5, (3.0, 3.5, 5.0)), (2.0, (−1.0, 0.0, 3.0)),
(2.1, (0.5, 2.0, 4.0)), (3.0, (0.5, 1.0, 3.0)), (3.9, (−0.5, 1.5, 2.0)),
(4.0, (−1.0, 1.0, 2.0)),

where (x_i, μ_i) are triangular fuzzy numbers, with $\mu_i = (l_i, m_i, r_i)$ and l_i and r_i are the lower and the upper end points of the support, while m_i represents the point where the membership function assumes the value 1.

The first problem consists in the sensitivity of independent variables to the units

of measure: changing the latter will change the possibility distribution of the model parameters and some of these may become crisp; this problem arises in the first method but is solved with what we call Method 2.
The effect of outlier, common to many ordinary regression methods, has recently found a partial solution by Peters [13], who make use of the concept of fuzzy interval.
In the end, about the question of infinite solutions, it should be pointed that the third method is the only, among the three we implemented, which offers a well defined solution.
We utilize the language C^{++} for interpolation because the software is linked to a preexisting one [1], and Fortran for regression in order to use the more robust simplex method IMSL routines.

5. Conclusions.

Finally we would underline that the whole tool of Fuzzy arithmetic, especially with the FPI and the FLR extensions, may offer a practical, simple and efficient way to manage imprecise informations for a wide range of applications. Within this framework it is possible to treat the very promising field of interproximation which will be the subject of future researches. Moreover a planned-friendly graphical user interface, an extension of FPI with other techniques, and a further improvement of the FLR usefulness, with a large number of practical examples will offer this tool as a complete and available alternative way to handle uncertainty.
Actually, all the routines we presented in this paper are available at the Mathematics Department of the University of Catania. More informations will be available writing to the following e-mail: floreno@dipmat.unict.it and novelli@agata.dipmat.unict.it.

REFERENCES

[1] A.M. Anile - S. Deodato - G. Privitera, *Implementing fuzzy arithmetic,* Fuzzy sets and Systems, 72 (1995), pp. 239-250.

[2] F. Cheng - B. Barsky, *Imterproximation using Cubic B-Spline Curves,* Computer Aided Design, pp. 359-373.

[3] A. Clematis - G. Privitera - M. Spagnuolo, *Calcolo di funzioni Fuzzy su architetture parallele,* Rapporto Tecnico Istituto per la Matematica Applicata n. 7196,

Genova, giugno 1996.

[4] C. De Boor, *A practical guide to splines,* Springer-Verlag, New York, 1978).

[5] D. Dubois - H. Prade, *Fuzzy Sets and Systems, Theory and Applications,* Academic Press, Cambridge Mass., 1980, pp. 53-57.

[6] E.R. Hansen, *Global Optimization using Interval Analysis,* Marcel Dekker, New York, 1992.

[7] B. Heshmaty - A. Kandel, *Fuzzy linear regression and its applications to forecasting in uncertain events,* Fuzzy Sets and Systems, 15 (1985), pp. 159-191.

[8] S. József, *On the effect of linear data transformations in possibilistic fuzzy linear regression,* Fuzzy Sets and Systems, 45 (1992), pp. 185-188.

[9] O. Kaleva, *Interpolation of fuzzy data,* Fuzzy Sets and System, 61 (1994), pp. 63-70.

[10] Kaufmann - M.M. Gupta, *Introduction to Fuzzy Arithmetic: Theory and Applications,* Van Nostrand Reinohold, New York, 1991.

[11] R. Lowen, *A fuzzy Lagrange interpolation theorem,* Fuzzy Sets and System, 34 (1990), pp. 33-38.

[12] H. Moskowitz - K. Kim, *On assessing the H value in fuzzy linear regression,* Fuzzy Sets and Systems, 58 (1993), pp. 303-327.

[13] G. Peters, *Fuzzy linear regression with fuzzy intervals,* Fuzzy Sets and Systems, 63 (1994), pp. 45-55.

[14] D.T. Redden - W. Woodall, *Properties of certain fuzzy linear regression methods,* Fuzzy Sets and Systems, 64 (1994), pp. 361-375.

[15] D. Savic - W. Pedrycz, *Evaluation of fuzzy linear regression models,* Fuzzy Sets and Systems, 39 (1991), pp. 145-160.

[16] T.W. Sederberg - R.T. Farouki, *Approximation by Interval Bezier Curves,* IEEE CG & A, September 1992, pp. 87-95.

[17] H. Tanaka, *Fuzzy data analysis by possibilistic linear models,* Fuzzy Sets and Systems, 24 (1987), pp. 363-375.

[18] H. Tanaka - H. Ishibuchi, *Identification of possibilistic linear systems by quadratic membership function of fuzzy parameters,* Fuzzy Sets and Systems, 41 (1991), pp. 145-160.

[19] H. Tanaka - S. Uegima - K. Asai, *Linear regression analysis with fuzzy model,* IEEE Trans. Systems Man Cybern., 12 (1982), pp. 903-907.

[20] L.A. Zadeh, *Fuzzy logic neural networks and soft computing,* Comm. ACM, 37 (1994).

[21] L.A. Zadeh, *Fuzzy sets,* Inform. and Control, 8 (1965), pp. 338-353.

CoRiMME,
Stradale Primosole 50,
I-95100 Catania (ITALY)