ALGORITHMS TO EXTEND CRISP FUNCTIONS AND THEIR INVERSE FUNCTIONS TO FUZZY NUMBERS

O.G. Duarte
Department of Electrical Engineering
Universidad Nacional de Colombia
Ciudad Universitaria Ed 453 Of 202
Bogotá, Colombia
ogduarte@ingenieria.ingsala.unal.edu.co

M. Delgado
Department of Computer Sciences
and A.I. - E.T.T.S.I. Informática,
Universidad de Granada,
Avda de Andalucía 38
C.P. 18071 Granada, Spain
mdelgado@decsai.ugr.es

I.Requena
Department of Computer Sciences
and A.I. - E.T.T.S.I. Informática,
Universidad de Granada,
Avda de Andalucía 38
C.P. 18071 Granada, Spain
irequena@decsai.ugr.es

Abstract

In this paper we show an algorithm to extend a kind ofcontinuous crisp functions to fuzzy numbers using the extension principle. Then we present two different solutions to the problem of extending inverse functions which we have called possible extension and necessary extension. Finally, using these solutions we have generated a family of intermediate extensions that let us define a parameter to measure the existence of the extended inverse functions.

Keywords: Fuzzy numbers, fuzzy relations, possible extension, necessary extension.

1 INTRODUCTION

Although fuzzy numbers can successfully represent the uncertainty of a numerical variable in a mathematical model, there are some difficulties representing the functions that relate these variables, mainly because with the most usual definition (used in this paper and shown later), the fuzzy numbers do not have a group algebraic structure.

Another difficulty is that mathematical models are generally designed to deal with non- fuzzy (crisp) numerical variables and therefore the functions in the models operate on crisp numbers. If we attempt to use fuzzy numbers to represent numerical variables, it is necessary to extend the crisp functions included in the model to fuzzy numbers. The usual way to extend a function is to use the Extension Principle postulated by Zadeh [4]; although this is generally an effective and useful strategy, in some cases it is not the best way to deal with inverse functions, because there are special difficulties associated with inverse functions; this has been studied in depth by Yager [3]. Bouchon- Meunier et al [1] have demonstrated that if a crisp function is invertible then fuzzy numbers are the only fuzzy sets that maintain the invertibility of the function.

Furthermore, as Giachetti [2] notes, when dealing with functions of more than one argument we can have diverse definitions of “invertibility” according to the application of the inverse function; we explain it in section 4.

The main aim of this paper is to present some algorithms that allow functions which relate crisp numerical variables in a mathematical model to be used when these variables are represented by fuzzy numbers, without the need to modify the original crisp functions. The algorithms related to inverse functions ensure the existence of a solution, but sometimes it is necessary to modify the problem conditions. furthermore, we develop a measure of the existence of the extended inverse function. The functions included in this paper are multiple argument functions, monotonically increasing in some of the arguments and monotonically decreasing in the others.

2 BASIC DEFINITIONS AND NOTATION

2.1 DEFINITION 1: FUZZY NUMBERS

We use in our algorithms the following representation of a Fuzzy Number: Let A be a number with membership function $A(x)$; the α-cuts of A are:

$$A_\alpha = \{x|A(x) > \alpha\} = [L_A(\alpha), R_A(\alpha)]\alpha \in (0,1] \quad (1)$$

A is a fuzzy number if and only if $L_A(\alpha)$ and $R_A(\alpha)$ satisfy that $L_A(\alpha)$ must be monotonically increasing and right continuous, $R_A(\alpha)$ must be monotonically decreasing and left continuous, and $L_A(1) \leq R_A(1)$

A may also be represented using the function $D_A(\alpha, d) : (0,1] \times \{-1, 1\} \to R$ as follows:

$$D_A(\alpha, d) = \begin{cases} L_A(\alpha) & \text{if } d = 1 \\ R_A(\alpha) & \text{if } d = -1 \end{cases}$$

The α-cuts of A may now be written:

$$A_\alpha = [D_A(\alpha, 1), D_A(\alpha, -1)]$$

Note that the representation of a fuzzy number using the function $D_A(\alpha, d)$ is conceptually the same as if we use α-cuts. In this paper we will use the function $D_A(\alpha, d)$ so that the algorithms are more compact.

In our algorithms we will suppose that we want to know a discrete number of α-cuts, associated with an ordered set $Alfa = \{\alpha_1, \alpha_2, ..., \alpha_p\}$ where $\alpha_i < \alpha_{i+1}$ and $\alpha_1 \geq 0$.

2.2 DEFINITION 2

In the rest of the paper we assume that $y = f(X) : U_1 \times U_2 \times \ldots \times U_n \rightarrow V$ is a continuous function, monotonically increasing with some of the n variables and monotonically decreasing with the others ($U_i, V \subseteq R$). We also assume that $x_k = f_k^{-1}(y, X_k) \quad X_k = \{x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n\}$ is the function that calculates the value of x_k when we know the output y and the other input variables x_i; furthermore, we assume that $f(X)$ and $f_k^{-1}(y, X_k)$ are well defined in all the possible combinations of their respective arguments. We define the function $d_f(i) : I \rightarrow \{1, -1\}$ (I is the index set) as follows:

$$d_f(i) = \begin{cases} 1 & \text{if } f(X) \text{ increases with } x_i \\ \text{-1} & \text{if } f(X) \text{ decreases with } x_i \end{cases}$$

2.3 DEFINITION 3

Let $A_1, A_2, \ldots, A_n$ be n fuzzy numbers defined on $U_1, U_2, \ldots U_n$ respectively using the functions $D_{Ai}(\alpha, d)$ (definition 1) with A_i representing the value of the argument x_i of $f(X)$.We define

$$\begin{aligned} D_X(\alpha, d_f) &= \{D_{Ai}(\alpha, d_f(i)); i = 1, \ldots, n\} \quad (2) \\ D_X(\alpha, -d_f) &= \{D_{Ai}(\alpha, -d_f(i)); i = 1, \ldots, n\} \end{aligned}$$

Each of the elements $D_{Ai}(\alpha, d_f(i))$ and $D_{Ai}(\alpha, -d_f(i))$ belong to the α-cut of A_i. Furthermore, they are the lowest or highest value of the α-cut of A_i depending on the increasing-decreasing condition of the variable x_i. So, $D_X(\alpha, d_f)$ is the set of values of X belonging to the respective α-cuts of $X_1, X_2, \ldots, X_n$ that gives y the lowest value. Moreover, $D_X(\alpha, -d_f)$ is the set of values of X belonging to the respective α-cuts of$X_1, X_2, \ldots, X_n$ that gives y the highest value. Note that $D_{Ai}(\alpha, d_f(i))$ is an increasing function of α, whereas $D_{Ai}(\alpha, -d_f(i))$ is a decreasing one. We also define

$$D_X(\alpha, d_f(k)) = \{D_{Ai}(\alpha, d_f(k)d_f(i)); i = 1, \ldots, n; i \neq k\}$$

$$D_X(\alpha, -d_f(k)) = \{D_{Ai}(\alpha, -d_f(k)d_f(i)); i = 1, \ldots, n; i \neq k\}$$

3 EXTENDING CRISP FUNCTIONS TO FUZZY NUMBERS

Let $y = f(X)$ be a crisp function with the characteristics indicated in definition 2. Suppose we want to extend it to fuzzy numbers; that is, we want to find a function that allows us to calculate y (or more exactly, a fuzzy number representing y) when we represent the n arguments of $f(X)$ using the fuzzy numbers $A_1, A_2, \ldots, A_n$. Due to the continuity of $f(X)$, we can extend the function $y = f(X)$ to fuzzy numbers using the Extension Principle with the α-cuts as follows:

$$y_\alpha = f(A_{1\alpha}, A_{2\alpha}, \ldots, A_{n\alpha})$$

The above expression implies that an α-cut of y is the set of all the values obtained when all the possible values of x_i belonging to the respective α- cuts of the input variables are used as arguments of $f(X)$. As $f(X)$ is monotonically increasing or decreasing with all its arguments, and since the α- cuts of the input variables are closed intervals, then y_α is also a closed interval whose extremes can be calculated using (3), according to definition 3.

$$y_\alpha = [f(D_X(\alpha, d_f)), f(D_X(\alpha, -d_f))] \qquad (3)$$

As stated in definition 3, $L_y(\alpha)$ is increasing and $R_y(\alpha)$ is decreasing; furthermore $L_{y1} \leq R_{y1}$. Due to the continuity of $f(x)$ then the conditions in (1) are satisfied, and y_α represents a fuzzy number. If we have a discrete representation as proposed in 2.1, then the algorithm 1 is used to extend $y = f(X)$ to fuzzy numbers.

Algorithm 1

1	For α_j, $j = 1, 2, \ldots, p$ calculate y_{α_j} using (3)

4 EXTENDING CRISP INVERSE FUNCTIONS TO FUZZY NUMBERS

Taking the same conditions as in section 3, suppose that we now want to extend the crisp inverse function $x_k = f_k^{-1}(y, X_k)$, with $X_k = \{x_1, x_2, \ldots, x_{k-1}, x_{k+1}, \ldots, x_n\}$ which calculates the value of the input variable x_k when we know the output y and the other input variables x_i. The solution to this problem may have at least two different approaches when each variable is represented by a fuzzy number:

- We want to know which values the variable x_k could take if we know the values of y, X_k . In this paper, this approach will be called the *possible extension* of the inverse function, and will be denoted by x_k^{pos}.

- We want to know which values the variable x_k <u>must take</u> in order to ensure that the output variable y takes certain predetermined values, when we know the values of the other input variables x_K. In this paper, this approach will be called the *necessary extension* of the inverse function, and will be denoted by x_k^{nec}.

4.1 POSSIBLE EXTENSION

The procedure used to obtain the possible extension consists in considering the crisp inverse function $x_k = f_k^{-1}(y, X_k)$ as a direct function, and extending it using the procedure described in section 3. If we take into account how x_k varies with regard to the other variables we can write an equation (4) to calculate $x_k^{pos}(\alpha)$.

$$\begin{aligned} x_k^{pos}(\alpha) &= [Lx_k^{pos}(\alpha), Rx_k^{pos}(\alpha)] \\ Lx_k^{pos}(\alpha) &= f_k^{-1}(D_y(\alpha, d_f(k)), D_{Xk}(\alpha, -d_{fk})) \\ Rx_k^{pos}(\alpha) &= f_k^{-1}(D_y(\alpha, -d_f(k)), D_{Xk}(\alpha, d_{fk})) \end{aligned} \quad (4)$$

On account of having extended the crisp function f_k^{-1} to fuzzy numbers using the procedure of section 3, $x_k^{pos}(\alpha)$ represents a fuzzy number; $D_y(\alpha, d_f(k))$, $D_{Xk}(\alpha, d_{fk})$ and $D_{Xk}(\alpha, -d_{fk})$ allow us to write in shorthand if x_k decreases or increases when the other variables increase.

If we have a discrete representation as proposed in 2.1, then the algorithm 2 is used to make a possible extension of $x_k = f_k^{-1}(y, X_k)$ to fuzzy numbers:

Algorithm 2

1	For α_j, $j = 1, 2, \ldots, p$ calculate $x_k^{pos}(\alpha_j)$ using (4)

4.2 NECESSARY EXTENSION

The procedure used to obtain the necessary extension consists in calculating the extremes of every α-cut of x_k using $x_k = f_k^{-1}(y, X_k)$ with the most restrictive extremes of the α-cuts of y and X_k. The expression used to calculate $x_k^{nec}(\alpha)$ is (5).

$$\begin{aligned} x_k^{nec}(\alpha) &= [Lx_k^{nec}(\alpha), Rx_k^{nec}(\alpha)] \\ Lx_k^{nec}(\alpha) &= f_k^{-1}(D_y(\alpha, d_f(k)), D_{Xk}(\alpha, d_{fk})) \\ Rx_k^{nec}(\alpha) &= f_k^{-1}(D_y(\alpha, -d_f(k)), D_{Xk}(\alpha, -d_{fk})) \end{aligned} \quad (5)$$

However, we cannot ensure that $Lx_k^{nec}(\alpha)$ will always be monotonically increasing, nor that $Rx_k^{nec}(\alpha)$ will always be monotonically decreasing, nor that $Lx_k^{nec}(\alpha) \le Rx_k^{nec}(\alpha)$ for every α, and as a result, (5) will not generally represent a fuzzy number. Only for certain fuzzy numbers Y will we have a fuzzy number represented by $x_k^{nec}(\alpha)$ in (5). The reason for this is explained in the following example:

Algorithm 3

1	For $\alpha_p = 1$ - calculate $Lx_k^{nec}(\alpha_p)$, $Rx_k^{nec}(\alpha_p)$ using (5) - if $Lx_k^{nec}(\alpha_p) > Rx_k^{nec}(\alpha_p)$ then widen the α-cut $Y\alpha_p = [L_y(\alpha_p), R_y(\alpha)_p)]$ in the left and right sides until we get a new α-cut $Y'\alpha_p = [L'_y(\alpha_p), R'_y(\alpha)_p)]$ such that $Lx_k^{nec}(\alpha_p) \le Rx_k^{nec}(\alpha_p)$
2	For α_j, $j = p-1, p-2, \ldots, 1$ - if $L_y(\alpha_j) > L'_y(\alpha_{j+1})$ let $L_y(\alpha_j) = L'_y(\alpha_{j+1})$ - if $R_y(\alpha_j) < R'_y(\alpha_{j+1})$ let $R_y(\alpha_j) = R'_y(\alpha_{j+1})$
3	For α_j, $j = p-1, p-2, \ldots, 1$ - calculate $Lx_k^{nec}(\alpha_j)$, using (5) - if $Lx_k^{nec}(\alpha_j) > Lx_k^{nec}(\alpha_{j+1})$ then calculate y^*_α as follows: $y^*_\alpha = f(D_X(\alpha^*_j, d_f))$ $D_X(\alpha^*_j, d_f) = \{D_X(\alpha^*_{ji}, d_f(i)d_f(k) \mid i = 1, 2, \ldots, n\}$ $\alpha^*_{ji} = \begin{cases} \alpha_j & \text{if } i \ne k \\ \alpha_{j+1} & \text{if } i = k \end{cases}$ - if $d_f(k) = 1$ then widen the α-cut $Y\alpha_j = [L_y(\alpha_j), R_y(\alpha)_j)]$ in the left side transforming it into the new α-cut $Y'\alpha_j = [y^*_\alpha, R_y(\alpha)_j)]$ - if $d_f(k) = -1$ then widen the α-cut $Y\alpha_j = [L_y(\alpha_j), R_y(\alpha)_j)]$ in the right side transforming it into the new α-cut $Y'\alpha_j = [L_y(\alpha_j), y^*_\alpha]$
4	For α_j, $j = p-1, p-2, \ldots, 1$ - calculate $Rx_k^{nec}(\alpha_j)$, using (5) - if $Rx_k^{nec}(\alpha_j) < Lx_k^{nec}(\alpha_{j+1})$ then calculate y^*_α as follows: $y^*_\alpha = f(D_X(\alpha^*_j, -d_f))$ $D_X(\alpha^*_j, -d_f) = \{D_X(\alpha^*_{ji}, -d_f(i)d_f(k) \mid i = 1, 2, \ldots, n\}$ - if $d_f(k) = -1$ then widen the α-cut $Y\alpha_j = [L_y(\alpha_j), R_y(\alpha)_j)]$ in the left side transforming it into the new α-cut $Y'\alpha_j = [y^*_\alpha, R_y(\alpha)_p)]$ - if $d_f(k) = 1$ then widen the α-cut $Y\alpha_j = [L_y(\alpha_j), R_y(\alpha)_j)]$ in the right side transforming it into the new α-cut $Y'\alpha_j = [L_y(\alpha_j), y^*_\alpha]$

Let us suppose that the fuzzy numbers $A_1, \ldots A_{k-1}, A_{k+1}, \ldots A_n$ represent the numerical values of the variables $x_1, \ldots, x_n$ and that each has a trapezoidal shape, and also that the fuzzy number Y represents the output variable y and has a singleton shape. We want to obtain $x_k^{nec}(\alpha)$: in other words, we want to obtain a fuzzy number A_k (whose α-cuts are $x_k^{nec}(\alpha)$) so that it ensures that the output will be a singleton, when the other input variables are trapezoidal ones. Obviously such a fuzzy number cannot be found, because the uncertainty included in the trapezoidal numbers cannot "just disappear".

To solve this problem, we propose algorithm 3, that verifies whether the fuzzy number Y is a coherent value with the uncertainties of the input variables X_k. If it is not, then the algorithm modifies the fuzzy number Y adding uncertainty (by widening its α-cuts) in a convenient way, so that we can ensure that Y and $x_k^{nec}(\alpha)$ will be fuzzy numbers. (Steps 2, 3 and 4 may be followed in the same iteration cycle)

4.3 FAMILY OF INTERMEDIATE EXTENSIONS

By comparing equations (4) and (5), it can be observed that the main difference between the possible and necessary extensions lies in the selection of the α-cut extreme of every variable in X_k used to calculate $x_k^{pos}(\alpha)$ or $x_k^{nec}(\alpha)$. This allows us to define intermediate extensions using intermediate values of the α-cuts, by using a parameter $r \quad (0 \leq r \leq 1)$. The possible extension will be associated with a value of $r = 0$ and the necessary solution with $r = 1$. The intermediate extensions is called $x_k^{int}(\alpha, r)$ and is calculated using a linear interpolation of the values used in (4) and (5) inside the algorithm 3. It is important to emphasize that possible extension always exists but that necessary extension does not. When necessary extension does not exist, there must be a threshold value of r, for example r_o, so that if $r < r_o$ then extension exists; r_o can then be used to measure the existence of a solution to the problem of extending the inverse function.

5 EXAMPLE

In the following example we will consider the crisp function $y = f(x_1, x_2) = \frac{x_1^2}{x_2}$ defined for strictly positive real numbers x_1,x_2,y, and their inverse function $x_1 = f_1^{-1}(y, x_2) = \sqrt{y x_2}$ If y, x_2 are represented by trapezoidal fuzzy numbers $y = T(1, 1, 1, 1)$, $x_2 = T(0.5, 0.9, 1.1, 1.5)$ then the values of x_1 can be obtained using the family of intermediate extensions of $f_1^{-1}(y, x_2)$. The result is shown in figure 1, but y has been modified as is shown in figure 2. Note in this example how the uncertainty of x_1,x_2, changes when r changes. Note also that in this example the threshold of existence of an extension is $r_o = 0.5$, because if r takes a value larger than 0.5 the algorithm has to modify y in order to obtain a solution.

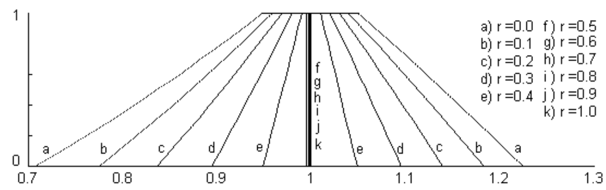

Figure 1: x_2 in example 1

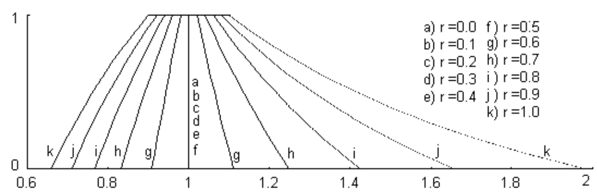

Figure 2: y in example 1

6 CONCLUSIONS AND REMARKS

In this paper we have presented some algorithms to extend crisp functions and their inverse functions to fuzzy numbers. The type of functions dealt with here are continuous functions monotonically increasing with some of their arguments, and monotonically decreasing with the others. We have also proposed some different solutions to the problem of extending crisp inverse functions: first, we proposed the possible and the necessary extensions, and then we proposed a family of intermediate extensions that varies slowly between the first two. Although some of the extensions may not exist, the algorithms proposed ensure that a solution is found, by conveniently modifying the condition of the problem.

References

[1] B. Bouchon-Meunier, O.Kosheleva, V. Kreinovich and H.T. Nguyen , Fuzzy Numbers are the only fuzzy sets that keep invertible operations invertible,*Fuzzy Sets and Systems* (1997) 155-164

[2] R. Giachetti, Evaluating Engineering functions with imprecise quantities *7th IFSA World Congress Prague* (1997) vol II 150-155

[3] R.R. Yager, On the lack of inverses in fuzzy arithmetic, *Fuzzy Sets ans Systems* (1980) 73- 82

[4] L.A. Zadeh, The concept of linguistic variable and its applications to approximate reasoning, Part I, *Inform. Sci.* 8 (1975) 199-249; Part II, *Inform. Sci.* 8 (1976) 301-357; Part III, Inform. Sci. 9 (1976) 43-80.