Pesquisa Operacional (2012) 32(2): 315-329
© 2012 Brazilian Operations Research Society
Printed version ISSN 0101-7438 / Online version ISSN 1678-5142
www.scielo.br/pope

# APPLICATION OF AN ITERATIVE METHOD AND AN EVOLUTIONARY

Ricardo Coelho Silva[1*], Luiza A.P. Cantão[2] and Akebo Yamakami[3]

Received February 4, 2011 / Accepted January 28, 2012

**ABSTRACT.** This work develops two approaches based on the fuzzy set theory to solve a class of fuzzy mathematical optimization problems with uncertainties in the objective function and in the set of constraints. The first approach is an adaptation of an iterative method that obtains cut levels and later maximizes the membership function of fuzzy decision making using the bound search method. The second one is a meta-heuristic approach that adapts a standard genetic algorithm to use fuzzy numbers. Both approaches use a decision criterion called satisfaction level that reaches the best solution in the uncertain environment. Selected examples from the literature are presented to compare and to validate the efficiency of the methods addressed, emphasizing the fuzzy optimization problem in some import-export companies in the south of Spain.

**Keywords**: fuzzy numbers, cut levels, fuzzy optimization, genetic algorithms.

## 1 INTRODUCTION

Mathematical programming is used to solve problems that involve minimization (or maximization) of the objective function in a function domain that can be constrained or not, as described in (Bazaraa *et al.*, 2006) and (Luenberger & Ye, 2008). The formulation of these problems needs to describe clear, certain and brief mathematical definitions both regarding the objective function and the set of constraints. This set of problems can be formalized in the following way:

$$\begin{array}{ll} \min & f(\mathbf{x}) \\ \text{s. to} & \mathbf{g}(\mathbf{x}) \leq \mathbf{b} \\ & \mathbf{x} \in \Omega \end{array} \tag{1}$$

*Corresponding author
[1]Institute of Science and Technology, Federal University of São Paulo – UNIFESP, Rua Talim, 330, 12231-280 São José dos Campos, SP, Brazil. E-mail: ricardo.coelhos@unifesp.br
[2]Environmental Engineering Department, São Paulo State University – UNESP, Av. Três de Março, 511, 18087-180 Sorocaba, SP, Brazil. E-mail: luiza@sorocaba.unesp.br
[3]Department of Telematics, School of Electrical and Computer Engineering, University of Campinas – UNICAMP, Av. Albert Einstein, 400, 13083-852 Campinas, SP, Brazil. E-mail: akebo@dt.fee.unicamp.br

where $\Omega \in \mathbb{R}^n$, $f : \Omega \to \mathbb{R}$ and $\mathbf{g} : \mathbb{R}^k \times \Omega \to \mathbb{R}^k$ Applications can be found in business, industrial, military and governmental areas, among others.

However, there are ambiguous and uncertain data in real-world optimization problems. In recent years, Theory of Fuzzy Sets (Zadeh, 1965) has shown great potential for modeling systems, which are non-linear, complex, ill-defined and not well understood. Fuzzy Theory has found numerous applications due to its ease of implementation, flexibility, tolerant nature to imprecise data, and ability to model non-linear behavior of arbitrary complexity. It is based on natural language and, therefore, is employed with great success in the conception, design, construction and utilization of a wide range of products and systems whose functioning is directly based on the reasoning of human beings.

The first works showing methods that solve optimization problems in fuzzy environments were described in (Tanaka *et al.*, 1974), (Verdegay, 1982) and (Zimmermann, 1983). The approaches there developed solve fuzzy linear programming problems. The fuzzy set theory is applied in a different branch of optimization problems, *i.e.*, the fuzzy graph theory to solve the minimum spanning tree problem, as described in (Almeida, 2006), and the shortest path problem, as described in (Hernandes *et al.*, 2009). Few works focus in developing methods that solve nonlinear programming problems in fuzzy environments. We can highlight some papers that solve problems with uncertainties in the set of constraints, as described in (Lee et al, 1999), (Silva et al, 2008), (Trappey et al, 1988) and (Xu, 1989). Other methods deal with the uncertainties present in some parameters that can be coefficients and/or decision variables, as described in (Berredo *et al.*, 2005), (Ekel *et al.*, 1998), (Ekel, 2002) and (Galperin & Ekel, 2005).

In this fuzzy environment, as it happens in the case of linear programming problems, a variety of fuzzy non-linear programming problems can be defined: non-linear programming problems with a fuzzy objective, *i.e.*, with fuzzy numbers defining the costs of the objective function; non-linear programming problems with a fuzzy goal, *i.e.*, with some fuzzy value to be attained in the objective; non-linear programming problems with fuzzy numbers defining the coefficients of the technological matrix and, finally, with a fuzzy constraint set, *i.e.*, with a feasible set defined by fuzzy constraints. Fuzzy numbers $\tilde{a}$ discussed here are defined by a triangular pertinence function $\mu_{\tilde{a}}(\mathbf{x}) : \mathbb{R} \to [0, 1]$ that associates to each $\mathbf{x} \in \mathbb{R}$ a pertinence level. In this setting, 0 expresses complete exclusion and 1 complete inclusion. Also, $\tilde{\mathbf{a}} \in F(\mathbb{R})$ is commonly referred as a fuzzy set over $\mathbb{R}$.

The goal of this work is proposing two methods, which are applied to solve nonlinear programming problems in a fuzzy environment. The uncertainties are presented in the costs of the objective function, represented as fuzzy numbers, and in the constraint set, allowing some violation of its bounds. The first method is an adaptation of Verdegay's method (Verdegay, 1982), which was developed to solve the linear case. The second is an adapted genetic algorithm that deals with fuzzy numbers (Liu & Iwamura, 1998a, 1998b). Both methods obtain a set of satisfactory solutions that represents the solution of the original fuzzy problems. Thus, the conventional nonlinear programming problem with fuzzy coefficients in the objective function and fuzzy constraints can be formalized in the following form:

$$\begin{array}{ll} \min & f(\tilde{\mathbf{a}};\mathbf{x}) \\ \text{s. to} & \mathbf{g}(\mathbf{x}) \leq_f \mathbf{b} \\ & x \in \Omega \end{array} \tag{2}$$

where $f\colon F(\mathbb{R}^n) \times \Omega \to F(\mathbb{R})$ and $\mathbf{g}\colon \mathbb{R}^k \times \Omega \to \mathbb{R}^k$. The parameter $\tilde{\mathbf{a}} \in R(\mathbb{R}^n)$ represents a vector of fuzzy numbers and $\leq_f$ represents the uncertainties in the constraints.

A defuzzification method, based on the first Yager's index, is used to obtain a real number from the fuzzy objective value. Although the fuzzification of the problem data allows the decision maker to handle its uncertainties, a real (crisp) answer is desirable and easier to interpret and implement. This defuzzification process is denoted by $Df(\cdot)$.

Due to the increasing interest in augmentation of fuzzy systems with learning and adaptation capabilities we find works that join it with some soft computing techniques. In (Cordón *et al.*, 2004), a brief introduction to models and applications of genetic fuzzy systems are presented. In addition, this work includes some of the key references about this topic. In (Coello, 2002), we can find a survey of state of the art theoretical and numerical techniques, which are used in some genetic algorithms to deal with constraint-handling optimization problems. The genetic algorithms are also used to solve multi-objective optimization problems and a hybrid approach, combining fuzzy logic and genetic algorithm, is proposed in (Sakawa, 2002).

This work is divided as follows. Section 2 introduces the adaptations of an iterative method that uses two phases to solve mathematical programming problems with fuzzy parameters in the objective function and uncertainties in the set of constraints; Section 3 introduces an adaptation of a genetic algorithm for the proposed fuzzy problems in this work; Section 4 presents a satisfaction level which tries to establish a tradeoff between $\alpha$-cut level and the solution of the objective function; Section 5 presents numerical simulations for selected problems and an analysis of the obtained results. Finally, concluding remarks are found in Section 6.

## 2 ITERATIVE METHOD

The proposed method in this section can be used to solve non-linear programming problems with uncertainties both in the objective function and in the set of constraints. It is an adaptation of the so-called Two-Phase Method, developed to solve problems with uncertainties in the set of constraints. It is described in (Silva *et al.*, 2007), which adapts classic methods that solve classic mathematical programming problems. Optimality conditions described in (Cantão, 2003) were introduced into this modified method in order to solve non-linear programming problems with fuzzy parameters in the objective function.

According to (Delgado *et al.*, 1989), we consider that the constraints defining the non-linear problem have a fuzzy nature, therefore allowing some violations in the accomplishment of such restrictions. Therefore if we denote each constraint as $g_i(x)$, the problem at hand can be expressed as

$$\begin{array}{ll} \min & f(\tilde{\mathbf{a}};\mathbf{x}) \\ \text{s. to} & g_i(\mathbf{x}) \le_f b_i \quad i \in I \\ & \mathbf{x} \in \Omega \end{array} \tag{3}$$

where the membership functions

$$\mu_i : \mathbb{R}^n \to [0,1], \quad i \in I$$

of the fuzzy constraints should be determined by the decision maker. It is patent that each membership function will give the membership (satisfaction) degree with which any $\mathbf{x} \in \mathbb{R}^n$ accomplishes the corresponding fuzzy constraint on which it is defined. This degree is equal to 1 when the constraint is perfectly accomplished (no violation), and decreases to zero according to greater violations. Finally, for non-admissible violations the accomplishment degree will equal to zero in all the cases. The violation that is related in the satisfaction degree is defined by $d_i$ for each constraint $i$. In the linear case (and formally also in the non-linear one), these membership functions can be formulated as follows

$$\mu_i(\mathbf{x}) = \begin{cases} 1 & g_i(\mathbf{x}) < b_i \\ 1 - \big(g_i(\mathbf{x}) - b_i\big)/d_i & b_i \le g_i(\mathbf{x}) \le b_i + d_i \\ 0 & g_i(\mathbf{x}) > b_i + d_i \end{cases}$$

In order to solve this problem in a two-phase method, first let us define for each fuzzy constraint, $i \in I$

$$X_i = \left\{ \mathbf{x} \in \mathbb{R}^n | g_i(\mathbf{x}) \le_f b_i,\ \mathbf{x} \in \Omega \right\}.$$

If $X = \cap_{i \in I} X_i$ then the former fuzzy nonlinear problem can be addressed in a compact form as

$$\min \left\{ f(\tilde{\mathbf{a}};\mathbf{x}) | \mathbf{x} \in X \right\}.$$

Note that $\forall \alpha \in [0,1]$, an $\alpha$-cut of the fuzzy constraint set will be the classical set

$$X(\alpha) = \left\{ \mathbf{x} \in \mathbb{R}^n | \mu_X(\mathbf{x}) > \alpha \right\}.$$

where $\forall \mathbf{x} \in \mathbb{R}^n$,

$$\mu_X(\mathbf{x}) = \inf\left[\mu_i(\mathbf{x})\right], \ i \in I.$$

Hence an $\alpha$-cut of the $i$-th constraint will be denoted by $X_i(\alpha)$. Therefore, if $\forall \alpha \in [0,1]$,

$$S(\alpha) = \left\{ \mathbf{x} \in \mathbb{R}^n | f(\tilde{\mathbf{a}};\mathbf{x}) = \min f(\tilde{\mathbf{a}};\mathbf{y}), \mathbf{y} \in X(\alpha) \right\}$$

the fuzzy solution of the problem will be the fuzzy set defined by the following membership function

$$S(\mathbf{x}) = \begin{cases} \sup\{\alpha : \mathbf{x} \in S(\alpha)\} & \mathbf{x} \in \cap_\alpha S(\alpha) \\ 0 & \text{otherwise} \end{cases}$$

$S(\mathbf{x})$ represents the biggest pertinence level (than the sup) for a given $\alpha$-cut, that minimizes the fuzzy problem.

Provided that $\forall\alpha \in [0, 1]$,

$$X(\alpha) = \bigcap_{i \in I} \left\{ \mathbf{x} \in \mathbb{R}^n | g_i(\mathbf{x}) \leq r_i(\alpha),\ x \in \Omega \right\}$$

with $r(\alpha) = b_i + d_i(1-\alpha)$, the operative solution to the former problem can be found, $\alpha$-cut by $\alpha$-cut, by means of the following auxiliary parametric nonlinear programming model

$$\begin{array}{ll} \min & f(\tilde{\mathbf{a}}; \mathbf{x}) \\ \text{s. to} & g_i(\mathbf{x}) \leq b_i + d_i(1-\alpha) \quad i \in I \\ & \alpha \in [0, 1],\ x \in \Omega \end{array} \tag{4}$$

It is easy to see that Problem (4) can have two possibilities: (i) it has a new decision variable $\alpha \in [0, 1]$, which helps to transform a fuzzy optimization problem into a classical one; or (ii) it has a parameter because we can discretize $\alpha$ between 0 and 1.

It is clear that one of them is chosen the first phase of this method ends at the start of the second one. In this case we will choose the parameter approach.

In the second phase we solve the parametric nonlinear programming problem determined in the previous step to each one of the different $\alpha$ values using conventional nonlinear programming techniques. We must find solutions of Problem (4) to each $\alpha$ that satisfy the Karush-Kuhn-Tucker's necessary and sufficient optimality conditions. One of the conventional techniques is to write the Lagrange function that is a transformation of Problem (1) as an unconstrained mathematical problem in the following form:

$$L(\tilde{\mathbf{a}}; \mathbf{x}, \eta) = f(\tilde{\mathbf{a}}; \mathbf{x}) + \eta^t \big(g_i(\mathbf{x}) - b_i - d_i(1-\alpha)\big), \quad \forall \mathbf{x} \in \Omega \tag{5}$$

where $\eta$ is the Lagrange multiplier for the inequality. The obtained results, for different $\alpha$ values, generate a set of solutions and then we use the Representation Theorem to integrate all of these particular $\alpha$-solutions. Note that the computational complexity depends on the optimization technique, such as Barrier and Penalty methods, chosen to solve the transformed problem. In addition, some real problems can be harder to solve them by using classical optimization techniques. So we decide to join the approach with a new decision variable and a genetic algorithm. The genetic algorithm will be presented in the next section and then these two approaches will be compared in the Section 5.

## 3 GENETIC ALGORITHM

Genetic algorithms belong to the class of evolutionary computation that simulates the process of natural evolution using stochastic optimization methods. They use the evolution principle as a search method in the solution of optimization problems. Several strategies to obtain a good evolution in genetic algorithms were described in (Goldberg, 1989) and (Michalewics, 1996).

Genetic algorithms have obtained considerable success during the last decades when applied to a wide range of problems, such as optimal control, transport, scheduling problems among others. The implementation of the algorithm here proposed was based on (Liu & Iwamura, 1998a; 1998b), with some modifications. The adapted genetic algorithm in this work uses only basic evolutionary strategies, namely: selection process, crossover operation and mutation operation. Later on we will describe a pseudo-code of this algorithm.

### 3.1 Structure representation

The approach used to represent a solution is the floating point implementation, where each chromosome has the same length as the solution vector. Here we use a vector $V^i = \left(x_1^i, x_2^i, \dots, x_n^i\right)$, with $i = 1, 2, \dots, popsize$ as a chromosome to represent a possible solution to the optimization problem, where $n$ is the dimension.

### 3.2 Initialization process

We define an integer $popsize = 10 \cdot n$ as the number of chromosomes and initialize *popsize* randomly for each chromosome. We choose an interior point to initialize the population, denoted by $V^0$, and select *popsize* individuals randomly in a given neighborhood of this point. We define a large positive number $\Gamma$ which is a step to be taken in a randomly selected direction. This number $\Gamma$ is used not only for the initialization process but also for the mutation operation. We randomly select a direction $d^i \in \mathbb{R}^{popsize}$ and define a chromosome $V^i = V^0 + \Gamma \cdot d^i$. This chromosome is added to the initial population if it represents a feasible solution; otherwise, we randomly select another $\Gamma$ in the interval $[0, \Gamma]$ until $V^0 + \Gamma \cdot d^i$ is feasible. We repeat this process *popsize* times and produce *popsize* initial feasible solutions $V^1, V^2, \dots, V^{popsize}$.

### 3.3 Fitness value

The fitness function evaluates the quality of each individual in the population at each iteration. The fitness of a given individual can be evaluated in several ways. Some forms of computing the fitness values can be found in (Goldberg, 1989) and (Michalewics, 1996). In this work, the value of the objective function was chosen as the fitness value for each individual. Since this value is a fuzzy number, then we apply a defuzzification method to find a classic number which best represents the fitness distribution that is Yager's first index. In (Klir & Folger, 1998) and (Pedrycs & Gomide, 1998), other defuzzification methods are presented.

### 3.4 Evaluation function

An evaluation function, denoted by $eval(V^i)$, is defined to assign a probability of reproduction to each individual of the population. It is reasonable to assume the use of the order relationship among the *popsize* chromosomes such that the *popsize* chromosomes can be rearranged from good to bad (best fitness values for the worse chromosomes). Let $a \in (0, 1)$ be a parameter in the genetic system. We can define the so-called rank-based evaluation function as follows:

$$eval(V^i) = a(1-a)^{i-1}, \quad i = 1, 2, \ldots, popsize\,.$$

We mention that $i = 1$ means the best individual, $i = popsize$ the worst individual, and

$$\sum_{i=0}^{popsize} eval(V^i) \approx 1\,.$$

## 3.5 Selection process

The selection process is based on spinning the roulette wheel *popsize* times, each time selecting a single chromosome for an auxiliary population in the following way:

1. Calculate the cumulative probability $q_i$ for each individual $V^i$,

$$\begin{aligned} q_0 &= 0, \\ q_i &= \sum_{j=1}^{i} eval(V^j), \quad i = 1, 2, \ldots, popsize; \end{aligned}$$

2. Generate a random real number **r** in $\left[0, q_{popsize}\right]$;
3. Select the $i$th individual $V^i$, $(1 \le i \le popsize)$ such that $q_{i-1} < r \le q_i$;
4. Repeat steps 2 and 3 *popsize* times and obtain *popsize* individuals.

## 3.6 Crossover operator

We define a parameter $P_c$ to determine the probability of crossover. This probability gives us the expected number $P_c \cdot popsize$ of chromosomes which undergo the operation. We denote the selected parents as $F^1, F^2, \ldots$ and divide them in pairs. At first, we generate a random number $c$ from the open interval $(0, 1)$, then we produce two children $C^1$ and $C^2$ using the crossover operator as follows:

$$\begin{aligned} C^1 &= c * F^1 + (1-c) * F^2, \\ C^2 &= c * F^1 + (1-c) * F^2. \end{aligned}$$

We check the feasibility of each child. If both children are feasible, then we replace the parents by them. If not, we keep the feasible one if it exists, and then reapply the crossover operator by regenerating the random number $c$ until two feasible children are obtained or a given number of cycles is finished. In this case, we only replace the parents by the feasible children. The individuals that were not chosen to do the crossover operator will used in the population of the next generation.

### 3.7 Mutation operator

A parameter $P_m$ is defined as the probability of mutation. This probability gives us the expected number of $P_m \cdot popsize$ of chromosomes which undergo the mutation operations. For each selected parent, denoted by $V^i = (x_1, \ldots, x_n)$ we choose a mutation direction $d^i \in \mathbb{R}^n$ randomly. If $M^i = V^i + \Gamma \cdot d^i$ is not feasible we set $\Gamma$ as a random number between 0 and $\Gamma$ until it is feasible. If the above process fails to find a feasible solution in a predetermined number of iterations, we set $\Gamma = 0$.

### 3.8 Pseudo-code for the genetic algorithm

Following selection, crossover and mutation, the new population is ready and we start a new iteration. The genetic algorithm will terminate after a given number of cyclic repetitions of the above steps. The pseudo-code of the genetic algorithm has the following form:

```
Algorithm 1 – adapted genetic algorithm
Input: Parameter popsize, $P_c$ and $P_m$
Output: The best individual in all generations.
Step 01: Initialize popsize individuals;
Step 02: Check feasibility of each individual;
WHILE number of generations is less than MAX_GENERATION DO
        Step 03: Compute the fitness of each individual;
        Step 04: Order the population, compute evaluation value;
        Step 05: Perform the selection process;
        Step 06: Use the crossover operator and check feasibility;
        Step 07: Use the mutation operator and check feasibility;
        Step 08: Update the population for the next generation;
END
```

## 4 SATISFACTION LEVEL

We define a satisfaction level representing the compromise between the objective function value and the satisfaction level of the pertinence function associated to the fuzzy inequalities. This level determines the lower bound for the membership value of the admissible solutions.

The solution of the Problem (4) belongs to the bounded interval for $\alpha \in [0, 1]$ that generates the upper and lower values of the form

$$\begin{aligned} \tilde{m} &= f(\tilde{\mathbf{a}}; \tilde{\mathbf{x}}(0)) = \min_{x \in X(0)} f(\tilde{\mathbf{a}}; \tilde{\mathbf{x}}) \\ \tilde{M} &= f(\tilde{\mathbf{a}}; \tilde{\mathbf{x}}(1)) = \min_{x \in X(1)} f(\tilde{\mathbf{a}}; \tilde{\mathbf{x}}), \end{aligned} \tag{6}$$

where $X(0)$ and $X(1)$ are the cut levels for $\alpha$ equal to 0 and 1, respectively. Numbers $\tilde{m}$ and $\tilde{M}$ are fuzzy numbers, being the result of arithmetical operations with the fuzzy coefficients present on the objective function (Kaufmann et al, 1991). In a fuzzy optimization problem we can establish the fuzzy goal as follows:

$$\mu_{\tilde{G}}(\mathbf{x}) = \frac{\tilde{m}}{f(\tilde{\mathbf{a}};\tilde{\mathbf{x}})}. \tag{7}$$

As expected, this fuzzy goal shows that when $f$ reaches its infimum $m$ the full membership ($\mu_{\tilde{G}} = 1$) is obtained; as $f$ increases $\mu_{\tilde{G}}$ approaches the null membership ($\mu_{\tilde{G}} = 0$) boundary. Clearly, the upper and lower limits of the fuzzy goal are given by

$$\begin{aligned} \mu^u_G &= 1 \\ \mu^l_G &= Df\left(\frac{\tilde{m}}{\tilde{M}}\right), \end{aligned} \tag{8}$$

where $Df(\cdot)$ is Yager's first index defuzzification method, as previously mentioned. However, if the objective is to maximize or the lower and upper values are negative, we can invert the division of the Equation (7), and then, the equation of the lower limit of the fuzzy goal is defined as

$$\mu^l_G = Df\left(\frac{\tilde{m}}{\tilde{M}}\right).$$

The minimum satisfaction level is determined by the quotient between the solution of the problem with totally violated constraints and the solution with totally satisfied constraints, see (Silva *et al.*, 2005) and (Silva *et al.*, 2006). This quotient is a fuzzy number that is determined by a fuzzy relation. The chosen defuzzification method is the modal value to determine the inferior bound.

The iterative method then performs a search in the interval $\left[\mu^l_G(\mathbf{x}), \mu^u_G(\mathbf{x})\right]$ to find the optimal $\alpha$-level according to (Bellman & Zadeh, 1970). The genetic algorithm uses the inferior $\alpha$-level to determine when a solution is feasible or not.

## 5 NUMERICAL EXPERIMENTS

The problems to evaluate the developed methods are presented in Subsection 5.1. There are three hypothetic formulations and a fuzzy optimization problem in some import-export companies in South-east Spain. Nevertheless, they are efficient to validate the study realized.

The computational results and a comparative analysis of the classic methods and the iterative methods responses are presented in Section 5.2.

For all examples, we use the following genetic algorithm parameters:

- Crossover probability $P_c = 0.2$;
- Mutation probability $P_m = 0.1$;
- Number of generations (iterations): 1000.

The tests were all performed on a Sun Blade 250 with two 1.28GHZ Ultra Sparc-IIIi processor, 4GB RAM running Solaris 9 operational system.

## 5.1 Formulation of the problem

In this paper we present some theoretical and real-world problems found in the literature in order to validate the proposed algorithms. We simulate four nonlinear programming problems: three theoretical problems and one real-world problem in some import-export companies in South-east Spain. All problemas have uncertainties on the parameters of the objective function.

### 5.1.1 Theoretic tests problems

The problems with inequality constraints, described in Table 1, were taken from (Schittkowski, 1987). Uncertainties were inserted into the parameters of the objective functions in the form of a 10% variation in the modal value. The maximum violation of each constraint of the problem is a hypothetical value and it was inserted to suppose a set of well-known constraints. The optimal solutions to the problems without uncertainties are presented in the columns $\bar{\mathbf{x}}$ and $f(\bar{\mathbf{x}})$ of Table 1.

**Table 1** – Theoretical problems consisting only of inequality constraints.

| Prob. | $f(\tilde{\mathbf{a}};\tilde{\mathbf{x}})$ | Fuzzy spread | $x_{init}$ | Constraints | Violation | Classic solution | |
|---|---|---|---|---|---|---|---|
| | | | | | | $\bar{\mathbf{x}}$ | $f(\bar{\mathbf{x}})$ |
| PG1 | $(x_1-\tilde{2})^2+$ $(x_2-\tilde{1})^2$ | 10% | [0.5, 0.5] | $x_1^2-x_2 \leq_f 0$ <br> $x_2^2-x_1 \leq_f 0$ | $d_1=0.35$ <br> $d_2=0.35$ | [1.0, 1.0] | 1 |
| PG2 | $x_1^2+x_2^2-$ $\tilde{4}x_1+\tilde{4}$ | 10% | [1.0, 2.5] | $x_2-x_1-2 \leq_f 0$ <br> $x_1-x_2-2 \leq_f 0$ <br> $-x_1 \leq 0$ <br> $-x_2 \leq 0$ | $d_1=1.0$ <br> $d_2=0.5$ <br> $d_3=0.0$ <br> $d_4=0.0$ | [0.5536, 1.3060] | 3.7989 |
| PG3 | $\tilde{9}x_1^2+x_2^2+$ $\tilde{4}x_1+\tilde{4}$ | 10% | [1.0, 1.0, 1.0] | $1-x_1x_2 \leq_f 0$ <br> $-x_2 \leq 0$ <br> $-x_3 \leq 0$ | $d_1=0.5$ <br> $d_2=0.0$ <br> $d_3=0.0$ | [0.5774, 1.7320, 0.0000] | 6 |

### 5.1.2 Real-world problem

The problem in some import-export companies in the south of Spain, described in Table 2, was taken from of (Jiménez *et al.*, 2006). The same parameters and uncertainties of this real-world problem were used in here, but uncertainties were inserted in the parameters of the objective functions. They are in the form of a 10% variation in the modal value, *e.g.* the number $\tilde{2}$ can vary up to 0.2 units positively or negatively. The optimal solutions to the problems without uncertainties are presented in the columns $\bar{x}$ and $f(\bar{x})$ of Table 2.

**Table 2** – Problem in some import-export companies in the south of Spain.

| Prob. | $f(\tilde{\mathbf{a}}; \tilde{\mathbf{x}})$ | $x_{init}$ | Constraints | Violation | Classic solution | |
|---|---|---|---|---|---|---|
| | | | | | $\bar{\mathbf{x}}$ | $f(\bar{\mathbf{x}})$ |
| PCSS | $\widetilde{23.0}x_1 + \widetilde{32.0}x_2 - \widetilde{0.04}x_1 - \widetilde{0.03}x_2$ | [0.0, 0.0] | $10x_1 + 6x_2 \leq_f 2500$<br>$5x_1 + 10x_2 \leq_f 2000$<br>$7x_1 + 10x_2 \leq_f 2050$ | $d_1 = 0.0$<br>$d_2 = 64.0$<br>$d_3 = 74.0$ | [84.8810, 145.5800] | 5686.9 |

## 5.2 Results and analysis

In Tables 1 and 2, we presented the problem formulations and their classical solutions. In this subsection we show the results obtained for the problems in sub-section 5.1 by the iterative method presented in section 2 and by the genetic algorithm introduced in 3.

Tables 3 and 5 refer to the results obtained on the first-phase of the method from Section 2 in two forms: (i) completely satisfied restrictions, including $\alpha = 1$ and, (ii) restrictions completely violated ($\alpha = 0$). Tables 4 and 6 bring the results from second-phase, Sections 2 and 3. The column corresponding to $f(\tilde{\mathbf{a}}; \mathbf{x})$ shows a 3-vector: first and third components are the spread, while second component is the modal value. Column $\mu$ presents the comparison between the $\alpha$-cut level and the solution of the objective function as discussed on Section 4. The closer $\mu$ is from 1 (one), the better the compatibility among the pertinences of both the objective and the restriction set.

**Table 3** – Maximum and minimum levels of tolerance for the set of constraints.

| Prob. | Constraints | Optimal of $f(\tilde{\mathbf{a}}; \mathbf{x})$ | | | Time |
|---|---|---|---|---|---|
| | | $\mathbf{x}$ | $f(\tilde{\mathbf{a}}; \mathbf{x})$ | $Df(\tilde{f})$ | |
| PG1 | Totally satisfied | [1.0005; 1.0004] | [0.8066; 0.9991; 1.2115] | [1.0041] | 2s |
| | Totally violated | [1.2074; 1.2073] | [0.5044; 0.6712; 0.8629] | [0.6774] | 1s |
| PG2 | Totally satisfied | [0.5530; 1.3052] | [3.1761; 3.7973; 4.4185] | [3.7973] | 4s |
| | Totally violated | [0.5769; 1.2322] | [2.9128; 3.5436; 4.1743] | [3.5436] | 15s |
| PG3 | Totally satisfied | [0.5780; 1.7287; 0.000] | [5.6945, 5.9952, 6.2959] | [5.9952] | 8s |
| | Totally violated | [0.5493; 1.6369; 0.0000] | [5.1237; 5.3953; 5.6669] | [5.3953] | 11s |

By examining the results presented in Table 3, we can calculate the minimum satisfaction level to each problem in Table 1 of fuzzy mathematical programming problems with uncertainties in objective function and in set of constants shown in this work.

Table 4 presents the results for the problem with inequality constraints, obtained by the iterative method and adapted genetic algorithm. For the problem PG1, the iterative method obtained better responses in every front, *i.e.*, lower defuzzification value and lower convergence time, while the satisfaction level is an admissible value. The genetic algorithm was slower than the iterative method and they were equal in PG2. Both obtained the same convergence time, but the adapted

**Table 4** – Results using iterative method and pure genetic algorithm.

| Prob. | Algorithm | Optimal of $f(\tilde{\mathbf{a}}; \mathbf{x})$ | | | | Time |
|---|---|---|---|---|---|---|
| | | $\mathbf{x}$ | $f(\tilde{\mathbf{a}}; \mathbf{x})$ | $Df(\tilde{f})$ | $\mu = \alpha$ | |
| PG1 | Section 2 | [1.058; 1.058] | [0.7091; 0.8908; 1.0974] | [0.8970] | 0.7575 | 4s |
| | Section 3 | [1.0608; 1.0454] | [0.5385; 0.8842; 1.3189] | [0.9071] | 0.7591 | 12s |
| PG2 | Section 2 | [0.5547; 1.3015] | [3.1607; 3.7826; 4.4045] | [3.7826] | 0.9361 | 11s |
| | Section 3 | [0.52202; 1.2306] | [2.9113; 3.6988; 4.4863] | [3.6988] | 0.9580 | 11s |
| PG3 | Section 2 | [0.5764; 1.7191; 0.000] | [5.6464; 5.9454; 6.2444] | [5.9454] | 0.9169 | 5s |
| | Section 3 | [0.5559; 1.7345; 0.0827] | [5.2659; 5.8510; 6.4361] | [5.8663] | 0.9197 | 8s |

genetic algorithm had better defuzzification value and satisfaction level. In PG3, the iterative method was faster, but the genetic obtained better defuzzification value and satisfactions level. We obtain satisfaction levels higher than 75% for all the problems. The good quality of the obtained solution is confirmed by the fact that the defuzzified values of the objective function for each problem are lower than those of the classic solution.

Table 5 depicts the optimal solutions of the real-world problem in two forms: (i) totally satisfied constraints; and (ii) totally violated constraints. Table 5 shows the results for Problem (4) imposing $\alpha = 1$, for case (i), and $\alpha = 0$, for case (ii).

**Table 5** – Maximum and minimum levels of tolerance for the set of constraints.

| Prob. | Constraints | Optimal of $f(\tilde{\mathbf{a}}; \mathbf{x})$ | | | Time |
|---|---|---|---|---|---|
| | | $\mathbf{x}$ | $f(\tilde{\mathbf{a}}; \mathbf{x})$ | $Df(\tilde{f})$ | |
| PCSS | Totally satisfied | [86.073; 144.75] | [4961.8; 5686.7; 6436.5] | [5694.6] | 2s |
| | Totally violated | [87.925; 150.85] | [5103.2; 5857.6; 6637.6] | [5866.1] | 1s |

In Table 6, we explore the results for the real-world problem. Note that the iterative method obtained a satisfaction level near 100%. However, the genetic algorithm presented the best fuzzy result and defuzzification value with lower processing time.

**Table 6** – Result to the problem with triangular fuzzy numbers.

| Prob. | Algorithm | Optimal of $f(\tilde{\mathbf{a}}; \mathbf{x})$ | | | | Time |
|---|---|---|---|---|---|---|
| | | $\mathbf{x}$ | $f(\tilde{\mathbf{a}}; \mathbf{x})$ | $Df(\tilde{f})$ | $\mu = \alpha$ | |
| PCSS | Section 2 | [84.813; 145.63] | [4962.2; 5686.9; 6436.5] | [5695.2] | 0.9992 | 6s |
| | Section 3 | [82.7938; 147.231] | [4965.9; 5691.2; 6441.3] | [5699.4] | 0.9172 | 2s |

The main analysis is in choosing the relation between objective function value and satisfaction level. This choice depends on the decision maker because he has a previous knowledge of the main objective to be reached. In this case, the genetic algorithm was faster than the iterative method.

## 6 CONCLUSION

The first method described in this work transforms the fuzzy non-linear programming problem in a parametric non-linear programming problem. The parameter is in the unit interval and we obtain the optimal solution of each discretized value of the parameter. The evolutionary algorithm described previously presents the basic steps of a standard genetic algorithm but there is a difference regarding the objective function value of the non-linear programming problem with fuzzy parameters. The satisfaction level was described in this work to determine a compromise between the fuzzy goal of the objective function and the permitted violation level of each one of the membership function of the constraints.

We developed an interactive method and a genetic algorithm to solve mathematical programming problems with uncertainties in objective function and in set of constraints. The iterative method uses the differentiation from the objective function that presented good responses to proposed problems. The obtained results were better than the classic results found in the literature. The genetic algorithm presented good responses to the same problems. However, they presented a satisfaction level smaller than 100%, *i.e.*, the optimum solution violates one or more constraints of the problem.

## REFERENCES

[1] ALMEIDA TA, YAMAKAMI A & TAKAHASHI MT. 2006. Sistema imunológico artificial para resolver o problema da árvore geradora mínima com parâmetros fuzzy. *Pesquisa Operacional*, **27**(1): 131–154.

[2] BAZARRA MS, SHERALI HD & SHETTY CM. 2006. *Nonlinear Programming – Theory and Algorithms*. 3rd ed., John Wiley & Sons, New York.

[3] BELLMAN RE & ZADEH LA. 1970. Decision-making in a fuzzy environment, *Management Science*, **17**(4): B141–B164.

[4] BERREDO RC, EKEL PY & PALHARES RM. 2005. Fuzzy preference relations in models of decision making. *Nonlinear Analysis*, **63**: e735–e741.

[5] CANTÃO LAP. 2003. Programação não-linear com parâmetros fuzzy. Tese de doutorado. FEEC – Unicamp, Campinas, Março.

[6] COELLO CAC. 2002. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of state of the art. *Computer Methods and Applied Mechanics and Engineering*, **191**: 1245–1287.

[7] CORDÓN O, GOMIDE F, HERRERA F, HOFFMANN F & MAGDALENA L. 2004. Ten years of genetic fuzzy systems: current framework and new trends. *Fuzzy Sets and Systems*, **141**(1): 5–31.

[8] DELGADO M, VERDEGAY JL & VILA M. 1989. A general model fuzzy linear programming. *Fuzzy Sets and Systems*, **29**: 21–29.

[9] EKEL PY. 2002. Fuzzy sets and models of decision making. *International Journal Computers & Mathematics with Applications*, **44**: 863–875.

[10] EKEL PY, PEDRICZ W & SCHINZINGER R. 1998. A general approach to solving a wide class of fuzzy optimization problems. *Fuzzy Sets and systems*, **97**: 46–66.

[11] GALPERIN EA & EKEL PY. 2005. Synthetic realization approach to fuzzy global optimization via gamma algorithm. *Mathematical and Computer Modelling*, **41**: 1457–1468.

[12] GOLDBERG DE. 1989. *Genetic Algorithms in Search, Optimization, and Machine Learning*, Addison-Wesley, Reading.

[13] HERNANDES F, BERTON L & CASTANHO MJP. 2009. O problema de caminho mínimo com incertezas e restrições de tempo. *Pesquisa Operacional*, **29**(2): 471–488.

[14] JIMÉNEZ F, CADENAS JM, SÁNCHEZ G, GÓMEZ-SKARMETA AF & VERDEGAY JL. 2006. Multiobjective evolutionary computation and fuzzy optimization. *International Journal of Approximate Reasoning*, **43**(1): 59–75.

[15] KAUFMANN A & GUPTA MM. 1991. *Introduction to Fuzzy Arithmetic: Theory and Applications*. Van Nostrand Reinhold.

[16] KLIR GJ & FOLGER TA. 1998. *Fuzzy Sets, Uncertainty and Information*, Prentice Hall, New York.

[17] LEE YH, YANG BH & MOON KS. 1999. An economic machining process model using fuzzy nonlinear programming and neural network. *International Journal of Production Research*, **37**(4): 835–847.

[18] LIU B & IWAMURA K. 1998a. Chance constrained programming with fuzzy parameters. *Fuzzy Sets and Systems*, **94**: 227–237.

[19] LIU B & IWAMURA K. 1998b. A note on chance constrained programming with fuzzy coefficients. *Fuzzy Sets and Systems*, **100**: 229–233.

[20] LUENBERGER DG & YE Y. 2008. *Linear and Nonlinear Programming*. 3rd ed., Addison-Wesley, Massachusetts.

[21] MICHALEWICS Z. 1996. *Genetic Algorithms + Data Structure = Evolution Programs*. 3rd ed., Springer, New York.

[22] PEDRYCS W & GOMIDE F. 1998. *An Introduction of Fuzzy Sets – Analysis and Design*. A Bardford Book.

[23] SAKAWA M. 2002. *Genetic algorithms and fuzzy multi-objective optimization*. Kluwer Academic Publishers.

[24] SCHITTKOWSKI K. 1987. *More Test Examples for Nonlinear Programming Codes*. Springer-Verlag.

[25] SILVA RC, CANTÃO LAP & YAMAKAMI A. 2005. Meta-heuristic to mathematical programming problems with uncertainties. In: *II International Conference on Machine Intelligence*, Tozeur-TN.

[26] SILVA RC, CANTÃO LAP & YAMAKAMI A. 2006. Adaptation of iterative methods to solve fuzzy mathematical programming problems. *Transactions on Engineering, Computing and Technology*, **14**: 330–335.

[27] SILVA RC, CANTÃO LAP & YAMAKAMI A. 2008. Relação entre modelos de programação não-linear com incerteza no conjunto de restrições. *Pesquisa Operacional*, **28**: 383–398.

[28] SILVA RC, VERDEGAY JL & YAMAKAMI A. 2007. Two-phase method to solve fuzzy quadratic programming problems. In: *2007 IEEE International Conference on Fuzzy Systems*, London-UK, 1–6.

[29] TANAKA H, OKUDA T & ASAI K. 1974. On fuzzy-mathematical programming. *Journal of Cybernetics*, **3**(4): 37–46.

[30] TRAPPEY JFC, LIU CR & CHANG TC. 1988. Fuzzy non-linear programming: theory and application in manufacturing. *International Journal of Production Research*, **26**(5): 975–985.

[31] VERDEGAY JL. 1982. Fuzzy mathematical programming. In: Fuzzy Information and Decision Procsses, Gupta MM & Sanches E (eds.). North-Holland Publishing Company, Amsterdan.

[32] XU C. 1989. Fuzzy optimization of structures by the two-phase method. *Computers & Structures*, **31**(4): 575–580.

[33] ZADEH LA. 1965. Fuzzy sets. *Information and Control*, **8**: 338–353.

[34] ZIMMERMANN HJ. 1983. Fuzzy mathematical programming. *Computer & Operation Research*, **10**(4): 291–298.