CAHIER DU LAMSADE

Laboratoire d'Analyse et Modélisation de Systèmes pour l'Aide à la Décision
(Université de Paris-Dauphine)
Unité Associée au CNRS n° 825

AN INTERACTIVE MULTIOBJECTIVE NONLINEAR PROGRAMMING PROCEDURE

CAHIER N° 99
avril 1990

H. M'SILTI
P. TOLLA

Received February 1990.

CONTENTS

	Pages
RESUME	I
ABSTRACT	I
1. INTRODUCTION	1
2. STANDARD MULTIOBJECTIVE FORMULATION	1
2.1 EXAMPLE	3
2.2 ATTEMPTS TO CALCULATE GLOBAL SOLUTION	9
3. OUTLINE OF THE METHOD	12
4. COMPUTER IMPLEMENTATION AND COMPUTATIONAL RESULTS	16
5. CONCLUSION	19
REFERENCES	20

UNE PROCEDURE INTERACTIVE POUR LA RESOLUTION D'UN PROBLEME NON-LINEAIRE MULTIOBJECTIF.

RESUME : Ce papier présente une méthode de résolution d'un problème non-linéaire multiobjectif. On y décrit une procédure interactive basée sur la méthode GRG (Gradient Réduit Généralisé) de J. Abadie (1978). On développe plusieurs heuristiques pour prendre en compte le problème de la recherche d'une solution globale. La méthode comporte trois grandes parties :
1) génération d'un sous-ensemble de l'ensemble des solutions efficaces ;
2) définition de l'information concernant les préférences du Décideur ;
3) détermination d'une solution de compromis basée sur des techniques de la programmation non-linéaire et des procédures de recherche de points de références (point Idéal, point Nadir). Le programme développé permet de résoudre des problèmes de taille moyenne sur micro-ordinateurs.

Mots-clés : Programmation non-linéaire multiobjectif, méthode interactive.

AN INTERACTIVE MULTIOBJECTIVE NONLINEAR PROGRAMMING PROCEDURE

ABSTRACT : This paper develops a method for interactive MultiObjective NonLinear Programming procedure (MONLP). It provides a detailed description of an efficient algorithm, and reports on promising computational results. It also discusses several alternatives strategies for implementing GRG code (Generalized Reduced Gradient), which is known as one of the " best " method for solving nonlinear optimization problems (Abadie, 1978). The method relies on three steps :
1) generation of a subset of feasible efficient solutions ;
2) interactive definition of information concerning the Decision Maker (DM) preference structure, relative to its outcomes ;
3) determination of a compromise solution using nonlinear optimization, global analysis and ideal point search procedure in the outcome space.
Following this methodology, it is possible for the DM to find its final solution. A micro-computer version (for medium problems) of the method is available.

Key-Words : Nonlinear multiobjective programming, interactive method.

1. INTRODUCTION

Multiobjective Mathematical Programming (MMP) is one way of considering, explicitely and simultaneously, multiple objectives in a mathematical programming framework. Many of the algorithms for solving multiobjective mathematical programs (MMPs) are defined as interactive procedures. Consequently these interactive approaches require quick response times and easy-to-use software. The purpose of this paper is to describe an interactive multiobjective nonlinear programming procedure for determining a " best compromise " solution.

The basic idea is to provide a mean to aid the DM without an explicit definition of a utility function in terms of its preference characteristics. Our decision situation involves a single DM who has three objectives. Without loss of generality, the objective functions may all be assumed to be maximized.

This article is organized as follows. In section 2 we define the problem and we discuss some specific questions and possible heuristics to resolve them. In section 3 we outline the method to be used. A brief report on computational experiments is given in section 4. We also discuss certain computational aspects of our method. Section 5 is devoted to conclusions and further developments.

2. STANDARD MULTIOBJECTIVE FORMULATION.

In this study, we consider MMPs in the following equivalent forms :

$$\begin{array}{l} \text{Max } f_1(x) \\ \text{Max } f_2(x) \\ \text{Max } f_3(x) \\ \text{s.t. } x \in A \end{array} \quad \text{or} \quad \begin{array}{l} \text{Max } F(x) = (f_1(x), f_2(x), f_3(x)) \\ \text{s.t.} \quad x \in A \end{array} \qquad (1)$$

where :

- $x \in R^n$, $x = (x_1, x_2, ..., x_n)^t$, is the vector of decision variables ;
- F(x) is the vector of objectives to be " maximized " ;
- A is the feasible region, defined by equalities and/or inequalities constraints

$g_j(x) \leq 0$; $j = 1,...,k$
$g_j(x) = 0$; $j = k+1,...,m$
$a_i \leq x_i \leq b_i$; $i = 1,...,n$

- $n \leq 200$ decision variables ; $m \leq 100$ constraints ;
- all the variables must have upper and lower bounds (which may be $+\infty$ or $-\infty$ respectively).

Let us suppose that :
- all nonlinear functions f_i ($i=1,2,3$) and g_j ($j=1,2,...,m$) are defined over A and differentiable everywhere ;
- each objective f_i is bounded inside A, and A is not necessarily convex (cf.Fig.1);
- there is no point in A at which all objectives are simultaneously maximized. In such a case, this point, called a " superior solution ", is the solution of problem (1) ;
- the functions f_i and g_j are not necessarily convex;
- E_f denotes the set of efficient solutions.

In MMPs, the objectives describe some aspects which are conflicting, and usually noncommensurable. Thus, it is difficult to combine them into a single objective. Consequently, the concept of optimality for single objective optimization problems cannot be applied for MMPs.

Fundamental to the MMP is the Pareto optimal solution. Conceptually, a Pareto optimal solution of MMPs is one where any improvement of one objective function can be achieved only at the expense of another criterion. Mathematically, different notions of Pareto optimality are defined (see for instance, Wight, 1982). An example is the concept of a properly efficient solution (Geoffrion, 1968), or a quasi-efficient solution (Arrow and Hahn, 1971 ; Lowe and al., 1984). Here we focus on the concept of a strong efficient solution (see for instance, Steuer, 1986, chap. 14, p.419-453).

Definition 1 : a point $x' \in A$ is said to be a strong efficient solution, or equivalently strong Pareto optimal solution to the MMP (1) if and only if it does not exist another $x \in A$ such that $f_i(x) \geq f_i(x')$ for all $i=1, 2, 3$ and $f_i(x) > f_i(x')$ for some $i=1, 2, 3$.

Definition 2 : a point $x' \in A$ is said to be a weak efficient solution to the MMP (1) if and only if it does not exist another $x \in A$ such that $f_i(x) \geq f_i(x')$ for all i.

Definition 3 : the ideal point for MMP (1) is a point in the outcome space, denoted by F^*, where $F^* = (f_1^*, f_2^*, f_3^*)^t$, $F^* \in R^3$, and such that f_i^* is the optimal objective function value for the problem (Pi)

$$\left| \begin{array}{l} \text{Max } f_i(x) \\ \\ \text{s.t. } x \in A \end{array} \right. \qquad \text{(Pi)}$$

In other words, $f_i^* = f_i(X_i^*)$, where X_i^* is the optimal solution of (Pi) , i=1,2,3 .

In most cases, there will be many efficient solutions. One of these will be preferred by the DM, in terms of its outcomes. However, determining what this solution is, requires further information from the DM concerning his preferences. One way of expressing this information is through the use of a value function over the multiple objectives of the problem (see for instance Jacquet-Lagrèze and al. 1987).

2.1 EXAMPLE : to illustrate the above definitions, consider the following example :

$$\left| \begin{array}{l} \text{Max } f_1 = x_1 + x_2 \\ \\ \text{Max } f_2 = x_2 - x_1 \\ \text{s.t.} \\ (x_1 - 6)^2 + (x_2 - 5)^2 \geq 16 \\ \\ \quad 0 \leq x_1 \leq 10 \\ \quad 0 \leq x_2 \leq 5 \end{array} \right.$$

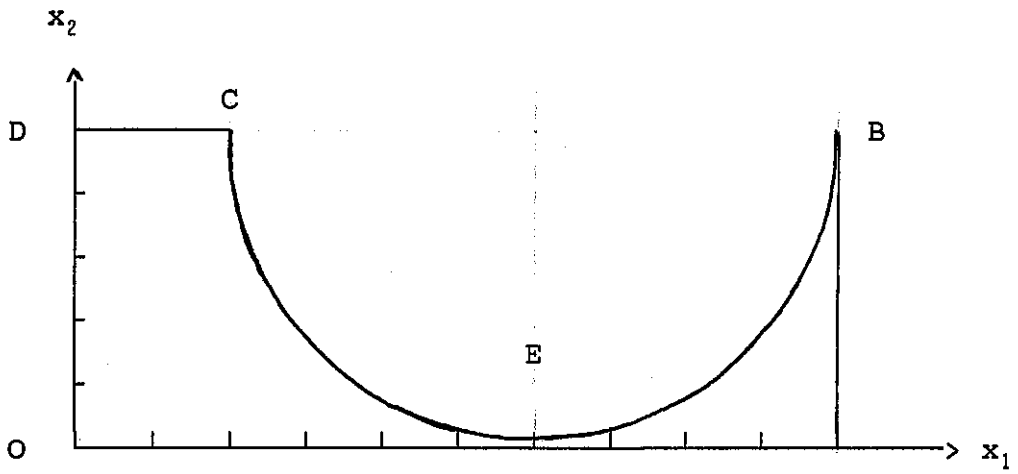


Fig. 1 : feasible set A in the decision variables space. A is non-convex.

The plot of the feasible set A is shown in Fig.1. The maximum solutions are B and D for f_1 and f_2, respectively. In Fig.2, we have plotted the values of the objective functions for this problem. Each feasible solution in Fig.1 has a corresponding point in Fig.2, for instance B'= F(B) = (15,-5).

Clearly, it appears that the strong efficient solutions set in the outcome space (Fig.2) is $E'_f = [D',C'] \cup B'$. $E'_f = F(E_f)$ where $E_f = [D,C] \cup B$, is the strong efficient set in the decision variables space. The compromise solution y is one of these, i.e. $y \in E_f$. E' is weakly dominated by B' and C'.

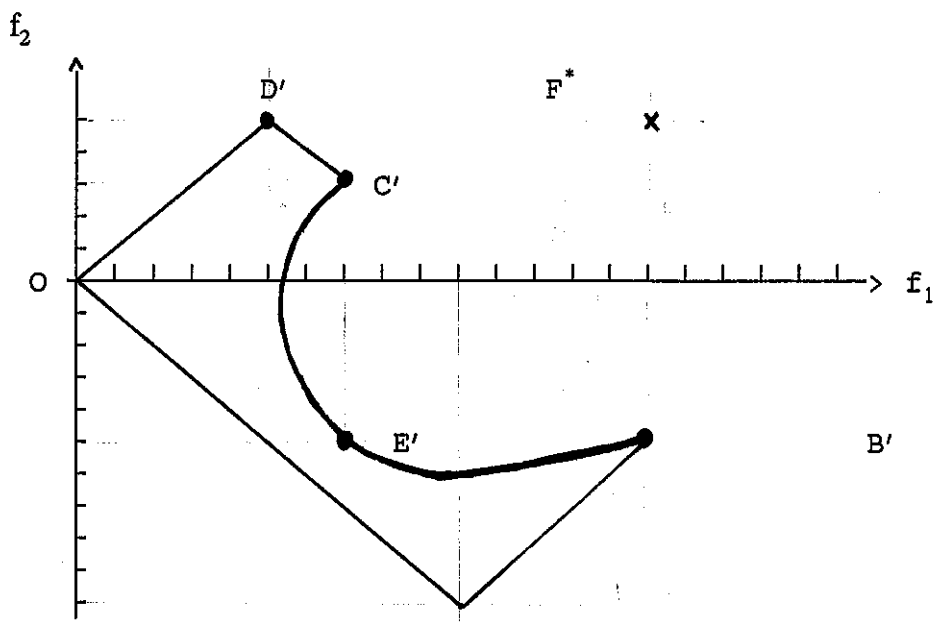


Fig. 2 : S = F(A) is the feasible set of value functions in the outcome space. S is non-convex. F^* = (15, 5) is the ideal point.

Let us consider the following table :

	x_1	x_2
X_1^*	10	5
X_2^*	0	5
X^I	5	5

where : $X_1^* = (x_{11}, x_{12})^t = (10,5)$ is the optimal solution of (P1) ; $f_1^* = 15$.
$X_2^* = (x_{21}, x_{22})^t = (0,5)$ is the optimal solution of (P2) ; $f_2^* = 5$.

$X^I = (x_1, x_2)^t$ is the initial arbitrary compromise solution, i.e. the solution that we propose to the DM as the initial compromise if he is himself unable to choose one compromise (if X^I is not feasible, a phase-one in GRG code finds a feasible point from X^I by means of a Raphson-Newton procedure).

Several heuristics will be developped in the " best compromise solution " search procedure, for example :

1) we can suppose here that the " final " solution will be such that $x_2 = 5$ because all the 2nd components are equal to 5, and thus we can rewrite the initial problem as $5 \leq f_1 \leq 7$; $3 \leq f_2 \leq 5$; $f_2 = 10 - f_1$.
$X^I = \alpha X_1^* + \beta X_2^*$ with $\alpha + \beta = 1$; $X^I=(5,5)^t$ for $\alpha = \beta = 0.5$ is not feasible and $F(X^I)=(10,0)$ is out of S (Fig.2).

2) let us denote by X^k the solution obtained at iteration k. If X^k is not accepted by the DM or the computer program, we set :
$X^{k+1} = X^k + \alpha (P - X^k)$; $0 < \alpha < 1$. In practice, any point can be taken, for example $P = X^I$. If X^{k+1} is not feasible a phase-one in GRG code finds a feasible point from X^{k+1}.

The purpose of this phase is to ask the DM, if possible, to compare different alternatives. The alternatives are expressed as scenarios in terms of their objective value functions in the outcome space. However, in order to aid the DM to choose between alternatives, such alternatives have not to be too similar. For example, given one solution X^k and $F(X^k)$, we specify a solution X^{k+1} such that a difference varying from 5 to 10% exists in all or some objective value functions. If the DM likes one alternative, we terminate the questionning process, we revise step-bounds if necessary, and we solve a NonLinear Program (see § 2.2 and § 3).

We attempt to be more precise about the " volume " of the region containing solution better than X^k :

1. The variables which are " approximately " known to be declared as constants can be excluded from the optimization process until an optimum has been found. The declaration of the constant can then be deleted and the process can be set up again with the constant considered now as a variable. An initial point for a new iteration can be chosen during the optimization process. This is possible because the DM is not usually interested in all variables.

2. If it is possible, we arrange the variables per classes, and we give different priorities to these classes (M'silti,1984). This " partition " allows us to solve the problem step wise, i.e. starting with the group having the highest priority, the other variables being set to zero or any constant value, we obtain a solution ; then we introduce another group into the problem, and so on. The two default " partitions " are : a) the class 1 which contains the nonlinear structural variables, the class 2 which contains the linear variables ; b) the class i which contains only variables concerning objective f_i (i=1,2,3). Priority decreases from class 1 to class i.

Let us consider the following example :

$$\text{Max } f_1 = x_1 + x_2 - x_4^2 x_1$$

$$\text{Max } f_2 = x_1 x_3^2 - x_4 + x_2$$

s.t.

$$x_1^2 + x_2 \leq 6$$

$$x_1 + x_2 + x_4 \leq 2$$

We have :

a) class 1 = (x_1 , x_3 , x_4) ; class 2 = (x_2)
b) class 1 = (x_1 , x_2 , x_4) ; class 2 = (x_1 , x_2 , x_3 , x_4)

This " partition " procedure (p.6) allows us first, to stop computation whenever the result is judged satisfactory, and secondly to correct progressively the DM's aspiration levels. Moreover, if the problem does not have any feasible solution, the DM has to adjust his requirements.

Several computational methods have been proposed for characterizing Pareto optimal solutions. Among them, we consider the augmented weighted Tchebycheff metric and the ϵ-constraint method.

For finding the points in $S \subset R^3$ closest to F^*, we use the augmented weighted Tchebycheff metric defined by :

$$||| F^* - F |||_\infty = || F^* - F ||_\infty^\theta + \delta.\sum_{i=1,..,3} | f_i^* - f_i | \qquad (2)$$

where $|| F^* - F ||_\infty^\theta$ is the weighted Tchebycheff norm ; the word "augmented" is due to the $\delta.\sum_{i=1,..,3} | f_i^* - f_i |$ term (for more details, see Steuer 1986).

By definition, we have : $|| F^* - F ||_\infty^\theta = \max_{i=1,..,3} (\theta_i | f_i^* - f_i |)$.

The weights $\theta_i \geq 0$ are used to define different weighted Tchebycheff metrics (see § 3 step 2).

The program for finding the closest points to F^* according to the weighted Tchebycheff metric is : $\text{Min}_x \text{Max}_i (\theta_i | f_i^* - f_i(x) |)$ or

$$\begin{array}{l} \text{Min}_{\alpha,x} \quad (\alpha) \\ \text{s.t.} \\ \quad \alpha \geq \theta_i(f_i^* - f_i) \quad ; i=1,2,3 \\ \quad f_i(x) = f_i \quad ; i=1,2,3 \\ \quad x \in A \end{array} \qquad (3)$$

The augmented weighted Tchebycheff program is :

$$\begin{array}{l} \text{Min}_{\alpha,x} \quad (\alpha + \delta. \sum_{i=1,..,3} (f_i^* - f_i)) \\ \text{s.t.} \\ \quad \alpha \geq \theta_i(f_i^* - f_i) \quad ; i=1,2,3 \\ \quad f_i(x) = f_i \quad ; i=1,2,3 \\ \quad x \in A \end{array} \qquad (4)$$

Because $(f_i^* - f_i)$ will never be negative, we can drop the absolute value signs in (2). The reason for the $\delta . \sum_{i=1,..,3} (f_i^* - f_i)$ term is to give to the contour a " slight slope " (cf. Steuer, 1986). In practice, we set δ sufficiently small.

The ϵ-constraint method is defined by :

$$\begin{array}{l} \text{Max } f_j(x) \\ \text{s.t. } x \in A \\ \quad f_i(x) \geq \epsilon_i \quad ; i \neq j \text{ and } i,j = 1,2,3 \end{array} \qquad (5)$$

By solving (4) or (5), all the Pareto optimal solutions to the MMPs involving non-convexities can be (theoretically) obtained. But since the range of the ϵ_i is not known numerically before, there is a possibility that the ϵ-constraint method becomes unfeasible for some parameter ϵ_i. Moreover, in (5), the objective function f_j is specially treated, and thus no all objective functions are equally treated. Consequently, we propose an hybrid program :

$$\begin{array}{l} \text{Min}_{\alpha,x} \quad (\alpha + \delta . \sum_{i=1,..,3} (f_i^* - f_i)) \\ \text{s.t.} \\ \quad \alpha \geq \theta_i (f_i^* - f_i) \quad ; i = 1,2,3 \\ \quad f_i(x) \geq \epsilon_i \quad ; i = 1,2,3 \\ \quad f_i(x) \leq \beta_i \epsilon_i \quad ; \beta_i = (0,1) \\ \quad x \in A \end{array} \qquad (6)$$

so that we search the " closest point " to F^* within a restricted region in the outcome space. If $\beta_i = 0$, we eliminate the corresponding constraint for f_i.

It should be emphasized here that strong Pareto optimality for an optimal solution of the problems (4) to (6) is not guaranteed without the following assumption for the solution : " X^* is a strong efficient solution to the MMP (1) if and only if X^* is a unique global optimal solution to problem (4) , (5) or (6) ".

2.2 ATTEMPTS TO CALCULATE GLOBAL SOLUTION OF PROBLEM (6) :

The mathematical methods available for solving non-convex programming problems generally yield local solutions. Moreover, there is no local criterion to decide if a local solution is global. Note that several global criteria for a global solution have been proposed, but except for very special cases, there is no numerically feasible method for computing it (cf. Horst, 1988). Due to the enormous difficulties inherent in global optimization problems and the computational cost for solving them, the methods devised for solving these problems are quite diverse. The solution procedure we propose uses mainly the behaviour of the local solutions in order to compute the " global solution ". It is a deterministic method based on the two following heuristics H1 and H2 (see, for instance, Törn, 1983 for stochastic algorithm):

H1. we use the fact that MMP's are specific problems where,for example, the DM can arrange its outcomes in the specified interval and we suggest a deterministic method which is well suited for problems that are known to have an optimal solution on a certain subset of the boundary of A and S or, in other words, where certain constraints are active at the global optimum. We use the following property : " each local optimum is associated with a set D_k obtained by reversing the sign of the active constraints " (cf. Horst 1988) :

Max (F(x) / $x \in A \cap V_k$) = Min (F(x) / $x \in D_k$) where the set V_k is a neighbourhood of X_k (local solution), and $D_k = A \cap U_k$. U_k is obtained by reversing the sign of " certain " active constraints, more precisely the constraints concerning the objective value function.

We replace (6) by : Max ($\alpha + \delta . \sum_{i=1,..,3} (f_i^* - f_i)$)

$$\text{s.t. } x \in A \cap U_k \qquad (7)$$

$$\alpha \leq \theta_i (f_i^* - f_i) \quad ; i = 1,2,3$$

H2. We use the general idea of the elimination of local solutions, once they are obtained, by addition of constraints which eliminate them. We consider here the following problem which determines a " feasible " solution for (1) :

$$
\begin{cases}
x \in A \\
f_j(x) \geq f_j(X_j) - \lambda_j \quad ; \quad j=1,2,3 \\
a_i \leq x_i \leq b_i \quad ; \quad i=1,...,n
\end{cases}
\tag{8}
$$

where X_j is a local optimum of f_j.

The added constraint is the following : $l(x) = \sum_{j=1,..,3} w_j \, . \, \nabla^t f_j(x) \, .dj$, where :
(i) x is an initial point for (8) ;
(ii) the parameters w_j and d_j determine respectively the step size and the direction of movement.

$d = (d_1 , d_2 , d_3)^t$ is an improving direction defined by the DM or computer ($\sum_{j=1,..,3} . \nabla^t f_j(x) \, .dj > 0$) ; for example, the direction d can be defined as follows : $d = x - X^I$, or $d = x - X^k$, or $d = (1,1,1)^t$ (see p.5). In practice, the scalar w_j controls the step-size ; it can be increased or decreased following the values of the gradient.

This " linearization " is only accurate when d is " small ", so upper and lower bounds are imposed : $-s \leq d \leq s$ where s is a 3-vector with all positive components ; s is called " the step-bound " (cf. Palacios-Gomez and al. 1982).

Finally, we consider the following Non-Linear Program (NLP(x,s)) :

$$
\begin{cases}
x \in A \\
f_j(x) \geq f_j(X_j) - \lambda_j \quad ; j=1,2,3 \\
l(x) \geq \Delta \\
a_i \leq x_i \leq b_i \quad ; i=1,...,n \\
\max (a - x, -s) \leq d \leq \min (b - x, s)
\end{cases}
\tag{9}
$$

where λ_j is specified by the DM , and $\Delta > 0$ ensures that d is an improving direction.

The NLP (9) is solved, yielding a solution y. If the DM accepts y, the step bounds may be increased. Otherwise, s is reduced to s' and NLP(x,s') is solved. This idea stressed the need for reducing the step-bound near an optimum, to control oscillations and to force convergence. The solution of problem (9) becomes an " initial " solution to global analysis. Clearly, if y is feasible for (9), then x is rejected as " strong efficient solution " and consequently x is not a compromise solution. However, we do not forget that the problem of finding a feasible point in (9) is itself an optimization problem, and is subject to the same difficulties than the original one.

We suggest here to implement a module in the computer program (GRG) which generalizes the basic idea of the GRG method (cf. Abadie, 1978). This primal method handles directly the constraints. It generates (starting at x^0 - feasible or non-feasible point in the decision variables space) a search improving direction within the "linear tangent variety" and it determines a new point (Fig.3) in this direction. Usually, this point is not feasible. A second shift is then executed for obtaining a new feasible point (by projection).

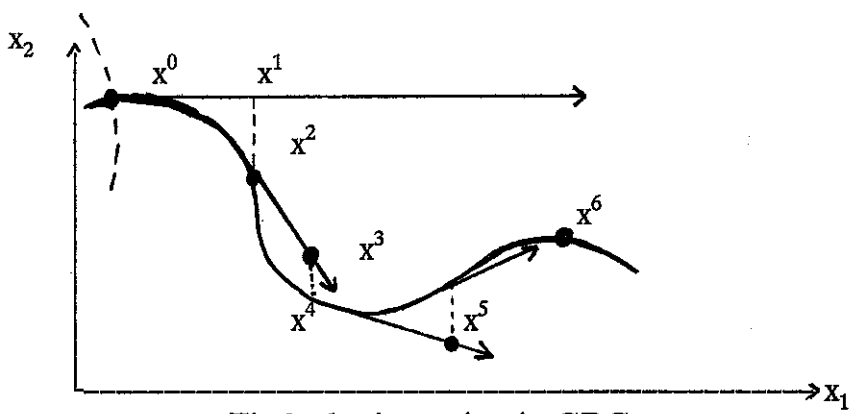


Fig.3 : basic version in GRG

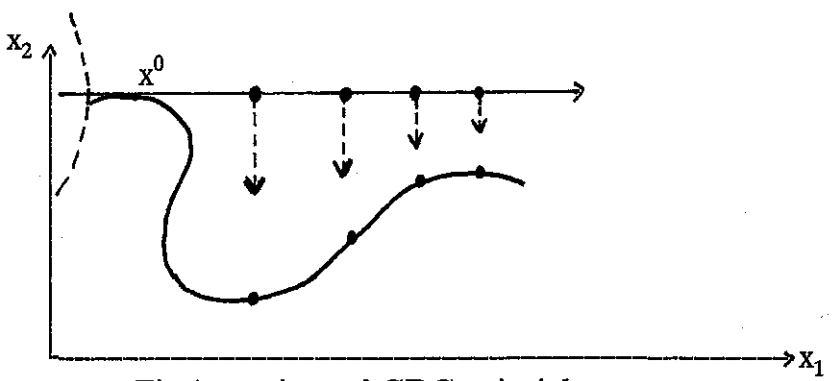


Fig.4 : variant of GRG principle

3. OUTLINE OF THE METHOD

The method involves the choice of an " reference point " and uses NLP procedure to find a suitably defined " closest point " . We use a reference point solution as a mechanism by which points localized hopefully in the neighbourhood of the final solution to the problem (1) may be generated, without requiring the DM to specify weights to find the closest point. The approach suggested here consists in searching nondominated solutions by moving from one nondominated solution to another (Fig.5 to 7).

Following the Hwang and Masud classification (1979), our technique requires a progressive articulation of preferences ; it involves an interactive process between the DM and the computer. More precisely, it involves a combination of two approaches : progressive articulation of preferences and a posteriori articulation of preferences (cf. Evans, 1984). The method is constructed in such a way that it generates a sequence of solutions, each solution preferred to its predecessor (Fig.5 to 7) :

Phase 1 : The algorithm is initiated typically through the finding of a " first " efficient point.

Phase 2 : The DM is then required to provide some information concerning his preferences structure over the multiple objectives, relative to the outcomes arising from this solution.

Phase 3 : Using this preference information, the algorithm sets up a new problem : the global optimality analysis.

If the current solution is accepted by the DM as a " best compromise solution " , then we stop ; otherwise, we ask the DM to define another " reference point". Iterations continue until the DM or the computer program decides that the current solution is " closed enough " to a best compromise. The idea is that the DM's selection of bounds on objectives realization can be interactively incorporated by constraints and step-bounds as new information. This step of the procedure, where we require the DM to determine the minimum and maximum values for each objective, at each iteration, is perhaps theoretically the most difficult, because it requires an ad hoc estimation of minimum acceptable realization (problems (6), (8), (9)).

The " feasible " solutions for the process are obtained by minimizing the distance from the " ideal point " according to the augmented weighted Tchebycheff norm.

Note that our algorithm can also be concerned with finding the efficient solutions to the problem (1).

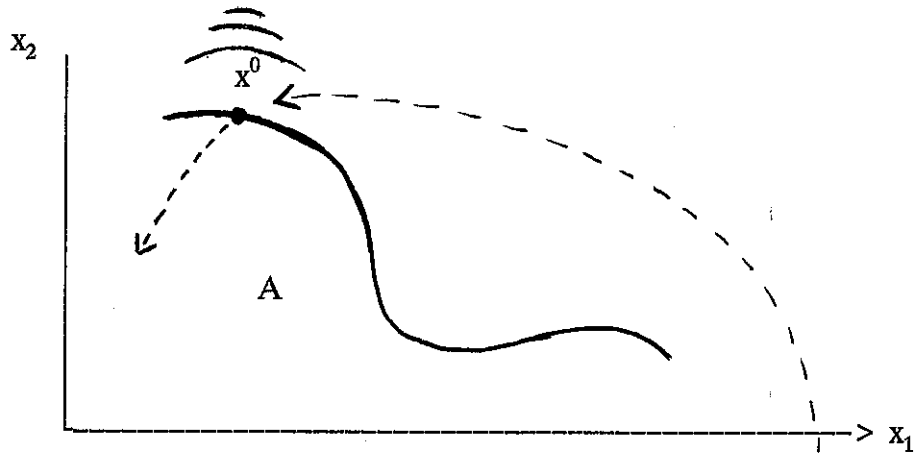


Fig.5 : global analysis in the decision variables space

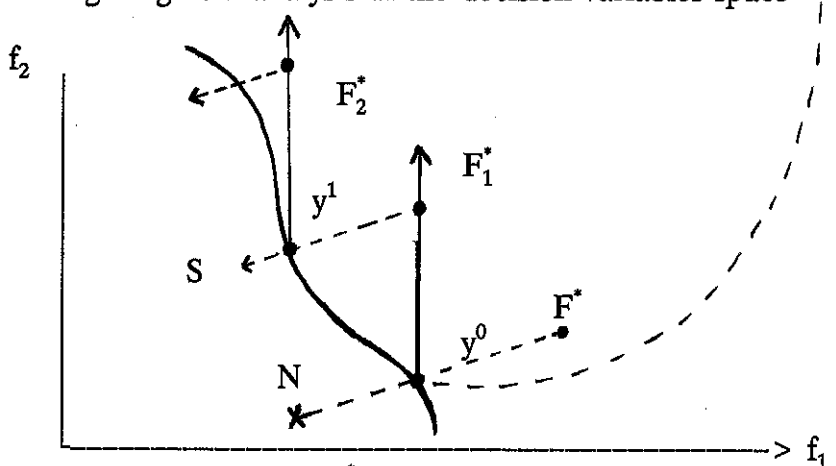


Fig.6 : ideal point F^* , and Nadir point are variables

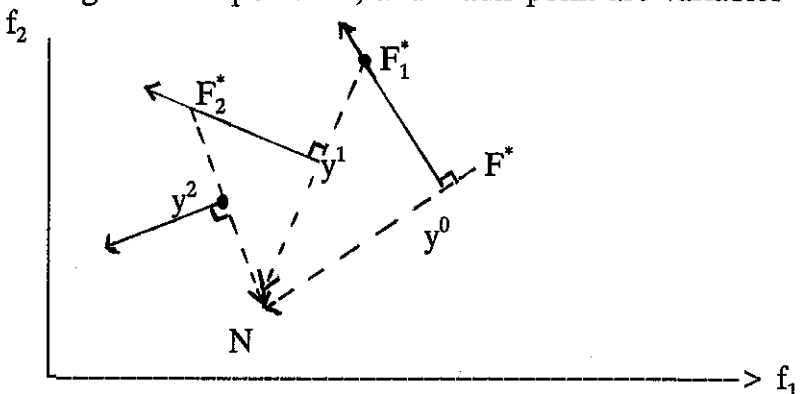


Fig.7 Nadir Point is constant

We now present the algorithm in a step-by-step way. It consists of sequential solution to a NLP problem. The resulting current solution is strongly efficient with respect to the set of feasible solutions.

Step 0 : Choose, for all objective functions f_i, an initial point x^0 ,feasible or non-feasible. Solve problem (Pi).
Let us denote by F^* the ideal point, $F^* = (f_1^*, f_2^*, f_3^*)$, where $f_i^* = f_i(X_i^*)$ and X_i^* is a optimal solution of f_i .

In order to reduce computing time, we avoid to optimize separately the objective functions f_i (i.e. (Pi)). We start from an optimal solution for one objective f_i in order to find an optimal solution to another. Various heuristics (for the choice of the objective order) enable us to reduce computing time.

Step 1 : Calculate the pay-off table of the MONLP problem and define the Nadir Point $N = (N_1, N_2, N_3)$, or another reference point. N_k is the smallest element in the k-th column of the pay-off table :

	f_1	f_2	f_3
X_1^*	f_1^*	$f_2(X_1^*)$	$f_3(X_1^*)$
X_2^*	$f_1(X_2^*)$	f_2^*	$f_3(X_2^*)$
X_3^*	$f_1(X_3^*)$	$f_2(X_3^*)$	f_3^*
N	N_1	N_2	N_3

Step 2 : Find the (first) strong efficient solution $y^0 = (y_1^0, y_2^0, y_3^0)$ closest to F^* in the sense of the augmented weighted Tchebycheff norm in the outcome space : Solve problem (6).

The weights $\theta_i > 0$ are chosen a priori to keep y^0 in the Nadir Point direction ; $1/\theta_i$ are direction coefficients of the straight line segment proceeding from F^* to the Nadir point (cf. Steuer, 1986, chap 14).

$$\theta_i = 1 / (f_i^* - N_i) \quad ; i=1,2,3$$

option : $f_i^* = f_i^* + \gamma_i$; where γ_i = 10% to 30% of f_i ; we now define f_i^* as $f_i^* := f_i^*$

The point y^0 becomes an initial point for generating another strong efficient point using a search parallel direction to (F^* - N), see Fig.6, or another, see Fig.7.

Step 3 : 3.1) If the DM prefers y^0 as a compromise solution, then go to step 4 ; if he does not like y^O , he may express a preference for another point (through its corresponding value in S) or his inability to choice a solution.

3.2) If he prefers at least one alternative to y^0, we save one such alternative, we denote it as F^*, and we go to step 1 ;

3.3) Otherwise, we apply rules (1) or (2) defined in § 2.1 (p.5) ; we denote the corresponding solution as F^* and we go to step 1.

Step 4 : (global analysis). A search procedure based on the problem (9) or (7) is used in order to decide whether y^0 is a global solution ; in this case, y^0 is declared as a " favourable " compromise solution.

4.1) Solve problem (9) or problem (7) ;

4.2) If the resulting solution is better than the previous one (in all objective value functions), it becomes the initial point for the next iteration ; we denote it as y^0 (y^0 is another compromise solution), and we go to step 3 ; the procedure continues until two successive solutions are " identical ".

4.3) If it does not exist another feasible efficient solution in the neighbourhood of y^0, we denote y^1 as the solution which minimizes the sum of unfeasibilities in (9). Ask the DM to choose between y^0 and y^1. If he chooses y^1 , we denote y^1 as F^* and we go to step 1. If he prefers y^0, we then stop ; if he is unable to choose, we affect y^0 - y^1 to y^0, and we go to step 1.

4. COMPUTER IMPLEMENTATION AND COMPUTATIONAL RESULTS.

This section describes a FORTRAN 77 implementation of our algorithm. Input to the system is in six parts :

1. a subroutine that computes the non-linear functions f_i for a given x ;
2. a subroutine that computes the non-linear functions g_j for a given x ;
3. a subroutine that computes the gradients ∇f_i for a given x ;
4. a subroutine that computes the Jacobian matrix $J=(\partial g_j / \partial x_i)$ for a given x ;
5. a file which contains all program parameters, options, initial values for x and step-bounds.
6. a file in special format, which specifies all other information about the problem ; this includes constant Jacobian elements, the position of non-constant elements, bounds on all variables, type of problem (max, min, " dense format", " sparse format "). For problems where there are few or no linear structural variables, the Jacobian matrix is said " dense " ; otherwise, the problem is called " sparse ". Note that Jacobian and gradient elements may be computed automatically, or by (optional) user supplied subroutines. The system stores the constraints matrix in a standard packed form (only non-zero elements stored). As the various input files are read in, the multiple options and values parameters are computed and printed out in a specified file.

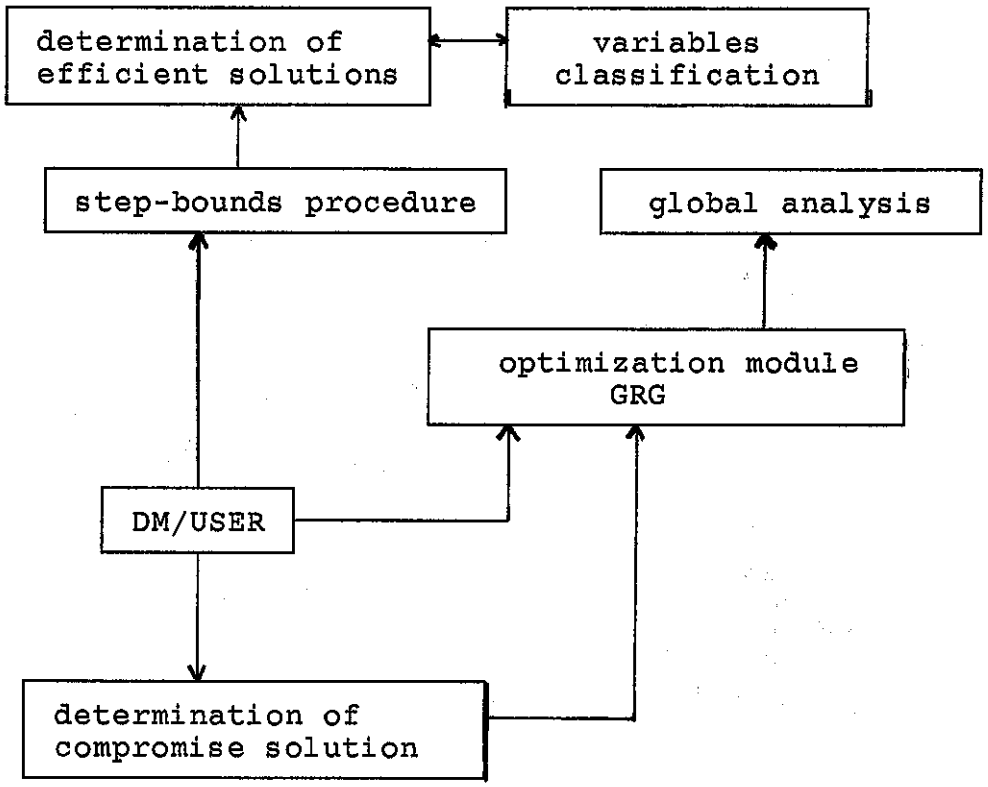
This method is operational on a micro-computer PC XT/AT . It seems particularly adapted as a part of a Decision Support System (DSS).

The code (GRG + ϵ-constraint + Tchebycheff metric + compromise search procedure) is written in a modular form. At this time, we attempt to redefine the GRG code structure in order to introduce new modules (sparse LU factorization,...) and to allow the DM to use its proper procedures.

This modular form involves several parts : an optimization module ; a decision-aid module for problem evaluation ; a graphic-aid module. Moreover, we define an interactive program for guiding the DM :

. it displays menus describing the current process ;
. it systematically controls all the codes and data ;
. it saves results (data, parameters, options, and any information) on the sequential files.

The software developed is as follows :



All runs were made on PC XT/AT computer, using extended precision (coprocessor is recommended). Initial values of the step-bounds are determined by the following rules :

1. the bounds are selected by the user ;
2. the initial bounds are computed by :
   $(s_0)_i = \max (\delta_1 . |(x_0)_i|, \delta_2)$, where $(s_0)_i$ and $(x_0)_i$ are the i-th components of s_0 and x_0, respectively.
3. the initial bounds are computed by :
   $(s_0)_i = \max (\delta_1 . (b_i - a_i) , \delta_3) ; \delta_1, \delta_2, \delta_3 = 10^{-3}$ to 10^{-6}

All problems tested were highly non-linear, and were input using the " dense " format. The reasons for termination are given in various " stopping criterion " : $||dj||_\infty < 0.001$; $||s_j|| < 0.001$; exceeded iteration limit ;..., without the stopping criteria due to the GRG code :

Problem	Number of variables			Number of Constr.	Average comput. time	Ref.
	Nonlinear	Linear	Total			
1	21	-	21	20	2′ 30′′	2. 10 15
2	30	-	30	30	5′	8. 17 18. 22.
3	40	-	40	11	1′	9. 10 18.
4	82	-	82	37	> 10′	15. 18.
5	86	15	101	77	> 12′	6. 7. 9. 18

The average computational time (with a sufficient weak error tolerance) on test problems 1 to 5 is the CPU time for obtaining about 10 efficient solutions.

5. CONCLUSION

There are several factors that imply an increase in future NLP applications (cf. GOICOECHEA and al. 1982). The results of this study are only preliminary ones and require further developments. There are probably many improvements which can be made in the MONLP strategy, specially the way of handling the unfeasible solution problem.

The methodology developed here enables us first, to build a description of a complex system, and second to provide decision-aid by setting up several evaluation scenarios, based on the different proposed alternatives. The main advantage of this approach is that the DM owns some information necessary for finding the solution (so, he can use the information contained in the dual variables values, which have specific meanings in economics and which are obtained as a by-product of GRG). Let us note however, that our analysis is only concerned with quantitative objectives ; moreover, it shows that the MONLP's can be solved and a compromise solution can be found.

Some extensions which one really needs are :
- parallelization of several optimization procedures ;
- communication between various components (modules) ;
- use of graphics to represent any solution in A or S (form which is familar to the DM) .

On the other hand, in the basic version of the method developed in § 3, a new constraint is added at each iteration to the set of existing constraints. It is important to develop methods that allow to drop certain constraints in order to decrease the size of the problem (1).

Results on rate of convergence concerning the acceptation of an efficient solution as globally optimal would also be useful. This rate is probably linear and depends on : number of degrees of freedom in the optimal solution, and how the step-bounds are reduced. It could be also interesting to perform some sensitivity analysis in step 3 depending on some imprecision of the parameters which define certain constraints.

References

[1] Abadie, J., " The GRG method for nonlinear programming ", in H.H. Greenberg (ed), Sijthoff & Noordhoff, Netherlands, 1978, p. 335-362.

[2] Abadie, J., and M'silti, A., " La programmation mathématique multicritère et la gestion des ressources en eau ", RAIRO, vol 22, n° 4, 1988, p.363-385.

[3] Arrow, K.J., and Hahn, F.H., " General competitive analysis ", Holden-Day, San-Francisco, 1971.

[4] Evans, G.W., " An overview of techniques for solving multiobjective mathematical programs ", Management Science, vol. 30, No 11, Nov 1984.

[5] Geoffrion, A.M., " Proper efficiency and the theory of vector maximization ", Journal of Mathematical Analysis and Applications, vol 22, n°3, 1968, p. 618-630.

[6] Graves G.W , Hatfield G.B and Whinston A.B, " Mathematical programming for Regional Water Quality management ", Water Resources Research, 8(2), 1972, p. 273-290.

[7] Graves G.W, Hatfield G.B and Whinston A.B, " Water Pollution control using by-pass piping " , Water Resources Research, 5(1), 1969, p. 13-47.

[8] Haimes Y.Y, Foley J.W and Yu W., " Computational results for water pollution taxation using multilevel approach ", Water Ressources Bulletin, 8(4), 1972, p. 761-772.

[9] Haimes Y.Y, Kaplan M.A and Husar M.A, " A multilevel approach to determining optimal taxation for the abatement of water pollution ", Water Resources Research, 8(4), 1972, p. 851-860.

[10] Himmelblau D.M., " Optimization of a Water Resource Systems by Nonlinear Programming ", Engineering Optimization, vol 2., 1977, p. 229-238.

[11] Horst, R.," Deterministic methods in constrained global optimization : some recent advances and new fields of applications" . Communication presented at EURO IX TIMS XXVIII, Paris, July 6.8, 1988.

[12] Hwang, C.L., and Masud, A.S., " Multiple objective decision making, methods and applications, Springer Verlag, Berlin, 1979, 351p.

[13] Jacquet-Lagrèze E., Meziani R. and Slowinski R. , " MOLP with an interactive assessment of a piecewise linear utility function ", European Journal of Operational Research , 31, 1987, p. 350-357.

[14] Linweaver F.P and Clarck C.S. " Costs of water transmission " , J. of the American Water Works Association, vol 56, 1964, p. 1519-1560.

[15] Lowe T.J, " On efficient solutions to multiple objective mathematical programs ", Management Science, Vol. 30, No.11, 1984.

[16] Mumford W, Czapski U. and Felow M. , " Dissipation of heat from a thermal effluent " , Water Resources Research, vol 9(4), 1973, p. 837-850.

[17] M'silti H. " Application de la Programmation Mathématique Multicritère et Nonlinéaire à la gestion des ressources en eau ", Université de Paris-Dauphine, thèse de 3^{e} cycle, 1984.

[18] Palacio-Gomez F., Lasdon L. and Engquist M., " Nonlinear Optimization by successive linear programming " , Management Science, vol 28, No 10, 1982.

[19] Steuer, R.E., " Multiple criteria optimization : theory, computation and applications ", John Wiley, 1986.

[20] Törn A., " A sampling-search clustering approach for solving scalar (local and global) and vector optimization problems ", Theory and Practice of MCDM, C. Carlson and Y.Kochetov (editors), North-Holland, 1983, p.119-142.

[21] Yotsukura N., Jackman A.P and Faust C.R, " Approximation of heat exchange at the air-water interface ", Water Resources Research, vol 9(1), 1973, p.118-128.

[22] White D.J. " Optimality and Efficiency " , J. Wiley and Sons, 1982.