International Journal For Technological Research In Engineering
Volume 7, Issue 6, February-2020 *ISSN (Online): 2347 - 4718*

CLASSICAL METHODS OF MULTI-OBJECTIVE OPTIMIZATION - A COMPARATIVE STUDY

Divesh Kumar Rohilla
Assistant professor
Mechanical Engineering, BK Birala Institute of Engineering &Technology, Pilani

Abstract: ***this research paper to qualitatively compare four popular classical methods of multi-objective optimization in terms of various performances matrices that represents solution quality and numerical efficiency. Two test problems were selected that cover a wide range of optimization problem types constrained vs. unconstrained, equality vs. inequality constraints and bounded vs. unbounded. One mathematical problem and one practical problem were chosen. Performance matrices chosen were the total number of function evaluations and variance in results. This work was motivated by the lack of clear comparison between the algorithms. As it will be shown, some methods are useful only if some conditions are met. For example, the weighted sum method is one of the most used methods. It fundamentally cannot represent non-convex regions, and has been found to be unable to represent well distributed solutions. ε-constraint method overcomes weaknesses of this method. For all algorithms, a built in MATLAB subroutine is used, and this made the computational implementation simpler***
Keywords: MATLAB, MOOP, SQP

I. INTRODUCTION

Optimization is the task of finding one or more solutions which correspond to minimizing (or maximizing) one or more specified objectives and which satisfy all constraints (if any) and design variables. Design variables are variables that a designer or engineer can freely choose, for example the thickness, material, width and the length of a part. The resulting stress, volume, natural frequency, free height and other performance measures are often considered either as objective functions or constraints. Objective functions are the system response that we wish to minimize, while constraints are limits that we impose on the system.A single-objective optimization problem involves a single objective function and usually results in a single solution, called an optimal solution. On the other hand, a multi-objective optimization task considers several conflicting objectives simultaneously. Global business environment challenges companies to deliver high quality products at low costs under shorter development lead- times. The challenge to produce the desired product first time and every time undeniably underlines the importance of product design and development function. Many in industrial sector accept the notion that more than 70% of the final product quality and cost are determined at the early design stage of a product development process. Moreover, this stage provides a great leverage to engineers to decide about product quality, reliability, cost, and other customer requirements in an environment composed of limited resources, divergent interests and annoying priorities. These multiple criteria need to be optimized for better trade-off at initial stage and following to which helps to make a successful product. Failing to address the multi-criteria issues often leads to wrong design, which in turn results in poor performance, premature failure, high cost, and customer dissatisfaction

This research focused on area of multi-objective optimization that was found to be lacking in literature. No clear comparison of classical methods of multi-objective optimization has been done looking at different types of algorithms, using quantitative performance measures. New designers wishing to use multi-objective optimization techniques are flooded with available solving methods, each clamming to be the superior algorithms.

There are lots of multi-objective optimization algorithms available in literature. Many methods produce a set of pareto solutions that demonstrate the trade-off between the objective functions. Other methods produce a single pareto solution, where the user's preferences are considered beforehand. A primary goal of multi-objective optimization is to model a decision maker's preferences (ordering or relative importance of objectives and goals), methods are categorized depending on how the decision maker articulates these preferences, (i) methods that involve a priori articulation of preferences, which implies that the user indicates the relative importance of the objective functions or desired goal before running the optimization algorithms, (ii) methods with a posteriori articulation of preferences, which entail selecting a single solution from a set of equivalent solutions [12]. Advantages and disadvantages of different methods are discussed throughout the paper. The methods compared all have their known strength and drawbacks.

This work was motivated by the lack of clear comparison between the algorithms. As it will be shown, some methods are useful only if some conditions are met. For example, the weighted sum method is one of the most used methods. It fundamentally cannot represent non-convex regions, and has been found to be unable to represent well distributed solutions. ε-constraint method overcomes weaknesses of this method.

For all algorithms, a built in MATLAB subroutine is used, and this made the computational implementation simpler.

the multi objective optimization algorithms that were compared. Five algorithms are presented: general multi-objective optimization method, the weighted sum method, the ϵ-constraints method, the goal programming method, and sequential quadratic programming.

For the four deterministic methods, the anchor points were automatically calculated before the multi-objective optimization portion of the algorithms was run. This eliminated the need to define the normalizing factors.

II. WEIGHTED SUM METHOD

The weighted sum method, as the name suggests, scalarizes a set of objective into a single objective by pre-multiplying each objective with a user supplied weight. This method is the simplest approach and is probably the most widely used classical approach. Faced with multiple objectives, this method is the most convenient one that comes in mind. For example, if we are faced with the two objectives of minimizing the cost of a product and minimizing the amount of wasted material in the process of fabricating the product, one naturally thinks of minimizing a weighted sum of these two objectives [18].

The weight of an objective is usually chosen in proportion to the objective's relative importance in the problem. For example, in the above mentioned two objective minimization problems, the cost of the product may be more important than the amount of wasted material. Thus, the user can set a higher weight for the material cost than for the amount of wasted material. However, setting up an appropriate weight vector also depends on the scaling of each objective function. We suppose that the weighting coefficients w_m are real numbers such that $w_m \geq 0$ for all $m = 1, 2. . . M$. it is also supposed that the weights are normalized, that is, $\sum_{m=1}^{M} w_m = 1$.

After the objectives are normalized, a composite objective function F(x) can be formed by summing the weighted normalized objectives and the multi-objective optimization problem is then converted to a single objective optimization problem as follows:

Minimize $$F(x) = \sum_{m=1}^{M} w_m f_m(x),$$

Subject to

$$g_j(x) \geq 0,$$
$$h_k(x) = 0,$$
$$x_i^{(L)} \leq x_i \leq x_i^{(U)},$$
$$j = 1, 2, ...J;$$
$$k = 1, 2, ...K;$$
$$i = 1, 2, ...n.$$

Where $f_m(x)$ is the normalized value of the m^{th} objective function, and M is the number of objective functions. The sub-problem is repeated for a number of different weights, and only one solution is obtained for each weight w_m. Of all multi-objective optimization methods, the weighted sum method is often the least computationally expensive. However, equally spaced weights do not guarantee equally spaced solutions in the objective space and the method can only represent well defined convex regions of a Pareto front. However, there are a number of difficulties with this approach. In handling mixed optimization problems, such as those with some objectives of the maximization type and some of the minimization type, all objective have to be converted into one type. Although different conversion procedures can be adopted, the duality principle is convenient and does not introduce any additional complexity. In most nonlinear multi-objective optimization problems, a uniformly distributed set of weight vectors need not find a uniformly distributed set of pareto-optimal solutions. Since this mapping is not usually known, it becomes difficult to set the weight vectors to obtain a pareto-optimal solution in a desired region in the objective space.

Despite the many methods for determining weights, a satisfactory, a priori selection of weights do not necessarily guarantee that the final solution will be acceptable; one may have to resolve the problem with new weights. In fact, weights must be functions of the original objectives, not constants, in order for a weighted sum to mimic a preference function accurately.

III. CONSTRAINT METHOD

In order to overcome the difficulties faced by the weighted sum approach in solving problems having non-convex objective spaces, the ε-constraint method is used. Haimeset al. (1971) suggested reformulating the multi-objective optimization problem by just keeping one of the objectives and restricting the rest of the objectives within user-specified values. The modified problem is as follows:

Minimize $f_\mu(x)$,

Subject to

$$f_m(x) \leq \varepsilon_m,$$
$$g_j(x) \geq 0,$$
$$h_k(x) = 0,$$
$$x_i^{(L)} \leq x_i \leq x_i^{(U)},$$
$$m = 1, 2, ..., M$$
$$j = 1, 2, ..., J;$$
$$k = 1, 2, ..., K;$$
$$i = 1, 2, ..., n.$$
$$m \neq \mu;$$

In the above formulation, the parameter ε_m represents an upper bound of the value of f_m and need not necessarily mean a small value close to zero. Different pareto-optimal solutions can be found by using different ε_m values. The same method can also be used for problems having convex or non-convex objective spaces alike. In terms of the information needed from the user, this algorithm is similar to the weighted sum approach. In the latter approach, a weight

vector representing the relative importance of each objective is needed. In this approach, a vector of ε values representing, in some sense, the location of the pareto-optimal solution is needed. However, the advantage of this method is that it can be used for any arbitrary problem with either convex or non-convex objective space.

The solution to the problem stated largely depends on the chosen ε vector. It must be chosen so that it lies within the minimum or maximum values of the individual objective function. Moreover, as the number of objectives increases, there exist more elements in the ε vector, thereby requiring more information from the user.

Goal Programming:

The ideas of goal programming were originally introduced in charnes etal. (1955), but the term goal programming was fixed in charnes and cooper (1961). It is one of the first methods expressly created for multi-objective optimization. The basic idea in goal programming is that the decision maker specifies (optimistic) aspiration levels for the objective functions and any deviations from these aspiration levels are minimized. An objective function jointly with an aspiration level forms a goal. We can say that , for example, minimizing the price of a product is an objective function, but if we want the price to be less than 500 dollars, it is a goal (and if the price must be less than 500 dollars, it is a constraint). The aspiration levels are assumed to be selected so that they are not achievable simultaneously.

It is worth noticing that the goals are of the same form as the constraints of the problem. This is why the constraints may be regarded as a subset of the goals. This way of formulating the problem is called generalized goal programming. After the aspiration levels have been specified, the following task is to minimize the under and overachievements of the objective function values with respect to the aspiration levels. We now have the multi-objective optimization problem in a form where we can minimize the deviational variables.

There are two methods for solving goal program: The weights method forms a single objective function consisting of the weighted sum of the goals and the preemptive methods optimizes the goals one at a time starting with the highest-priority goal and terminating with the lowest, never degrading the quality of a higher priority goal. Here we consider only weights method.

Goal programming problem is as follows:

Suppose that the goal programming model has n goals and that the i^{th} goal is given as

Minimize $Z = G_i, \quad i = 1, 2, \ldots, n.$

The combined objective function used in the weights method is then defined as:

Minimize $Z = w_1G_1 + w_2G_2 + \ldots\ldots + w_nG_n$

Subject to

$$f_i(x_i) - d_i^+ + d_i^- = 0.$$

Where d_i^- = negative deviation from i^{th} goal (underachievement), and

d_i^+ = positive deviation from i^{th} goal (overachievement),

$G_i = (d_i^- + d_i^+).$

Goal programming is considered as powerful technique for solving multi-objective optimization and its application has been greatly enhanced in recent years. It is being applied in various functional areas, including academic planning, financial planning, hospital administration, media solution, aggregate production planning, manpower planning, etc. many researchers who solves design problems with conflicting objective have found applying goal programming. All goal programming formulations are minimization models. The distinguishing feature of goal programming is its capability of handling a number of goals with different priorities simultaneously in the objective function. The goals are satisfied in an ordinal sequence.

Despite its popularity and wide range of applications, there is no guarantee that it provides a Pareto optimal solution. In addition, goal programming has additional variables and nonlinear equality constraints, both of which can be troublesome with larger problems. A weighted goal programming constitutes a subclass of goal programming, in which weights are assigned to the deviation of each objective from its perspective goal. The preemptive goal programming approach is similar to the lexicographic method in that the deviations $|d_j| = d^+_j + d^-_j$ for the objectives are ordered in terms of priority and minimized lexicographically. Weighted goal programming and preemptive goal programming provide Pareto optimal solutions if the goals form a Pareto optimal point or if all deviation variables, d^+_j for functions being increased and d^-_j for functions being reduced, have positive values at the optimum [15]. The latter condition suggests that all of the goals must be unattainable. Generally, however, Weighted and preemptive goal programming can result in non-Pareto optimal solutions.

Sequential Quadratic Programming (SQP):

This is the most widely used algorithm for non-linearly constrained optimisation. The Karush-Kuhn-Tucker conditions are en-forced in an iterative manner. An approximate Quadratic Programming sub-problem is solved at major iteration. The solution to the QP problem gives a search direction. Using the search direction a line search is carried out. At major iteration an approximation of the Hessian is updated using the BFGS method. The Quadratic Programming problem solved at iteration of SQP is an approximation of the original problem with linear constraints and quadratic objective:

$$\min_d f(x_k) + \nabla f(x_k)^T d + \frac{1}{2} d^T \nabla^2 L(x_k) d$$

Subject to

$$g_i(x_k) + \nabla g_i(x_k)^T d \le 0;$$
$$h_i(x_k) + \nabla h_i(x_k)^T d = 0.$$

Where the Lagrangian function L is defined as:

$$L(x) \; @f(x) + \lambda^T h(x) + \mu^T g(x)$$

Where λ is a vector of approximate Lagrange multipliers and μ is a vector of approximate KKT multipliers.

Instead of using the true Hessian of the Lagrangian, $\nabla^2 L(x)$ is replaced by H_k, which is the BFGS approximation of the Hessian:

$$H_{k+1} = H_k + \frac{q_k q_k^T}{q_k^T p_k} - \frac{H_k p_k p_k^T H_k}{p_k^T H_k p_k}$$

Where:

$$p_k = x_{k+1} - x_k,$$
$$q_k = \nabla f(x_{k+1}) + \lambda^T \nabla h(x_{k+1}) - (\nabla f(x_k) + \lambda^T \nabla h(x_k))$$

The solution to the QP sub-problem produces a search direction vector d_k, which is used to form a new iterate x_{k+1}.

$$x_{k+1} = x_k + \alpha d_k$$

A line search is carried out to choose α such that the following penalty function (also called merit function) is minimised:

$$\psi(x) = f(x) + \rho\left[\sum_{i=1}^{p}|h_i(x)| + \sum_{i=1}^{m}\max\{0, g_i(x)\}\right]$$

where ρ is a penalty factor. This line search sub-problem helps to improve the convergence of the algorithms and it balances the sometimes conflicting objectives of reducing the objective function and of satisfying the constraints.

SQP Algorithm:
Step.1. Given the current iterate x_k, and a current approximate Hessian H_k, solve the QP sub problem to obtain the search direction d_k. Notice that the solution of the QP sub problem provides estimates of the multipliers λ and μ.
Step.2. Given d_k, solve the line search problem to minimise the merit function $\psi(x)$ and then find the next iterate x_{k+1}.
Step.3. Update the Hessian approximation H_{k+1} using the BFGS formula.
Step.4. Check if the convergence criterions satisfied, if not set $k = k+1$ and go back to *Step 1*.
Sequential quadratic programming (SQP) methods tried to solve a nonlinear program directly instead of converting it to a sequence of unconstrained problems. The basic approach is same to Newton's method for unconstrained minimization: A local model of the optimization problem is constructed and solved at each step, approaching a step towards the solution of the original problem. In unconstrained minimization, the local model is quadratic and the objective function must be approximated. A quadratic model for the objective function and a linear model of the constraint are used in an SQP. A nonlinear programming in which the objective function is quadratic and the constraints are linear is called a quadratic programming (QP). An QP is solved at every iteration by an SQP method.

IV. RESULTS AND COMPARISON

The objective of this research was to perform a comprehensive comparison between multi-objective optimization methods on both mathematical and practical problems, determine the efficiency of each method and to determine which method is appropriate for which type of problem. In order to assess the efficiency and effectiveness of each method, two problems were chosen from the literature. The test problem set represents various types of optimization (equality and inequality constraints, bounded and unbounded) [11].

4.1 Test Problem 1

The first problem was proposed by [11]. The problem contains two equality constraints with five unbounded design variables, as follows:

$$J_1 = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2$$

Minimize $J_2 = 3x_1 + 2x_2 - \frac{x_3}{3} + 0.01(x_4 - x_5)^3$

Subject to

$$x_1 + 2x_2 - x_3 - 0.5x_4 + x_5 = 2$$
$$4x_1 - 2x_2 + 0.8x_3 + 0.6x_4 + 0.5x_5^2 = 0$$
$$x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 \le 10 \qquad i =$$
$$3x_1 + 2x_2 - \frac{x_3}{3} + 0.01(x_4 - x_5)^3 = 0$$
$$-\infty \le x_i \le \infty$$

1, 2, …., 5.

4.2 Test Problem 2

To demonstrate the proposed approach, the study considers leaf spring design. In order to have a better understanding a brief description suspension is discussed first. The leaf spring suspension is a popular choice for the rear of the trucks and some utility vehicles. It has been used for some heavy duty trucks front suspensions and the rear of the passenger cars. It is made from the layers of the spring steel bolted together through the centre of the leaves. It has several leaves – simply adding leaves increases the load carrying capacity of the suspension.
The top leaf typically is the longest leaf and each end of the leaf formed into an eye, into which rubber bushing is installed. The spring eyes are bolted to the chassis in front and attached at the rear through a hinge joint called a shackle.
The shackle permits the spring to effectively change its length as it flexes to absorb the impacts. The leaf spring also attaches through U-bolts to the solid rear axle. So it locates the axle housing in both lateral and fore-aft position, no

additional linkage is necessary, an important function. This permits a simple suspension design with obvious packaging benefits.

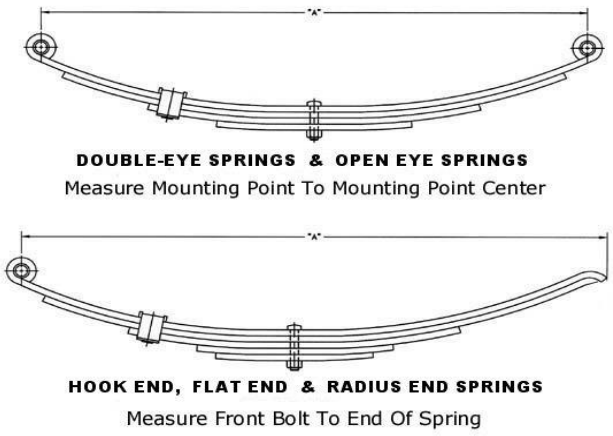

Fig.4.1. Leaf Spring

Leaf spring is a flexible element suspension system, which is able to store the energy applied to it in the forms of loads and displacements. It is designated to absorb bumps and irregularities of road surface, which are transmitted from wheels, tires and axles. Further, the spring allows the suspension arms, links and axles to travel in vertical plane, a function called wheel travel. The movement leaf spring is termed as oscillation and the stiffness of the spring is called spring rate. Basically, higher the spring rate, the stiffer a vehicle will ride. Softer spring rate and longer wheel travel leads to a smooth ride. The length, width, thickness of the leaves and severity of arch are all physical factors, which affect the spring rate and in turn the spring performance.

4.2.1 Design for Leaf Spring:

There are different types of leaf springs as: elliptic, semi-elliptic, flat cantilever etc. The semi-elliptical springs are the most common variety. It can be considered as to cantilevers. Leaf springs are initially curved and straighten under load. Such an arrangement decreases the stresses as an assembled leaf spring is subjected to preliminary deformation opposite to that caused by the forces acting upon it during operation in a machine. The free end of the spring is modified for the attachment of the load. The practical design of the leaf spring is based on the assumption of a beam of uniform strength. The design expressions for semi-elliptic type of leaf spring are as given below:

1. Stress: The springs are small elastic beam assemblies loaded under bending stresses. The expression for stress is given as:

$$Stress = \frac{3 \times P \times l}{2 \times n \times b \times t^2} \quad \text{(MPa)}$$

During the cycle of absorbing and storing energy the stress in the spring must not exceed a certain maximum value in order to avoid settling or premature failure.

2. Spring Rate: In design of leaf spring, the spring rate plays an important role. Spring rate in simplest sense is defined as the measurement of the load required to compress the spring at a given distance. The expression for spring rate is given as:

$$\text{Spring Rate} = \frac{8 \times E \times n \times b \times t^3 \times stiff.factor}{3 \times l^2} \quad \text{(Nm)}$$

It is a gauge of spring strength and flexibility "X" amount of rate is required to support "X" amount of weight. For example, if it takes 100 pounds to compress a spring one inch, it would take to 200 pounds to compress the spring two inches. Basically, higher the spring rate, the stiffer the vehicle will ride and lower the spring rate, the softer will be the vehicle ride. The spring rate is influenced by several factors including material grades, leaf thickness, leaf lengths, leaf end configuration, shackle angle, number of leaves and internal friction.

3. Free Height: The free height dictates the arch of the leaf spring, which creates added strength and recoil action. The expression for free height is given as:

$$\text{Free Height} = \frac{l^2}{8 \times R_0} \quad \text{(m)}$$

4. Weight: Leaf spring contributes about 10 – 20% of the un-sprung weight (the weight, which is not supported by the suspension system). Hence even a small reduction in the weight of the leaf spring will lead to improvements in the passenger comfort, vehicle cost and increased fuel efficiency. The expression for weight is given as:

$$Weight = \delta \times n \times l \times b \times t \quad \text{(N)}$$

Where n is number of leaves, l denotes length, b represents breadth, t denotes thickness of each leaf, and R_0 is radius of curvature of leaf spring. Other nomenclatures include δ as material density = $78*10^3$ N/m^3, P is load on spring = 4500 N, E as elastic modulus = $2.10*10^3$ MPa. The Stiffness factor value is assumed to be equal to 1. The above design equations clearly indicate that they are mainly influenced by the control parameters such as length, breadth, thickness, number of leaves and radius of curvature.

For a specific automotive vehicle, the range of these control parameters is selected from the manufacturer's catalogue and is presented in table:

Table.4.1. Range of Design Variables

S.N	Control Parameters	Range
1.	Number of Leaves (n)	6
2.	Length (l)	$700 \le l \le 3000$ mm
3.	Breadth (b)	$40 \le b \le 120$ mm
4.	Thickness (t)	$6 \le t \le 25$ mm
5.	Radius of Curvature (R_0)	350 mm

Less stress and softer spring rate is preferred for smooth ride and longer spring life. From the design expressions for stress and spring rate it is observed that for minimizing stress the values for number of leaves *n*, breadth *b* and thickness *t* must be as large as possible and value of length *l* must be as small as possible. However, for softer spring rate value of length *l* should be as large as possible and values of number of leaves *n*, breadth *b* and thickness *t* should be as small as possible. Therefore, these decision variables have opposite effect on stress spring rate. Hence it is clear that the requirements of stress and spring rate are not commensurate and are therefore conflicting. There are tradeoffs in the sense that sacrificing requirement of one goal will tend to produce greater return on others. In this situation where all the performance characteristics are to be improved simultaneously the problem has to be formulated as multi-objective optimization problem [13].

4.2.3 Multi-Objective Optimization Model:
Multi-objective optimization model is formulated to obtained optimal values of the control parameters. The four quality characteristics are considered as objective functions and as constraints with target values of each.

Minimize

$$F_1 = n \times \delta \times l \times b \times t$$

$$F_2 = \frac{l^2}{8 \times R_0}$$

$$F_3 = \frac{8 \times n \times St \times E \times b \times t^3}{3 \times l^2}$$

$$F_4 = \frac{3 \times P \times l}{2 \times n \times b \times t^2}$$

Subject to:

$$n \times \delta \times l \times b \times t \leq 83.63$$

$$\frac{l^2}{8 \times R_0} \leq 0.10795$$

$$\frac{8 \times n \times St \times E \times b \times t^3}{3 \times l^2} \leq 251.72$$

$$\frac{3 \times P \times l}{2 \times n \times b \times t^2} \leq 895.56$$

$$0.7 \leq l \leq 3.0$$

$$0.04 \leq b \leq 0.120$$

$$0.006 \leq t \leq 0.025$$

$$n = 6$$

$$R_0 = 0.7$$

Each quality characteristics is assigned a target value. The target value for each quality characteristic is obtained by solving them individually as a single objective optimization model [13].

Numerical Result:
The objective of this research was to perform a comprehensive comparison between multi-objective methods on both mathematical and practical problems, to determine the efficiency of each method and to determine which method is appropriate for which type of problem. In order to assess the efficiency and effectiveness of each method, two problems were chosen from the literature. The test problem set represents various types of optimization (equality and inequality constraints, bounded and unbounded).
Four numerical performance metrics and one visual criterion were chosen for quantitative and qualitative comparisons: (1) function count, (2) current step, (3) current point, (4) function value, (5) graphical representation of results.
For consistency, the same parameter values of the MATLAB optimization function "*fminimax*" were used in all optimization algorithms and the default values were used for all optimization parameters. This section shows the numerical results by all algorithms for all test problems.

V. CONCLUSION

This paper As has been stressed many times thus far, a large variety of methods exists for multi-objective optimization problems and none of them can be claimed to be superior to the others in every aspect. Selecting a multi objective method is a problem with multi objective itself. Thus some matters of comparison and selection between the methods are worth considering. The theoretical properties of the methods can rather easily be compared. However, in addition to theoretical properties, practical applicability is hard to determine without experience and experimentation.
More fruitful information relating to the question of method selection would likely emerge if computational applications were more extensively reported. Unfortunately not too many actual computational applications of multi-objective optimization techniques have been published. Instead, methods have mainly been presented without computational experiences or with simple academic test problems. Most of the applications presented are merely proposals for applications or they deal with highly idealized problems for most multi-objective optimization methods a natural reason is the difficulty in finding with real decision makers. A complicating fact is also the enormous diversity of decision makers.
One more thing to keep in mind is that for the most part only successful applications are published. This means that a complete picture of the applicability of methods cannot be drawn, on the basis of the experiences reported. The evident lack of benchmarking type test problems for non linear multi-objective optimization complicates the comparison of different methods. Naturally, some methods are useful for some problems and other methods for other types of problems. However, bench mark problems could be used to point out such behavior.
Each of four methods showed their own strength and

weakness. For a convex pareto front, the weighted sum method gave an idea of the pareto front with minimal computational burden; however, it could not find any solutions on non-convex resigns. The weighted sum algorithm was the simplest one of the primary reasons for its frequent use in the various fields of multi-objective optimization.

Different pareto optimal solutions can be found by using different ε values. The same methods can also be used for problem having convex or non-convex objective space alike. In terms of the information needed from the user, this algorithm is similar to the weighted sum approach. In the latter approach, a weight vector representing the relative importance of each objective is needed. In this approach, a vector of ε values representing the sense, the location of the pareto-optimal solution is needed. However, the advantage of this method is that it can be used for any arbitrary problem with either convex or non-convex objective space.

Goal programming yields only an efficient, rather than optimum solution to the problem. In essence, what goal programming does is to find a solution that simply satisfies the goals of the model with no regard to optimization. Such "deficiency" in finding an optimum solution may raises doubts about the viability of goal programming as an optimizing technique. Sequential quadratic programming (SQP) methods attempt to solve a nonlinear program directly rather than convert it to a sequence of unconstrained minimization problems.

This research quantitatively and qualitatively compared the performance of the four widely used multi-objective optimization methods using two test problems. The results indicated that there is no single algorithm that outperforms all others in all performance metrics. Indeed there is a "trade-off" in various performance metrics among the tested multi-objective optimization methods, and the choice of the proper algorithm should be made considering the type and complexity of the problem at hand, computing resource, and the user's proficiency in programming.

REFERENCES

[1] J. R. H. CARVALHO and P. A. V. FERREIRA (1995), Multiple-criterion Control: a Convex Programming Approach, Aukmaticn, Vol. 31, No. I, pp. lDZ5-1029.

[2] S.K. Das, A. Goswami *, S.S. Alam (1998), Multi-objective transportation problem with interval cost, source and destination parameters, European Journal of Operational Research 117.

[3] Eduardo Fernandez, Juan Carlos Leyva (2002), a method based on multi-objective optimization for deriving a ranking from a fuzzy preference relation, European Journal of Operational Research 154.

[4] Bruno Urli, Raymond Nadeau (2002), PROMISE/scenarios: An interactive method for multi-objective stochastic linear programming under partial uncertainty, European Journal of Operational Research 155.

[5] R.T. Marler and J.S. Arora (2003), Survey of multi-objective optimization methods for engineering, Struct Multidisc Optim 26, 369–395.

[6] E.K. Burke, J.D. Landa Silva (2004), the influence of the fitness evaluation method on the performance of multi-objective search algorithms, European Journal of Operational Research 169 (2006) 875–897.

[7] Alexander Engau, Margaret M. Wiecek (2005), generating e-efficient solutions in multi-objective programming, European Journal of Operational Research 177 (2007) 1566–1579.

[8] H. Fang, M. Rais-Rohani, Z. Liu, M.F. Horstemeyer (2005), A comparative study of meta-modeling methods for multi-objective crashworthiness optimization, Computers and Structures 83 (2005) 2121–2136.

[9] R.O. Parreiras, J.A. Vasconcelos (2006), A multiplicative version of Promethee II applied to multi-objective optimization problems, Department of Electrical Engineering, Federal University of Minas Gerais (UFMG), Av. Antoˆnio Carlos, 6627, 31270-010 BH, MG, Brazil, European Journal of Operational Research 183 (2007) 729–740.

[10] Blasco, J.M. Herrero, J. Sanchis, M. Martínez (2007), A new graphical visualization of n-dimensional Pareto front for decision-making in multi-objective optimization, Predictive Control and Heuristic Optimization Group, Department of Systems Engineering and Control, Universidad Politécnica de Valencia,Camino de Vera S/N, 46022 Valencia, Spain Information Sciences 178 (2008) 3908–3924

