Document downloaded from:

http://hdl.handle.net/10251/125654

This paper must be cited as:

Martínez Iranzo, MA.; Herrero Durá, JM.; Sanchís Saez, J.; Blasco, X.; García-Nieto, S. (2009). Applied Pareto multi-objective optimization by stochastic solvers. Engineering Applications of Artificial Intelligence. 22(3):455-465. https://doi.org/10.1016/j.engappai.2008.10.018

The final publication is available at

http://doi.org/10.1016/j.engappai.2008.10.018

Copyright Elsevier

Additional Information

Applied Pareto multi-objective optimization by stochastic solvers

Miguel Martínez-Iranzo, Juan M. Herrero, Javier Sanchis,

Xavier Blasco, Sergio García-Nieto

Predictive Control and Heuristic Optimization Group

Department of Systems Engineering and Control

Polytechnic University of Valencia

Camino de Vera 14, 46022 - Valencia, Spain

e-mail: mmiranzo@isa.upv.es http://ctl-predictivo.upv.es

Abstract

It is well known that many engineering design problems with different objectives, some of which can be opposed to one another, can be formulated as multi-objective functions and resolved with the construction of a Pareto front that helps select the desired solution. Obtaining a correct Pareto front is not a trivial question, because it depends on the complexity of the objective functions to be optimized, the constraints to keep within and, in particular, the optimizer type selected to carry out the calculations. This paper presents new methods for Pareto front construction based on stochastic search algorithms (GAs and MOGAs) that enable a very good determination of the Pareto front and fulfill some interesting specifications. The advantages of these applied methods will be proven by the optimization of well-known benchmarks for metallic supported I-beam and gearbox design.

Key words: Multi-objective optimization, Pareto front, Engineering design,

1 Introduction

Many engineering design problems can be translated into multi-objective optimization (MO) problems. This is particularly important when the objectives may be opposed to one another. For example, the volume or the cost of a structure may be opposed to its deflation. Examples can be found in almost any branch of science where MO problems and decision making are present.

MO techniques offer advantages over single-objective optimization techniques because they can provide a solution with different trade-offs among different individual objectives of the problem, so that the Decision Maker (DM) can select the best final solution [10, 11]. Solving an MO problem is, in general, associated with the construction of a Pareto front. Each point of this front represents one solution in the objective function space of the MO problem. Therefore, given any pair of solutions as vectors of their objective function values, an improvement of one component involves a deterioration of the others. Thus, there is no point of a front that is better than another point in this front (non-dominated points), and the rest of the objective space points are also dominated by one or more points of the front (dominated points).

The construction of a Pareto front can be very complex, even impossible to construct, depending on the nature of the MO problem to be solved. The presence of multi-modal objective functions and non-convex constraints can complicate the task of the optimizer. Specifically, the use of numeric optimizers based on non-linear programming (NLP) derived from Gauss-Newton methods

can present problems when determining Pareto fronts - since NLP depends on the starting point chosen for the search, given its nature as a local optimizer.

Although Messac et al. presents in [14] good alternatives for the construction of Pareto fronts using NLP optimizers, clear limitations in the case of solving bi-objective problems are shown [12]. These limitations are even greater for three-objective and multi-objective problems in general, therefore more research in that direction could be challenged.

Using stochastic optimizers, which deals with multi-modal and non-convex problems, can be a very effective way to solve these limitations. As design problems are performed off-line, the computational cost involved using this kind of optimizer to solve MO problems is not a key point in this research.

This paper presents two approaches to solve MO engineering design problems with stochastic optimizers. Firstly, a Genetic Algorithm (GA) is used as a substitute of the NLP in the Normalized Normal Constraint method (NNC) [13, 14]. The NNC method focuses on achieving well distributed Pareto front solutions. Secondly, Pareto solutions with a Smart distribution around the Pareto Front are generated using the ϵ-MOGA algorithm. In a smart distribution, the density of Pareto points is usually non-uniform, depending on the trade-offs among the objectives [13]. The greater the trade-off, the greater the density of points achieved. However, in regions with insignificant trade-off, fewer Pareto solutions are needed, and therefore the number of Pareto solutions to consider is reduced, making the choice of a final solution easier.

The paper is organized as follows: section 2 presents the mathematical foundations of the NNC method for MO problems. In section 3, deficiencies of the NNC method are discussed and their solutions are proposed with a method

based on the previous method - called the WNNC method. Two application examples of MO design problems are solved in section 4, presenting well-distributed Pareto fronts. The following section, offers a synopsis of the $\not\nearrow$ MOGA algorithm and discusses its natural ability to solve MO problems with smart characteristics. Comparison with WNNC results are also presented in this section. And finally, section 6 provides the concluding remarks.

2 An introduction to the Normalized Normal Constraint Method

The NNC method was originally formulated for bi-objective optimization [14], and was developed for the three-objective case in [13]. This method produces reasonably well distributed Pareto fronts.

The NNC method begins formulating an MO problem as follows:

$$min_{\mathbf{x}} \; [\mu_1(\mathbf{x}) \quad \mu_2(\mathbf{x}) \cdots \mu_n(\mathbf{x})] \tag{1}$$

subject to:

$$\begin{aligned} g_q(\mathbf{x}) &\leq 0, \qquad (1 \leq q \leq r) \\ h_e(\mathbf{x}) &= 0, \qquad (1 \leq e \leq s) \\ x_{li} \leq x_i &\leq x_{ui}, \quad (1 \leq i \leq n_x) \end{aligned} \tag{2}$$

Where $\mathbf{x}$ is the vector of the n_x design variables to be optimized; $g_q(\mathbf{x})$ and $h_e(\mathbf{x})$ are each of the r inequality and s equality problem constraints respectively; and x_{li} and x_{ui} are the lower and upper constraint limits in the n_x dimensions of the search space D, respectively.

This problem does not have a single solution and, therefore the Pareto optimal set $\mathbf{\Theta}_P$ (solutions where none dominate the others) must be found.

Pareto dominance is defined as follows:

A solution $\mathbf{x}^1$ dominates another solution $\mathbf{x}^2$, denoted by $\mathbf{x}_1 \prec \mathbf{x}_2$, if

$$\forall i, k \in [1 \dots n], \mu_i(\mathbf{x}^1) \leq \mu_i(\mathbf{x}^2) \ \wedge \ \exists k : \mu_k(\mathbf{x}^1) < \mu_k(\mathbf{x}^2) \ .$$

Therefore, the Pareto optimal set $\mathbf{\Theta}_P$, is given by

$$\mathbf{\Theta}_P = \{\mathbf{x} \in D \mid \ \neq \exists \ \ \tilde{\mathbf{x}} \in D \ : \ \tilde{\mathbf{x}} \prec \mathbf{x}\} \ . \tag{3}$$

The Pareto set, $\mathbf{\Theta}_P$, is unique and normally includes infinite solutions. Hence a set $\mathbf{\Theta}_P^*$, with a finite number of elements from $\mathbf{\Theta}_P$, should be obtained [1] .

The NNC method begins calculating the minimum in each objective function, $\mu_i(\mathbf{x}^{i*})$ $(i = 1, 2, ..., n)$. Thus, the solutions of the following n optimization problems are calculated as

$$\mathbf{x}^{i*} = \arg\min_{\mathbf{x}} \ \mu_i(\mathbf{x}) \quad \text{subject to (2)} \tag{4}$$

Using the solutions $\mathbf{x}^{i*}$, the *anchor points* μ^{i*} are formed as vectors of n components:

$$\begin{aligned} \mu^{i*} &= [\mu_1^{i*} \quad \mu_2^{i*} \quad \dots \quad \mu_n^{i*}]^T = \\ &= [\mu_1(\mathbf{x}^{i*}) \quad \mu_2(\mathbf{x}^{i*}) \quad \dots \quad \mu_n(\mathbf{x}^{i*})]^T \end{aligned} \tag{5}$$

Notice that the anchor points will determine the extremes of the Pareto frontier. In addition, two characteristic points are calculated: the *utopia point*, μ^U,

[1] Notice that $\mathbf{\Theta}_P^*$ is not unique.

composed of the best components [2] of each anchor point:

$$\begin{aligned}\mu^U &= [\mu_1^U \quad \mu_2^U \quad \dots \quad \mu_n^U]^T = \\ &= [\mu_1(\mathbf{x}^{1*}) \quad \mu_2(\mathbf{x}^{2*}) \quad \dots \quad \mu_n(\mathbf{x}^{n*})]^T\end{aligned} \tag{6}$$

and the *nadir point* written as the worst design objective values of the anchor points:

$$\mu^N = [\mu_1^N \quad \mu_2^N \quad \dots \quad \mu_n^N]^T \tag{7}$$

where

$$\mu_j{}^N = \max[\mu_j(\mathbf{x}^{1*}) \quad \dots \quad \mu_j(\mathbf{x}^{n*})]; \quad (j = 1, \dots, n) \tag{8}$$

To avoid scaling deficiencies, the optimization is performed in the normalized objective space (Fig. 1). In order to obtain the required normalized space, let vector $L = [l_1, l_2, \cdots, l_n]^T$ the difference between the nadir point and the utopia point, $L = \mu^N - \mu^U$. Hence, the following equation is used to perform the mapping:

$$\overline{\mu}_i(\mathbf{x}) = \frac{\mu_i(\mathbf{x}) - \mu_i^U}{l_i}, \quad i = (1, \dots, n) \tag{9}$$

Let vectors $\overline{N}_k$, for $k = 1, \dots, n-1$, defined as

$$\overline{N}_k = \overline{\mu}^{n*} - \overline{\mu}^{k*} \tag{10}$$

where each vector $\overline{N}_k$ represents the direction from the normalized anchor point for the n objective, $\overline{\mu}^{n*}$, to the normalized anchor point corresponding

[2] Since it corresponds to all objectives simultaneously at their best possible values - however, it cannot be reached.

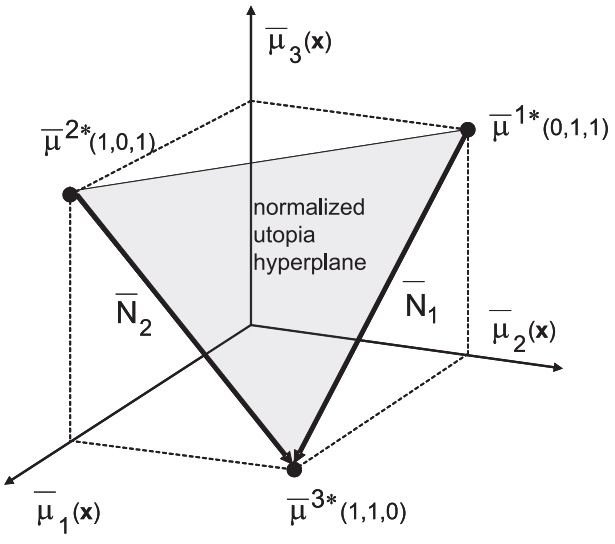

Figure 1. Normalized utopia hyperplane for a three objective problem.

to the k objective, $\overline{\mu}^{k*}$.

Along the direction of each vector $\overline{N}_k$ and for a prescribed number of divisions m_k, a normalized increment δ_k is defined as

$$\delta_k = \frac{1}{m_k - 1}, \quad k = 1, \ldots, n-1 \tag{11}$$

which will lead to a resulting segment size of $\delta_k \parallel \overline{N}_k \parallel$ for each vector $\overline{N}_k$.

Once the vectors $\overline{N}_k$ have been segmented, a set of distributed points $\overline{X}_{pj}$ over the normalized utopia hyperplane can be calculated as [3]

$$\overline{X}_{pj} = \sum_{k=1}^{n} \alpha_{kj} \overline{\mu}^{k*} \tag{12}$$

where

$$\begin{aligned} \alpha_{kj} &= [0, 1, \ldots, m_k - 1]\delta_k, \quad k = 1, \ldots, n-1 \\ \alpha_{nj} &= 1 - \sum_{k=1}^{n-1} \alpha_{kj} \end{aligned} \tag{13}$$

[3] For a tri-objective problem, the normalized utopia hyperplane is a triangle.

and the coefficients α_{kj} must fulfill the following property

$$\sum_{k=1}^{n-1} \alpha_{kj} \leq 1 \tag{14}$$

The NNC method states that the solution of the MO problem (1) can be transformed into a series of single objective minimizations, but in the normalized domain. Thus, for each point $\overline{X}_{pj}$ belonging to the normalized utopia hyperplane, a single optimization problem can be stated as:

$$\min_{\mathbf{x}} \ \overline{\mu}_n(\mathbf{x}) \tag{15}$$

subject to:

$$\begin{aligned} g_q(\mathbf{x}) &\leq 0, \qquad (1 \leq q \leq r) \\ h_e(\mathbf{x}) &= 0, \qquad (1 \leq e \leq s) \\ x_{li} \leq x_i &\leq x_{ui}, \ \ (1 \leq i \leq m_x) \end{aligned} \tag{16}$$

$$\overline{N}_k^T(\overline{\mu} - \overline{X}_{pj}) \leq 0, \quad k = 1, 2, \ldots n-1 \tag{17}$$

$$\overline{\mu} = [\overline{\mu}_1(\mathbf{x}), \cdots, \overline{\mu}_n(\mathbf{x})]^T \tag{18}$$

Therefore, the Pareto frontier can be obtained solving the series of optimization problems stated as (15). However, the frontier constructed in this way can include non Pareto or local Pareto points [4]. These local Pareto points could be marked differently when presenting results to an engineer, but at the same time, they provide information about the boundary of the constraint design

[4] Local Pareto points are those that are not locally dominated by any other point. Non Pareto points are even dominated locally.

space, and can always can be avoided. In [12] an algorithm with memory and bidirectional search for bi-objective problems, and in ([14]) a filtering process after optimization are proposals to eliminate these local and non Pareto solutions.

3 The wide NNC method for multi-objective optimization

The NNC method includes two undesirable features that result in the generation of unevenly distributed, and even incomplete, Pareto fronts. The first aspect is related to the calculation of normalized utopia hyperplanes and the distribution of points for more than two objectives. The second aspect is the absence of a guarantee that the generated set of solutions will represent the complete Pareto front for problems with more than two objectives. More specifically, the NNC method leaves some regions of the Pareto frontier unexplored.

This section provides important developments that overcome the limitations of the originally developed NNC method. Specifically, the new developments ensure that the generated Pareto set represents the complete Pareto frontier. It enables the generation of an entire discrete description of the Pareto frontier, which is a powerful feature that the original NNC method did not possess.

In [17] application of the NNC method for MO is discussed, particulary the impossibility of generalizing the transformation from the utopia plane to a normalized plane. It states an alternative enhanced normalized normal constraint (ENNC) method, which can be applied to MO problems of any number of objectives when they have disparate scales. The approach involves the use

of an exact matrix transformation between the objective space and the normalized space to obtain a normalized space with equal scales, instead of a linear transformation (9):

$$\overline{\mu}^{i*} = \mathbf{T}(\mu^{i*} - \mu^{U}) \tag{19}$$

Without loss of generality, the linear transformation matrix $\mathbf{T}$ for a three objective optimization problem can be calculated from

$$\begin{bmatrix} \overline{\mu}^{1*} \\ \overline{\mu}^{2*} \\ \overline{\mu}^{3*} \end{bmatrix}_{9x1} = \begin{bmatrix} \mathbf{T} & [0] & [0] \\ [0] & \mathbf{T} & [0] \\ [0] & [0] & \mathbf{T} \end{bmatrix}_{9x9} \begin{bmatrix} \mu^{1*} - \mu^{U} \\ \mu^{2*} - \mu^{U} \\ \mu^{3*} - \mu^{U} \end{bmatrix}_{9x1} \tag{20}$$

Matrix $\mathbf{T}$ enables mapping any point of the objective function space with disparate scales to the normalized space with equal scales, and vice versa. [5] However, it does not guarantee that the orthogonal projections of the points over the normalized utopia hyperplane take in the whole Pareto front.

Figure 2 shows how the orthogonal projection of the normalized utopia hyperplane leaves parts of the Pareto front unexplored. This fact implies that the set of distributed points $\overline{X}_{pj}$ over this plane - calculated as in (12) - is insufficient for the correct determination of the whole Pareto front.

The proposed method, called wide normalized normal constraint method (WNNC) can overcame this problem. WNNC modifies the approach used to generate

[5] Notice that, for the three objective case, Eq. (20) results in an independent system of 3^3 equations. In general, for MO problems of n objectives, the independent system (20) will always be of n^n equations.

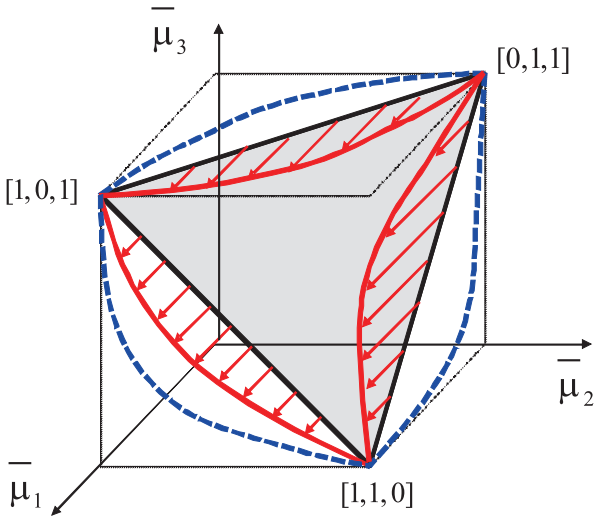

Figure 2. Solid: Normalized utopia hyperplane projection over Pareto front generated by the NNC method. Dashed: the complete Pareto front.

the points $\overline{X}_{pj}$ on the normalized utopia hyperplane, by simply resizing the normalized utopia hyperplane section. If the normalized utopia hyperplane is enlarged to the first quadrant limits, its projection will enclose the entire Pareto front in the normalized space. Doing this will ensure that Pareto points can be obtained using the NNC method anywhere in the hypercube.

In a three objective case, this new hyperplane, termed an enlarged normalized utopia hyperplane, is formed by the triangle limited by the vertices (figure 3):

$$\begin{aligned} \overline{\mu}_E^{1*} &= [2 \quad 0 \quad 0] \\ \overline{\mu}_E^{2*} &= [0 \quad 2 \quad 0] \\ \overline{\mu}_E^{3*} &= [0 \quad 0 \quad 2] \end{aligned} \tag{21}$$

And the new set of distributed points $\overline{X}_{pj}$ over the enlarged normalized utopia hyperplane can be now calculated as:

$$\overline{X}_{pj} = \sum_{k=1}^{n} \alpha_{kj} \overline{\mu}_E^{k*} \tag{22}$$

Notice that a greater number of points will be calculated as the enlarged utopia hyperplane covers a larger surface area. This means that the computational burden of the method will be increased.

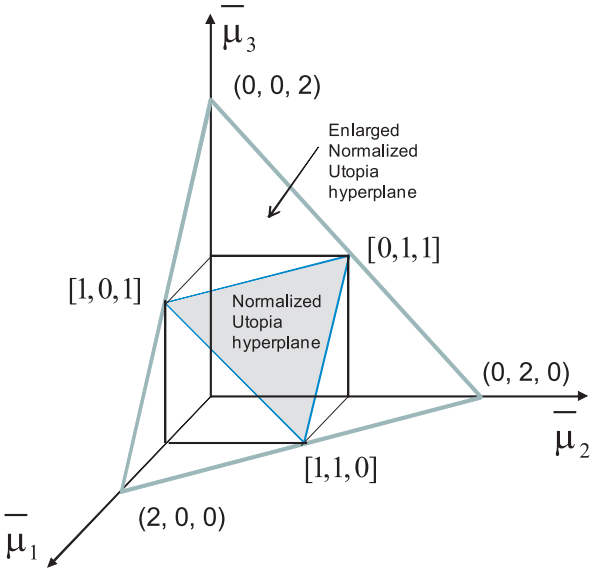

Figure 3. Enlarged normalized utopia hyperplane for a three objective problem.

Let's consider an example that shows the effectiveness of the proposal. This numerical example is presented in [13] and defined as

$$\begin{aligned}
&\min_{\mathbf{x}} \; [\mu_1(\mathbf{x}) \quad \mu_2(\mathbf{x}) \quad \mu_3(\mathbf{x})] \\
&\text{subject to:} \\
&\mu_1(\mathbf{x}) = x_1 \\
&\mu_2(\mathbf{x}) = x_2 \\
&\mu_3(\mathbf{x}) = x_3 \\
&(\mu_1(\mathbf{x}) - 1)^4 + (\mu_2(\mathbf{x}) - 1)^4 + (\mu_3(\mathbf{x}) - 1)^4 - 1 \leq 0
\end{aligned} \tag{23}$$

Figure 4 shows the normalized utopia hyperplane and the orthogonal projections when the NNC method is applied in its original form. Figure 5 shows the results in the normalized space when the enlarged normalized utopia plane is used with the WNNC method. Notice how new points belonging to the Pareto front now appear. It is important to note that the search in each orthogonal direction by means of an NLP algorithm (such as SQP [6]) did not produce adequate results in the upper area of the enlarged triangle. Therefore, a GA was used [7], together with a filter to eliminate non-Pareto points.

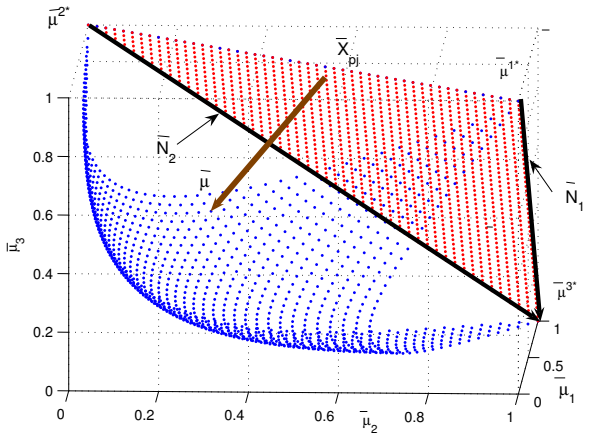

Figure 4. Solution of example (23) with the original NNC method.

4 Engineering application examples

In this section, we consider two engineering design problems that show the effectiveness of proposed methods for generating Pareto frontiers. The first problem, is a benchmark used in the area of civil engineering. The second example entails the optimization of a gearbox related with mechanical engineering. Notice that the examples are limited to two and three objectives in

[6] Sequential Quadratic Programming.

[7] The characteristics of the GA are presented in the appendix.

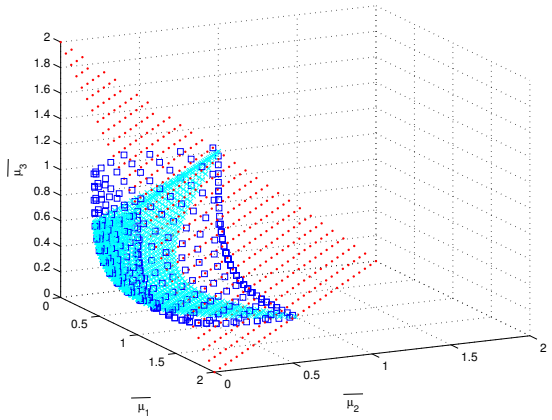

Figure 5. Enlarged utopia hyperplane (dots). The whole projection over the Pareto front that WNNC generates (squares).

order to present the results graphically.

4.1 Example 1: Simply supported I-beam design

The supported I-beam was originally proposed in [4]. This problem has four design parameters, $\mathbf{x}$, related to different longitudinal magnitudes of the I-beam (figure 6). It should minimize both the total cross-sectional area, $f_1(\mathbf{x})$, and its deflection at the midspan ,$f_2(\mathbf{x})$, under the applied loads P and Q. The constants of this problem are $P = 600\ kN$, $Q = 50\ kN$, $E = 2 \cdot 10^4\ kN/cm^2$ (Young's module), $\sigma = 16\ kN/cm^2$ (maximum stress), $L = 200\ cm$.

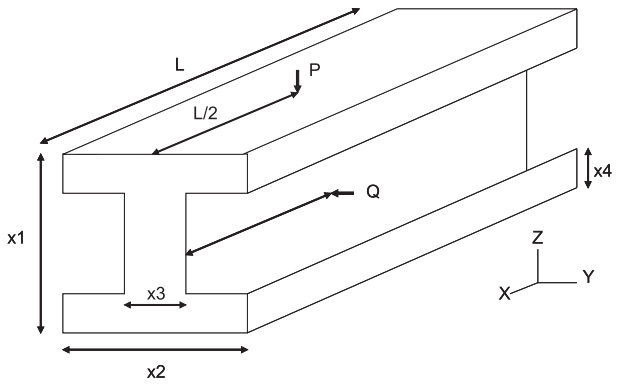

Figure 6. Simply supported I-beam of example 1.

Therefore, the MO design problem can be stated as:

$$\min_{\mathbf{x}} \; [f_1(\mathbf{x}) \quad f_2(\mathbf{x})]$$

$$\mathbf{x} = [x_1, x_2, x_3, x_4]$$

subject to

$$\frac{0.3Px_1}{x_3(x_1-2x_4)^3+2x_2x_4[4x_4^2+3x_1(x_1-2x_4)]} + \frac{0.3Qx_2}{(x_1-2x_4)x_3^3+2x_2^3x_4} \leq 0.001\sigma$$

$$10 \leq x_1 \leq 80$$

$$10 \leq x_2 \leq 50$$

$$0.9 \leq x_3 \leq 5$$

$$0.9 \leq x_4 \leq 5$$

where

$$f_1(\mathbf{x}) = 2x_2x_4 + x_3(x_1 - 2x_4)$$

$$f_2(\mathbf{x}) = \frac{PL^3}{48EI}$$

$$I = \frac{x_3(x_1-2x_4)^3+2x_2x_4[4x_4^2+3x_1(x_1-2x_4)]}{12}$$

Figure 7 presents the domain of feasible solutions where it is possible to identify the Pareto front. This result is achieved by exhaustive calculations and it seeks to serve as a confirmation when the Pareto front is obtained by the WNNC method.

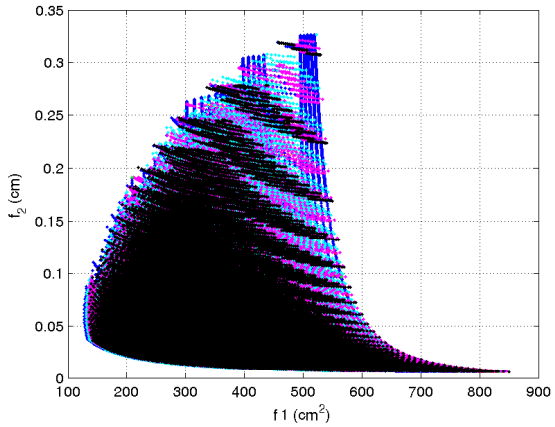

Figure 7. Example 1: Domain of feasible solutions.

Figure 8 presents the Pareto front obtained using WNNC. To generate this front a GA was used in each orthogonal search (15), instead of an NLP method (such as SQP) as originally proposed in the NNC method. Since the Pareto front reproduces the lower limit of the domain of feasible solutions presented in figure 7, it is possible to affirm that this front has been found correctly.

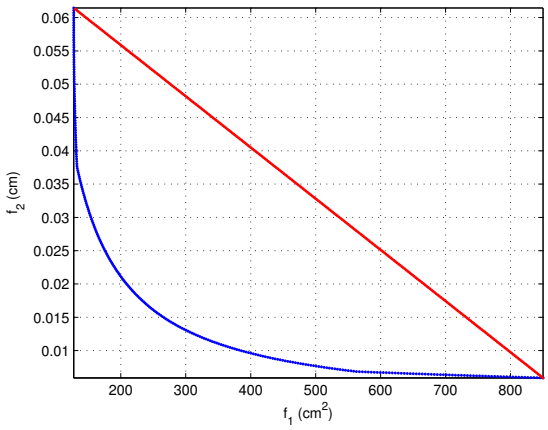

Figure 8. Example 1: Pareto front generated by the WNNC method.($m_1 = m_2 = 50$)

The optimization results from [6] - using a fuzzy multi-objective optimization method (FMO) - are also listed in Table 1 for comparison. This table shows how some points selected from the WNNC solution dominate the points obtained using the FMO method [8]. It means that the proposed method offers better optimum results.

Table 1

Optimization results of example 1 with FMO and WNNC methods.

	FMO (x_1, x_2, x_3, x_4)	FMO (f_1, f_2)	WNNC (x_1, x_2, x_3, x_4)	WNNC (f_1, f_2)
1	(80,26.1303,1.4637,4.7086)	(349.3860,0.0128)	(80,50.0,0.9,2.8160)	(348.5352,0.0111)
2	(80,49.9860,1.2242,2.3464)	(326.7680,0.0126)	(80,50.0,0.9,2.5837)	(325.7217,0.0119)
3	(80,50.0000,1.1312,2.2856)	(313.8876,0.0130)	(80,50.0,0.9,2.4565)	(313.2271,0.0125)
4	(80,35.7683,1.0355,3.0966)	(297.9494,0.0138)	(80,50.0,0.9,2.2934)	(297.2155,0.0132)
5	(80,50.0000,0.9000,2.0820)	(276.4525,0.0143)	(80,50.0,0.9,2.0820)	(276.4525,0.0143)

4.2 Example 2 - Gearbox design

In [8] a gearbox design is presented as a bi-objective optimization problem. In this paper, the same problem, but reformulated as in [6] with three objectives, is solved. The objectives are: to minimize the speed reducer volume (cm^3), and minimize the stress in the shafts 1 and 2. The MO problem has seven design variables constrained by their upper and lower limits. Furthermore, the problem is subjected to a number of constraints imposed by gear and

[8] Only point 5 is not dominated.

shaft design practices. Figure 9 and table 2 describe the physical meaning of variables and constraint functions.

Below, the mathematical formulation of the example is described:

$\min_{\mathbf{x}} \; [f_1(\mathbf{x}) \quad f_2(\mathbf{x}) \quad f_3(\mathbf{x})]$

$\mathbf{x} = [x_1, x_2, x_3, x_4, x_5, x_6, x_7]$

subject to

$g_1(\mathbf{x}) = 27x_1^{-1}x_2^{-2}x_3^1 - 1 \leq 0$

$g_2(\mathbf{x}) = 397.5x_1^{-1}x_2^{-2}x_3^{-2} - 1 \leq 0$

$g_3(\mathbf{x}) = 1.93x_2^{-1}x_3^{-1}x_4^3x_6^{-4} - 1 \leq 0$

$g_4(\mathbf{x}) = 1.93x_2^{-1}x_3^{-1}x_5^3x_7^{-4} - 1 \leq 0$

$g_5(\mathbf{x}) = x_2x_3 - 40 \leq 0$

$g_6(\mathbf{x}) = x_1x_2^{-1} - 12 \leq 0$

$g_7(\mathbf{x}) = 5 - x_1x_2^{-1} \leq 0$

$g_8(\mathbf{x}) = 1.9 - x_4 + 1.5x_6 \leq 0$

$g_9(\mathbf{x}) = 1.9 - x_5 + 1.1x_7 \leq 0$

$g_{10}(\mathbf{x}) = \frac{A_1}{B_1} - 1300 \leq 0$

$g_{11}(\mathbf{x}) = \frac{A_2}{B_2} - 850 \leq 0$

$2.6 \leq x_1 \leq 3.6$

$0.7 \leq x_2 \leq 0.8$

$17 \leq x_3 \leq 28$

$7.3 \leq x_4 \leq 8.3$

$7.3 \leq x_5 \leq 8.3$

$2.9 \leq x_6 \leq 3.9$

where

$$f_1(\mathbf{x}) = 0.7854x_1x_2^2(10x_3^2/3 + 14.9334x_3 - 43.0934)$$

$$-1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3)$$

$$+0.7854(x_4x_6^2 + x_5x_7^2)$$

$$f_2(\mathbf{x}) = \frac{A_1}{B_1}$$

$$A_1 = \sqrt{(745x_4x_2^{-1}x_3^{-1})^2 + 1.69e^7},\ B_1 = 0.1x_6^3$$

$$f_3(\mathbf{x}) = \frac{A_2}{B_2}$$

$$A_2 = \sqrt{(745x_5x_2^{-1}x_3^{-1})^2 + 1.575e^8},\ B_2 = 0.1x_7^3$$

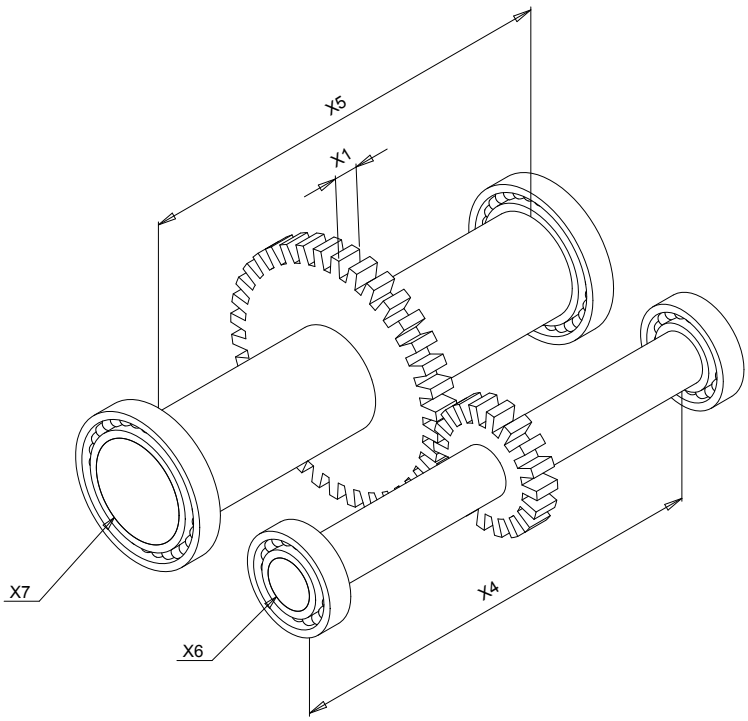

Figure 9. Example 2: Graphical representation of the gearbox.

In this example, the anchor, utopia, and nadir points are respectively:

Table 2

Physical variables and gear constraints.

x_1	Gear face width (cm)
x_2	Teeth module (cm)
x_3	Number of teeth on pinion
x_4	Distance between bearing 1 (cm)
x_5	Distance between bearing 2 (cm)
x_6	Diameter of shaft 1 (cm)
x_7	Diameter of shaft 2 (cm)
$g_1(x)$	Bending stress of teeth
$g_2(x)$	Constant stress of teeth
$g_3(x)$	Transverse displacement of shaft 1
$g_4(x)$	Transverse displacement of shaft 2
$g_5(x)$	Generated torque constraint
$g_6(x)$	Generated torque constraint
$g_7(x)$	Generated torque constraint
$g_8(x)$	Generated torque constraint
$g_9(x)$	Generated torque constraint
$g_{10}(x)$	Stress of shaft 1
$g_{11}(x)$	Stress of shaft 2

$$\mu^{1*} = [2948.2 \quad 1308.0 \quad 850.6]$$
$$\mu^{2*} = [6012.9 \quad 694.7 \quad 809.9]$$
$$\mu^{3*} = [5950.8 \quad 1048.5 \quad 754.5]$$

$$\mu^{U} = [2948.2 \quad 694.7 \quad 754.5]$$
$$\mu^{N} = [6012.9 \quad 1308.0 \quad 850.6]$$

Using the simple linear transformation (9), the following normalized anchor points result:

$$\begin{aligned} \overline{\mu}^{1*} &= [0 \quad 1 \quad 1] \\ \overline{\mu}^{2*} &= [1 \quad 0 \quad 0.5767] \\ \overline{\mu}^{3*} &= [0.9797 \quad 0.5769 \quad 0] \end{aligned} \tag{24}$$

which do not have the desired form of

$$\begin{aligned} \overline{\mu}^{1*} &= [0 \quad 1 \quad 1] \\ \overline{\mu}^{2*} &= [1 \quad 0 \quad 1] \\ \overline{\mu}^{3*} &= [1 \quad 1 \quad 0] \end{aligned} \tag{25}$$

To obtain the anchor points of 25, an exact matrix transformation as in 19 must be applied, with

$$\mathbf{T} = \begin{bmatrix} 0.3296 & 0.0289 & -0.1847 \\ 0.0697 & 2.2347 & -3.8552 \\ 0.0698 & -0.5921 & 14.1837 \end{bmatrix} \cdot 10^{-3}$$

derived from the system of equations (20).

The Pareto front generated using the NNC method and the exact matrix transformation with $\mathbf{T}$ as normalization procedure is shown in figure 10.

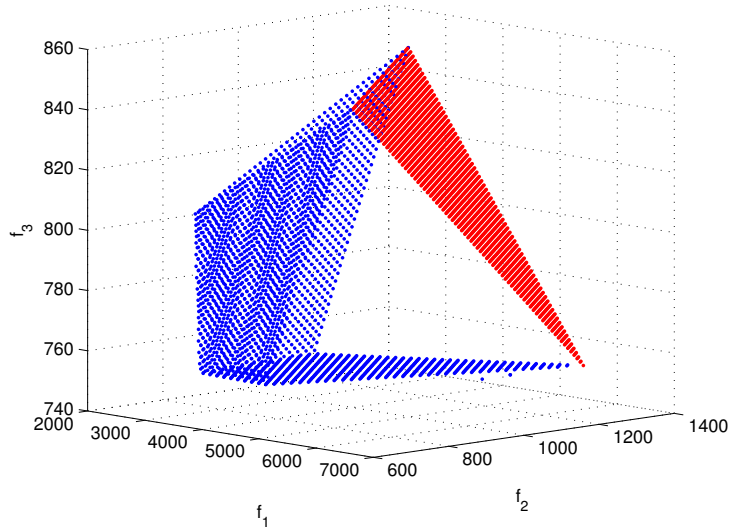

Figure 10. Example 2: Pareto front and utopia plane generated with NNC corrected with transformation matrix $\mathbf{T}$. ($m_1 = m_2 = 75$).

As stated in section 3, using the enlarged utopian hyperplane expands the obtainable Pareto solutions because unexplored regions of the Pareto front are reduced. This fact can be observed in figure 11 which depicts the results in the objectives space. Once more, a GA is used in each orthogonal search (15) to generate the front, instead of an NLP method (such as SQP) as this gradient-based method produces inaccurate results.

Differences between the use of the NNC method with the normalized utopia hyperplane and the proposed WNNC can be observed in figure 12.

As in the previous example, the optimization results from [6] - using the FMO method - are listed in Table 3 for comparison. In this case, the WNNC results again significantly improved on those obtained by FMO [9].

[9] For the sake of simplicity, the values of the design variables $\mathbf{x}$ corresponding to the selected points of the Pareto front are not included.

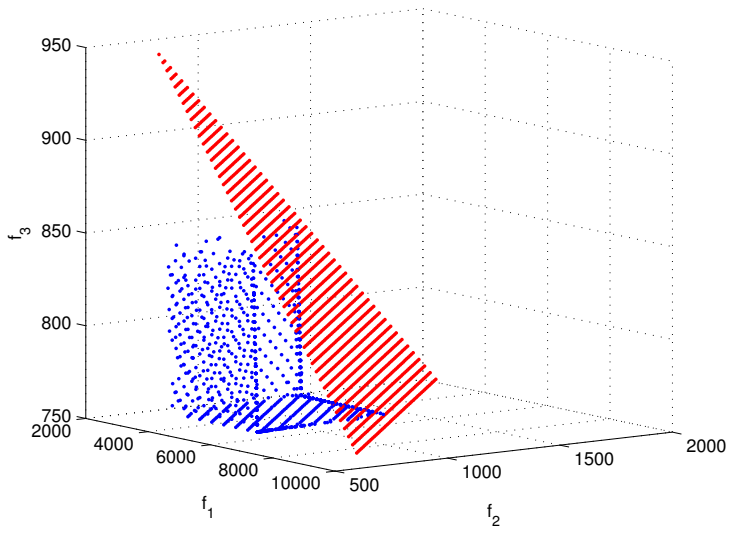

Figure 11. Example 2: Pareto front and enlarged utopia plane generated with WNNC ($m_1 = m_2 = 50$).

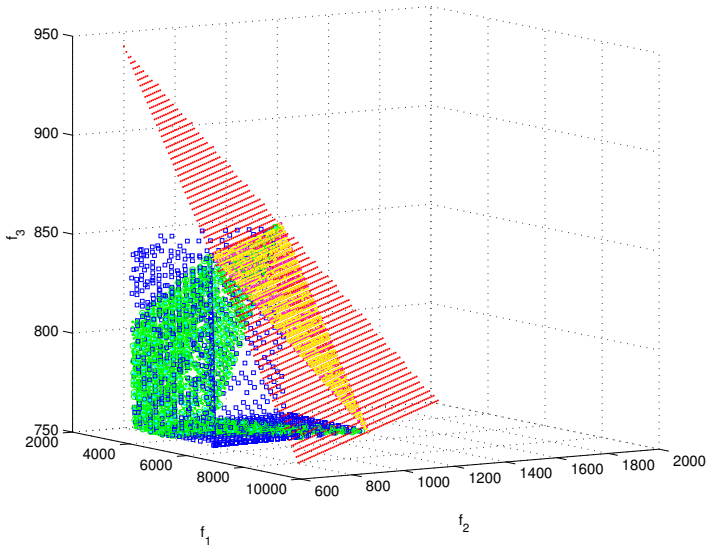

Figure 12. Example 2: Comparison of solutions generated by the WNNC method (squares) and the NNC method (dots).

5 MOGA algorithm with 'smart' characteristics

Previous sections have shown the WNNC algorithm can generate complete and evenly distributed Pareto fronts. This uniform distribution is achieved

Table 3

FMO vs. WNNC optimization results for the gearbox design.

	FMO	WNNC
	(f_1, f_2, f_3)	(f_1, f_2, f_3)
1	(4361.3, 1004.5, 797.4)	(4202.9, 844.8, 754.7)
2	(4588.8, 870.0, 810.8)	(4508.7, 695.7, 754.7)
3	(3765.5, 1089.3, 793.0)	(3614.9, 795.9, 754.8)
4	(4821.6, 757.7, 762.9)	(4508.7, 695.7, 754.7)
5	(3425.0, 879.8, 797.6)	(3412.1, 829.4, 754.7)
6	(3762.0, 939.8, 775.7)	(3569.0, 895.4, 754.8)
7	(4001.6, 822.1, 775.7)	(3466.5, 729.9, 754.8)
8	(3812.9, 702.0, 793.0)	(3664.9, 696.9, 760.1)

solving an increased number of single optimization problems - one per each solution generated. This computational burden can be relaxed using other types of algorithms which generate solutions with smart distributions. Smart means that the density of the points belonging to the Pareto front depends on its rate of change. Front zones with lower rates of change have fewer points than higher rated zones.

One of the algorithms with smart characteristics is the ϵ-MOGA. It is an elitist multi-objective evolutionary algorithm based on the concept of ϵ-dominance [9] which is used to control the content of the archive $A(t)$ where the problem solutions are stored.

Other alternatives, also based on the concept of ϵ-dominance are the ϵ-MOEA [7] and the ϵ-MOGA [5] algorithms. In [7], a comparison between the ϵ-MOEA method and other well known algorithms such as NSGA-II, PESA, SPEA2, etc. is performed, showing the superiority of the ϵ-MOEA. In [5] the ϵ-MOGA - an earlier version of the $\not\epsilon$MOGA - is compared with the ϵ-MOEA algorithm. The main differences between the two algorithms are:

- Different genetic operators are used on each algorithm. For crossover,ϵ-MOEA uses the SDX operator but ϵ-MOGA uses linear recombination and gaussian mutation.
- ϵ-MOEA just creates one individual per iteration. In ϵ-MOGA, n individuals per iteration are created. This makes its parelization easier.

Furthermore, with $\not\epsilon$MOGA, the anchor points - the Pareto front limits - are not required to determine the archive size as in ϵ-MOEA. In $\not\epsilon$MOGA, this size is settled with the n_box parameter and the cell width ϵ is adapted dynamically [10] . As a result, in addition to the genetic nature of its operators, the $\not\epsilon$MOGA algorithm achieves a better characterization of the Pareto fronts than the ϵ-MOEA approach.

On the other hand, $\not\epsilon$MOGA tries to ensure that $A(t)$ converges towards an ϵ-Pareto set, $\mathbf{\Theta}_P^*$, in a smartly distributed manner along the Pareto front, $\mu(\mathbf{\Theta}_P)$, with limited memory resources. It also adjusts the limits of the front $\mu(\mathbf{\Theta}_P^*)$ dynamically and prevents the anchor points being lost.

To reach this goal, the objective space is split into a fixed number of boxes

[10] Details of the $\not\epsilon$MOGA algorithm and its parameters are described in appendix B.

n_box_i. Hence, for each dimension $i \in [1 \dots n]$, n_box_i cells of ϵ_i width are created where

$$\epsilon_i = (\mu_i^{max} - \mu_i^{min})/n\_box_i,$$

$$\mu_i^{max} = \max_{\mathbf{x} \in \mathbf{\Theta}_P^*} \mu_i(\mathbf{x}),\ \mu_i^{min} = \min_{\mathbf{x} \in \mathbf{\Theta}_P^*} \mu_i(\mathbf{x}).$$

This grid preserves the diversity of $\mu(\mathbf{\Theta}_P^*)$, since one box can be occupied by only one solution, and at the same time it produces a smart distribution as will be shown later.

The concept of ϵ-dominance is defined as follows. For a solution $\mathbf{x} \in D$, $box_i(x)$ is defined by

$$box_i(\mathbf{x}) = \left\lceil \frac{\mu_i(\mathbf{x}) - \mu_i^{min}}{\mu_i^{max} - \mu_i^{min}} \cdot n\_box_i \right\rceil \ \forall i \in [1 \dots n]. \tag{26}$$

A solution $\mathbf{x}^1$ with value $\mu(\mathbf{x}^1)$ ϵ-dominates the solution $\mathbf{x}^2$ with value $\mu(\mathbf{x}^2)$, denoted by $\mathbf{x}^1 \prec_\epsilon \mathbf{x}^2$, if and only if

$$\mathbf{box}(x^1) \prec \mathbf{box}(x^2) \vee \left(\mathbf{box}(x^1) = \mathbf{box}(x^2) \wedge x^1 \prec x^2\right) .$$

where $\mathbf{box}(\mathbf{x}) = \{box_1(\mathbf{x}), \dots, box_s(\mathbf{x})\}$.

Hence, a set $\mathbf{\Theta}_P^* \subseteq \mathbf{\Theta}_P$ is ϵ-Pareto, if and only if

$$\forall \mathbf{x}^1, \mathbf{x}^2 \in \mathbf{\Theta}_P^*, \mathbf{x}^1 \neq \mathbf{x}^2,\ \mathbf{box}(\mathbf{x}^1) \neq \mathbf{box}(\mathbf{x}^2) \wedge \mathbf{box}(\mathbf{x}^1) \not\prec_\epsilon \mathbf{box}(\mathbf{x}^2) \tag{27}$$

Therefore, ϵ-MOGA is responsible for updating the content of $A(t)$ by saving only ϵ-dominant solutions that do not share the same box. When two mutually ϵ-dominant solutions compete, the solution that prevails in $A(t)$ will be the closest to the center of the box. It is thereby possible to prevent solutions belonging to adjacent boxes (neither one dominating the other)and from being too close to each other, thus encouraging a smart distribution.

Figure 13 illustrates the ϵ-dominance idea. For a bi-objective case, this figure depicts what $\mathbf{\Theta}_P^*$ would be when $n\_box_1 = n\_box_2 = 10$ is used. The values ϵ_1 and ϵ_2 depend on the limits of the front (μ_1^{min}, μ_2^{min}, μ_1^{max} and μ_2^{max}), which are dynamically adjusted as they are located in accordance with the solutions. It can be seen that the distribution of solutions comprised by $\mu(\mathbf{\Theta}_P^*)$, along the front, depends on the slope, the greatest number of points accumulating in the central area (indicated by a dotted line) where the slope is greatest.

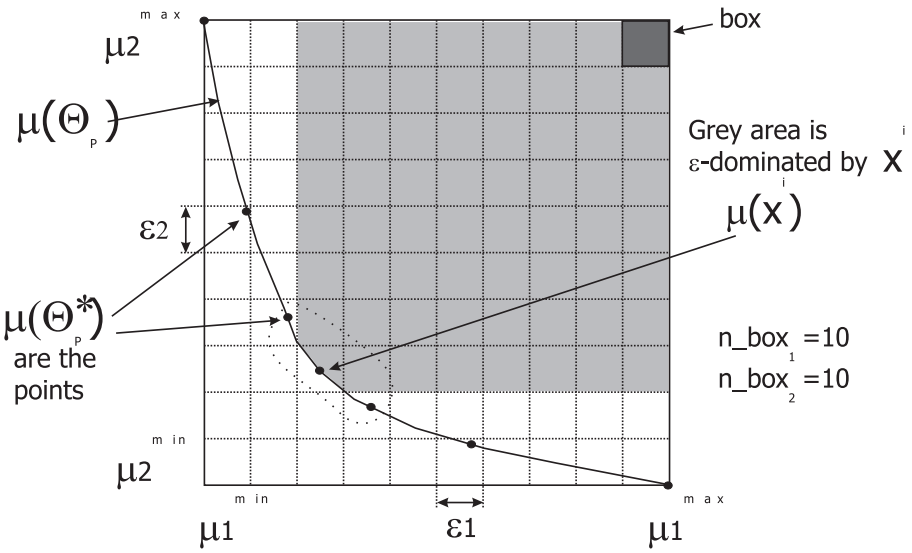

Figure 13. Two objective optimization example. Dots: ϵ-Pareto front $\mu(\mathbf{\Theta}_P^*)$; μ_1^{min}, μ_2^{min}, μ_1^{max}, μ_2^{max}: Pareto front limits; ϵ_1, ϵ_2: box widths and n_box_1, n_box_2; number of boxes for each dimension.

The aim of ε-MOGA is to achieve a $\mathbf{\Theta}_P^*$ with the greatest possible number of solutions in order to adequately characterize the Pareto front. Although the number of possible solutions will depend on the shape of the front, and on n_box_i, it will not exceed the following level

$$|\Theta_{P_\epsilon}^*| \leq \frac{\prod_{i=1}^{n} n\_box_i + 1}{n\_box_{max} + 1}, \quad n\_box_{max} = \max_i n\_box_i \tag{28}$$

which is advantageous, as it is possible to control the maximum number of

solutions that will characterize the Pareto front.

Furthermore, thanks to the definition of the box, the anchor points μ^{i*} are assigned a value of $box_i(\mathbf{x}^{i*}) = 0$, since $\mu_i(\mathbf{x}^{i*}) = \mu_i^{min}$. Therefore, no solution $\mathbf{x}$ can ϵ-dominate them, as their $box_i(\mathbf{x}) \geq 1$ by applying the box definition.

In order to check the ϵ-MOGA performance, the same examples shown in section 4 have been solved. The parameters of the ϵ-MOGA algorithm were set to:

- $Nind_G = 4$ and $Nind_P = 100$.
- $t_{max} = 10000$ for example 1, 50000 for example 2.
- $P_{c/m} = 0.1$.
- $n\_box_i = 40$.

Figure 14 shows the front $\mu(\mathbf{\Theta}_P^*)$ obtained for the I-beam design problem compared with that generated by the WNNC method. Notice how the ϵ-MOGA algorithm has characterized the Pareto front with just sixteen solutions (including the anchor points) distributed in a smart manner. In this case, the trade-off is lower near the anchor points than near the ideal point, so the density solutions in the first area are larger than in the latter area. Characterizing the Pareto front with fewer points makes the subsequent decision-making task easier, as well as involving a lower computational cost than when using WNNC with GA.

For the gearbox design problem, figure 15 shows the $\mu(\mathbf{\Theta}_P^*)$ together with the WNNC results. In this case, the ϵ-MOGA algorithm has characterized the Pareto front - in a smart manner - by means of 54 points. The number of points has been considerably reduced (in comparison with the WNNC results)

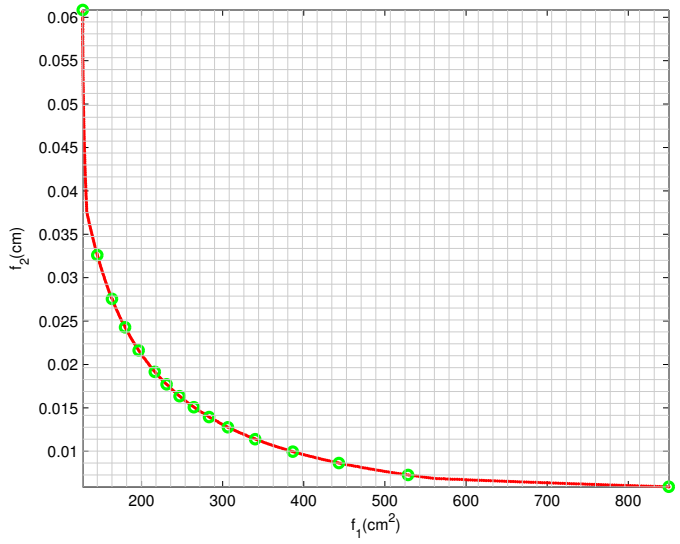

Figure 14. Pareto front solution of example 1. Generated by ε-MOGA (circles) and by WNNC (solid).

since the front solution presents large areas where the trade-off is insignificant.

6 Conclusions

In this paper the NNC method for multi-objective optimization has been analysed and its drawbacks identified. A new method with two new improvements over the original NNC method has been proposed: the definition of an exact linear transformation between μ and $\overline{\mu}$ spaces for correct calculation of the normalized anchor points - and the definition of an enlarged utopian hyperplane to expand the obtainable Pareto solutions. Hence, unexplored regions of the Pareto front have been reduced, but the computational cost of the method has been increased.

The WNNC method is applied to two engineering design examples, and in both cases, a GA has been used because NLP optimization was insufficient

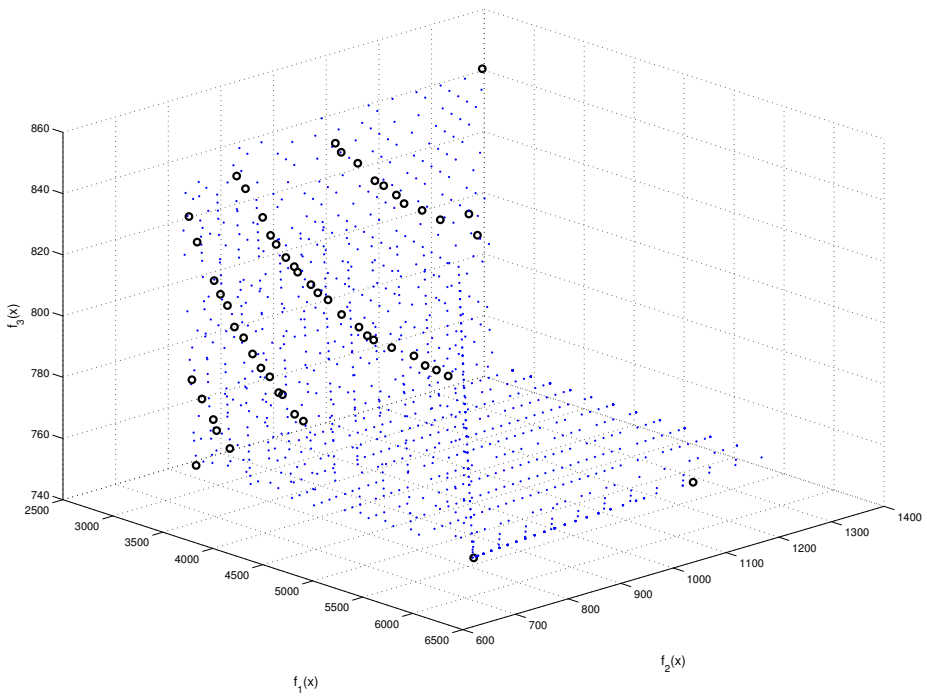

Figure 15. Pareto front solution of example 2. Generated by ϵ-MOGA (circles) and by WNNC (dots).

for achieving good results. A comparison with other optimal solutions of the examples are presented too, showing a significant improvement in some cases.

Finally, the ϵ-MOGA algorithm is presented and its results are compared with the WNNC. This algorithm tries to ensure that the final solution converges towards an ϵ-Pareto set, in a smart distributed manner along the Pareto front with limited memory resources. Therefore, the computational burden is reduced and the subsequent decision-making task made easier.

Acknowledgments

This research has been partially financed by GV06-026 Generalitat Valenciana and DPI2005-07835, MEC (Spain)-FEDER.

Appendix A GA characteristics

In this paper, the implementation of GA has the following characteristics:

- Real value codification [15], i.e. each gene has a real value so the chromosome is an array of real values.
- $\mu(\mathbf{x})$ is not directly used as a cost function. A 'ranking' operation is performed [3, 1]. The first individuals are sorted into decreasing $\mu(\mathbf{x})$ value, and then, $\mu(\mathbf{x})$ is replaced by its position in such a distribution. Each individual has a new cost function value $\mu'(\mathbf{x})$. The ranking operation prevents clearly dominant individuals from prevailing too soon, thus exhausting the algorithm.
- Selection is made by the operator known as *Stochastic Universal Sampling (SUS)* [2]. The probability of survival of an individual, $P(\mathbf{x}_i)$, is guaranteed to be:

$$P(\mathbf{x}_i) = \frac{\mu'(\mathbf{x}_i)}{\sum_{j=1}^{Nind} \mu'(\mathbf{x}_j)} \tag{29}$$

  Where $Nind$ is the number of individuals.
- For crossover, the *intermediate recombination* operator is used [16]. Offspring chromosomes $\mathbf{x}'_1$ and $\mathbf{x}'_2$ are obtained through the following operation on the parents' chromosomes ($\mathbf{x}_1$ and $\mathbf{x}_2$):

  $\mathbf{x}'_1 = \alpha_1 \cdot \mathbf{x}_1 + (1 - \alpha_1)\mathbf{x}_2$

$$\mathbf{x}_2' = \alpha_2 \cdot \mathbf{x}_2 + (1 - \alpha_2)\mathbf{x}_1$$
$$\alpha_1 \in [-d, 1 + d]$$
$$\alpha_2 \in [-d, 1 + d]$$

The operation can be performed on the whole chromosome, or on each gene separately. In the latter case, random parameters α_1 and α_2 have to be generated for each gene - increasing search capability but with a higher computational cost.

Implemented GA has been adjusted as follows:

· $\alpha_1 = \alpha_2$ and generated for each chromosome.

· $d = 0$.

Crossover probability is set to $P_c = 0.8$.

- The mutation operation is made with a probability of $P_m = 0.1$ and a normal distribution with standard deviation set at 20% of the search space range.

Appendix B – MOGA algorithm

The algorithm is composed of three populations:

(1) The main population $P(t)$ explores the searching space D during the algorithm iterations (t). Population size is $Nind_P$.

(2) Archive $A(t)$ stores the solution $\mathbf{\Theta}_P^*$. Its size $Nind_A$ is variable but bounded (see equation (28)).

(3) Auxiliary population $G(t)$. Its size is $Nind_G$, which must be an even number.

The pseudocode of the $\not\nearrow\!\!-MOEA$ algorithm is given by:

```
1. t:=0
2. A(t):=∅
3. P(t):=ini_random(D)
4. eval(P(t))
5. A(t):=store_ini(P(t),A(t))
6. while t<t_max do
7.      G(t):=create(P(t),A(t))
8.      eval(G(t))
9.      A(t+1):=store(G(t),A(t))
10.     P(t+1):=update(G(t),P(t))
11.     t:=t+1
12. end while
```

The main steps of the algorithm are as follows:

Step 3. $P(0)$ is initialized with $Nind_P$ individuals (solutions) that have been randomly selected from searching space D.

Step 4 and 8. Function **eval** calculates objective functions for each individual in $P(t)$ (step 4) and $G(t)$ (step 8).

Step 5. Function **store**$_{ini}$ checks individuals in $P(t)$ that might be included in the archive $A(t)$, as follows:

(1) Non-dominated $P(t)$ individuals are detected, $\mathbf{\Theta}_{ND}$.

(2) Pareto front limits μ_i^{max} and μ_i^{min} are calculated from $\mu(\mathbf{x}), \forall \mathbf{x} \in \mathbf{\Theta}_{ND}$.

(3) Individuals in $\mathbf{\Theta}_{ND}$ are analyzed, one by one, and those that are not ϵ-dominated by individuals in $A(t)$, will be included in $A(t)$.

Step 7. With each iteration, the function **create** creates $G(t)$ as follows:

(1) Two individuals are randomly selected, $\mathbf{x}^P$ from $P(t)$, and $\mathbf{x}^A$ from $A(t)$.

(2) A random number $u \in [0 \ldots 1]$ is generated.

(3) If $u > P_{c/m}$ (probability of crossing/mutation), $\mathbf{x}^P$ and $\mathbf{x}^A$ are crossed over by means of the extended linear recombination technique.

(4) If $u \leq P_{c/m}$, $\mathbf{x}^P$ and $\mathbf{x}^A$ are mutated using random mutation with Gaussian distribution and then included in $G(t)$.

This procedure is repeated $Nind_G/2$ times until $G(t)$ is filled up.

Step 9. Function **store** checks, one by one, which individuals in $G(t)$ must be included in $A(t)$ on the basis of their location in the objective space (see figure 16). Thus $\forall\, \mathbf{x}^G \in G(t)$

(1) If $\mu(\mathbf{x}^G)$ belongs to the area $Z1$ and is not ϵ-dominated by any individual from $A(t)$, it will be included in $A(t)$ (if its box is occupied by an individual not ϵ-dominated too, then the individual lying furthest away from the center box will be eliminated). Individuals from $A(t)$ which are ϵ-dominated by $\mathbf{x}^G$ will be eliminated.

(2) If $\mu(\mathbf{x}^G)$ belongs to the area $Z2$ then it is not included in the archive, since it is dominated by all individuals in $A(t)$.

(3) If $\mu(\mathbf{x}^G)$ belongs to the area $Z3$, the same procedure is applied as was used with the function $\mathbf{store}_{ini}$ but now applied over a population $P'(t) = A(t) \bigcup \mathbf{x}^G$, that is, $\mathbf{store}_{ini}(P'(t), \emptyset)$. In this procedure new Pareto front limits and ϵ_i widths could be recalculated.

(4) If $\mu(\mathbf{x}^G)$ belongs to the area $Z4$, all individuals from $A(t)$ are deleted since they all are ϵ-dominated by $\mathbf{x}^G$. $\mathbf{x}^G$ is included and objective space limits are $\mu(\mathbf{x}^G)$.

Step 10. Function **update** updates $P(t)$ with individuals from $G(t)$. Every individual $\mathbf{x}^G$ from $G(t)$ replaces an individual $\mathbf{x}^P$ that is randomly selected

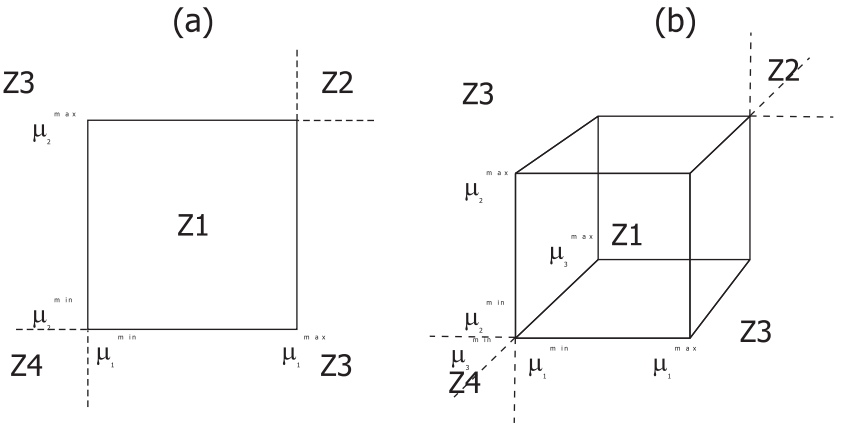

Figure 16. Objectives space areas (Z) that function **store** uses: (a) two-dimensional case (b) three-dimensional case.

from the individuals in $P(t)$ that are dominated by $\mathbf{x}^G$. $\mathbf{x}^G$ will not be included in $P(t)$ if there is no individual in $P(t)$ dominated by $\mathbf{x}^G$.

Finally, individuals from $A(t)$ comprise $\mathbf{\Theta}_P^*$, the smart characterization of the Pareto front.

References

[1] T. Back. *Evolutionary Algorithms in Theory and Practice*. Oxford University Press, New York, 1996.

[2] J.E. Baker. Reducing bias and inefficiency in the selection algorithms. In *Proceedings of the Second International Conference on Genetic Algorithms*, pages 14–21. Lawrence Erlbaum Associates, 1987.

[3] F.X. Blasco. *Model based predictive control using heuristic optimization techniques. Application to non-linear and multivariable processes*. PhD thesis, Universidad Politécnica de Valencia, Valencia, 1999 (In Spanish).

[4] P. Hajela and C.J. Shih. Multiobjective optimum design in mixed integer and discrete design variable problems. *AIAA Journal*, 28(4):670–675, 1990.

[5] J.M. Herrero. *Robust identification of non-linear systems using evolutionary algorithms*. PhD thesis, Polytechnic University of Valencia, Valencia (Spain), 2006 (In Spanish).

[6] H.Z. Huang, Y.K Gu, and X. Du. An interactive fuzzy multi-objective optimization method for engineering design. *Eng. Appl. Artif. Intel.*, 19(5):451–460, 2006.

[7] S. Mishra K. Deb, M. Mohan. Evaluating the ϵ-domination based multi-objective evolutionary algorithm for a quick computation of pareto-optimal solutions. *Evolutionary computation*, 13(4):501–526, 2005.

[8] A. Kurapati and S. Azarm. Immune network simulation with multiobjective genetic algoritms for multidisciplinary design optimization. *Eng. Opt.*, 33:245–260, 2000.

[9] M. Laumanns, L. Thiele, K. Deb, and E. Zitzler. Combining convergence and diversity in evolutionary multi-objective optimization. *Evolutionary*

computation, 10(3):263–282, 2002.

[10] M. Martínez, J. Sanchis, and X. Blasco. Algoritmos genéticos aplicados al diseño de controladores robustos. *Revista de Automática e Informática Industrial (RIAI). In Spanish.*, 3(1):39–51, 2006.

[11] M. Martínez, J. Sanchis, and X. Blasco. Multiobjective controller design handling human preferences. *Eng. Appl. Artif. Intel.*, 19(8):927–938, 2006.

[12] M. Martínez, J. Sanchis, and X. Blasco. Global and well-distributed pareto frontier by modified normalized normal constraint methods for bicriterion problems. *Struct. Multidiscip. Optimization*, 34:197 – 209, 2007.

[13] C.A. Mattson, A.A. Mullur, and A. Messac. Smart pareto filter: Obtaining a minimal representation of multiojective design space. *Eng. Optimiz.*, 26(6):721–740, 2004.

[14] A. Messac, A. Ismail-Yahaya, and C.A. Mattson. The normalized normal constraint method for generating the pareto frontier. *Struct. Multidiscip. Optim.*, 25:86–98, 2003.

[15] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer series Artificial Intelligence. Springer, 3rd edition, 1996.

[16] H. Mühlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm. continuous parameter optimization. *Evolutionary Computation. The MIT Press*, 1(1):25–49, 1993.

[17] J. Sanchis, M. Martínez, X. Blasco, and J.V. Salcedo. A new perspective on multiobjective optimization by enhanced normalized normal constraint method. *Struct. Multidiscip. Optimization (DOI:10.1007/s00158-007-0185-4)*, 2007.