# Designer's Preferences and Multi–objective Preliminary Design Processes

Dragan Cvetković and Ian Parmee
Plymouth Engineering Design Centre, University of Plymouth, UK
email: {dcvetkovic,iparmee}@plymouth.ac.uk

**Abstract**

In this paper we present a method based on preference relations for transforming non–crisp (qualitative) relationships between objectives in multi–objective optimisation into quantitative attributes (i.e. numbers).This is integrated with two multi–objective genetic algorithms (GAs): weighted sums GA and a method for combining the Pareto method with weights. Examples of the preference relations application together with traditional genetic algorithms are also presented. Some complexity issues are discussed and analysed.

## 1 Introduction

When dealing with industrial design problems, it rapidly becomes apparent that there are significant differences between so called 'textbook optimisation problems' and 'real–world applications'. In both cases, in multi–objective optimisation we have a multi–objective function to optimise under a set of constraints:

$$\max_{\boldsymbol{x}}(f_1(\boldsymbol{x}),\ldots,f_k(\boldsymbol{x})) \quad \text{s.t.} \quad g_1(\boldsymbol{x},\boldsymbol{p}) \leq 0,\ldots,g_l(\boldsymbol{x},\boldsymbol{p}) \leq 0 \tag{1}$$

This problem is well known and a number of approaches, both non–genetic [1] and genetic algorithm [2] approaches exist. An additional problem is that not all objectives are equally important which necessitates the use of weights or preferences. We have applied appropriate genetic algorithms for multi–objective optimisation and described a number of the design problems in [3].

According to the psychological study in [4, p. 254], people generally have problems in reporting accurately on the weights they assign to the various stimulus factors in making evaluations. The designer cannot always completely objectively define the preferences regarding the objectives which have to be optimised. A common situation is a subjective statement "objective A is much more important than objective B" but without any quantitative representation. In this paper we address this problem and integrate our developed preference methods to different preliminary design techniques. Some of the work presented here has been presented in [5].

# 2 Fuzzy preferences

One general problem of the (very popular) weighted sums based multi–objective optimisation approach is how to specify weights. Figure 2.1 illustrates the problem that we face when optimising 2 objectives. The example is based on objectives $y_4$ and $y_9$ of the BAe airframe problem briefly described in section 4.3 and in more detail in [3]. The rightmost point has been obtained by setting the weight $w_4$ of $y_4$ to 0 and the weight $w_9$ of $y_9$ to 1, whereas the leftmost point has been obtained for $(w_4, w_9) = (1, 0)$. The points in–between are obtained by varying $w_4$ and $w_9$ under the constraint $w_4 + w_9 = 1$. Note that in the case of convex Pareto fronts, we can obtain every point of Pareto front just by varying those weights. In this figure there are only two objectives but more usually there are 20 or more objectives to consider at the same time. We will present an approach here that can simplify the weights assignment problem.

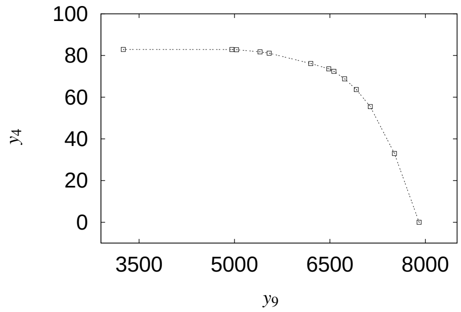



**Figure 2.1:** Influence of weights on BAe function

In our approach, for every two objectives we ask the designer to specify one of the following characterisations:

- Much less important ($\ll$)
- Less important ($\prec$)
- Equally important ($\approx$)
- Don't care (#)
- More important ($\succ$)
- Much more important ($\gg$)

We can easily extend the number of degrees of importance (such as slightly more important, vastly more important etc.).

## 2.1 Treatment of *don't care* relation

In the following text, *don't care* is treated exactly as *equally important*. There is a psychological justification for this: if we do not care in respect of two objectives, then we also do not care which one provides better results. Alternatively, we can use the following: "Given a partial preorder structure, it is always possible to replace the incomparabilities by preferences or indifferences in order to obtain a complete preorder structure" [6, p. 113].

Roy [7] distinguishes two types of hesitations:

**weak preference:** can not decide between $a \succ b$ and $a \approx b$, being sure that $b \not\succ a$;

**incomparability:** can not decide between $a \succ b$ and $b \succ a$.

He states that any hesitation may come from:

- the existence in the decision–maker's mind of zones of uncertainty, half–held belief, or conflicts and contradictions;
- the vaguely defined quality of the decision–maker;
- the fact that the engineers who built the model ignore, in part, how the decision–maker compares $a$ and $b$;
- the imprecision, uncertainty, inaccurate determination of the maps $g(a)$ and $g(b)$ by means of which $a$ and $b$ are compared.

## 2.2 Method used and properties of relations

In general, $k$ objectives require in the worst case $k(k-1)/2$ questions, but usually less is enough to construct the fuzzy preference relation $R$. The question of complexity will be discussed in section 3.2 below.

Once we know the fuzzy preference relation $R \subseteq A^2$, we can compute the complete ordering on $A$ using the relation graph $G = (A, R)$ and the concept of 'leaving score' [8, p. 151] and the corresponding order:

$$S_L(a,R) \stackrel{\text{def}}{=} \sum_{c \in A \setminus \{a\}} R(a,c) \tag{2}$$

$$a \geq_L b \text{ if and only if } S_L(a,R) \geq S_L(b,R) \tag{3}$$

We define the following relations:

| relation | intended meaning |
|---|---|
| $\approx$ | is equally important |
| $\prec$ | is less important |
| $\ll$ | is much less important |
| $\neg$ | is not important |
| $!$ | is important |

The required properties are:

- Relation $\approx$ is an *equivalence relation* (reflexive, symmetric and transitive);
- Relations $\prec$ and $\ll$ are *strict orders* (irreflexive and transitive);
- Relation $\approx$ is *congruent* with $\ll$ and $\prec$;
- Relation $\ll$ is sub–relation of $\prec$ i.e. $x \ll y \Rightarrow x \prec y$;
- Miscellaneous properties:

$$!x \vee \neg x, \quad !y \wedge \neg x \Rightarrow x \ll y, \quad \neg x \wedge \neg y \Rightarrow x \approx y, \quad x \prec y \wedge y \ll z \Rightarrow x \ll z$$

Predicates $\succ$ (is more important) and $\gg$ (is much more important) are then defined in the obvious way.

### 2.2.1 *Some philosophical aspects, contradictions and group preferences*

**Transitivity, yes or no?** Whether preferences should be transitive is a very much discussed question in the Logic of Preferences field [9]. Some of the authors argue that human preferences are not necessarily transitive and give very good examples of non–transitivity of preference relations [10].

However, being pragmatic, we can argue that transitivity reduces the number of questions (e.g. if $A$ is more important than $B$, and $B$ more important than $C$, it is automatically inferred that $A$ is more important than $C$). Transitivity is also used to avoid cyclic preferences $A \prec B$, $B \prec C$, $C \prec A$ or indifferences $A \approx B$, $B \approx C$ and $A \prec C$.

Further, intransitivity of preferences could yield to contradictions. Contradiction (in classical mathematical logic) is something we try to avoid because by having contradictions, every possible conclusion is derivable. There are some branches of logic that try to bypass this situation i.e. *logic of relevance* [11] and *Paraconsistent logics* [12].

**Group preferences** In most real–world cases, the designer is not just *one* person but a group of people usually with different and sometimes contradictory opinions relating to what is important and what is not. This causes an already hard decision–making process to be even harder.

We can use group preferences to aggregate single designer's preferences, but there are some problems as stated by Arrow's Impossibility Theorem [13].

Arrow's problem is roughly as follows: Given the ranking of a set of alternatives by each individual in a decision making group, what should the grouping ranking for these alternatives be? He postulated some very reasonable assumptions concerning the aggregation of individuals' rankings, and then he investigated their composite implications. These assumptions are as follows [14, p. 523]: Complete Domain, Positive Association of Social and Individual Ordering, Independence of Irrelevant Alternatives, Individual's Sovereignty, and Nondictatorship. Arrow proved that there is no rule for combining the individual's rankings that is consistent with these seemingly innocuous assumptions.

One interpretation of the Theorem is that, in general, there is no procedure for combining individual rankings into a group ranking that does not explicitly address the question of interpersonal comparison of preferences.

# 3 Algorithm and Complexity

## 3.1 Description of the algorithm

- Let the set of objectives be $O = \{o_1, \ldots, o_k\}$. Construct the equivalence classes $\{C_i \mid 1 \leq i \leq m\}$ of the equivalence relation $\approx$ and choose one element $x_i$ from each class $C_i$ giving set $X = \{x_1, \ldots, x_m\}$ where $m \leq k$.
- Use the following valuation $v$:
  - If $a \ll b$ then $v(a) = \alpha$ and $v(b) = \beta$

– If $a \prec b$ then $v(a) = \gamma$ and $v(b) = \delta$
– If $a \approx b$ then $v(a) = v(b) = \varepsilon$.[1]

**Note:** Taking into account the intended meaning of the relations, we can assume that $\alpha < \gamma < \varepsilon = 1/2 < \delta < \beta$ and $\alpha + \beta = \gamma + \delta = 1$. We are not limited to constants $\alpha$, $\beta$, $\gamma$, $\delta$ and $\varepsilon$. The above valuation is the simplest possible case but we can as well use the more complex one $\alpha = h(u,t)$ etc, where $u$ is some parameter, $t$ is time (as we can change our preferences as the time goes on), and $h$ is some real-valued function. Their order and property $\alpha + \beta = \gamma + \delta = 1$ is what matters.

- Initialise two matrices $R$ and $R_a$ of size $m \times m$ to the identity matrix $\mathbf{E}_m$. They will be used in the following way:

$$\begin{aligned} x_i \ll x_j &\Leftrightarrow R(i,j) = \alpha, R(j,i) = \beta \Leftrightarrow R_a(i,j) = 0, R_a(j,i) = 2 \\ x_i \prec x_j &\Leftrightarrow R(i,j) = \gamma, R(j,i) = \delta \Leftrightarrow R_a(i,j) = 0, R_a(j,i) = 1 \\ x_i \approx x_j &\Leftrightarrow R(i,j) = \varepsilon, R(j,i) = \varepsilon \Leftrightarrow R_a(i,j) = 1, R_a(j,i) = 1 \end{aligned} \tag{4}$$

**Note:** This valuation already provides an indication of how to generalise preferences to have $s$ stages instead of only 5 (from "much less important" to "much more important"): if $x_i$ is (say) $s'$ times more important the $x_j$, we will simply assign $R_a(i,j) = s'$ and $R_a(j,i) = 0$ etc.

- Perform the following procedure:
  1. For each pair of objectives $x_i$ and $x_j$ s.t. $R_a(i,j) + R_a(j,i) = 0$ and $i \neq j$:
     (a) ask whether $x_i \ll x_j$, $x_i \prec x_j$, $x_j \ll x_i$ or $x_j \prec x_i$ and using (4) set $R_a(i,j)$ and $R_a(j,i)$ accordingly;
     (b) Using Warshall's algorithm [15], compute transitive closure of $R_a$;
  2. Using (4), calculate matrix $R$ from $R_a$;
- For each $x_i \in \mathcal{X}$ compute weight as a normalised leaving score:

$$w(x_i) = \frac{S_L(x_i, R)}{\sum_{x_j \in \mathcal{X}} S_L(x_j, R)}.$$

and for each $y \in \mathcal{C}_i$ set $w(y) = w(x_i)$.

**Example 1** Let $O = \{o_1, \ldots, o_6\}$, and $o_1 \approx o_2$ and $o_3 \approx o_4$. We have

$$\mathcal{C}_1 = \{o_1, o_2\},\ \mathcal{C}_2 = \{o_3, o_4\},\ \mathcal{C}_3 = \{o_5\},\ \mathcal{C}_4 = \{o_6\}$$

and $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$ where $x_i \in \mathcal{C}_i$ for $1 \leq i \leq 4$. Let $R = R_a = \mathbf{E}_4$ — identity $4 \times 4$ matrix.

Suppose that the first question gives the answer $x_2 \ll x_1$, the second question gives the answer $x_3 \prec x_1$ and the third one $x_1 \prec x_4$. The fourth question gives the answer $x_2 \ll x_3$ and since for each pair $(i,j)$ we have $R_a(i,j) + R_a(j,i) \neq 0$, we have enough information (without computing transitive closure we would have to ask 6 questions

[1] Since we work with class represents, this is only possible if $a = b$.

and additionally handle non–consistent answers), and we can construct the matrix $R$. Suppose that $\alpha = 0.05$, $\beta = 0.95$, $\gamma = 0.35$, $\delta = 0.65$ and $\varepsilon = 0.5$. Then

$$R = \begin{bmatrix} \varepsilon & \beta & \delta & \gamma \\ \alpha & \varepsilon & \alpha & \alpha \\ \gamma & \beta & \varepsilon & \gamma \\ \delta & \beta & \delta & \varepsilon \end{bmatrix} = \begin{bmatrix} 0.50 & 0.95 & 0.65 & 0.35 \\ 0.05 & 0.50 & 0.05 & 0.05 \\ 0.35 & 0.95 & 0.50 & 0.35 \\ 0.65 & 0.95 & 0.65 & 0.50 \end{bmatrix}.$$

Further,

$$S_L(x_1,R) = 1.95,\ S_L(x_2,R) = 0.15,\ S_L(x_3,R) = 1.65 \text{ and } S_L(x_4,R) = 2.25$$

and the order of importance is $x_2 \lll x_3 \prec x_1 \prec x_4$. Weights $w_i$ are finally calculated and normalised so that they sum to 1:

$$w_1 = w_2 = 0.2407, w_3 = w_4 = 0.0185, w_5 = 0.2037, w_6 = 0.2778.$$

## 3.2 Scalability and complexity issue

One important question is "how does the preference method scale with increased number of objectives?". It is easy to see that in the worst–case scenario for $k$ objectives we need $n_q^*(k) = k \cdot (k-1)/2$ questions, but usually transitivity and the other preference properties reduce this number. In order to find the average number of questions needed we have performed the following test: for a given number of objectives $k$ random answers to each of the questions is given, and the number of questions needed to construct the complete preference matrix is counted. The tests have been performed for $4 \leq k \leq 100$, each repeated 100 times, and average value $\tilde{n}_q$ and its standard deviation $\sigma_q(k)$ have been calculated. The results are presented in Table 3.1. For a comparison, the maximal theoretical number of questions $n_q^*$ is presented in column 4. As we can see whereas $n_q^* \sim O(k^2)$, we have the following approximate formula

$$\tilde{n}_q \approx (0.85 + \frac{1}{40} \cdot \sqrt{k}) \cdot k \cdot \ln k \tag{5}$$

i.e.

$$\tilde{n}_q \sim O(\sqrt{k} \cdot k \cdot \ln k + k \cdot \ln k) \sim O(\sqrt{k} \cdot k \cdot \ln k) \tag{6}$$

The value of (5) is presented as a last column in Table 3.1.
**Note:** The worst–case scenario (i.e. the designer has to answer all $n_q^*$ questions) happens when the designer answers all the questions with the answers in the 'same direction' (e.g. $y_1 \succ y_2, y_1 \gg y_3, \ldots, y_1 \succ y_k, y_2 \gg y_3, \ldots, y_2 \succ y_k, \ldots, y_{k-1} \succ y_k$) so that the transitivity can not reduce the number of questions. In some cases asking the questions in different order might help e.g. for order $x_1 \succ x_2 \succ x_3$ instead of asking $x_1?x_2$, $x_1?x_3$, $x_2?x_3$ which all give answer $\succ$ we can just ask $x_1?x_2$, $x_2?x_3$ and since both answers are $\succ$ we automatically infer $x_1 \succ x_3$. In this case, using 'depth first' traversing instead of 'breadth first' traversing does help. Of course, we do not know in advance the designer's opinions. Dynamically changing the order of questions as they are being answered is a matter of further research.

| $k$ | $\tilde{n}_q$ | $\sigma_q(k)$ | $n^*_q(k)$ | (5) | $k$ | $\tilde{n}_q$ | $\sigma_q(k)$ | $n^*_q(k)$ | (5) |
|---|---|---|---|---|---|---|---|---|---|
| 4 | 4.36 | 0.6745 | 6 | 4.99 | 18 | 50.04 | 2.6282 | 153 | 49.74 |
| 5 | 6.37 | 0.4852 | 10 | 7.29 | 21 | 62.25 | 2.8793 | 210 | 61.67 |
| 6 | 8.89 | 0.5842 | 15 | 9.80 | 25 | 79.51 | 2.6723 | 300 | 78.46 |
| 7 | 11.78 | 0.7047 | 21 | 12.48 | 30 | 101.48 | 2.3376 | 435 | 100.70 |
| 8 | 14.83 | 1.5378 | 28 | 15.32 | 35 | 125.68 | 3.0281 | 595 | 124.18 |
| 9 | 17.61 | 1.0040 | 36 | 18.29 | 40 | 150.77 | 3.3900 | 780 | 148.75 |
| 10 | 20.84 | 2.0137 | 45 | 21.39 | 50 | 202.42 | 3.2820 | 1225 | 200.84 |
| 11 | 23.80 | 1.2144 | 55 | 24.61 | 60 | 257.78 | 3.2492 | 1770 | 256.38 |
| 12 | 26.48 | 1.1145 | 66 | 27.93 | 75 | 345.49 | 4.7958 | 2775 | 345.35 |
| 15 | 37.30 | 1.7552 | 105 | 38.46 | 100 | 501.31 | 5.6527 | 4950 | 506.57 |

**Table 3.1:** The number of questions $\tilde{n}_q$ needed for $k$ objectives with random answers. Average over 100 runs.

# 4 Applications of preferences

The concept of preferences and of the relative importance of the objectives can be integrated with Genetic Algorithms in at least two different situations:

1. weighted sum based optimisation, and
2. Pareto optimisation.

In both cases the preference method is used to calculate the weights as required.

Our method integrated with weighted sums has a significant advantage over the traditional weighted sum based optimisation methods since the user doesn't have to express the weights quantitatively but qualitatively (within a few categories) which is much easier. More details are given in [5].

However, we are not only limited on objectives: we can also use preferences to assign different priorities and/or importance to constraints, scenarios, agents in agent–based design etc.

## 4.1 Pareto optimisation based methods

Comparing Pareto principle based multi-objective optimisation with lexicographic order based optimisation, we see two extremes concerning the objective importance: in the case of Pareto optimisation, all objectives are considered simultaneously (and equally important) whereas in the case of lexicographical order the first objective is the most important one and only if we get the same results for the first objective, do we then consider the second objective etc. In this section we try to develop an optimisation method based on the Pareto principle where we can specify the relative importance of objectives.

As in the case of weighted sums based methods, relative importance of objectives in this weighted Pareto method could be specified using weights (quantitatively) or they could be combined with the above developed preference method that would translate qualitative specification into quantitative. Without this combination, our modified Pareto method suffers from the same problem as the weighted sum method: how to specify weights in the case of 15–20 or more objectives.

## 4.2 Definition of weighted Pareto method

Lin [16] distinguishes between orders on $k$–dimensional vectors:

$$\boldsymbol{x} \geq \boldsymbol{y} \quad \text{if and only if} \quad (\forall i \leq k)(x_i \geq y_i) \tag{7}$$

$$\boldsymbol{x} \geqslant \boldsymbol{y} \quad \text{if and only if} \quad (\boldsymbol{x} \geq \boldsymbol{y}) \wedge (\exists j \leq k)(x_j > y_j) \tag{8}$$

He notes that the orders (7) and (8) are definable in terms of each other and therefore ordering a set in $\mathbf{R}^k$ by $\geqslant$ is equivalent to ordering a set by $\geq$ [16, p. 46]. With this remark in mind, we define *non–dominance* (usually defined using (8)) in the following way:

**Definition 1** We say that (in object space) the vector $\boldsymbol{x} = (x_1, \ldots, x_k)$ is *non–dominated* by vector $\boldsymbol{y} = (y_1, \ldots, y_k)$, denoted $\boldsymbol{x} \succeq \boldsymbol{y}$, if $x_i \geq y_i$ for all $1 \leq i \leq k$. In other words,

$$\boldsymbol{x} \succeq \boldsymbol{y} \Leftrightarrow \frac{1}{k}\sum_{i=1}^{k} I_{\geq}(x_i, y_i) \geq 1, \qquad \text{where } I_{\geq}(x,y) = \begin{cases} 1, & x \geq y \\ 0, & x < y \end{cases} \tag{9}$$

We can generalise (9) (assuming $\sum_{i=1}^{k} w_i = 1$):

$$\boldsymbol{x} \succeq_w \boldsymbol{y} \text{ if and only if } \sum_{i=1}^{k} w_i \cdot I_{\geq}(x_i, y_i) \geq 1, \tag{10}$$

or we can even put some threshold $\tau \leq 1$:

$$\boldsymbol{x} \succeq_w^{\tau} \boldsymbol{y} \text{ if and only if } \sum_{i=1}^{k} w_i \cdot I_{\geq}(x_i, y_i) \geq \tau. \tag{11}$$

**Note:** The relation $\succeq_{\boldsymbol{w}}^{\tau}$ is transitive as a product of transitive (component–wise) orders and has all the usual features of an order relation. Also, we assume that the weights do not change during the optimisation process.

**Definition 2** We call relation $\succeq_w$ defined by (10) ***w***–*non-dominance* and the relation $\succeq_w^{\tau}$ defined by (11) $(\boldsymbol{w}, \tau)$–*non-dominance*. The *Pareto front* is defined as a maximal set of non–dominated elements (according to a given order $\succeq$) and this definition is naturally extended to $\boldsymbol{w}$–Pareto front and to $(\boldsymbol{w}, \tau)$–Pareto front for a given vector of weights $\boldsymbol{w}$ and threshold $\tau$ i.e. according to the order $\succeq_{\boldsymbol{w}}$ and $\succeq_{\boldsymbol{w}}^{\tau}$ given by (10) and (11) respectively. We assume that at least one of the inequalities is strict.

**Note:** The standard dominance relation is a special case of (11) for $\boldsymbol{w} = (\frac{1}{k}, \frac{1}{k}, \ldots, \frac{1}{k})$ and $\tau = 1$.

Vector $\boldsymbol{w}$ could be either specified directly by the designer or it can be calculated from preferences which would help the designer to work in more qualitative terms without the burden to reason if the weight should be set of 0.1 or to 0.09.

The Pareto front method combined with genetic algorithms is a very powerful optimisation method since it maintains the diversity of population. However, it can be computationally expensive.

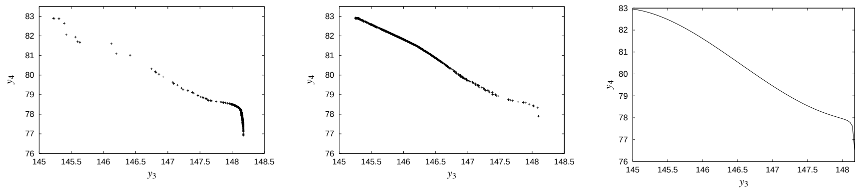
## 4.3 BAe Function and GA/Pareto Optimisation

This section describes applications of the weighted Pareto method described in section 4.2. The British Aerospace (BAe) design problem is presented in [3]. Briefly, we have 9 input variables $x_1, \ldots, x_9$ and 13 objectives $y_1, \ldots, y_{13}$. The interaction between objectives $y_4$ and $y_9$ is specially interesting as they strongly conflict.

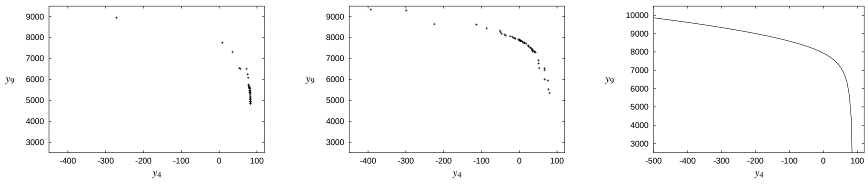
However, the BAe design problem is not only an optimisation problem, in fact optimisation is a rather small part of it. The problems of conceptual design relate to the fuzzy nature of initial design concepts and the many different variants that engineer wishes to try. Computers should be able to help him explore these variants whilst also suggesting some others as well [3, 17]. Therefore, interaction with the design team is very important. Our goal is to assist the designer during conceptual design process more in the sense of MCDA (multiple criteria decision aid), than MCDM (multiple criteria decision making) [18]. See [19] for more details.

The core of the genetic algorithm we have used is based on the Breeder Genetic Algorithm [20]. It utilises genetic operators suitable for real–valued chromosomes (arithmetic crossover, exponential mutation etc.), and is adapted to use techniques for multi–objective optimisation [3].

Figure 4.1(a)&(b) show two $\boldsymbol{w}$–Pareto fronts of $y_3$ versus $y_4$ for different preferences i.e. for different weights, whereas Figure 4.1(c) show the shape of the complete Pareto front. Figure 4.1(d)&(e) show two $\boldsymbol{w}$–Pareto fronts of $y_4$ versus $y_9$ for different preferences i.e. for different weights, whereas Figure 4.1(f) show the shape of the complete Pareto front. Varying the Pareto threshold and the weights of each objective, we can obtain different Pareto fronts. Knowing the behaviour of the Pareto front with respect to those parameters, would enable us to vary the parameters during the genetic algorithm run to identify those parts of the Pareto front that are of special interest (considering the density of points in given regions etc.). There are some other approaches that use biased sharing in order to explore parts of the Pareto front [21].



(a) $y_3 \gg y_4$ (b) $y_3 \ll y_4$ (c) complete front



(d) $y_4 \succ y_9$ (e) $y_4 \ll y_9$ (f) complete front

**Figure 4.1:** (a)–(c) $\boldsymbol{w}$–Pareto front of $(y_3, y_4)$ of the BAe function and complete Pareto front. (d)–(f) $\boldsymbol{w}$–Pareto front of $(y_4, y_9)$ of the BAe function and complete Pareto front.

# 5 Numerical Examples

## 5.1 Example with 8 objectives

Let us consider the case where we are interested in the results of more then 2 objectives. Suppose that we are optimising the following 8 objectives of the BAe function: $y_3, y_4, y_5, y_6, y_7, y_8, y_9, y_{13}$ and that we specify the following answers concerning preferences (those are the exact questions the program asks — there are 6 questions only for 5 distinctive classes of objectives):

$$
\begin{aligned}
&y_3 \approx y_4,\ y_5 \approx y_6,\ y_7 \approx y_8, \\
&y_3 \succ y_5,\ y_3 \gg y_7,\ y_3 \prec y_9,\ y_3 \succ y_{13},\ y_5 \succ y_7,\ y_5 \prec y_{13}
\end{aligned}
$$

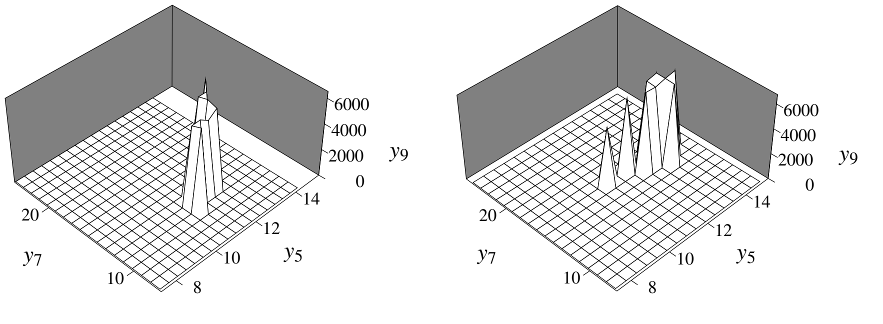
This give us the following preference order:

$$y_9 \succ y_3 \approx y_4 \succ y_{13} \succ y_5 \approx y_6 \gg y_7 \approx y_8$$

and using the same valuation as in Example 1, we have the following weights:

$$w_3 = w_4 = 0.1722,\ w_5 = w_6 = 0.1126,\ w_7 = w_8 = 0.053,\ w_9 = 0.1921,\ w_{13} = 0.1325$$

Performing weighted Pareto GA optimisation (with Pareto set size limited to 400) we end up with an 8–dimensional surface with some 3D slices presented in Figure 5.1. For the comparison, the right hand side contains the same 3D slice without preferences (i.e. using standard Pareto method). Please note that these are just 3–dimensional slices of an 8–dimensional surface. We are still facing the problem of $n$–dimensional surface presentation for $n > 4$. Although we are actually looking at a very small part of the 8D Pareto front, we can nevertheless notice the shift towards the smaller values of $y_7$ in Figure 5.1(a) as the objective $y_7$ is the least important of all. Similarly, the values of $y_5$ are less in Figure 5.1(a) than in Figure 5.1(b). This could be easily explained by noticing that the weight factor for $y_9$ (0.1921) is almost twice the weight of $y_5$ (0.1126) and almost four times the weight of $y_7$ (0.053).




(a) (b)

**Figure 5.1:** 3D slices of 8D Pareto front: (a) $(y_5, y_7, y_9)$ with preferences, (b) $(y_5, y_7, y_9)$, without preferences.

### 5.2 Example with 13 objectives

Another large experiment involves *all* 13 objectives $y_1, y_2, \ldots, y_{12}, y_{13}$ of the BAe function. Suppose that the designer has the following preferences (as asked by the program — 24 questions for 13 objectives (instead of theoretically maximal 78), or 21 question for 10 non–equivalent objectives instead of 45 — agreeing with results from section 3.2):

$y_3 \approx y_4,\ y_5 \approx y_6,\ y_7 \approx y_8,$
$y_1 \succ y_2,\ y_1 \prec y_3,\ y_1 \succ y_5,\ y_1 \succ y_7,\ y_1 \ll y_9,\ y_1 \succ y_{10},\ y_1 \gg y_{11},\ y_1 \succ y_{12},\ y_1 \prec y_{13},$
$y_2 \prec y_5,\ y_2 \succ y_7,\ y_2 \prec y_{10},\ y_2 \prec y_{11},\ y_2 \succ y_{12},\ y_3 \ll y_9,\ y_3 \prec y_{13},$
$y_5 \succ y_{10},\ y_5 \succ y_{11},\ y_7 \succ y_{12},\ y_9 \succ y_{13},\ y_{10} \gg y_{11}$

This gives the following order:

$$y_9 \succ y_{13} \succ y_3 \approx y_4 \succ y_1 \succ y_5 \approx y_6 \succ y_{10} \gg y_{11} \succ y_2 \succ y_7 \approx y_8 \succ y_{12}$$

and the weights vector: (0.1003, 0.0334, 0.1054, 0.1054, 0.0951, 0.0951, 0.0283, 0.0283, 0.1414, 0.09, 0.0386, 0.0231, 0.1157).

The run was performed using a GA with a population size of 500 for 500 generations and limiting the size of Pareto front to 1000. Objectives $\{y_1, y_2, y_{10}, y_{11}, y_{12}\}$ are being minimised, all others are maximised. Each objective is normalised to $[0, 1]$ range. Limited space prohibits the inclusion of graphs similar to those of Figure 5.1, but as before, results ilustrate shift towards the larger values of $y_9$ and $y_5$ in the case of preferences (since $y_9 \succ y_5 \gg y_{12}$).

We must stress that in our method the increase in number of objectives decreases the difference in importance of particular objectives (i.e. the weights–difference of $A$ and $B$ in preference $A \gg B$ is much higher if we are working with two objectives instead of 10) so asymptotically weighted Pareto method performs the same as the ‘classical’ Pareto method i.e. with no preferences at all.

# 6 Conclusion

In this paper one method for transforming qualitative characterisation of objective relative importance into quantitative characterisation is presented. One algorithm is given that implements the transformation. Integration with traditional and GA based multi–objective optimisation method is discussed and a novel Pareto optimisation method combination with weights via preferences developed. Some real–world applications of preferences in the new Pareto based method are presented. Complexity of the procedure is analysed. In the future work we will try to further develop the preference model, to reduce the number of questions, and to integrate it more tightly into the real world applications.

# References


[1] Ching-Lai Hwang and Abu Syed Md. Masud. *Multiple Objective Decision Making – Methods and Applications*. Springer Verlag, Berlin, 1979.

[2] David A. Van Veldhuizen and Gary B. Lamont. Multiobjective evolutionary algorithm research: A history and analysis. Technical Report TR-98-03, Air Force Institute of Technology, Wright–Paterson AFB, October 1998.

[3] Dragan Cvetković, Ian C. Parmee, and Eric Webb. Multi–objective optimisation and preliminary airframe design. In Ian C. Parmee, editor, *Adaptive Computing in Design and Manufacture*, pages 255–267. Springer–Verlag, 1998.

[4] Richard E. Nisbett and Timothy DeCamp Wilson. Telling more then we can know: Verbal reports on mental processes. *Psychological Review*, 84(3):231–259, May 1977.

[5] Dragan Cvetković and Ian C. Parmee. Use of preferences for ga–based multi–objective optimisation. In W. Banzhaf and J. Daida et al., editors, *GECCO–99: Proceedings of the Genetic and Evolutionary Computation Conference*, pages 1504–1510, Orlando, Florida, USA, July 1999. Morgan Kaufmann.

[6] Philippe Vincke. Basic concepts of preference modelling. In Bana e Costa [18], pages 101–118.

[7] Bernard Roy. The outranking approach and the foundation of ELECTRE method. In Bana e Costa [18], pages 156–183.

[8] János Fodor and Marc Roubens. *Fuzzy Preference Modelling and Multicriteria Decision Support*. Kluwer Academic Publishers, Dordrecht, The Netherlands, 1994.

[9] Georg Henrik von Wright. The logic of preference. An essay. Edinburgh, 1963.

[10] A. Tversky. Intransitivity of preferences. *Psychological Review*, 76:31–48, 1969.

[11] A. R. Anderson and N. D. Belnap, Jr. *Etailment. The Logic of Relevance and Necessity*, volume I. Princeton University Press, 1975.

[12] G. Priest, R. Routley, and J. Norman, editors. *Paraconsistent Logic. Essays on the Inconsistent*. Philosophia Verlag, München, Germany, 1989.

[13] K. J. Arrow. *Social Choice and Individual Values*. John Wiley & Sons, 1951.

[14] Ralph L. Keeney, Howard Raifa, and Richard F. Meyer. *Decisions with Multiple Objectives: Preferences and Value Tradeoffs*. John Wiley & Sons, 1976.

[15] Stephen Warshall. A theorem on Boolean matrices. *Journal of the ACM*, 9(1):11–12, 1962.

[16] J. G. Lin. Maximal vectors and multi–objective optimization. *Journal of Optimization Theory and Application*, 18(1):41–64, 1976.

[17] Ian C. Parmee. Exploring the design potential of evolutionary search, exploration and optimisation. In P. J. Bentley, editor, *Evolutionary Design by Computers*, pages 119–143. Morgan Kaufmann, San Francisco, CA, 1999.

[18] Carlos A. Bana e Costa, editor. *Readings in Multiple Criteria Decision Aid*. Springer–Verlag, Berlin, 1990.

[19] Ian C. Parmee, Dragan Cvetković, Andrew H. Watson, and Christopher R. Bonham. Multi–objective satisfaction within an interactive evolutionary design environment. *Evolutionary Computation*, 8(2), 2000.

[20] Heinz Mühlenbein and Dirk Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm I: Continuous parameter optimization. *Evolutionary Computations*, 1(1):25–49, 1993.

[21] Kalyanmoy Deb. Multi–objective evolutionary algorithms: Introducing bias among Pareto–optimal solutions. KanGAL report 99002, Indian Institute of Technology, Kanpur, India, 1999.