In EUFIT'98, Aachen, Germany, pp. 397–401

Evolutionary Design and Multi–objective Optimisation

Dragan Cvetković and Ian Parmee
Plymouth Engineering Design Centre
University of Plymouth
Drake Circus, Plymouth PL4 8AA, UK
Phone: +44 - 1752 233 508, Fax: +44 - 1752 233 529
email: {dcvetkovic,iparmee}@plymouth.ac.uk

ABSTRACT: In this paper we explore established methods for optimising multi-objective functions whilst addressing the problem of preliminary design. Methods from the literature are investigated and new ones introduced. All methods are evaluated within a collaborative project for whole system airframe design and the basic problems and difficulties of preliminary design methodology are discussed (Cvetković, Parmee and Webb 1998).

Our Genetic Algorithm is expanded to integrate different methods for optimising multi–objective functions. All presented methods are also analysed in the context of whole system design, discussing their advantages and disadvantages.

The problem of qualitative versus quantitative characterisation of relative importance of objectives (such as 'objective A is much more important then objective B') in multi–objective optimisation framework is also addressed and some relationships with fuzzy preferences (Fodor and Roubens 1994) and preference ordering established. Several new predicates between objectives ($\prec$, $\ll$, $\approx$ etc.) are introduced (Cvetković and Parmee 1998). Some examples of application in GA based multi–objective framework are also presented.

1 Introduction

In multi–objective optimisation, we have a function to optimise:

Definition 1 Let $n > 0$, $k > 0$, $\mathcal{D} = \mathcal{X}_1 \times \mathcal{X}_2 \times \ldots \times \mathcal{X}_n \subseteq \mathbf{R}^n$, and $\mathcal{R} = \mathcal{Y}_1 \times \mathcal{Y}_2 \times \ldots \times \mathcal{Y}_k \subseteq \mathbf{R}^k$. Let further $f_i : \mathcal{D} \mapsto \mathcal{Y}_i$ for $1 \leq i \leq k$ and finally $\mathbf{F} : \mathcal{D} \mapsto \mathcal{R}$, so that $\mathbf{F}(\boldsymbol{x}) = (f_1(\boldsymbol{x}), \ldots, f_k(\boldsymbol{x}))$.

Our goal is to optimise function $\mathbf{F}(\boldsymbol{x})$ under additional constraints i.e.

$$\max_{\boldsymbol{x}} \mathbf{F}(\boldsymbol{x}) \tag{1}$$

$$g_1(\boldsymbol{x},\boldsymbol{p}) \leq 0, \ldots, g_l(\boldsymbol{x},\boldsymbol{p}) \leq 0 \tag{2}$$

where $\boldsymbol{p} = (p_1, \ldots, p_u)$ are additional (real–valued) parameters. This problem is well known and a number of non–genetic (Hwang and Masud 1979, Osyczka 1984) and genetic algorithm approaches exists (Schaffer 1985, Horn and Nafpliotis 1993, Fonseca and Fleming 1995). An additional problem is that not all objectives are equally important which necessitates the use of weights and preferences.

However, from previous experience and initial discussion with the industrial partner the problem becomes even harder due to the following additional meta–objectives (Cvetković et al. 1998, Parmee 1998):

- We have objectives and we have constraints. The difference between them is very fuzzy and some of them will move from objectives to constraints or vice versa. Some constraints are hard, some not; some will change or disappear whilst others may be introduced as the problem knowledge base expands.
- In many cases the variable ranges are fuzzy and there is a requirement for exploration outside of the default regions.
- The output should contain both optimal solutions and suggestions for extending ranges and/or inclusion/removals of constraints.
- A set of results is required which the engineer can analyse off-line.

- The end–user (designer) should not be confused with the number of parameters and possibilities the (optimisation) program offers, we must avoid cognitive overload.
- The engineer wishes to interact with the search process by sampling results after N functions evaluations, and adapting parameters and/or constraints.

1.1 The BAe model

Our project is based on a collaboration with British Aerospace (BAe) and utilises their CAPS system. CAPS (Computer aided project studies) is an integrated computer software suite, developed at BAe for use by engineers and designers during the earliest investigation stages of a new aircraft project. BAe has developed a MINICAPS model based upon the full CAPS representation for use by the PEDC. At the moment there are 8 input and 9 output parameters (i.e. $n = 8$ and $k = 9$ in Definition 1). All constraints so far are on variable domains. In the further text we will usually denote input variables by $x_1,\ldots,x_8$ and the objectives $y_1,\ldots,y_9$.

The following sections describe some well established multi–objective optimisation methods and discuss their advantages and disadvantages concerning preliminary design.

2 Optimisation methods

2.1 Weighted sums

Weighted sums is a method of scalarisation of vector functions (Hwang and Masud 1979, p.32), (Osyczka 1984): Instead of the function $\mathbf{F}(\boldsymbol{x})$ from Definition 1, we optimise the function $\mathbf{F}_{\boldsymbol{w}}(\boldsymbol{x}) = \sum_{i=1}^{k} w_i \cdot f_i(\boldsymbol{x})$ for some weights $\boldsymbol{w} = (w_1,\ldots,w_k)$.

Usually one assigns weights according to the importance of objectives: more important objective will get a higher weight, less important objectives will get lower weights. However, since not all the objectives have the same range of values, they must further be normalised. A further problem is distinguishing between maximisation and minimisation of objectives. See (Cvetković et al. 1998) for details.

Advantages using weights:

- Reduces multi–objective function to a single–objective function;
- Can use traditional optimisation methods.

Disadvantages using weights:

- Problem how to set weights — results are very sensitive to weights ratio;
- Must normalise objectives — computationally expensive in general;
- Not suitable for preliminary design where fitness landscape changes a lot.

The last disadvantage is a very serious one. In the conceptual design phase, the probability of objectives and constraints variation is high, and thus the fitness landscape will change therefore, necessitating the re–calculation of our weights again and again. So, this method is not very feasible for our purpose. We will however use it in the further text in order to demonstrate different techniques and results obtained using them.

2.2 Lexicographic method

The lexicographic method for multi-objective optimisation is based on lexicographic ordering (Ben-Tal 1979):

Definition 2 We will say that point $\boldsymbol{x} \in \mathbf{R}^n$ is *less in lexicographic order* then the point $\boldsymbol{y} \in \mathbf{R}^n$, written $\boldsymbol{x} \preceq \boldsymbol{y}$ if either $\boldsymbol{x} = \boldsymbol{y}$ or the first nonzero component of $\boldsymbol{x} - \boldsymbol{y}$ is negative.

Obviously, since the sorting of individuals is done using relation $\preceq$, the first objective is the most important one, the second objective is a bit less important etc. so one has to be very aware of the importance of the particular objectives. This method is quite opposite the to Pareto sorting method, presented in the following section, that treats all objectives equally.

Advantages of using lexicographic order:

- Lexicographic order is a total order — there is only one maximum according to this order;
- It is easy to implement — sequential optimisation: first the most important one, then from the set that satisfy the first objective optimise on second etc;

Disadvantages of using lexicographic order:

- Importance of each objective is crucial;
- Relative importance of each objective must be known from the very beginning.

The last one is also a serious drawback in preliminary system design where the designer doesn't always know the importance of particular objective. One usually uses *random lexicographic order*: the order of objectives we sort on is chosen randomly and varies from generation to generation. We have implemented this method within a GA and some results are presented in Table 1.

2.3 Pareto optimisation

Definition 3 Consider a vector function $\mathbf{F} : \mathcal{D} \mapsto \mathcal{R}$ from Definition 1. We will say that a point $\boldsymbol{x} \in \mathcal{D}$ *Pareto–dominates* a point $\boldsymbol{y} \in \mathcal{D}$ with respect to function $\mathbf{F}$, denoted $\boldsymbol{y} \leq^P_{\mathbf{F}} \boldsymbol{x}$, if $\wedge_{i=1}^{k}(f_i(\boldsymbol{y}) \leq f_i(\boldsymbol{x}))$ and at least one of inequalities is strict.

We say that point $\boldsymbol{x}_P \in \mathcal{D}$ is *Pareto–optimal* or *non–dominated* (for a given function $\mathbf{F}$) if there is no point $\boldsymbol{y} \in \mathcal{D}$ that Pareto–dominates $\boldsymbol{x}$ i.e. $(\neg\exists \boldsymbol{y} \in \mathcal{D})(\boldsymbol{x}_P \leq^P_{\mathbf{F}} \boldsymbol{y})$

Set $\mathcal{F} \subseteq \mathcal{D}$ is called the *Pareto front* with respect to function $\mathbf{F}$ if every element $\boldsymbol{x} \in \mathcal{F}$ is Pareto optimal with respect to function $\mathbf{F}$. In other words, the Pareto front is the maximal set of non–dominated elements.

In the following text we will drop indices $\mathbf{F}$ and P and will just write $\leq$ instead of $\leq^P_{\mathbf{F}}$.

Advantages of Pareto method:

- Takes all objectives into consideration simultaneously;
- Every element of Pareto front is a good solution;
- Maintains the diversity of solutions;

Disadvantages of Pareto method:

- Computationally expensive: to maintain the front, we need to store and constantly update a large set of solutions;
- Not very good and intuitive if the number of objectives is large;

We have used the Pareto method with Genetic Algorithms in the selections phase in the following way:

Pareto tournament As standard tournament selection but the chosen individual will be according to Pareto ordering;

Pareto sort Sort according to Pareto ordering and then, if both individuals are dominant (or non dominant), sort according to the fitness value;

Pareto truncation Choose parents only among the non–dominated parents.

It is interesting to observe how the number of non-dominated elements in the population increases. Figure 1, presents a diagram of number of non–dominated elements in population versus generation number. Population size used is 100. We have used standard tournament of size 2 as a selection method. The results are averaged over 50 runs and standard deviation is presented using error bars. We were optimising on y_3, y_4 and y_9 simultaneously. What we can see immediately is that after some 150 generations more then 50% of the populations is formed by non–dominated individuals and that the selection pressure gradually decreases with the number of generations.

In the case of conceptual design this means that the Pareto front (the set the designer is usually interested in) will be much too large for a human to handle. Even with a small population of 100 individuals, after 400 generations, the size of the Pareto front increases to 80 i.e. 80 solutions (more or less different) to deal with. One method for dealing with this problem is using some form of Pareto ranking. Two different methods are described in the section 2.3.1.

2.3.1 **Pareto ranking**

Instead of just a dominant/non-dominant scheme for Pareto sorting that doesn't distinguish elements very well (as we can see from the Figure 1 where at the end of the run more then 80% are non–dominated) and because of the arguments presented above, we can use a finer method: Pareto ranking (Cvetković et al. 1998).

Definition 4 *Pareto rank r* in a set $\mathbf{X} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\}$ is assigned in the following way:

$$
\begin{array}{l}
(\forall \boldsymbol{x} \in \mathbf{X}) r(\boldsymbol{x}) \leftarrow 1 \\
(\forall \boldsymbol{x} \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\})(\forall \boldsymbol{y} \in \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n\} \setminus \{\boldsymbol{x}\}) \text{ // sequentially!} \\
\quad \text{If } (r(\boldsymbol{x}) = r(\boldsymbol{y}) \wedge \boldsymbol{x} > \boldsymbol{y})\ r(\boldsymbol{x}) \leftarrow r(\boldsymbol{x}) + 1 \\
\quad \text{If } (r(\boldsymbol{x}) = r(\boldsymbol{y}) \wedge \boldsymbol{x} < \boldsymbol{y})\ r(\boldsymbol{y}) \leftarrow r(\boldsymbol{y}) + 1
\end{array}
$$

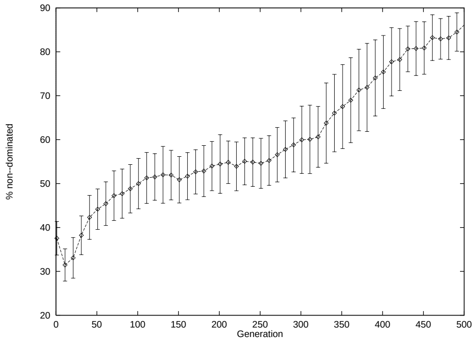

Figure 1: Non–dominated part of the population.

An alternative ranking method is the following one (Srinivas and Deb 1995):

1. At first stage, all non–dominated elements are assigned the maximal rank M and removed from the set;
2. At the second stage, all the non-dominated elements of the remaining set are assigned rank M-1 and removed;
3. etc. until the remaining set is empty.

2.4 Results

If we use weights and the method of positive ideal solution for assigning weights, we get the results shown in Table 1. Because of the interaction between objectives, we have put additional constraints that every objective that we optimise on must be greater then 0.001. We present average $\bar{y}$ and standard deviation $\sigma(y)$ over 50 runs.

Those results clearly show that choosing different optimisation methods and different objectives can give totally different results that are, globally, not necessary better or worse, just different.

What?	How?	$\bar{y}_3$	$\sigma(y_3)$	$\bar{y}_4$	$\sigma(y_4)$	$\bar{y}_9$	$\sigma(y_9)$
	PAR	119.72	4.27	0.68	3.65	7747.15	134.69
y_3, y_4, y_9	LEX	115.61	5.17	0.42	1.33	7826.03	100.81
	WEI	146.84	≈ 0	76.93	≈ 0	6070.39	0.64
	PAR	5.62	6.27	-459.52	28.27	10220.1	43.41
y_3, y_9	LEX	0.34	0.64	-482.91	2.72	10249.8	10.47
	WEI	139.56	0.08	52.26	0.19	6959.5	5.93
	PAR	117.33	4.41	0.29	1.19	7811.8	106.24
y_4, y_9	LEX	113.68	5.35	0.17	0.80	7846.14	84.12
	WEI	145.72	≈ 0	76.16	0.002	6198.94	0.29
	PAR	-1.74	0.02	-491.83	1.25	10262.8	0.96
y_9	LEX	-1.74	0.01	-491.98	0.79	10263.4	0.45
	WEI	-1.74	0.01	-492.19	0.81	10263.4	0.31
y_1	PAR	59.67	≈ 0	9.81	≈ 0	4787.75	0.02
...	LEX	59.67	≈ 0	9.81	≈ 0	4787.73	0.03
y_9	WEI	114.66	0.16	37.19	0.21	5769.41	15.22

Table 1: Results for Pareto (PAR), random lexicographic (LEX) and weighted sum (WEI) optimisations.

3 Preferences and their applications

We are addressing the fuzziness in the relative importance specification and issues such as

- Objective A is much more important then objective B ($A \gg B$)
- Objective A is more important then objective B ($A \succ B$);
- Objective A is equally important to objective B ($A \approx B$) etc.

This method could be used together with weighted sums based optimisation methods or with Pareto based optimisation (Cvetković and Parmee 1998). We will require the following basic properties of those relations:

- Relation $\approx$ is an equivalence relation;
- Relations $\succ$ and $\gg$ are (strict) orders;
- Relation $\gg$ is sub-relation of relation $\succ$;
- etc. See (Cvetković and Parmee 1998) for details and the full list of properties.

After establishing the relative importance of the objectives, ordering between the objectives could be computed using so called 'leaving score' of the relation R (Fodor and Roubens 1994, p. 151) and the order induced by it:

$$S_L(a,R) \stackrel{\text{def}}{=} \textstyle\sum_{c\in A\setminus\{a\}} R(a,c)$$
$$a \geq_L b \text{ if and only if } S_L(a,R) \geq S_L(b,R)$$

Example 1 Suppose that we are optimising BAe function but interested only on objectives z_3, z_4, z_7, z_8 and z_9 (where z_i is the objective y_i normalised to $[0,1]$) and that further we have the following preferences:

$$z_3 \approx z_4,\ z_7 \approx z_8,\ z_3 \gg z_7,\ z_9 \succ z_3.$$

We get the following ordering:

$$z_9 \succ \{z_3,z_4\} \gg \{z_7,z_8\} \gg \{z_1,z_2,z_5,z_6\}$$

where all elements in $\{,\}$ are equally important ($z_3 \approx z_4$, etc.).

Using the algorithm from (Cvetković and Parmee 1998) and leaving score order, we get the following weights: $w(z_9) = 0.3636$, $w(z_3) = w(z_4) = 0.2995$, $w(z_7) = w(z_8) = 0.0227$, $w(z_1) = w(z_2) = w(z_5) = w(z_6) = 0$.

4 Conclusions

In this paper we have addressed some issues concerning multi–objective optimisation from the aspect of preliminary design. Some techniques are explored and their advantages and disadvantages discussed and results on BAe function presented. Preferences for ranking objectives have been introduced and some basic properties have been presented. One example show how they could be used. Their application in weighted sums and Pareto based multi–objective optimisation will be a matter of further investigations.

Acknowledgements: This work is supported by UK Engineering and Physical Sciences Research Council (EPSRC) and British Aerospace (BAe). We thank them for the continuing support.

References

Ben-Tal, A.: 1979, Characterisation of Pareto and lexicographic optimal solutions, *in* G. Fandel and T. Gal (eds), *Proceedings of the Third Conference on Multiple Criteria Decision Making Theory and Application*, number 177 in *Lecture Notes in Economics and Mathematical Systems*, Springer Verlag, pp. 1–11.

Cvetković, D. and Parmee, I. C.: 1998, Preferences and weights for ga–based multi-objective optimisation, *Technical report*, PEDC, University of Plymouth, UK.

Cvetković, D., Parmee, I. C. and Webb, E.: 1998, Multi–objective optimisation and preliminary airframe design, *in* I. C. Parmee (ed.), *Adaptive Computing in Design and Manufacture: The Integration of Evolutionary and Adaptive Computing Technologies with Product/System Design and Realisation*, Springer–Verlag, pp. 255–267.

Fodor, J. and Roubens, M.: 1994, *Fuzzy Preference Modelling and Multicriteria Decision Support*, Kluwer Academic Publishers.

Fonseca, C. M. and Fleming, P. J.: 1995, An overview of evolutionary algorithms in multiobjective optimization, *Evolutionary Computation* **3**(1), 1–16.

Horn, J. and Nafpliotis, N.: 1993, Multiobjective optimization using the niched Pareto genetic algorithm, *Technical Report IlliGAL Report No. 93005*, Illinois Genetic Algorithm Laboratory.

Hwang, C.-L. and Masud, A. S. M.: 1979, *Multiple Objective Decision Making – Methods and Applications*, Springer Verlag, Berlin.

Osyczka, A.: 1984, *Multicriterion Optimization in Engineering with FORTRAN Programs*, Ellis Horwood.

Parmee, I. C.: 1998, Exploring the design potential of evolutionary search, exploration and optimisation, *in* P. Bentley (ed.), *Evolutionary Design by Computers*, John Wiley. In press.

Schaffer, J. D.: 1985, Multiple objective optimization with vector evaluated genetic algorithms, *in* J. J. Grefenstette (ed.), *Proceedings of the First International Conference on Genetic Algorithms*, Lawrance Erlbaum Associates, pp. 93–100.

Srinivas, N. and Deb, K.: 1995, Multiobjective optimization using nondominant sorting in genetic algorithms, *Evolutionary Computation* **2**(3), 221–248.