Research Paper

Struct Multidisc Optim 29, 149–158 (2005)
DOI 10.1007/s00158-004-0465-1

Adaptive weighted-sum method for bi-objective optimization: Pareto front generation[⋆]

I.Y. Kim and O.L. de Weck

Abstract This paper presents a new method that effectively determines a Pareto front for bi-objective optimization with potential application to multiple objectives. A traditional method for multiobjective optimization is the weighted-sum method, which seeks Pareto optimal solutions one by one by systematically changing the weights among the objective functions. Previous research has shown that this method often produces poorly distributed solutions along a Pareto front, and that it does not find Pareto optimal solutions in non-convex regions. The proposed adaptive weighted sum method focuses on unexplored regions by changing the weights adaptively rather than by using a priori weight selections and by specifying additional inequality constraints. It is demonstrated that the adaptive weighted sum method produces well-distributed solutions, finds Pareto optimal solutions in non-convex regions, and neglects non-Pareto optimal solutions. This last point can be a potential liability of Normal Boundary Intersection, an otherwise successful multiobjective method, which is mainly caused by its reliance on equality constraints. The promise of this robust algorithm is demonstrated with two numerical examples and a simple structural optimization problem.

Key words multiobjective optimization, weighted sum method, adaptive algorithms, Normal Boundary Intersection (NBI), truss optimization, Pareto front generators

Received: 31 October 2003
Revised manuscript received: 11 June 2004
Published online: 7 September 2004

I.Y. Kim and O.L. de Weck[✉]

Department of Aeronautics & Astronautics, Engineering Systems Division, Massachusetts Institute of Technology, Cambridge, Massachusetts 02139, USA
e-mail: kiy@mit.edu, deweck@mit.edu

[⋆] Presented as paper AIAA-2004-1680 at the 45th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Palm Springs, California, April 19–22, 2004

1 Introduction

This section provides a brief review of multi-objective optimization and discusses the shortcomings of the weighted-sum method which is still – and may always remain – the most frequently used technique.

1.1 Multiobjective optimization

Engineering design often deals with multiple, possibly conflicting, objective functions or design criteria. For example, one may want to maximize the performance of a system while minimizing its cost. Such design problems are the subject of multiobjective optimization and can generally be formulated as a Multiple Objective Nonlinear Program (MONLP) of the form:

$$\begin{aligned}
&\min\ \mathbf{J}(\mathbf{x},\mathbf{p})\\
&\text{s.t.}\ \ \mathbf{g}(\mathbf{x},\mathbf{p}) \le 0\\
&\mathbf{h}(\mathbf{x},\mathbf{p}) = 0\\
&x_{i,LB} \le x_i \le x_{i,UB}\ \ (i=1,\dots,n)\\
&\mathbf{J} = [J_1(\mathbf{x}) \cdots J_z(\mathbf{x})]^T\\
&\mathbf{x} = [x_1 \cdots x_i \cdots x_n]^T\\
&\mathbf{g} = [g_1(\mathbf{x}) \cdots g_{m_1}(\mathbf{x})]^T\\
&\mathbf{h} = [h_1(\mathbf{x}) \cdots h_{m_2}(\mathbf{x})]^T
\end{aligned} \tag{1}$$

where $\mathbf{J} = [J_1, J_2, \dots, J_z]^T$ is an objective function vector, $\mathbf{x}$ is a design vector, $\mathbf{p}$ is a vector of fixed parameters, $\mathbf{g}$ is an inequality constraint vector, and $\mathbf{h}$ is an equality constraint vector. In this case there are z objectives, n design variables, m_1 inequality constraints and m_2 equality constraints. Additionally, the design variables may be bounded by side constraints assuming that $x_i \in \mathbb{R}$.

The most popular way of solving the MONLP or vector minimization problem is to reduce it to a scalar problem of the form:

$$\min \tilde{J} = \sum_{i=1}^{z} \frac{\lambda_i}{sf_i} J_i \tag{2}$$

where $\tilde{J}$ is an aggregated, weighted sum of the individual objectives and sf_i and λ_i are the scale factor and weight of the i-th objective, respectively. Typically, weights are chosen such that $\sum_{i=1}^{z} \lambda_i = 1$ and $\lambda_i \geq 0$ leading to a convex combination of objectives. The special case of two objectives is the focus of this paper.

1.2 Literature review

After Pareto (1906) introduced the concept of non-inferior solutions in the context of economics, Stadler (1979, 1984) began to apply the notion of Pareto optimality to the fields of engineering and science in the 1970s. The applications of multiobjective optimization in engineering design grew over the following decades. One of the most widely used methods for solving multiobjective optimization problems is to transform a multiple objective (vector) problem into a series of single-objective (scalar) problems, see (2). When an appropriate set of solutions is obtained by the single-objective optimizations, the solutions can approximate a Pareto front ($z = 2$) or Pareto surface ($z > 2$) in objective space. The weighted-sum method is a traditional, popular method that parametrically changes the weights among objective functions to obtain the Pareto front. Initial work on the weighted-sum method can be found in Zadeh (1963) with many subsequent applications and citations. Koski (1988), for example, studied the weighted sum method in the context of multicriteria truss optimization. Multiobjective optimization applications on aircraft control systems design can be found in Schy and Giesy (1988).

Marglin (1967) developed the ε-constraint method, where one individual objective function is minimized with an upper level constraint imposed on the other objective functions (Steuer 1986). Lin (1976) developed the equality constraint method that minimizes objective functions one by one by simultaneously specifying equality constraints on the other objective functions. Heuristic methods are also used for multiobjective optimization; Suppapitnarm *et al.* (1999) applied simulated annealing to multiobjective optimization, and multiobjective optimization by Genetic Algorithms can be found in Goldberg (1989), Fonseca and Fleming (1995), and Tamaki *et al.* (1996) among others. Messac and Mattson (2002) used physical programming for generating a Pareto front, and they (2003) introduced the concept of s-Pareto fronts for concept selection. Messac and Mattson (2002, 2004) also developed the normal constraint method, which generates evenly distributed Pareto solutions along the entire Pareto front for n-dimensional problems. Das and Dennis (1998) proposed the Normal Boundary Intersection (NBI) method where a series of single-objective optimizations is solved on normal lines to the utopia line. The NBI method gives fairly uniform solutions and can treat problems with non-convex regions on the Pareto front. It achieves this by imposing equality constraints along equally spaced lines or hyperplanes in the multidimensional case.

As discussed in a number of studies by Messac and Mattson (2002), Das and Dennis (1997), and Koski (1985), the traditional weighted-sum approach has two main drawbacks. First, an even distribution of the weights among objective functions does not always result in an even distribution of solutions on the Pareto front. Indeed, in real applications, solutions quite often appear only in some parts of the Pareto front, while no solutions are obtained in other parts. Second, the weighted-sum approach cannot find solutions on non-convex parts of the Pareto front, although such non-dominated solutions (Pareto optimal solutions) do often exist. This is due to the fact that the weighted-sum method is often implemented as a convex combination of objectives, where the sum of all weights is constant and negative weights are not allowed. Increasing the number of weights by reducing the step size does not solve this problem. Eventually, this may result in selection of an inferior solution by missing important solutions in the non-convex regions.

Despite the drawbacks aforementioned, it is true that the weighted-sum approach is extensively used because it is simple to understand and easy to implement. Also, the weight itself reflects the relative importance (preference) among the objective functions under consideration.

We propose a new adaptive method, based on the weighted-sum approach, for multiobjective optimization. In this approach, the weights are not predetermined, but they evolve according to the nature of the Pareto front of the problem. Starting from a large weight step size, $\Delta\lambda$, a coarse representation of the solution is generated and regions where more refinement is needed are identified. The specific regions are then designated as a feasible region for sub-optimization by imposing inequality constraints in the objective space. The typical weighted-sum multiobjective optimization is performed in these regions. When all the regions of the Pareto front reach a pre-specified resolution, the algorithm terminates. The methodology is formulated and demonstrated for bi-objective optimization where there are two objective functions. The potential for extension to a greater number of objectives is briefly discussed.

2 Adaptive weighted-sum method: overview

2.1 Fundamental concepts

Figure 1 shows the concept of the adaptive weighted-sum (AWS) method, compared with the typical weighted-sum approach. The true Pareto front is represented by a solid line, and the solution points obtained by multiobjective

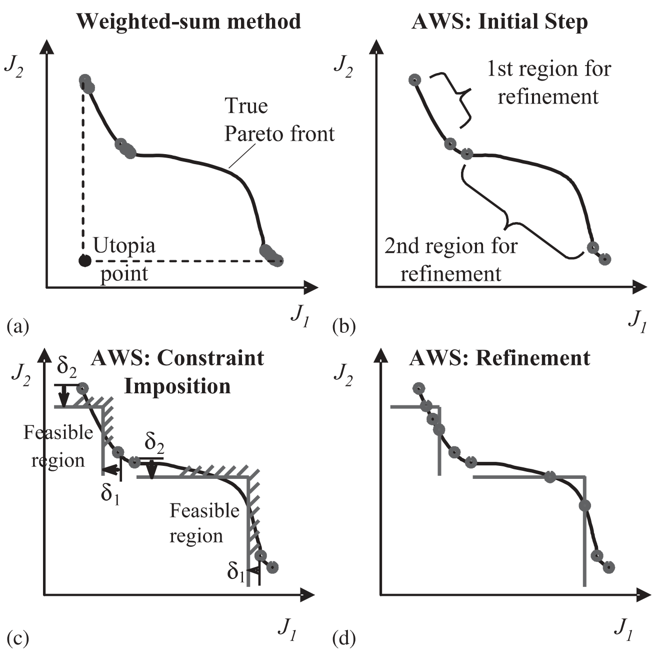


Fig. 1 (a) Weighted-sum method, (b) Initial step of AWS, (c) AWS constraint imposition, (d) Pareto front refinement

optimization are denoted by round black dots. In this example, the whole Pareto line is composed of two parts: a relatively flat convex region and a distinctly concave region. A typical way to solve the problem is to use the weighted-sum method, which is stated as:

$$\begin{aligned}&\min\ \lambda\frac{J_1(\mathbf{x})}{sf_{1,0}(\mathbf{x})}+(1-\lambda)\frac{J_2(\mathbf{x})}{sf_{2,0}(\mathbf{x})}\\&\text{s.t. } h(\mathbf{x})=0 \text{ and } g(\mathbf{x})\le 0\\&\text{and } \lambda\in[0,1]\end{aligned} \tag{3}$$

where J_1 and J_2 are two objective functions to be mutually minimized, $sf_{1,0}$ and $sf_{2,0}$ are normalization factors for J_1 and J_2, respectively, and λ is the weighting factor which reveals the relative importance between J_1 and J_2.

When the typical weighted-sum method is used, as shown in Fig. 1(a), most solutions concentrate near the anchor points and the inflection point, and no solutions are obtained in the concave region. The figure illustrates the two typical drawbacks of the weighted-sum method:

– Generally, the solutions are not uniformly distributed.
– The weighted-sum method cannot find solutions that lie in non-convex regions of the Pareto front. Increasing the number of steps of the weighting factor does not resolve this problem.

These are the main reasons that restrict the usage of the weighted-sum method despite its simplicity and insight into the relative importance among objective functions. The ill-behaved nature of the method is frequently observed in realistic design optimization problems.

Figure 1(b)–(d) illustrates the fundamental concepts and overall procedure of the proposed adaptive weighted-sum method. It starts from a small number of divisions with a large step size of the weighting factor, λ, using the traditional weighted-sum method (Fig. 1(b)). By calculating the distances between neighboring solutions on the front in objective space, regions for further refinement are identified. Only these regions then become the feasible regions for optimization by imposing additional inequality constraints in the objective space (Fig. 1(c)). Each region has two additional constraints that are parallel to each of the objective function axes. The constraints are constructed such that their distances from the solutions are δ_1 and δ_2 in the inward direction of J_1 and J_2, respectively. A sub-optimization is solved in each of the regions using the traditional weighted-sum technique, and a new solution set is identified. Again, regions for further refinement are selected by computing the distances between two adjacent solutions (Fig. 1(d)). The procedure is repeated until a termination criterion is met. The maximum segment length along the entire Pareto front is one measure for the convergence. The detailed procedure is elaborated in the following section.

2.2 Detailed discussion

The adaptive weighted-sum method can effectively solve multiobjective optimization problems whose Pareto front has (i) convex regions with non-uniform curvature, (ii) non-convex regions of non-dominated solutions, and (iii) non-convex regions of dominated solutions. First, for a multiobjective optimization problem of non-uniform curvature Pareto front, most solutions obtained with the usual weighted-sum method are concentrated in the region whose curvature is relatively high. Figure 2(a) shows that very few solutions are obtained in the flat region when the usual weighted-sum method is used. Because the segment length between $\mathbf{P_1}$ and $\mathbf{P_2}$ is larger than others, a feasible region for further refinement is established in the segment, in the adaptive weighted-sum method. An optimization is then conducted only within this region, and more Pareto optimal solutions are ob-

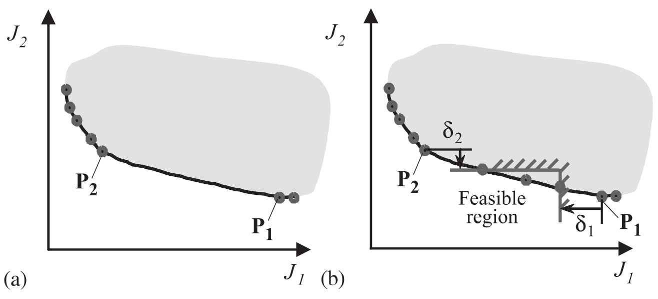


Fig. 2 Adaptive weighted-sum method for convex Pareto front: (a) solutions with weighted-sum method only, (b) additional refinement with AWS

tained there. This makes the distribution of solutions more uniform, as shown in Fig. 2(b).

In the second case of a non-convex region containing non-dominated solutions, there exist Pareto optimal solutions in the region that the usual weighted-sum approach cannot reach. In Fig. 3(a), no solutions are obtained between $\mathbf{P}_1$ and $\mathbf{P}_2$ if the usual weighted-sum method is used. On the other hand, the adaptive weighted-sum method finds solutions because the optimization is conducted only in the non-convex region, as shown in Fig. 3(b). The region is explored by imposing inequality constraints that are offset from $\mathbf{P_1}$ and $\mathbf{P_2}$ by distances δ_1 and δ_2 in the direction of J_1 and J_2, respectively. In this case, only two solutions are obtained at the points where the Pareto front and the inequality constraints intersect.

In the third case of concave regions containing only dominated solutions, there are no Pareto optimal solutions in the region between $\mathbf{P}_1$ and $\mathbf{P}_2$, as shown in Fig. 4. No solution must be identified between $\mathbf{P}_1$ and $\mathbf{P}_2$ in this case. Indeed, the adaptive weighted-sum method does not return solutions in this case, because there is no feasible region within the imposed constraints, whereas the normal boundary intersection (NBI) method typically produces dominated solutions in this case.

In summary, the adaptive weighted-sum method produces evenly distributed solutions, finds Pareto optimal solutions in non-convex regions, and neglects non-Pareto optimal solutions in non-convex regions.

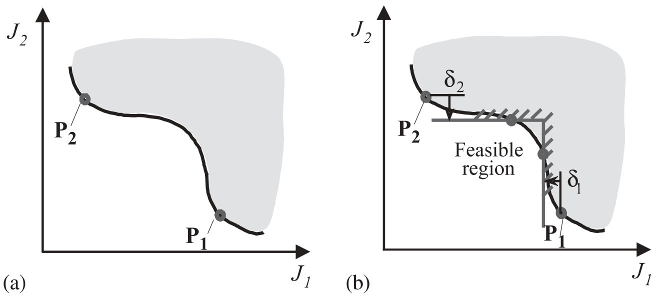


Fig. 3 Adaptive weighted-sum method for non-convex Pareto regions of non-dominated solutions: (a) original solutions, (b) additional solutions obtained with AWS

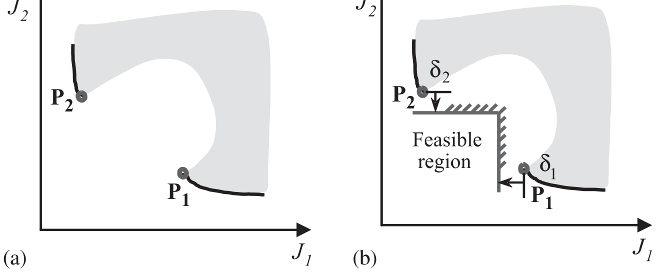


Fig. 4 Adaptive weighted-sum method for non-convex Pareto regions of dominated solutions: (a) original solutions, (b) no additional solutions found

3 Adaptive weighted-sum method: procedures

In this section, the detailed procedure for implementing the adaptive weighted-sum method is described. The description is valid for the bi-objective case.

Step 1: Normalize the objective functions in the objective space. When $\mathbf{x}^{i*}$ is the optimal solution vector for the single-objective optimization of J_i, the normalized objective function $\bar{J}_i$ is obtained as,

$$\bar{J}_i = \frac{J_i - J_i^U}{J_i^N - J_i^U} \tag{4}$$

where $\mathbf{J}^U$ is the utopia point, defined as

$$\mathbf{J}^U = \left[J_1\left(\mathbf{x}^{1*}\right) \quad J_2\left(\mathbf{x}^{2*}\right) \right] , \tag{5}$$

and $\mathbf{J}^N$ is the nadir point, defined as

$$J_i^N = \max \left[J_i\left(\mathbf{x}^{1*}\right) \quad J_i\left(\mathbf{x}^{2*}\right) \right] . \tag{6}$$

Step 2: Perform multiobjective optimization using the usual weighted-sum approach with a small number of divisions, n_{initial}. The uniform step size of the weighting factor λ is determined by the number of divisions:

$$\Delta\lambda = \frac{1}{n_{\text{initial}}} \tag{7}$$

By using a large step size of the weighting factor, $\Delta\lambda$, a small number of solutions is obtained.

Step 3: Compute the lengths of the segments between all the neighboring solutions. Delete nearly overlapping solutions. It occurs often that several nearly identical solutions are obtained when the weighted-sum method is used. The Euclidian distances between these solutions are nearly zero, and among these, only one solution is needed to represent the Pareto front. In the computer implementation, if the distance among solutions is less than a prescribed distance (ε), then all solutions except one are deleted.

Step 4: Determine the number of further refinements in each of the regions. The longer the segment is, the more it needs to be refined. The refinement is determined based on the relative length of the segment:

$$n_i = round\left(C \frac{l_i}{l_{avg}} \right) \quad \text{for the } i\text{th segment} \tag{8}$$

where n_i is the number of further refinements for the ith segment, l_i is the length of the ith segment, l_{avg} is the average length of all the segments, and C is a constant of the algorithm. The function '*round*' rounds off to the nearest integer.

Step 5: If n_i is less than or equal to one, no further refinement is conducted in the segment. For other segments whose number of further refinements is greater than one, go to the following step.

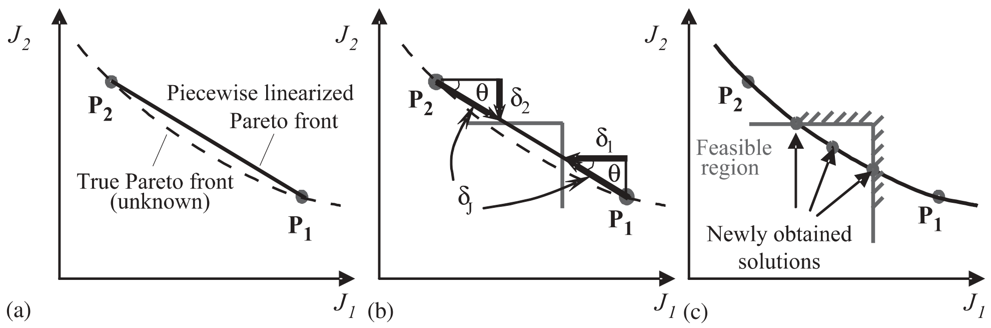


Fig. 5 Determining the offset distances, δ_1 and δ_2, based on δ_J

Step 6: Determine the offset distances from the two end points of each segment. First, a piecewise linearized secant line is made by connecting the end points, $\mathbf{P}_1$ and $\mathbf{P}_2$, see Fig. 5(a). Then, the user selects the offset distance along the piecewise linearized Pareto front, δ_J. The distance δ_J determines the final density of the Pareto solution distribution, because it becomes the maximum segment length during the last phase of the algorithm.

In order to find the offset distances parallel to the objective axes, the angle θ in Fig. 5(b) is computed as

$$\theta = \tan^{-1}\left(-\frac{P_1^y - P_2^y}{P_1^x - P_2^x}\right) \tag{9}$$

where P_i^x and P_i^y are the x (J_1) and y (J_2) positions of the end points, $\mathbf{P}_1$ and $\mathbf{P_2}$, respectively.

Then, δ_1 and δ_2 are determined with δ_J and θ as follows,

$$\delta_1 = \delta_J \cos\theta \;\text{ and }\; \delta_2 = \delta_J \sin\theta \tag{10}$$

Step 7: Impose additional inequality constraints and conduct sub-optimization with the weighted-sum method in each of the feasible regions. As shown in Fig. 5(b), the feasible region is offset from $\mathbf{P}_1$ and $\boldsymbol{P_2}$ by the distance of δ_1 and δ_2 in the direction of J_1 and J_2. Performing sub-optimization in this region, the problem is stated as

$$\min \quad \lambda\frac{J_1(\mathbf{x})}{sf_{1,0}(\mathbf{x})} + (1-\lambda)\frac{J_2(\mathbf{x})}{sf_{2,0}(\mathbf{x})}$$

s.t.

$$J_1(\mathbf{x}) \le P_1^x - \delta_1$$

$$J_2(\mathbf{x}) \le P_2^y - \delta_2$$

$$h(\mathbf{x}) = 0\,, \quad g(\mathbf{x}) \le 0\,, \quad \lambda \in [0,1] \tag{11}$$

where δ_1 and δ_2 are the offset distances obtained in Step 6, P_i^x and P_i^y are the x and y position of the end points, and $sf_{1,0}$ and $sf_{2,0}$ are scaling factors. The uniform step size of the weighting factor λ_i for each feasible region is determined by the number of refinements, n_i, obtained in Step 4:

$$\Delta\lambda_i = \frac{1}{n_i} \tag{12}$$

The segments in which no converged optimum solutions are obtained are removed from the segment set for further refinement, because in this case these regions are non-convex and do not contain Pareto optimal solutions.

Step 8: Compute the length of the segments between all the neighboring solutions. Delete nearly overlapping solutions. If all the segment lengths are less than a prescribed maximum length, δ_J, terminate the optimization procedure. If there are segments whose lengths are greater than the maximum length, go to Step 4 and iterate.

4 Numerical examples

Three numerical examples are presented in this section to demonstrate the performance of the adaptive weighted sum method. All optimizations were performed with the the Sequential Quadratic Programming (SQP) method in MATLAB.

4.1 Example 1: convex Pareto front

The first example is a multiobjective optimization problem that was investigated in the context of the NBI method development by Das and Dennis (1998). The problem statement is

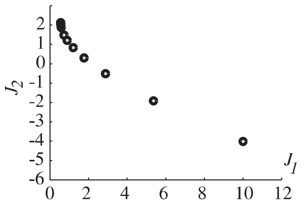
$$\text{minimize} \quad \begin{bmatrix} J_1 = x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 \\ J_2 = 3x_1 + 2x_2 - \frac{x_3}{3} + 0.01(x_4 - x_5)^3 \end{bmatrix}$$

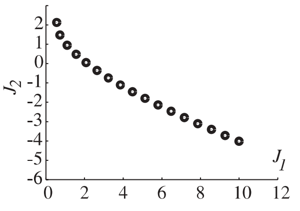
$$\text{subject to } \; x_1 + 2x_2 - x_3 - 0.5x_4 + x_5 = 2\,,$$

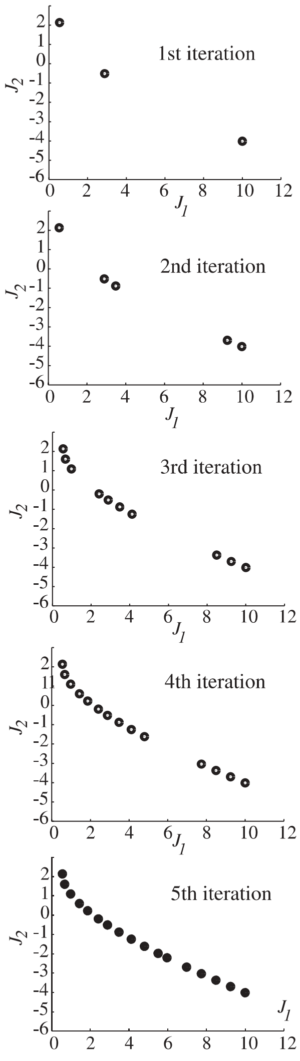
$$4x_1 - 2x_2 + 0.8x_3 + 0.6x_4 + 0.5x_5^2 = 0\,,$$

$$x_1^2 + x_2^2 + x_3^2 + x_4^2 + x_5^2 \le 10 \tag{13}$$

The Pareto front of this problem is convex, but the curvature is not uniform. Figure 6(a) shows the optimal solution obtained by the usual weighted-sum method. The number of solutions on the Pareto front is 17, but most of the solutions are concentrated in the left upper region. The NBI method gives a very good approximation


(a) WS (Weighted Sum Method)


(b) NBI (Normal Boundary Intersection) method



(c) AWS (Adaptive Weighted Sum Method) converges in 5 iterations

Fig. 6 Results for multiobjective optimization with a convex Pareto front (Example 1)

of the Pareto front by obtaining evenly distributed solutions, as shown in Fig. 6(b). The adaptive weighted-sum method converges in five iterations, obtaining fairly well-distributed solutions (Fig. 6(c)).

The offset distance selected on the Pareto front, δ_J, is 0.1; and the offset distances, δ_1 and δ_2, are calculated by (10). Table 1 provides a quantitative comparison of solutions in terms of computational cost (CPU time) and variance of segment lengths. The weighted-sum method, the NBI method, and the adaptive-weighed sum (AWS) method are compared for the case of 17 solutions on the Pareto front.

Although the weighted-sum method is fast, its variance is very large. The NBI method has better performance both in terms of CPU time and secant length variance compared to the adaptive weighted-sum method in this example. At this point it is not obvious why one might further pursue the adaptive weighted-sum method. It has been observed that the NBI method usually performs better in the cases of well-conditioned multiobjective optimization problems with a convex Pareto front. However, the uniformity of the solutions obtained by the adaptive weighted-sum method is satisfactory according to the maximum length criterion, and the adaptive weighted-sum method shows better performance in more complex problems, as demonstrated in the following example. The relatively heavy computational cost of the adaptive weighted-sum approach is due to additional calculations, such as obtaining the distances between adjacent solutions and selecting segments for further refinement. This overhead will be less significant for large problems, where the cost of objective function evaluations typically dominates.

Table 1 Comparison of the results for Example 1

	WS	NBI	AWS
No. of solutions	17	17	17
CPU time (sec)	1.71	2.43	3.83
Length variance ($\times 10^{-4}$)	266	0.23	2.3

4.2 Example 2: non-convex Pareto front

In the previous example, the Pareto front was convex, and the problem associated with the usual weighted-sum approach was only that the solution distribution was not uniform. However, if the Pareto front is not convex, the weighted-sum approach does not find concave parts, regardless of step size. In this example, a multiobjective optimization problem that has a partially non-convex Pareto front and that is not well conditioned is considered. The problem statement is:

$$\max \quad [J_1 \quad J_2]^T$$

$$J_1 = 3\,(1-x_1)^2\, e^{-x_1^2-(x_2+1)^2} - 10\left(\frac{x_1}{5} - x_1^3 - x_2^5\right) \times$$
$$e^{-x_1^2-x_2^2} - 3e^{-(x_1+2)^2-x_2^2} + 0.5\,(2x_1 + x_2)$$

$$J_2 = 3\,(1+x_2)^2\, e^{-x_2^2-(1-x_1)^2} - 10\left(-\frac{x_2}{5} + x_2^3 + x_1^5\right) \times$$
$$e^{-x_2^2-x_1^2} - 3e^{-(2-x_2)^2-x_1^2}$$

$$\text{subject to } -3 \le x_i \le 3\,, \quad i = 1, 2 \tag{14}$$

The solutions obtained by the usual weighted-sum method are shown in Fig. 7. This figure shows the efficient designs in the design space on the left and the Pareto optimal solutions in the objective space on the right. The entire range in the objective space is obtained by a full combinatorial analysis. The difficulty in performing optimization for this non-linear problem is that the convergence to an optimal solution is highly dependent on an initial starting point and determining the starting point

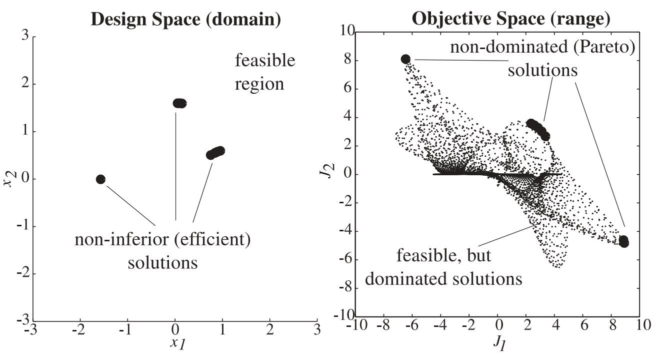


Fig. 7 Usual weighted-sum method for multiobjective optimization with a non-convex Pareto front: Example 2

is not straightforward. The solution dependence on the initial starting point is even more severe in the case of the NBI method and the adaptive weighted-sum method than the usual weighted-sum method. This is because the two methods use additional constraints and so it is difficult to find feasible regions that satisfy all the constraints. In the usual weighted-sum method, three points ([1.5 0], [1 1] and [0 2]) are used as a starting point, and the best among the solutions is selected. As shown in Fig. 7, trying these three initial starting points always yields the optimum solutions for the usual weighted-sum method. However, the solutions cluster around three small regions. The vast area of the two concave regions is not revealed by the traditional weighted-sum method, which confirms the second drawback of the method mentioned in Sect. 1.

The NBI method and adaptive weighted-sum method successfully find solutions in the non-convex regions. However, the solution dependence on the initial starting point is a serious concern for these methods. So, full combinatorial trials of initial starting points were conducted to better understand this issue. The domain is discretized into grids of size, $\varDelta x_1$ and $\varDelta x_2$, and the optimization is started from the grid points. The best solution is then selected from among all the solutions obtained. Four different cases of starting grid resolution were tested for the NBI method and the AWS method:

- Case 1: $\varDelta x_1 = \varDelta x_2 = 2.0$
- Case 2: $\varDelta x_1 = \varDelta x_2 = 1.5$
- Case 3: $\varDelta x_1 = \varDelta x_2 = 1.0$
- Case 4: $\varDelta x_1 = \varDelta x_2 = 0.5$

The solutions obtained from the NBI method for each of all four cases are shown in Fig. 8. In all four cases, one non-Pareto solution is obtained, which is dominated by its two neighboring solutions. Because of this problem, a Pareto filter needs to be applied a posteriori for all results obtained with the NBI method. In addition, some sub-optimal solutions are obtained: three suboptimal solutions for Case 1 and one suboptimal solution for Case 3. These solutions are dominated and have apparently converged to local maxima, despite the abundance of starting points across the domain.

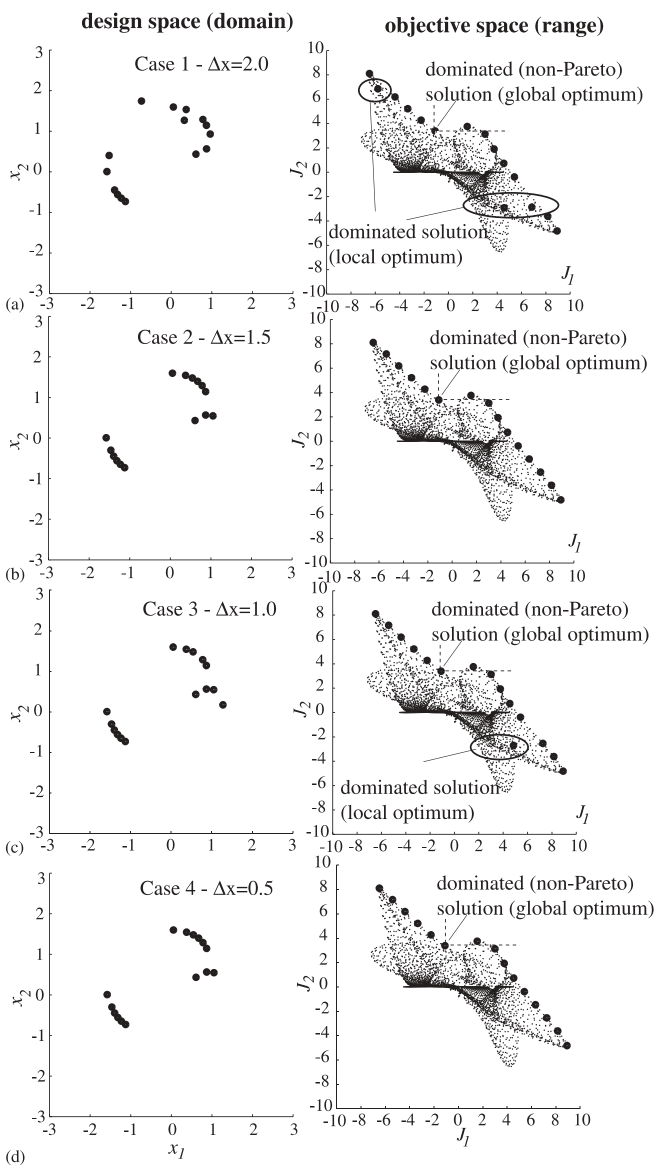


Fig. 8 Results from the NBI method for multiobjective optimization with a non-convex Pareto front: Example 2

When the adaptive weighted-sum method is used, on the other hand, all the solutions obtained are truly Pareto optimal, as shown in Fig. 9. Only one case is represented in the figure because the solutions are identical for all four cases. The offset distance on the Pareto front, δ_J, is 0.1. Note that non-Pareto optimum or suboptimal solutions are not obtained with the adaptive weighted-sum method, as it should be. The reason for the method's robustness in finding Pareto optimal solutions is that it uses inequality constraints rather than equality constraints, which makes it easier to find feasible solutions during optimization.

This example demonstrates the advantages of the adaptive weighted-sum method: (i) it finds solutions of even distribution; (ii) it can find solutions on non-convex regions; (iii) non-Pareto solutions in non-convex regions are not considered as optimal, because they are not in the feasible region bounded by the additional constraints.

Table 2 Comparison of the results for Example 2

	WS	NBI				AWS			
Initial starting point case		Case 1	Case 2	Case 3	Case 4	Case 1	Case 2	Case 3	Case 4
No. of solutions	15	15	15	15	15	15	15	15	15
CPU time (sec)	0.4	17.8	24.5	52.9	165.6	28.1	44.0	87.6	289.2
Length variance ($\times 10^{-4}$)	632	11	3.6	8.8	3.6	4.3	4.3	4.3	4.3
No. of suboptimal solutions	0	3	0	1	0	0	0	0	0
No. of non-Pareto solutions	0	1	1	1	1	0	0	0	0

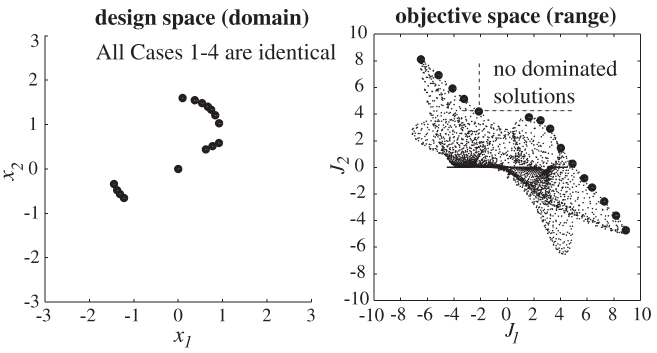


Fig. 9 Results obtained with AWS for multiobjective optimization with a non-convex Pareto front: Example 2. Case 1, Case 2, Case 3 and Case 4 give the same results

AWS is potentially more robust in finding optimum solutions than other methods that use equality constraints. The solution comparison for each method for this example is provided in Table 2.

4.3 Example 3: three-bar truss problem

Finally, the adaptive weighted-sum method is applied to the three-bar truss problem first presented by Koski (1985). Figure 10 illustrates the problem and shows the

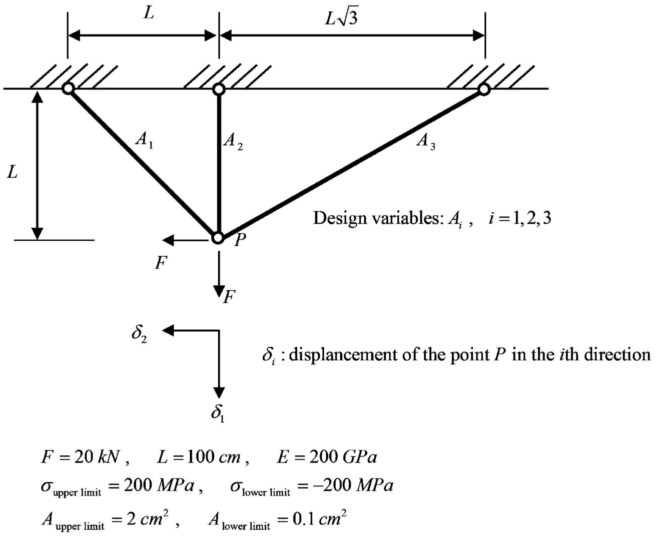

L
$L\sqrt{3}$
A_1
A_2
A_3
P
F
Design variables: A_i , $i = 1,2,3$
δ_1
δ_2
δ_i : displancement of the point P in the ith direction
$F = 20\,kN$, $L = 100\,cm$, $E = 200\,GPa$
$\sigma_{\text{upper limit}} = 200\,MPa$, $\sigma_{\text{lower limit}} = -200\,MPa$
$A_{\text{upper limit}} = 2\,cm^2$, $A_{\text{lower limit}} = 0.1\,cm^2$

Fig. 10 The three-bar truss problem

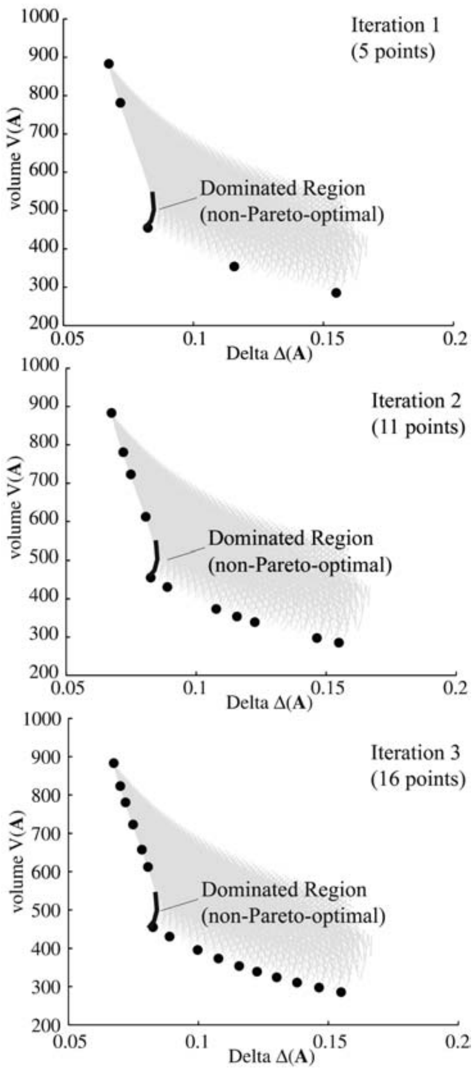


Fig. 11 Optimization history of the adaptive weighted-sum method for Example 3 with $\delta_J = 0.1$. It converges in three phases

values of the parameters used. A horizontal load and a vertical load are applied at point P, and the objective functions are the total volume of the truss and the displacement of point P.

The mathematical problem statement is:

$$\text{minimize} \quad \begin{bmatrix} \text{volume}\ (\mathbf{A}) \\ \Delta(\mathbf{A}) \end{bmatrix}$$

$$\begin{aligned}&\text{subject to}\quad \sigma_{\text{lower limit}} \le \sigma_i \le \sigma_{\text{upper limit}} \quad i=1,2,3\\ &A_{\text{lower limit}} \le A_i \le A_{\text{upper limit}} \quad i=1,2,3\\ &\text{where}\ \Delta = 0.25\delta_1 + 0.75\delta_2\\ &\text{and}\ \mathbf{A} = [A_1 \quad A_2 \quad A_3]\end{aligned} \tag{15}$$

The Pareto front for this example is non-convex, and the Pareto line is separated into two regions by a segment of dominated solutions, as shown in Fig. 11. The adaptive weighted-sum method with an offset of $\delta_J = 0.1$ is used. The optimization history is shown in the figure. The adaptive weighted-sum method converged in three phases, and the solutions are quite evenly distributed.

Note that no solution is obtained in the non-Pareto region, without using a Pareto filter. If one changes the value of the offset distance, δ_J, the density of final solutions changes. Figure 12 shows the two results when 0.2 and 0.05 are used as the offset distance, δ_J. The adaptive weighted-sum method gives 8 and 32 evenly distributed Pareto solutions for each case. Again in this example, the distribution is nearly uniform; the Pareto optimal solutions on the non-convex region are identified; and the non-Pareto optimal solutions are ignored. The parameter δ_J is used to tune the desired density of Pareto points generated by the algorithm.

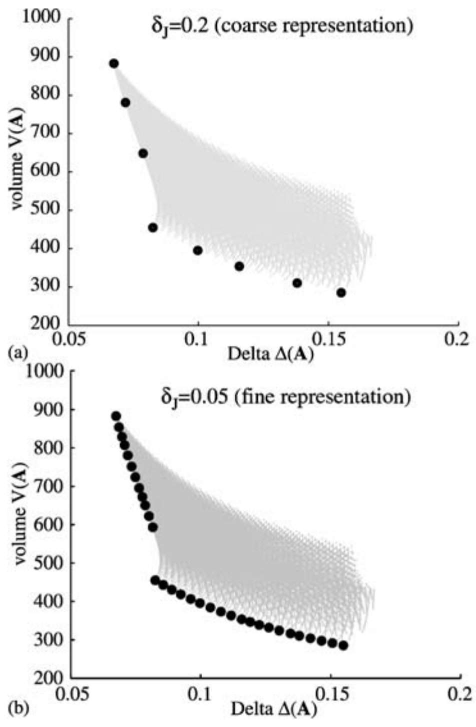


Fig. 12 Solutions for different offset distances in Example 3

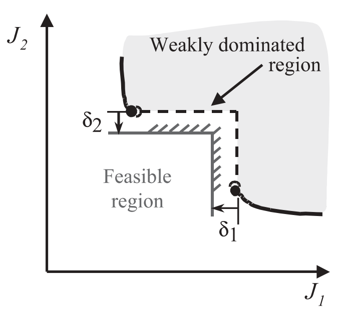


Fig. 13 Degenerate case of a Pareto front with a weakly dominated region

5 Discussion

The adaptive weighted-sum (AWS) method effectively approximates a Pareto front by gradually increasing the number of solutions on the front. In that sense it gradually "learns" the shape of the Pareto front and concentrates computational effort where new information can be gained most effectively. This is in contrast to other Pareto generation methods such as traditional weighted-sum or NBI, which generally explore the Pareto front in a predetermined fashion. Because it adaptively determines where to refine further, the adaptive weighted-sum method produces well-distributed solutions. In addition, performing optimization only in feasible regions by imposing additional inequality constraints enables the method to find Pareto solutions in non-convex regions. Because the feasible region includes only the regions of non-dominated solutions, it automatically neglects non-Pareto optimal solutions. It is potentially more robust in finding optimal solutions than other methods where equality constraints are applied.

There are four important parameters that the user must set: the offset distance (δ_J); the Euclidean distance for determination of overlapping solutions (ε) used in Step 3 and Step 8; the constant for further refinement (C) used in Step 4; and the number of the Pareto front segments in the initial iteration (n_{initial}).

The offset distance, δ_J, determines the final solution distribution density and can be chosen independently of other parameters. Values between 0.05 and 0.2 in the normalized objective space are recommended. The smaller δ_J is, the denser the final solution distribution becomes. The overlapping solution distance ε must be smaller than δ_J. In this paper, ε is 50% of the magnitude of δ_J, and well-distributed solutions are obtained. The multiplier C must be chosen carefully. If it is too small, no further refinement will be conducted in subsequent iterations, and the optimization will terminate prematurely. If it is excessively large, many overlapping solutions will be generated, and the computational cost will increase. It is our experience that the optimization progresses well with reasonable computing time when C is between 1 and 2. The initial number of Pareto front divisions, n_{initial}, must be selected in the same way. A small n_{initial} will not lead the optimization to subsequent iterations of further refinement, but on the other hand, the computational cost will become too expensive with a large value of n_{initial}. A proper range, found in several examples here, is between three and ten. It is noted that the optimization behavior depends on the parameter selection to some extent. Currently the parameters can be chosen only heuristically, and more study is needed to investigate this issue. In particular, n_{initial} and C should be selected in consideration of each other. For example, when a small n_{initial} is used, a large C would help prevent premature convergence.

It is found that the adaptive weighted-sum (AWS) method cannot handle a degenerate problem of a Pareto front that has a weakly dominated region, as shown in Fig. 13. In this special case, the horizontal and vertical lines are weakly dominated, and they are not included in the feasible region for further optimizations regardless of the size of δ_1 or δ_2. Such an extreme case, however, is not likely to be experienced in practice.

This article does not claim superiority of the adaptive weighted-sum method over other methods such as NBI in all cases. Rather the method presents itself as a potential addition to the growing suite of Pareto generators, with potential advantages for ill-conditioned problems. Further work is needed to understand the nature of this advantage in terms of starting points, imposition of inequality constraints versus equality constraints and computational cost. It must also be said that while the traditional weighted-sum method has known limitations, it remains the method offering greatest transparency to non-expert users. The adaptive weighted-sum approach is an effective extension of traditional weighted-sum optimization, but some of the transparency is invariably hidden from the user due to the adaptive scheme. In addition, the adaptive weighted-sum method needs to be applied to multidimensional multiobjective optimization problems where there are more than two objective functions. Some multiobjective optimization algorithms perform well for bivariate problems, but scale poorly to multiple objectives. It remains to be seen how well adaptive weighted-sum (AWS) optimization can be scaled to problems of higher dimensionality. Practical applications of increased complexity will also be solved by the adaptive weighted-sum method.

Acknowledgements This work was supported by the Postdoctoral Fellowship Program of the Korean Science & Engineering Foundation (KOSEF) and by a grant from the MIT Karl Chang Innovation fund.

References

Das, I.; Dennis, J.E. 1997: A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Struct Optim* **14**, 63–69

Das, I.; Dennis, J.E. 1998: Normal-Boundary Intersection: A New Method for Generating Pareto Optimal Points in Multicriteria Optimization Problems. *SIAM J Optim* **8**, 631–657

Fonseca, C.; Fleming, P. 1995: An overview of evolutionary algorithms in multiobjective optimization. *Evol Comput* **3**, 1–18

Goldberg, D.E. 1989: *Genetic Algorithms in Search, Optimization and Machine Learning*. Boston, MA: Addison Wesley

Koski, J. 1985: Defectiveness of weighting method in multicriterion optimization of structures. *Commun Appl Numer Methods* **1**, 333–337

Koski, J. 1988: Multicriteria truss optimization. In: Stadler, W. (ed.) *Multicriteria Optimization in Engineering and in the Sciences*, New York: Plenum

Lin, J. 1976: Multiple objective problems: Pareto-optimal solutions by method of proper equality constraints. *IEEE Trans Autom Control* **21**, 641–650

Marglin, S. 1967: *Public Investment Criteria*. Cambridge, MA: MIT Press

Messac, A.; Mattson, C.A. 2002: Generating Well-Distributed Sets of Pareto Points for Engineering Design using Physical Programming. *Optim Eng* **3**, 431–450

Messac, A.; Ismail-Yahaya, A.; Mattson, C.A. 2003: The Normalized Normal Constraint Method for Generating the Pareto Frontier. *Struct Multidisc Optim* **25**, 86–98

Mattson, C.A.; Messac, A. 2003: Concept Selection Using s-Pareto Frontiers. *AIAA J* **41**, 1190–1204

Messac, A.; Mattson, C.A. 2004: Normal Constraint Method with Guarantee of Even Representation of Complete Pareto Frontier. *AIAA J* **42**

Pareto, V. 1906: *Manuale di Economia Politica*, Societa Editrice Libraria, Milano, Italy. Translated into English by Schwier, A.S. 1971: *Manual of Political Economy*, New York: Macmillan

Stadler, W. 1979: A Survey of Multicriteria Optimization, or the Vector Maximum Problem. *JOTA* **29**, 1–52

Stadler, W. 1984: Applications of Multicriteria Optimization in Engineering and the Sciences (A Survey). In: Zeleny, M. (ed.) *Multiple Criteria Decision Making – Past Decade and Future Trends*. Greenwich, CT: JAI

Steuer, R.E. 1986: *Multiple Criteria Optimization: Theory, Computation and Application*. New York: Wiley

Suppapitnarm, A.; Seffen, K.A.; Parks, G.T.; Clarkson, P.J. 2000: A simulated annealing algorithm for multiobjective optimization. *Eng Optim* **33**(1), 59–85

Schy, A.A.; Giesy, D.P. 1988: Multicriteria Optimization for Design of Aircraft Control Systems. In: Stadler, W. (ed.) *Multicriteria Optimization in Engineering and in the Sciences*, New York: Plenum, p.225–262

Tamaki, H.; Kita, H.; Kobayashi, S. 1996: Multiobjective optimization by genetic algorithms: a review. *1996 IEEE International Conference on Evolutionary Computation*, ICEC 1996, Nagoya, Japan

Zadeh, L. 1963: Optimality and Non-Scalar-Valued Performance Criteria. *IEEE Trans Autom Control* **8**, 59–60