H. Nakayama
Y. Yun
M. Yoon

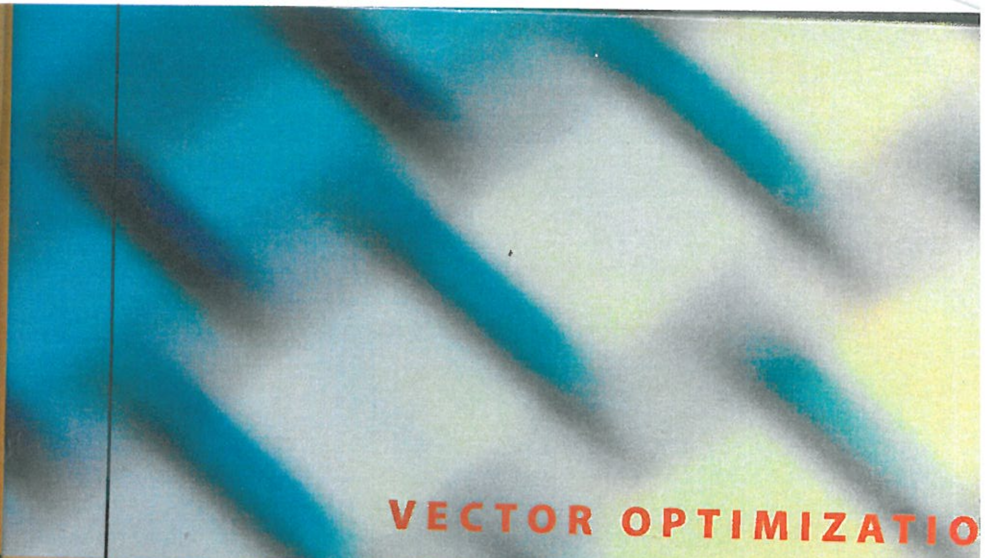
Sequential Approximate Multiobjective Optimization Using Computational Intelligence

This book highlights a new direction of multiobjective optimization, which has never been treated in previous publications. When the function form of objective functions is not known explicitly as encountered in many practical problems, sequential approximate optimization based on meta-models is an effective tool from a practical viewpoint. Several sophisticated methods for sequential approximate multiobjective optimization using computational intelligence are introduced along with real applications, mainly engineering problems, in this book.

ISBN 978-3-540-88909-0

Nakayama · Yun · Yoon

Sequential Approximate Multiobjective Optimiza
Using Computational Intelligence



VECTOR OPTIMIZATIO

Hirotaka Nakayam
Yeboon Yu
Min Yoo

Sequential Approximate Multiobjective Optimization Using Computational Intelligence

Remark 2.2. If less level of f is more preferable, and if we aim to find a solution which yields as less level of f as possible even though it clears the goal level $\overline{f}$, then we should formulate as follows:

$$\begin{array}{lll} \underset{x,\,z}{\text{minimize}} & \displaystyle\sum_{i=1}^{r} w_i z_i & \qquad\qquad (\text{GP}'') \\ \text{subject to} & f_i(x) - \overline{f}_i \leqq z_i, \quad i = 1, \ldots, r, & \\ & x \in X \subset \mathbb{R}^n . & \end{array}$$

Note that the new variable z is not necessarily nonnegative. The stated problem is equivalent to multiobjective programming problem:

$$\begin{array}{ll} \underset{x}{\text{minimize}} & \displaystyle\sum_{i=1}^{r} w_i f_i(x) \\ \text{subject to} & x \in X \subset \mathbb{R}^n . \end{array}$$

However, by (GP″), we can treat f_i not only as an objective function but also as a constraint by setting $z_i = 0$. This technique will be used to interchange the role of objective function and constraint in subsequent sections.

From a viewpoint of decision making, it is important to obtain a solution which the decision maker accepts easily. This means it is important whether the obtained solution is suited to the value judgment of the decision maker. In goal programming, the adjustment of weights w_i is required to obtain a solution suited to the value judgment of the decision maker. However, this task is very difficult in many cases as will be seen in the following.

2.2 Why is the Weighting Method Ineffective?

In multiobjective programming problems, the final decision is made on the basis of the value judgment of DM. Hence it is important how we elicit the value judgment of DM. In many practical cases, the vector objective function is scalarized in such a manner that the value judgment of DM can be incorporated.

The most well-known scalarization technique is the linearly weighted sum

$$\sum_{i=1}^{r} w_i f_i(x).$$

The value judgment of DM is reflected by the weight. Although this type of scalarization is widely used in many practical problems, there is a serious drawback in it. Namely, it cannot provide a solution among sunken parts of Pareto surface (frontier) due to *duality gap* for nonconvex cases. Even for

convex cases, for example, in linear cases, even if we want to get a point in the middle of line segment between two vertices, we merely get a vertex of Pareto surface, as long as the well-known simplex method is used. This implies that depending on the structure of problem, the linearly weighted sum cannot necessarily provide a solution as DM desires.

In an extended form of goal programming, e.g., *compromise programming* [165], some kind of metric function from the goal $\overline{f}$ is used as the one representing the preference of DM. For example, the following is well known:

$$\left(\sum_{i=1}^{r} w_i |f_i(x) - \overline{f}_i|^p \right)^{1/p} . \tag{2.1}$$

The preference of DM is reflected by the weight w_i, the value of p, and the value of the goal $\overline{f}_i$. If the value of p is chosen appropriately, a Pareto solution among a sunken part of Pareto surface can be obtained by minimizing (2.1). However, it is usually difficult to predetermine appropriate values of them. Moreover, the solution minimizing (2.1) cannot be better than the goal $\overline{f}$, even though the goal is pessimistically underestimated.

In addition, one of the most serious drawbacks in the weighted sum scalarization is that people tend to misunderstand that a desirable solution can be obtained by adjusting the weight. It should be noted that there is no positive correlation between the weight w_i and the value $f(\hat{x})$ corresponding to the resulting solution $\hat{x}$ as will be seen in the following example.

Example 2.1. Let $f_1 := y_1$, $f_2 := y_2$ and $f_3 := y_3$, and let the feasible region in the objective space be given by

$$\{ (y_1, y_2, y_3) \mid (y_1 - 1)^2 + (y_2 - 1)^2 + (y_3 - 1)^2 \leqq 1 \} .$$

Suppose that the goal is $(\overline{y}_1, \overline{y}_2, \overline{y}_3) = (0, 0, 0)$. The solution minimizing the metric function (2.1) with $p = 1$ and $w_1 = w_2 = w_3 = 1$ is $(y_1, y_2, y_3) = (0.42265, 0.42265, 0.42265)$. Now suppose that DM wants to decrease the value of f_1 a lot more and that of f_2 a little more, and hence modify the weight into $w_1' = 10$, $w_2' = 2$, $w_3' = 1$. The solution associated with the new weight is $(0.02410, 0.80482, 0.90241)$. Note that the value of f_2 is worse than before despite that DM wants to improve it by increasing the weight of f_2 up to twice. Someone might think that this is due to the lack of normalization of weight. Therefore, we normalize the weight by $w_1 + w_2 + w_3 = 1$. The original weight normalized in this way is $w_1 = w_2 = w_3 = 1/3$ and the renewed weight by the same normalization is $w_1' = 10/13$, $w_2' = 2/13$, $w_3' = 1/13$. We can observe that w_2' is less than w_2. Now increase the normalized weight w_2 to be greater than $1/3$. To this end, set the unnormalized weight $w_1 = 10$, $w_2 = 7$ and $w_3 = 1$. With this new weight, we have a solution $(0.18350, 0.42845, 0.91835)$. Despite that the normalized weight $w_2'' = 7/18$ is greater than the original one $(=1/3)$, the obtained solution is still worse than the previous one.

As is readily seen in the above example, it is usually very difficult to adjust the weight in order to obtain a solution as DM wants. This difficulty seems to be caused due to the fact that there is no positive correlation between the weight and the resulting solution in cases with more than two objective functions. Therefore, it seems much better to take another probe for getting a solution which DM desires. The aspiration level of DM is promising as the probe. Interactive multiobjective programming techniques based on aspiration levels have been developed so that the drawbacks of the traditional weighting method may be overcome [13,157]. In Sect. 2.3, we shall discuss the *satisficing trade-off method* developed by one of the authors (Nakayama [100]) as an example.

2.3 Satisficing Trade-off Method

In the *aspiration level approach*, the aspiration level at the kth iteration $\overline{\boldsymbol{f}}^k$ is modified as follows:

$$\overline{\boldsymbol{f}}^{k+1} = T \circ P(\overline{\boldsymbol{f}}^k)$$

Here, the operator P selects the Pareto solution nearest in some sense to the given aspiration level $\overline{\boldsymbol{f}}^k$. The operator T is the trade-off operator which changes the kth aspiration level $\overline{\boldsymbol{f}}^k$ if DM does not compromise with the shown solution $P(\overline{\boldsymbol{f}}^k)$. Of course, since $P(\overline{\boldsymbol{f}}^k)$ is a Pareto solution, there exists no feasible solution which makes all criteria better than $P(\overline{\boldsymbol{f}}^k)$, and thus DM has to trade-off among criteria if he wants to improve some of criteria. Based on this trade-off, a new aspiration level is decided as $T \circ P(\overline{\boldsymbol{f}}^k)$. Similar process is continued until DM obtains an agreeable solution. This idea is implemented in DIDASS [51] and the satisficing trade-off method [100]. While DIDASS leaves the trade-off to the heuristics of DM, the satisficing trade-off method provides a device based on the sensitivity analysis which will be stated later.

2.3.1 On the Operation P

The operation which gives a Pareto solution $P(\overline{\boldsymbol{f}}^k)$ nearest to $\overline{\boldsymbol{f}}^k$ is performed by some *auxiliary scalar optimization*. It has been shown in [130] that the only one scalarization technique, which provides any Pareto solution regardless of the structure of problem, is of the Tchebyshev type. As was stated before, however, the Tchebyshev type scalarization function yields not only a Pareto solution but also a weak Pareto solution. Since weak Pareto solutions have a possibility that there may be another solution which improves a criteria while others being fixed, they are not necessarily "efficient" as a solution in

decision making. In order to exclude weak Pareto solutions, we apply the *augmented Tchebyshev scalarization function*:

$$\max_{1 \leq i \leq r} w_i(f_i(\boldsymbol{x}) - \overline{f}_i) + \alpha \sum_{i=1}^{r} w_i f_i(\boldsymbol{x}), \tag{2.2}$$

where α is usually set a sufficiently small positive number, say 10^{-6}. The weight w_i is usually given as follows: Let f_i^* be an ideal value which is usually given in such a way that $f_i^* < \min\{f_i(\boldsymbol{x}) \mid \boldsymbol{x} \in X\}$, and let f_{*i} be a nadir value which is usually given by

$$f_{*i} = \max_{1 \leq j \leq r} f_i(\boldsymbol{x}_j^*),$$

where

$$\boldsymbol{x}_j^* = \arg \min_{\boldsymbol{x} \in X} f_j(\boldsymbol{x}).$$

For this circumstance, we set

$$w_i^k = \frac{1}{\overline{f}_i^k - f_i^*} \tag{2.3}$$

or

$$w_i^{k'} = \frac{1}{f_{*i} - f_i^*}. \tag{2.4}$$

The minimization of (2.2) with (2.3) or (2.4) is usually performed by solving the following equivalent optimization problem, because the original one is not smooth:

$$\begin{aligned} &\underset{\boldsymbol{x},\, z}{\text{minimize}} && z + \alpha \sum_{i=1}^{r} w_i f_i(\boldsymbol{x}) && \text{(AP)} \\ &\text{subject to} && w_i^k \left(f_i(\boldsymbol{x}) - \overline{f}_i^k \right) \leqq z, \quad i = 1, \ldots, r && (2.5) \\ & && \boldsymbol{x} \in X. \end{aligned}$$

Remark 2.3. Note the weight (2.3) depends on the kth aspiration level, while the one by (2.4) is independent of aspiration levels. The difference between solutions to (AP) for these two kinds of weight can be illustrated in Fig. 2.1. In the auxiliary min–max problem (AP) with the weight by (2.3), $\overline{f}_i^k$ in the constraint (2.5) may be replaced with f_i^* without any change in the solution. For we have

$$\frac{f_i(\boldsymbol{x}) - f_i^*}{\overline{f}_i^k - f_i^*} = \frac{f_i(\boldsymbol{x}) - \overline{f}_i^k}{\overline{f}_i^k - f_i^*} + 1.$$