SIAM J. OPTIM.
Vol. 27, No. 2, pp. 1010–1034
© 2017 Society for Industrial and Applied Mathematics

A NEW SCALARIZATION TECHNIQUE AND NEW ALGORITHMS TO GENERATE PARETO FRONTS*

R. S. BURACHIK†, C. Y. KAYA‡, AND M. M. RIZVI§

Abstract. We propose a new scalarization technique for nonconvex multiobjective optimization problems and establish its theoretical properties. By combining our new scalarization approach with existing grid generation techniques, we design new algorithms for constructing reliable approximations of the Pareto fronts of three- and four-objective optimization problems. Our algorithms can be extended for problems with more objective functions to minimize. Four three-objective algorithms are formed by pairing up the new scalarization technique and three existing scalarization techniques with a grid of weights generated over the convex hull of individual minima (the CHIM grid) due to Das and Dennis. Four more algorithms are obtained by using, instead of the CHIM grid, the grid in the successive boundary generation algorithm (the SBG grid) by Mueller-Gritschneder, Graeb, and Schlichtmann. The new algorithms seem to perform better than existing algorithms, in terms of computational time and accuracy in constructing the boundary and the interior of the Pareto front. The advantages are tested, in particular, through a problem whose Pareto front has a hole in it. A rocket injector design problem with four objective functions illustrates the effectiveness of our new approach further.

Key words. multiobjective optimization, scalarization techniques, Pareto front, grid generation techniques, rocket injector design

AMS subject classifications. 90C29, 90C30

DOI. 10.1137/16M1083967

1. Introduction.

The problem of optimizing several and conflicting objective functions simultaneously over a feasible region is referred to as a multiobjective optimization problem (MOP). The conflicting nature of the objectives makes it impossible to find a solution that individually optimizes all of the objective functions at the same time. Therefore, the intent of a MOP is to figure out the best possible trade-off among these criteria. In many real-world problems, decisions indeed have to be made based on noncomparable, or conflicting, multiple objective functions (see [2, 13, 27]). Points satisfying a trade-off among conflicting objective functions are referred to as *Pareto points*, and the set consisting of the images of these points in the objective or value space is called a *Pareto front*. In the literature, "Pareto" is sometimes replaced by "efficient" or "nondominating," and "front" is replaced by "set."

1.1. Scalarization techniques.

A common approach to solving a MOP is, first, to convert the MOP into a single-objective optimization problem by combining the multiple objectives through parameters, which is referred to as *parametric*, or *parameter-based*, *scalarization*. Then the resulting single-objective, or scalarized, problem is solved by standard optimization methods and software. There exist many scalarization techniques which ensure that, under certain conditions, the

*Received by the editors July 11, 2016; accepted for publication (in revised form) December 27, 2016; published electronically June 6, 2017.
http://www.siam.org/journals/siopt/27-2/M108396.html

†School of Information Technology and Mathematical Sciences, University of South Australia, Mawson Lakes, SA 5095, Australia (regina.burachik@unisa.edu.au).

‡Corresponding author. School of Information Technology and Mathematical Sciences, University of South Australia, Mawson Lakes, SA 5095, Australia (yalcin.kaya@unisa.edu.au).

§School of Information Technology and Mathematical Sciences, University of South Australia, Mawson Lakes, SA 5095, Australia, and Department of Mathematics, University of Chittagong, Chittagong, Bangladesh (rizmm001@mymail.unisa.edu.au).

solution obtained in this way is a Pareto point. Among the most common techniques are the well-known *weighted-sum* [17, 32], *epsilon-constraint* [5], *Pascoletti–Serafini* [26], and its generalization, *modified Pascoletti–Serafini* [8, 12, 14] methods. We have recently proposed the *weighted-constraint method* [4, 28], which is another parameter-based scalarization technique, as an alternative to the above-listed techniques, in particular for problems with disconnected feasible sets, i.e., disconnected domains. Comprehensive discussions of other scalarization techniques can be found in [6, 7, 9, 10, 11, 15, 19, 22, 23, 24].

With the exception of the weighted sum method, the above-listed scalarization methods yield a Pareto minimizer of a nonconvex MOP, as long as the Pareto front is simply connected (e.g., if the front is a continuous curve for two objectives or if the front is a surface containing no holes for three objectives). However, if the front is not simply connected, then few of these scalarization techniques are likely to be successful in giving a Pareto minimizer and to be effective in constructing an approximation of the Pareto front. Furthermore, if the domain, or the feasible set, of the problem is not simply connected, then all of these methods, except the *weighted-constraint method* [4, 28], can be expected to encounter severe difficulties.

The set of all Pareto points of a MOP is in general an infinite set. Hence, one can only hope to obtain a discrete approximation of this set, so that a practitioner could then make a decision as to which Pareto point should be chosen, based on some additional criterion. A discrete approximation of the Pareto front is typically obtained over a discrete set, or *grid*, of values of the weights.

1.2. Normal boundary intersection and sequential boundary generation methods. We hereby give brief descriptions of the grid generation techniques, because they will play an important role in designing the algorithms. The normal boundary intersection (NBI) method, which was proposed by Das and Dennis [8], has probably been the most popular approach in the past decade or so for constructing an approximation of the Pareto front. In the NBI method, first the individual minima, i.e., the minimum values of each objective function individually, are found. Then the convex hull of these individual minima, called CHIM, is generated. Note that the CHIM is a simplex: in the case of two objectives it is a line segment and in the case of three objectives, a triangle.

In the NBI method, the CHIM is discretized uniformly, corresponding to a grid of weights. At each *CHIM grid* point corresponding to a fixed vector of weights the modified Pascoletti–Serafini scalarization of the problem is solved to obtain a Pareto point. Geometrically speaking, one finds a point in the Pareto front intercepted by a "ray" which emanates from the grid point in the objective space and is normal to the CHIM. Graphic illustrations of this procedure are given in [8], versions of which we give later in Figure 1, for two and three objectives, respectively. As acknowledged in [8], to guarantee a ray's intersection with the front, one should have that (i) the front is simply connected (no holes or gaps are present in the front) and (ii) a projection of the Pareto front on the CHIM should contain the CHIM.

For the case of two objective functions, if the condition (i) above is satisfied, so is the condition (ii), automatically. However, for three or more objective functions, the condition (ii) would often not be satisfied and this would result in a rather poor approximation of the *boundary* and the points nearby the boundary of the Pareto front. To address this difficulty, or disadvantage, of the NBI method, the so-called successive approach was proposed by Mueller-Gritschneder, Graeb, and Schlichtmann [25].

The successive approach differs from the NBI method in two main aspects:

- The boundary of the Pareto front is approximated truly, by solving subproblems involving the subsets of the set of objective functions in a sequential manner. In each subproblem, a grid of weights is generated by solving a linear programming (LP) problem, again in a sequential manner.
- The Pascoletti–Serafini method, instead of its modified version, is used for scalarization.

As an example, consider the minimization of the vector of three objective functions, $[f_1, f_2, f_3]$. Here we do not show the dependence on the domain variable x for clarity. First the individual minima are found by solving the subproblems of minimizing f_1, f_2, and f_3, separately. Then the two-objective subproblems of minimizing $[f_1, f_2]$, $[f_1, f_3]$, and $[f_2, f_3]$ are solved over linear uniform grids of weights for each subproblem by using the Pascoletti–Serafini scalarization at each grid point. The union of the discretized Pareto fronts of these subproblems yields a discretized boundary of the Pareto front of the three-objective problem of minimizing $[f_1, f_2, f_3]$. Next, the grid of weights in two dimensions (the third weight can be obtained by subtracting the sum of the other two from 1) is generated by solving an LP problem such that the grid points are uniformly distributed not only in the boundary but also in the *interior*,[1] in the space of weights. Finally, over the grid of weights generated in the way described above, the Pascoletti–Serafini method is used for scalarization at each grid point so as to solve the scalarized problems and obtain a discrete approximation of the Pareto front.

In the case of minimizing four or more objective functions, the above procedure is continued for the remaining subsets of the objective functions by also taking the union of the Pareto fronts of the subproblems so as to form the boundary of the upper level problem. Each time the boundary at a given level (by solving subproblems involving the same number of objective functions) is formed, the grid of weights is generated in the same fashion by solving an LP problem.

We will call the approach by Mueller-Gritschneder, Graeb, and Schlichtmann [25], which we have outlined above, the *sequential boundary generation* (SBG) method because of the sequential manner in which the Pareto front is constructed. The SBG method is demonstrated in [25] to be in general more effective in generating the boundary of the Pareto front and as a result the Pareto front itself, compared with the NBI method. We call the grid generated by the SBG method the *SBG grid*. Since generation of the SBG grid requires sequential solutions of LP problems, it requires, in general, more computational time than the CHIM grid.

Many engineering applications, like the analog circuit sizing discussed in [25], require the knowledge of the true boundary of the Pareto front because designs must often meet extreme performances. So, in the case when the major aim is to generate the boundary of the Pareto front rather than the whole front, constructing the SBG grid appears to be more relevant, although the construction is in general more time-consuming. Later on, we provide an example which involves a four-objective rocket injector design problem to exemplify the advantage of using the SBG grid.

The SBG method uses the same assumption as the NBI method in that the Pareto front is simply connected. It further assumes that the set of Pareto-optimal solutions and the set of weakly Pareto-optimal solutions are the same. Simple connectedness of the Pareto front indeed appears to be a crucial requirement for the NBI method, as

[1]Note that the terms "boundary" and "interior" are used here loosely, but their meaning should be intuitively clear from the context, as in similar articles in the literature. They should not be confused with the topological definitions of boundary and interior.

we also assert in the numerical experiments we carry out in this paper. However, the SBG method, conceivably owing to the Pascoletti–Serafini scalarization method that it uses, does not need the "simple connectedness" assumption, as elaborated also in a recent article [21].

1.3. A new scalarization technique and new algorithms. In the present paper, we propose to employ a new scalarization technique with the CHIM and SBG grids, giving rise to new, effective algorithms, in particular for problems which do not necessarily have simply connected Pareto fronts or even simply connected feasible sets. We call our new scalarization technique the *feasible-value-constraint* approach, because, in the description of the scalarized problem, we constrain the objective function values by a value at a fixed feasible point.

We present theoretical properties of the new scalarization technique so as to ensure that a solution of the scalarized problem for a fixed vector of weights is a Pareto minimizer and that there is a weight associated with each Pareto minimizer such that the Pareto minimizer solves the scalarized problem. By these results, we can be sure that there is a surjection from the space of weights to the set of Pareto minimizers so that a suitable grid of weights, when used with the new scalarization, can yield a reliable discrete approximation of the Pareto front.

We devise algorithms for three- and four-objective problems. We pair up the CHIM and SBG grids with the (1) feasible-value-constraint, (2) weighted-constraint, (3) Pascoletti–Serafini, and (4) modified Pascoletti–Serafini scalarization techniques. Each pairing gives a different algorithm: Algorithms 1–4 pair up the CHIM grid with the four scalarization techniques listed above, and Algorithms 5–8 pair up the SBG grid with the same scalarization techniques, respectively, all for three-objective problems. For a four-objective version of these algorithms, we only pair up the feasible-value-constraint scalarization and the SBG grid, resulting in Algorithm 9.

We test Algorithms 1–9 on example problems. The first problem has three objectives and the second problem has four. The first problem has a Pareto front with a hole in the middle. For this problem, Algorithms 1 and 7 (the CHIM and SBG grids with the feasible-value-constraint scalarization) are observed to be the best performers in yielding a reliable approximation of the front, although Algorithm 7 needs a longer computational time. Finally, we apply Algorithm 9 to generate the boundary of the Pareto front of the four-objective problem which concerns a rocket injector design given in [18, 30].

In a great majority of the papers in the literature, including the articles [8, 25] which introduced the NBI and SBG methods, respectively, numerical experiments and real-life applications involve only two- or three-objective problems, but not four-objective ones. Likewise, in [18, 30] which we mentioned above, although the rocket injector design problem is intrinsically posed as a four-objective problem, a "reduced" three-objective problem is solved, after an assertion that two of the objectives are "correlated." We solve the same problem as a truly four-objective optimization problem and obtain a Pareto front boundary which is at least slightly different from that in [18, 30]. This casts doubt on the assertion above. Avoidance of four-objective problems is common practice, presumably because, with four objectives, numerical implementation is more challenging and visualization is impossible.

The layout of the paper is as follows. We present the problem description along with some preliminaries in section 2. The Pascoletti–Serafini, modified Pascoletti–Serafini and weighted-constraint techniques are recalled in section 3. We introduce our new scalarization technique in section 4, where we also establish the theoretical

results for the new scalarization approach. Algorithms 1–8 are provided in detail in section 5, while Algorithm 9 is described in the appendix. Experimental results are given for these algorithms in section 6. Section 6.2 presents the results for the rocket injector design. Finally a conclusion is presented in section 7.

2. Preliminaries. We recall now some standard notation and tools for multiobjective optimization problems. Let $\mathbb{R}^n$ be the n-dimensional Euclidean space. Denote by $\mathbb{R}_+$ the set of nonnegative real numbers and by $\mathbb{R}_{++}$ the set of strictly positive numbers. For $\ell, m \in \mathbb{N}$, we consider the following multiobjective optimization, problem (P):

$$\text{(P)} \qquad \begin{cases} \min & f(x) := [f_1(x), \ldots, f_\ell(x)] \\ \text{s.t.} & x \in X := \{x \in \mathbb{R}^n \mid g_j(x) \leq 0, \;\; j = 1, \ldots, m\}, \end{cases}$$

where the functions $f_i : \mathbb{R}^n \to \mathbb{R}$, $i = 1, \ldots, \ell$, and $g_j : \mathbb{R}^n \to \mathbb{R}$, $j = 1, \ldots, m$, are continuous. We will assume that the functions f_i are bounded below on the constraint set X, with a known lower bound. In this situation, there is no loss of generality in imposing that

$$\min_{i=1,\ldots,\ell} \{\inf_{x \in X} \; f_i(x)\} > 0. \tag{2.1}$$

The solutions of (P) are called *efficient points* [31] or *Pareto points* [24]. A less restrictive concept of solution of (P) is the one of a *weak efficient* point. To recall the definition of these types of solutions, we use the following standard notation. Given $u, v \in \mathbb{R}^\ell$ we say that

$$\begin{gathered} u \leqq v \text{ iff } u_i \leq v_i \, \forall \, i = 1, \ldots, \ell \text{ and} \\ \exists \, j \text{ such that } u_j < v_j \, ; \\ u < v \text{ iff } u_i < v_i \, \forall \, i = 1, \ldots, \ell \, . \end{gathered} \tag{2.2}$$

Definition 2.1. *A point $\bar{x}$ is said to be efficient for problem* (P) *iff there is no $x \in X$ such that $f(x) \leqq f(\bar{x})$. A point $\bar{x}$ is said to be weak efficient for problem* (P) *iff there is no $x \in X$ such that $f(x) < f(\bar{x})$. We denote the set of efficient and weak efficient points of problem* (P) *as E(P) and WE(P), respectively.*

Suppose that $\bar{x}_{f_i}$ is a minimizer of f_i over X, $i = 1, \ldots, \ell$. Then the vector $\bar{x}_f = [\bar{x}_{f_1}, \ldots, \bar{x}_{f_\ell}]$ is called an *ideal vector*. We refer to the vector $f(\bar{x}_{f_i})$, $i = 1, \ldots, \ell$, as a vector of *individual minima*. A reference vector, called a *utopia vector*, as defined next, is often considered, which is strictly better than every Pareto point.

Definition 2.2. *A utopia vector $u^* \in \mathbb{R}^\ell$ associated with problem* (P) *is defined as $u_i^* := z_i^* - \varepsilon_i$, where $\varepsilon_i > 0 \;\forall\; i = 1, \ldots, \ell$ and z_i^* is the optimal value of the optimization problem*

$$\text{(P}_i\text{)} \qquad \min_{x \in X} \; f_i(x)$$

for $i = 1, \ldots, \ell$. If $\bar{x}_{f_i}$ is a minimizer of problem (P$_i$)*, then $z_i^* = f(\bar{x}_{f_i})$, $i = 1, \ldots, \ell$, are vectors of individual minima.*

3. Classical scalarization techniques. In this section, we recall classical scalarization approaches of MOP which can generate Pareto points in the front. We describe only those approaches which are relevant to our new scalarization method.

Define the set of positive weights,

$$W := \left\{ w \in \mathbb{R}^\ell \mid w_i > 0\,,\ \sum_{i=1}^{\ell} w_i = 1 \right\}.$$

The weighted-constraint approach. Problem (P) is converted into a single objective optimization problem by associating a weight, i.e., a parameter, to each objective function and using the weighted objective functions as constraints. The main advantage of this approach, introduced in [4], is that it can tackle nonconvex problems with disconnected Pareto fronts and disconnected feasible sets. The technique is based on the following fact: For any fixed $k \in \{1, \ldots, \ell\}$ and fixed $w \in W$, consider

$$(\mathrm{P}_w^k) \qquad \begin{cases} \min\limits_{x \in X} & f_k(x) \\ \text{subject to} & w_i\, f_i(x) \le w_k\, f_k(x)\,, \quad i = 1, \ldots, \ell, \ \ i \ne k\,. \end{cases}$$

If $x \in X$ solves (P_w^k) $\forall\ k$, then x is a weakly efficient point (for more details, see [4, section 3]).

The Pascoletti–Serafini approach. Another form of the parameter-based scalarization approach, introduced by Pascoletti and Serafini [26], is widely used for generating points in the Pareto front. This is the technique incorporated into the SBG method in [25]. This method is also known as *goal-attainment method* (see [7, 9, 24]). For a given $w \in W$, the scalarization is stated as follows:

$$(\mathrm{PS}) \qquad \min_{(\alpha, x) \in \mathbb{R} \times X} \alpha, \quad \text{s.t. } w_i\,(f_i(x) - u_i) \le \alpha\,, \ \ \forall\, i = 1, \ldots, \ell\,,$$

where $\alpha \in \mathbb{R}$ is a new variable and u is a utopia vector associated with problem (P). In [12], it is proved that every solution of problem (PS) is a weakly efficient point, which in turn is implemented for generating an approximation of the Pareto front.

The modified Pascoletti–Serafini approach. In this scalarization approach, which can be found in [12,14], the inequality in problem (PS) is simply replaced by equality. It was first introduced as part of the NBI method in [8]. For a given $w \in W$, the method solves the following optimization problem:

$$(\mathrm{MPS}) \qquad \min_{(\alpha, x) \in \mathbb{R} \times X} \alpha, \ \ \text{s.t. } w_i\,(f_i(x) - u_i) = \alpha\,, \ \ \forall\ \ i = 1, \ldots, \ell\,,$$

It turns out that problem (MPS) may yield a solution which is not necessarily weakly efficient but dominated (see [12, 14]). In the process of constructing a Pareto front using this approach, a "weed-out" process is applied to discard dominated points.

4. The feasible-value-constraint approach. We propose here a new scalarization technique, called the *feasible-value-constraint approach*, a description of which is given as follows. First, for some $\hat{x} \in X$, we set the weight

$$(4.1) \qquad w_i := \frac{1/f_i(\hat{x})}{\sum\limits_{j=1}^{\ell} 1/f_j(\hat{x})}\,,$$

which is well defined, thanks to (2.1). It is easy to check that $w \in W$ and

$$w_k f_k(\hat{x}) = w_i f_i(\hat{x}) \ \forall i = 1, 2, \ldots, \ell, \ \ i \neq k\,. \tag{4.2}$$

We will assume from now on that a weight w_i always verifies (4.1). We define the associated scalar problem as

$$(\mathrm{P}^k_{\hat{x}}) \qquad \begin{cases} \min\limits_{x \in X} & f_k(x) \\ \text{subject to} & w_i f_i(x) \leq w_k f_k(\hat{x}), \ \ i = 1, 2, \ldots, \ell, \ \ i \neq k\,. \end{cases}$$

The only difference $(\mathrm{P}^k_{\hat{x}})$ has from (P^k_w) is that the weighted objective functions are constrained by a (constant) weighted feasible value, which results in a problem with a simpler structure. If $\hat{x}$ in problem $(\mathrm{P}^k_{\hat{x}})$ above is replaced by x, then one simply gets problem (P^k_w).

Define $S^k_{\hat{x}} := \{x \in X \mid x \ \text{ solves } \ (\mathrm{P}^k_{\hat{x}})\}$.

4.1. Theoretical results for $(\mathbf{P}^{\boldsymbol{k}}_{\hat{\boldsymbol{x}}})$. Next we present results characterizing efficiency and weak efficiency in terms of solutions of problem $(\mathrm{P}^k_{\hat{x}})$.

THEOREM 4.1. *Assume $\bar{x} \in E(P)$ and w is chosen as in* (4.1). *Then $\bar{x} \in S^k_{\bar{x}}\ \forall\ k$.*

Proof. Assume that the components of w are as in (4.1) with $\hat{x} = \bar{x}$. Suppose that $\bar{x}$ does not solve $(\mathrm{P}^k_{\bar{x}})$ for some k; then $\exists x \in X$ such that

$$f_k(x) < f_k(\bar{x})$$

and

$$w_i f_i(x) \leq w_k f_k(\bar{x}) = w_i f_i(\bar{x}) \ \ \forall\, i \neq k,$$

where we also used (4.2) for $\hat{x} = \bar{x}$ and the fact that w_i and w_k are positive. Therefore

$$f_k(x) < f_k(\bar{x}) \qquad \text{and} \qquad f_i(x) \leq f_i(\bar{x}) \ \ \forall\, i \neq k\,,$$

which contradicts the fact that $\bar{x} \in E(P)$. Hence $\bar{x} \in S^k_{\bar{x}}\ \forall\ k$. ∎

Theorem 4.1 is no longer true when $\bar{x}$ is weakly efficient, as the following example illustrates.

Example 4.2. Consider problem (P) with $f_1(x) := (x+1)^2+1$, $f_2(x) := (x+2)^2+1$ and $f_3(x) := 4$, subject to $x \in \mathbb{R}$. Here, $\bar{x} = 0$ is a weak efficient solution. Consider the weights $w_1 = 10/19$, $w_2 = 4/19$ and $w_3 = 5/19$, where $w_i f_i(\bar{x}) = 20/19$, $i = 1, 2, 3$. For $k = 1$, problem $(\mathrm{P}^1_{\bar{x}})$ can be written as

$$\begin{aligned} \min \quad & f_1(x) \\ \text{subject to} \quad & \\ & w_2 f_2(x) - w_1 f_1(\bar{x}) \leq 0\,, \\ & w_3 f_3(x) - w_1 f_1(\bar{x}) \leq 0\,. \end{aligned}$$

The feasible region is $\{x \mid -4 \leq x \leq 0\}$ and $\bar{x} = 0$ is not an optimal solution of $(\mathrm{P}^1_{\bar{x}})$.

The converse of Theorem 4.1 is true for efficient points.

THEOREM 4.3. *Let $\hat{x} \in X$ and assume that $\bar{x} \in S^k_{\hat{x}}\ \forall\ k$ and w is chosen as in* (4.1). *Then $\bar{x} \in E(P)$.*

Proof. Fix $\hat{x} \in X$ and w chosen as in (4.1) and assume $\bar{x} \in S^k_{\hat{x}}\ \forall\ k$. This implies that $\bar{x}$ is feasible for $(\mathrm{P}^k_{\hat{x}})$ for all k. Hence, $\forall\ k$ we can write

$$w_i f_i(\bar{x}) \leq w_k f_k(\hat{x}),\ \ i = 1, 2, \ldots, \ell \quad \text{and} \quad i \neq k\,.$$

If $\bar{x} \notin E(P)$, then there exists $\tilde{x} \in X$ such that $f_i(\tilde{x}) \leq f_i(\bar{x})\ \forall\ i$ and $\exists k$ such that $f_k(\tilde{x}) < f_k(\bar{x})$. Therefore,

$$w_i f_i(\tilde{x}) \leq w_i f_i(\bar{x}) \leq w_k f_k(\hat{x})\ \ \forall i \neq k \qquad \text{and} \qquad f_k(\tilde{x}) < f_k(\bar{x})\,.$$

Therefore $\bar{x} \notin S^k_{\hat{x}}$. This contradiction implies that $\bar{x} \in E(P)$. ∎

When $S^k_{\hat{x}}$ is a singleton we obtain the following result.

THEOREM 4.4. *Let $\hat{x} \in X$, and w be chosen as in* (4.1). *If for some k, $\bar{x}$ is the unique solution of $(P^k_{\hat{x}})$, then $\bar{x} \in E(P)$.*

Proof. Take $\bar{x}$ and k as in the hypothesis, and suppose $\bar{x}$ is not an efficient solution. Then there exists $x \in X$ such that

$$f_i(x) \leq f_i(\bar{x})\ \ \forall i \tag{4.3}$$

and $\exists$ index r such that

$$f_r(x) < f_r(\bar{x}). \tag{4.4}$$

Case I ($r = k$). From (4.4) we have

$$w_r f_r(x) < w_r f_r(\bar{x})\,. \tag{4.5}$$

From (4.3) and the fact that $\bar{x}$ is feasible for $(\mathrm{P}^k_{\hat{x}})$, we have

$$w_i f_i(x) \leq w_i f_i(\bar{x}) \leq w_k f_k(\hat{x})\ \ \forall i \neq k\,; \tag{4.6}$$

therefore, (4.5) and (4.6) contradict the fact that $\bar{x}$ is a solution of $(\mathrm{P}^k_{\hat{x}})$.

Case II ($r \neq k$). From (4.3), (4.4), and the fact that $\bar{x}$ is feasible for $(\mathrm{P}^k_{\hat{x}})$, we have

$$w_i f_i(x) \leq w_i f_i(\bar{x}) \leq w_k f_k(\hat{x}),\ \ i = 1, 2, \ldots, \ell \quad \text{and} \quad i \neq r, k,$$

and

$$w_r f_r(x) < w_r f_r(\bar{x}) \leq w_k f_k(\hat{x})\,;$$

therefore, x is feasible for $(\mathrm{P}^k_{\hat{x}})$. Moreover, by (4.3) we have

$$w_k f_k(x) \leq w_k f_k(\bar{x})\,.$$

Since $x \neq \bar{x}$ and $(\mathrm{P}^k_{\hat{x}})$ has unique solution $\bar{x}$, we arrived at a contradiction. Hence $\bar{x} \in E(P)$. ∎

To ensure that a solution generated by $(\mathrm{P}^k_{\hat{x}})$ is efficient, either k different problems have to be solved or we have to obtain a unique solution of $(\mathrm{P}^k_{\hat{x}})$ for some k. In general, uniqueness is not easy to verify. However, if the set X is convex and the objective functions are strictly convex, then we know that the solution is unique and without further checking we can claim that the set of solutions of $(\mathrm{P}^k_{\hat{x}})\ \forall\ k$ has at most one element. By Theorem 4.4, this unique element is an efficient point.

COROLLARY 4.5. *Let $\hat{x} \in X$ and w be chosen as in* (4.1). *If for some k, f_k is strictly convex and $(\mathrm{P}^k_{\hat{x}})$ has a solution $\bar{x}$, then $\bar{x} \in E(P)$.*

Proof. Since $\bar{x}$ solves $(\mathrm{P}^k_{\hat{x}})$ and f_k is strictly convex, the set $S^k_{\hat{x}}$ is a singleton. Therefore, by Theorem 4.4, $\bar{x} \in E(P)$. ∎

Without convexity assumptions, a solution of $(\mathrm{P}^k_{\hat{x}})$ for any k would be weakly efficient. This is asserted in our next result.

THEOREM 4.6. *Fix $\hat{x} \in X$ and let w be chosen as in* (4.1). *Assume $\bar{x} \in S^k_{\hat{x}}$ for some k. Then $\bar{x} \in WE(P)$.*

Proof. Let $\bar{x} \in X$ solve $(\mathrm{P}^k_{\hat{x}})$ for some $k \in \{1, \ldots, \ell\}$. Then we have

$$w_k f_k(\bar{x}) \leq w_k f_k(x)$$

for every x such that

$$w_i f_i(x) \leq w_k f_k(\hat{x}) = w_i f_i(\hat{x}), \quad i = 1, 2, \ldots, \ell \quad \text{and} \quad i \neq k\,. \tag{4.7}$$

Since $\bar{x}$ is feasible for $(\mathrm{P}^k_{\hat{x}})$, by (4.7) we have

$$w_i f_i(\bar{x}) \leq w_k f_k(\hat{x}) = w_i f_i(\hat{x}), \quad i = 1, 2, \ldots, \ell \quad \text{and} \quad i \neq k\,. \tag{4.8}$$

Assume that $\bar{x} \in X$ is not a weak efficient point of (P); then there is a $\tilde{x} \in X$ such that

$$f_j(\tilde{x}) < f_j(\bar{x}), \quad j = 1, 2, \ldots, \ell\,. \tag{4.9}$$

Two cases have to be considered.

Case I. Assume that $\tilde{x} \in X$ is feasible for $(\mathrm{P}^k_{\hat{x}})$. In this case we have that

$$w_i f_i(\tilde{x}) \leq w_k f_k(\hat{x}), \quad i = 1, 2, \ldots, \ell \quad \text{and} \quad i \neq k\,. \tag{4.10}$$

Since $w_k > 0$, from (4.9) we have

$$w_k f_k(\tilde{x}) < w_k f_k(\bar{x}). \tag{4.11}$$

Hence (4.10) and (4.11) contradict the fact that $\bar{x} \in X$ is a solution of $(\mathrm{P}^k_{\hat{x}})$. Hence Case I cannot happen, and we are left with the following case.

Case II. Assume that $\tilde{x} \in X$ is not feasible for $(\mathrm{P}^k_{\hat{x}})$. Therefore, $\exists\, i \neq k$ such that

$$w_i f_i(\tilde{x}) > w_k f_k(\hat{x}) = w_i f_i(\hat{x})\,.$$

Also, by (4.8) we have that

$$w_i f_i(\hat{x}) \geq w_i f_i(\bar{x})\,.$$

These two expressions yield

$$w_i f_i(\tilde{x}) > w_i f_i(\bar{x})\,,$$

so for this index i we have

$$f_i(\tilde{x}) > f_i(\bar{x})\,.$$

This contradicts (4.9) for $j = i$. Therefore, $\bar{x}$ must be weakly efficient. ∎

5. Algorithms. In this section, we give full descriptions of Algorithms 1–8, all devised for three objectives. Algorithm 9, which works for four-objective problems, is provided in Appendix A.

Algorithms 1–4 use the CHIM grid and Algorithms 5–8 the SBG grid. The algorithms within each of these two major groups differ from one another depending on the scalarization technique that is used. As also reviewed in section 1.3, in Algorithms 1–4 the following scalarization techniques are used, respectively:

(i) the feasible-value-constraint scalarization (problem ($P_{\hat{x}}^k$)),
(ii) the weighted-constraint scalarization (problem (P_w^k)),
(iii) the Pascoletti–Serafini scalarization (problem (PS)) and
(iv) the modified Pascoletti–Serafini scalarization (problem (MPS)).

Following up the descriptions of the CHIM and SBG grids that were provided in section 1.2, geometric depictions are given in Figures 1 and 2. Figures 1(a) and (b) illustrate the CHIM for two objectives (as a convex hull of the two individual minima denoted IM) and demonstrate two different ways of discretizing the CHIM and so forming a CHIM grid. One approach (Figure 1(a)) takes a common base point and passes rays emanating from the common point. The other approach (Figure 1(b)), which is the same as that in [8], takes equidistant base points and passes parallel rays, which are normal to the CHIM, through each base point. Figure 1(c) illustrates a (triangular) CHIM grid for three objectives in the weight space, which can be obtained by extending to dimension three either of the two-objective cases in Figures 1(a) and (b).

Figure 2 illustrates the use of an SBG grid where the new feasible-value-constraint scalarization has been used.

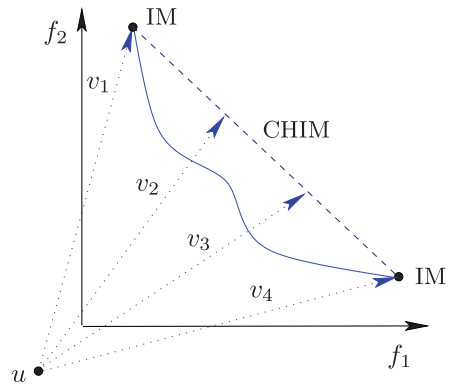
5.1. Three-objective algorithms using the CHIM grid. In Step 2 of Algorithms 1–4, each objective function, subject to the original constraints of the problem, is minimized individually. This yields the individual minima of the Pareto front.

In Step 3 of Algorithms 1–4, a CHIM grid is generated. The incremental grid points in the partition correspond to rays emanating from a single utopia point (base point), as in Figure 1(a). If the utopia point is chosen to be sufficiently away from the Pareto front, these rays will become almost parallel to one another with a more or less equal spacing between each consecutive ray. Almost parallel rays and a regular weight partition are expected to result in a set of almost uniformly spaced points on the Pareto front. Ease of computer programming of the algorithms with three or more objective functions justifies further the choice of the scheme in Figure 1(a).

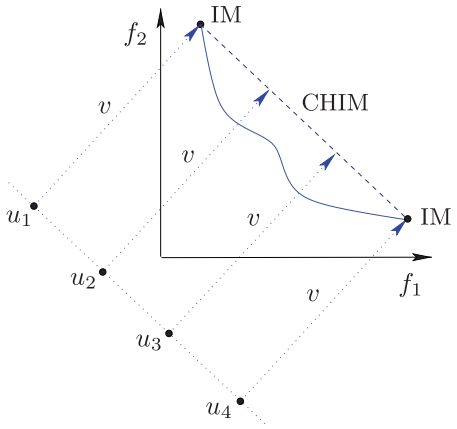
The CHIM grid is constructed by using the weights corresponding to the points of individual minima. Figure 3, which is an extension to three objectives of the diagram in [20] for two objectives, illustrates these points of individual minima and provides the equations of the rays emanating from the utopia point (u_1, u_2, u_3) and passing through the points of individual minima. Suppose that $\bar{x}_{f_i}$ solves (P_i), $i = 1, 2, 3$. Then $f_i^* := f_i(\bar{x}_{f_i})$, $i = 1, 2, 3$. Two points of individual minima are already displayed as (red) full dots in Figure 3. The third point of individual minimum, which cannot be shown in the figure, is simply $(f_1(\bar{x}_{f_3}), f_2(\bar{x}_{f_3}), f_3^*)$, and the equations of the ray passing through this point are given by

$$c_1 (f_1 - u_1) = c_2 (f_2 - u_2),$$
$$c_1 (f_1 - u_1) = (1 - c_1 - c_2) (f_3 - u_3),$$

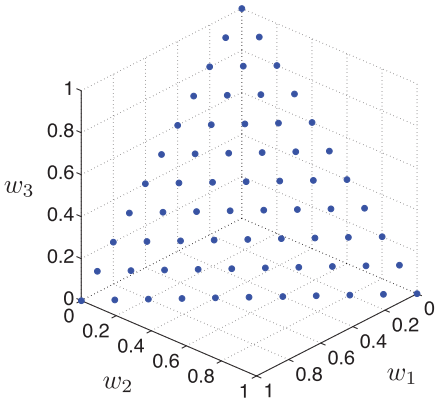
where f_1, f_2, and f_3 are variables. For the first individual minimum point $(f_1^*, f_2(\bar{x}_{f_1}), f_3(\bar{x}_{f_1}))$, the weights are given as $w_1 = a_1$, $w_2 = a_2$, and $w_3 = 1 - a_1 - a_2$.



(a) A common base point u and different directions v_i.



(b) Same direction v and different base points u_i.



(c) A triangular CHIM grid of weights

FIG. 1. *Illustration of the CHIM and CHIM grids.*

Evaluation of the pertaining equations at the point $(f_1^*, f_2(\bar{x}_{f_1}), f_3(\bar{x}_{f_1}))$ gives

$$\begin{aligned} a_1\,(f_1^* - u_1) &= a_2\,(f_2(\bar{x}_{f_1}) - u_2),\\ a_1\,(f_1^* - u_1) &= (1 - a_1 - a_2)\,(f_3(\bar{x}_{f_1}) - u_3)\,, \end{aligned}$$

which can be solved for a_1 and a_2 as

$$a_k = \frac{\prod_{\substack{j=1\\ j\neq k}}^{3} (f_j(\bar{x}_{f_1}) - u_j)}{\prod_{j=1,2}(f_j(\bar{x}_{f_1}) - u_j) + \prod_{j=2,3}(f_j(\bar{x}_{f_1}) - u_j) + \prod_{j=1,3}(f_j(\bar{x}_{f_1}) - u_j)} =: \varphi_k(\bar{x}_{f_1}) \tag{5.1}$$

for $k = 1, 2$, where the functions $\varphi_k : \mathbb{R}^3 \to \mathbb{R}$, $k = 1, 2$, have been defined for convenience. The weights at each individual minimum point can then be written as

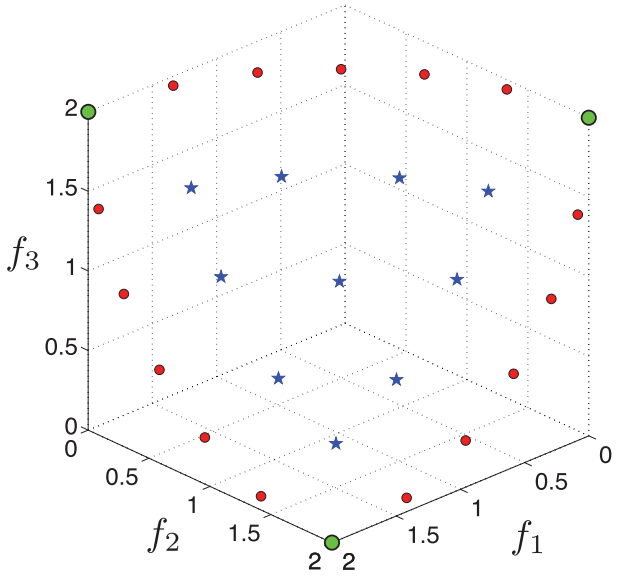


FIG. 2. *An SBG grid. Green big circles indicate individual minima, red small circles depict boundary Pareto points, and blue stars depict interior points of the grid.*

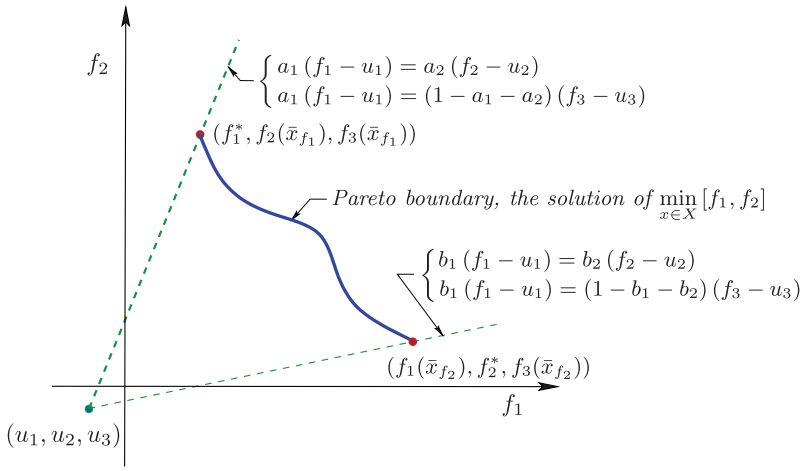


FIG. 3. *Determination of the individual minima weights* (a_1, a_2) *and* (b_1, b_2) *in the* $f_1f_2f_3$*-space. The* f_3*-axis points out of the page perpendicularly.*

$$a_k = \varphi_k(\bar{x}_{f_1})\,, \quad b_k = \varphi_k(\bar{x}_{f_2})\,, \quad c_k = \varphi_k(\bar{x}_{f_3})\,, \tag{5.2}$$

for $k = 1, 2$.

The indexing that is used in Step 3 of Algorithms 1–4 for the weights over the CHIM grid is illustrated in Figure 4. The scalarized problems in Step 4 of Algorithms 1–4 are solved at each node of the grid in the following manner: first, at nodes from $(0, 0)$ to $(0, 4)$, i.e., nodes with $i_1 = 0$ and $i_2 = 0, 1, 2, 3, 4$, sequentially; second, at nodes from $(1, 0)$ to $(1, 3)$; third, at nodes from $(2, 0)$ to $(2, 2)$; fourth, at nodes $(3, 0)$ and $(3, 1)$; and finally, at $(4, 0)$.

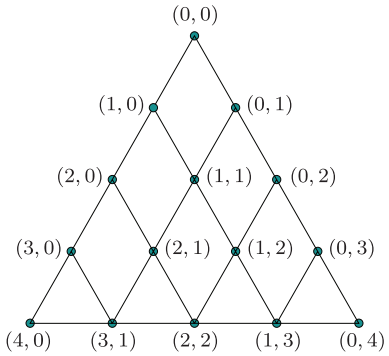


FIG. 4. *The indexing* (i_1, i_2) *over the CHIM grid for Step* 3 *of Algorithms* 1–4.

Algorithm 1 (CHIM grid + $(\mathrm{P}^k_{\hat{x}})$).

Step 1 (*Input*)
Choose a utopia point $u = (u_1, u_2, u_3)$. Set the number of partition points N in the $(N+1)(N+2)/2$ grid points. Set $s = 0$.

Step 2 (*Determine the individual minima*)
Find $\bar{x}_{f_i}$ that solves Problem $(\mathrm{P}_i) := \min_{x \in X} f_i(x)$, $i = 1, 2, 3$.
Set $\bar{F} := [f_1(\bar{x}_{f_3}), f_2(\bar{x}_{f_3}), f_3(\bar{x}_{f_3})]$.

Step 3 (*Generate weights over the CHIM grid*)
Compute a_k, b_k and c_k, $k = 1, 2$, using (5.1)–(5.2).
For $i_1 := 0, 1, \ldots, (N-1)$
{
  Let $\hat{w}^{i_1}_k := a_k + i_1(c_k - a_k)/N, \quad k = 1, 2,$
  and $\tilde{w}^{i_1}_r := b_r + i_1(c_r - b_r)/N, \quad r = 1, 2$.
  For $i_2 := 0, 1, \ldots, (N - i_1)$
  {
    Let $w^{(i_1,i_2)}_\ell := \hat{w}^{i_1}_\ell + i_2(\tilde{w}^{i_1}_\ell - \hat{w}^{i_1}_\ell)/(N - i_1), \quad \ell = 1, 2$.
  }
}

Step 4 (*Select* $\hat{x}$ *and solve the scalarized problems*)
For $i_1 := 0, 1, \ldots, (N-1)$
{
  For $i_2 := 0, 1, \ldots, (N - i_1)$
  {
    Let $w_k = w^{(i_1,i_2)}_k$, $k = 1, 2$. Let $w = (w_1, w_2, 1 - w_1 - w_2)$.
    (a) If possible, find $\hat{x}$ that solves the auxiliary problem
    $(\mathrm{P}_{\mathrm{aux}}) : \min\limits_{x \in X} w_k(f_k(x) - u_k)$, s.t. $w_i(f_i(x) - u_i) = w_k(f_k(x) - u_k)$,
    $i = 1, 2, 3, \ i \neq k$, for any $k \in \{1, 2, 3\}$. If it is not possible to solve $(\mathrm{P}_{\mathrm{aux}})$, then skip parts (b) and (c) below.
    (b) With the given $\hat{x}$ and w, find $\bar{x}_k$ that solves problem $(\mathrm{P}^k_{\hat{x}})$, for $k = 1, 2, 3$.
    (c) (*Determine efficient or weak efficient points*)
    Let $s = s + 1$.
    (i) If $\bar{x}_1 = \bar{x}_2 = \bar{x}_3$, then set $\bar{x} = \bar{x}_1$ (an efficient point) and

$F(s) := [f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x})]$.

(ii) If $\bar{x}_1 = \bar{x}_2 = \bar{x}_3$ does not hold, then one has either two or three weak efficient points:

- If $\{\bar{x}_1, \bar{x}_2, \bar{x}_3\} = \{\bar{y}, \bar{z}\}$, then set $F(s) := [f_1(\bar{y}), f_2(\bar{y}), f_3(\bar{y})]$, $F(s+1) := [f_1(\bar{z}), f_2(\bar{z}), f_3(\bar{z})]$, and $s = s + 1$.
- If $\bar{x}_1$, $\bar{x}_2$ and $\bar{x}_3$ are all different from one another, then they are all weak efficient points and so set $F(s + k - 1) := [f_1(\bar{x}_k), f_2(\bar{x}_k), f_3(\bar{x}_k)]$, $k = 1, 2, 3$, and $s = s + 2$.

}

}

Step 5 (*Output*)

Set $F(s+1) := \bar{F}$.

The array of Pareto points, F, is an approximation of the Pareto front, with $(s+1)$ elements.

Remark 5.1. The number of distinct Pareto minima generated by Algorithm 1 is in general more than just $(N+1)(N+2)/2$, because of the multiple Pareto points found in Step 4(c)(ii).

In Step 4(a) of Algorithm 1, having to solve an additional problem in order to find a feasible point $\hat{x}$ may at first look like a disadvantage. However, Step 4(a) provides a feasible solution (i.e., $\hat{x}$) which is located in the Pareto front. As a result, Step 4(b) would consume less computational time in finding Pareto points. It is possible that ($\mathrm{P}_{\mathrm{aux}}$) may not have a solution because of the ray not intersecting the feasible objective values, in which case parts (b) and (c) are skipped.

Step 4(c)(i) of Algorithm 1 generates efficient points when the solutions of problems ($\mathrm{P}^1_{\hat{x}}$), ($\mathrm{P}^2_{\hat{x}}$), and ($\mathrm{P}^3_{\hat{x}}$) are the same, which is a consequence of Theorem 4.3. Otherwise, by Theorem 4.6, all points are weak efficient, which is a case treated in Step 4(c)(ii). The latter case typically occurs when the Pareto front is disconnected. The prospect of finding three Pareto points for a single choice of grids/weights is another significant advantage of this method. In addition, the chance of three Pareto points for a single choice of weights is also possible for disconnected domain if Step 4(a) finds a solution, which again points to this advantage of the new approach.

Algorithm 2 (CHIM grid + ($\mathbf{P}^k_w$)).

Steps 1–3. Do the same as in Steps 1–3 of Algorithm 1.

Step 4 (*Solve the scalarized problems*)

For $i_1 := 0, 1, \ldots, (N-1)$

{

For $i_2 := 0, 1, \ldots, (N - i_1)$

{

Let $w_k = w_k^{(i_1, i_2)}$, $k = 1, 2$. Let $w = (w_1, w_2, 1 - w_1 - w_2)$.

(a) Find $\bar{x}_k$ that solves problem (P^k_w), $k = 1, 2, 3$.

(b) *Determine the weak efficient points*:

Let $s := s + 1$.

(i) If $\bar{x}_1 = \bar{x}_2 = \bar{x}_3$, then set $\bar{x} = \bar{x}_1$ (a weak efficient point) and $F(s) := [f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x})]$.

(ii) If $\bar{x}_1 = \bar{x}_2 = \bar{x}_3$ does not hold, then, of $\bar{x}_1$, $\bar{x}_2$, and $\bar{x}_3$, identify a point as a weak efficient point if it is not dominated by any of the other two. Set $r := 0$.

For $k = 1, 2, 3$

{
If x_k is weak efficient, then set
$F(s+r) = [f_1(\bar{x}_k), f_2(\bar{x}_k), f_3(\bar{x}_k)]$ and $r = r+1$.
}
Set $s = s + r - 1$.
}
}

Step 5. Do the same as in Step 5 of Algorithm 1.

Note that, in Step 4(b) of Algorithm 2, which parallels Step 4(c) of Algorithm 1, we implement the results from [4, Proposition 3.3] (see also [28, p. 112] for further details).

Algorithm 3 (CHIM grid + (PS)).

Steps 1–3. Do the same as in Steps 1–3 of Algorithm 1.

Step 4 *(Solve the scalarized problem)*

For $i_1 := 0, 1, \ldots, (N-1)$
{
For $i_2 := 0, 1, \ldots, (N - i_1)$
{
Let $w_k = w_k^{(i_1, i_2)}$, $k = 1, 2$. Let $w = (w_1, w_2, 1 - w_1 - w_2)$.
Find $\bar{x}$ and α that solves problem (PS). Let $s := s + 1$.
Set $F(s) := [f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x})]$.
}
}

Step 5. Do the same as in Step 5 of Algorithm 1.

Algorithm 4 (CHIM grid + (MPS)).

Steps 1–5. Do the same as in Steps 1–5 of Algorithm 3, except that in Step 4 of Algorithm 3, solve problem (MPS) instead of problem (PS).

5.2. Three-objective algorithms using the SBG grid. It is noted that the SBG approach only allows one to pass parallel rays on the front; this means that the SBG grid requires rays with the same direction but different base points. On the other hand, the scalarization approaches described in sections 3 and 4 allow one to pass rays into the front from a fixed base point with different directions.

In Step 2 of Algorithm 5 below, the individual minima of the objectives are computed. In Step 3, the discretized boundary of the Pareto front of the two-objective combination is constructed. That is, Pareto fronts of the objective pairs $\{f_1, f_2\}$, $\{f_2, f_3\}$, and $\{f_1, f_3\}$ are generated which are the true boundaries of the Pareto front of problem (P). In Step 4, the base points are defined by solving LP problems as in [25, section 7.4]. In Step 5, the interior of the Pareto front is approximated by solving the scalarized problems.

Let e_k be the kth standard basis vector of $\mathbb{R}^3$, e.g., $e_2 = [0, 1, 0]$.

Algorithm 5 (SBG grid + ($\mathbf{P}^k_{\hat{x}}$)).

Step 1 (*Input*)
Set the number of partition points N.

Step 2 (*Determine the individual minima*)
Find $\bar{x}_{f_k}$ that solves problem $(\mathrm{P}_k) := \min_{x \in X} f_k(x)$ for $k = 1, 2, 3$.
Set $F(k) := [f_1(\bar{x}_{f_k}), f_2(\bar{x}_{f_k}), f_3(\bar{x}_{f_k})]$ and $W(k) := e_k$ for $k = 1, 2, 3$.
Set $s = k + 1$.

Step 3 (*Generate Pareto fronts of the objective pairs*)
Set the direction vector $v^{(i,k)}$ *in the* $f_i f_k$*-space,* $i = 1, 2$, $k = 2, 3$, $i \neq k$:

Let $v^{(i,k)} := [v_1^{(i,k)}, v_2^{(i,k)}]$.
Set $v_1^{(i,k)} = \max\{f_i(\bar{x}_{f_i}), f_i(\bar{x}_{f_k})\} - f_i(\bar{x}_{f_i})$
and $v_2^{(i,k)} := \max\{f_k(\bar{x}_{f_i}), f_k(\bar{x}_{f_k})\} - f_k(\bar{x}_{f_k})$.
Auxiliary weights for a parallel ray in the f_if_k-space:
Let $c = 1/(v_1^{(i,k)} + v_2^{(i,k)})$, $\tilde{w}_1^{(i,k)} = cv_1^{(i,k)}$, $\tilde{w}_2^{(i,k)} = 1 - \tilde{w}_1$.
For $j := 1, \ldots, (N-1)$
{
(a) (*Generate Pareto points for $\{f_1, f_2\}$*)
Let $w_1 = (N-j)/N$; $w_2 = 1 - w_1$; $w_3 = 0$.
Set $W(s) := [w_1, w_2, w_3]$ and $s = s + 1$.
Form a base point for $\{f_1, f_2\}$:
Let $u_1 = w_1 f_1(\bar{x}_{f_1}) + w_2 f_1(\bar{x}_{f_2})$; $u_2 = w_1 f_2(\bar{x}_{f_1}) + w_2 f_2(\bar{x}_{f_2})$.
(i) If possible, find $\hat{x}$ that solves the auxiliary problem
$(\mathrm{P_{aux}}) := \min_{x \in X}\ \tilde{w}_k^{(1,2)}(f_k(x) - u_k^{(1,2)})$ s.t. $\tilde{w}_i^{(1,2)}(f_i(x) - u_i^{(1,2)}) = \tilde{w}_k^{(1,2)}(f_k(x) - u_k^{(1,2)})$, $i = 1, 2$, $i \neq k$, for any $k \in \{1, 2\}$. If it is not possible to solve $(\mathrm{P_{aux}})$, then skip parts (ii) and (iii) below.
(ii) Find $\bar{x}_1$ and $\bar{x}_2$ that solve problems $(\mathrm{P}_{\hat{x}}^1)$ and $(\mathrm{P}_{\hat{x}}^2)$, respectively.
(iii) *Determine efficient or weak efficient points*:
- If $\bar{x}_1 = \bar{x}_2 = \bar{x}$ (an efficient point), then set $F(s) := [f_1(\bar{x}), f_2(\bar{x}), f_3(\bar{x})]$ and $s = s + 1$.
- If $\bar{x}_1 \neq \bar{x}_2$, then $\bar{x}_1$ and $\bar{x}_2$ are weak efficient points and so set $F(s + k - 1) := [f_1(\bar{x}_k), f_2(\bar{x}_k), f_3(\bar{x}_k)]$, $k = 1, 2$, $W(s + 1) = W(s)$ and $s = s + 2$.

(b) (*Generate Pareto points for $\{f_2, f_3\}$*)
Let $w_1 = 0$; $w_2 = (N-j)/N$; $w_3 = 1 - w_2$.
Do the same as in the rest of Step 3(a) for the new objective pair $\{f_2, f_3\}$ instead of $\{f_1, f_2\}$.
(c) (*Generate Pareto points for $\{f_1, f_3\}$*)
Let $w_1 = (N-j)/N$; $w_2 = 0$; $w_3 = 1 - w_1$.
Do the same as in the rest of Step 3(a) for the new objective pair $\{f_1, f_3\}$ instead of $\{f_1, f_2\}$.
}

Step 4 (*Form base points in the $f_1f_2f_3$-space*)
Solve an LP problem as in [25, section 7.4] to get an array of base points $U(r) = [u_1(r), u_2(r), u_3(r)]$, $r = 1, \ldots, (N-1)(N-2)/2$.

Step 5 (*Generate interior Pareto points for $\{f_1, f_2, f_3\}$*)
Set the direction vector v and weights for a parallel ray in the $f_1f_2f_3$-space:
Let $v := [v_1, v_2, v_3]$. Set $v_j := \max\{f_j(\bar{x}_{f_1}), f_j(\bar{x}_{f_2}), f_j(\bar{x}_{f_3})\} - f_j(\bar{x}_{f_j})$, $j = 1, 2, 3$.
Let $c = 1/(v_1 + v_2 + v_3)$, $w_1 = cv_1$, $w_2 = cv_2$, and $w_3 = 1 - w_1 - w_2$.
For $r = 1, \ldots, (N-1)(N-2)/2$
{
(a) If possible, find $\hat{x}$ that solves the auxiliary problem
$(\mathrm{P_{aux}}) := \min_{x \in X}\ w_k(f_k(x) - u_k(r))$, s.t. $w_i(f_i(x) - u_i(r)) = w_k(f_k(x) - u_k(r))$, $i = 1, 2, 3$, $i \neq k$, for any $k \in \{1, 2, 3\}$. If it is not possible to solve $(\mathrm{P_{aux}})$, then skip parts (b) and (c) below.
(b) With the given $\hat{x}$ and w, find $\bar{x}_k$ that solves problem $(\mathrm{P}_{\hat{x}}^k)$, $k = 1, 2, 3$.
(c) (*Determine efficient or weak efficient points*)
Do the same as in Step 4(c) of Algorithm 1.
}

Step 6 (*Output*)

The array of Pareto points, F, is an approximation of the Pareto front.

Algorithms 6–8 (SBG grid + (P_w^k) or (PS) or (MPS)). In Steps 1–4 of Algorithms 6 and 7, do the same as in Steps 1–4 of Algorithm 5, and in Step 5, do the same as in Step 4 of Algorithms 2 and 3, respectively, except in Steps 3 and 5 of Algorithm 5, solve problem (P_w^k) or (PS), respectively, instead of problem ($\mathrm{P}_{\hat{x}}^k$). In Algorithm 8, do the same as in Algorithm 7, except solve problem (MPS) instead of problem (PS).

5.3. A four-objective algorithm using the SBG grid. The four-objective algorithm is an extension of the three-objective Algorithm 5 (with the SBG grid), and its description is provided in Appendix A. Note that, with four objectives, we do not consider (P_w^k) and the other scalarization techniques any longer because, in view of the numerical experiments, the new feasible-value-constraint scalarization given in problem ($\mathrm{P}_{\hat{x}}^k$) is superior over the other techniques—see section 6.1.

6. Numerical experiments. In this section, we test and compare Algorithms 1–9, described in section 5 and Appendix A, by means of three- and four-objective example problems. A discussion of the two-objective case can be found in [28, section 5.4]. The boundary of the Pareto front in the two-objective case is just a finite (and usually a small) number of points (in the case of a connected Pareto front the boundary of the front is a set of two points). Therefore, the task of approximating the boundary of the Pareto front is particularly challenging for the three- and four-objective cases.

We have implemented Algorithms 1–9 and modeled the example problems by using MATLAB. The MATLAB optimization solver `fmincon` has been used in solving the scalarized problems, with default options and the SQP method as the optimization algorithm. In the generation of the SBG grid, MATLAB `linprog` has been used, with default options, for solving the associated linear programming (LP) problems. The computations have been performed on an HP ENVY 14 laptop with 4 GB RAM and core i5 at 1.60 GHz.

6.1. A three-objective test problem. For an algorithm to be deemed successful, it is expected to approximate all parts of the Pareto surface reasonably well and in particular help one identify the true boundary of the Pareto surface. To compare the performances of the algorithms, we consider a three-objective problem which has a Pareto front with a boundary that is difficult to construct. The feasible region of this problem is defined as the intersection of a sphere with the complement of an ellipsoid. The Pareto front of the problem is bounded and multiply connected, but not simply connected, in that there is a hole inside the Pareto front. This poses a difficulty for the existing techniques, as will be seen in the experiments. The problem description is given as follows:

$$\begin{aligned} \min \quad & [x_1, x_2, x_3] \\ \text{subject to} \quad & (x_1-2)^2+(x_2-2)^2+(x_3-2)^2 \le 4, \\ & (1.05)^2(x_1+x_2+x_3)^2 - 4(x_1^2+x_2^2+x_3^2) + 7.18 \le 0, \\ & 0 \le x_i \le 4, \quad i=1,2,3. \end{aligned}$$

Note that $f_1(x) = x_1$, $f_2(x) = x_2$, and $f_3(x) = x_3$. We chose the point $(-10,-10,-10)$ to be the utopia point.

With the same number of CHIM grid points, each algorithm requires substantially different amounts of CPU time. In order to make fair comparisons of the algorithms,

TABLE 1
The three-objective test problem: numerical performance of Algorithms 1–4.

Algorithms	No. of grid points	No. of subproblems	CPU time (sec)	No. of Pareto points	No. of non-Pareto points
1	231	231×4=924	47	688	Nil
2	406	406×3=1218	47	434	10
3	666	666	43	663	3
4 (NBI method)	1081	1081	52	186	895 (51 nonweedable)

TABLE 2
The three-objective test problem: numerical performance of Algorithms 5–8.

Algorithms	No. of subproblems	CPU time (sec)	No. of Pareto points	No. of non-Pareto points
5	60×3+171(LP)+171×4=1035	226	463	2
6	69×2+231(LP)+231×3=1062	235	383	30
7 (SBG method)	105+561(LP)+561=1227	240	666	Nil
8	105+561(LP)+561=1227	236	240	426 (50 nonweedable)

the CHIM grid for each algorithm was chosen to be different in size (some finer, i.e., with more grid points, and some coarser) in such a way that the amount of CPU time each algorithm took was similar (between 43 and 52 seconds). The information on the number of grid points, the CPU time, the number of subproblems solved, and the solution points obtained is listed in Table 1, for Algorithms 1–4. Figure 5 depicts the Pareto points generated by each algorithm, to approximate the Pareto front.

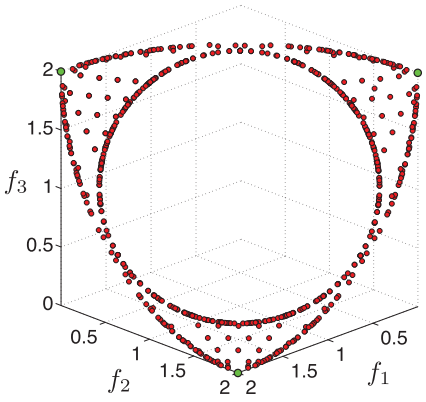
Although Algorithms 1 and 3 both generate (almost equally) many more Pareto points compared with Algorithms 2 and 4, the approximation of the Pareto front by Algorithm 1, as shown in Figure 5(a), is much more succesful, especially in terms of revealing the outer and inner boundaries of the Pareto front.

In computational multiobjective optimization, it is common practice to eliminate a computed point, which, after being compared with the other computed points, is deemed as dominated. This process is referred to as a *weeding out* process [9]. It was possible to weed out the dominated, or non-Pareto, points generated by Algorithms 2 and 3 (see Figure 5(b) and (c)), but not those generated by Algorithm 4 (Figure 5(d)).

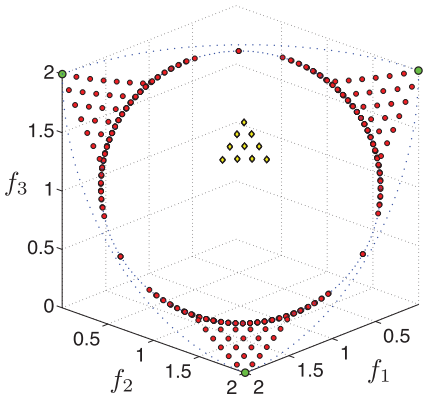
The poor performance of Algorithm 4, i.e., the NBI method, with this problem is understandable, because the NBI method requires the Pareto front to be multiply connected, which is not the case here. We include the NBI method for the sake of completeness, as well as to illustrate the challenging nature of problems with Pareto fronts, or feasible sets, which are not multiply connected.

Recall that Algorithms 5–8 employ the SBG grid. As in the case of Algorithms 1–4, the grid for each of Algorithms 5–8 was chosen to be of varying size so that computations for constructing an approximation of the Pareto front can be completed in similar lengths of CPU time (226 to 240 seconds), as seen in Table 2.

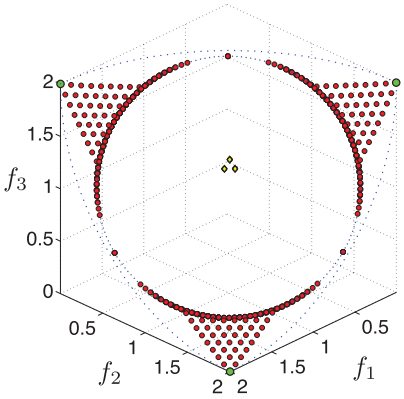
Algorithm 7, which implements the SBG method, finds the highest number of Pareto points, as reported in Table 2. Algorithm 5, which employs the new feasible-value-constraint scalarization technique, finds the second highest number of Pareto



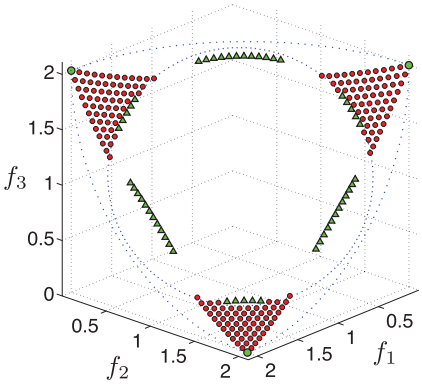
(a) Algorithm 1.



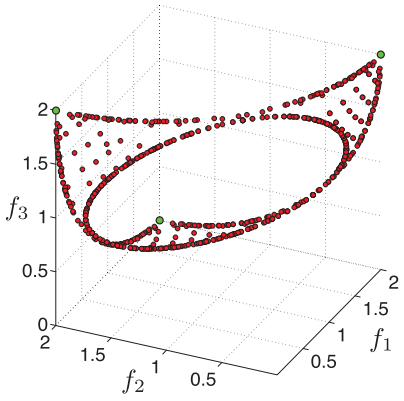
(b) Algorithm 2.



(c) Algorithm 3.

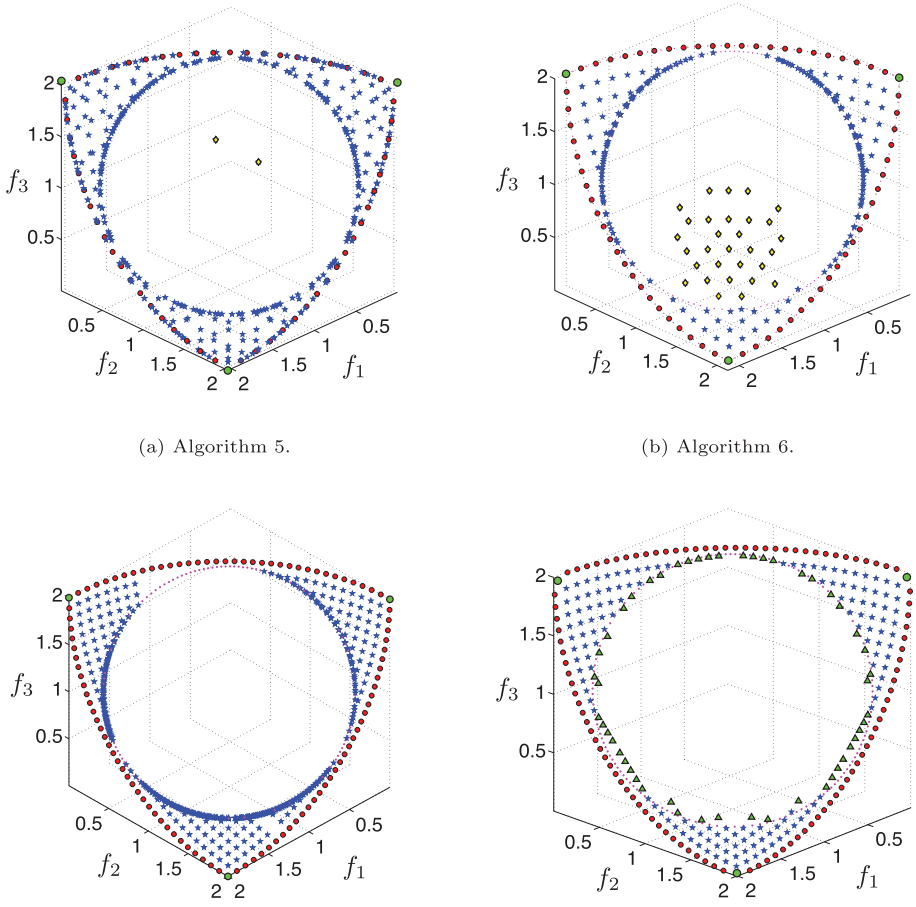


(d) Algorithm 4 (NBI method).



(e) Rotated view of the front in (a).

FIG. 5. *The three-objective test problem—Pareto front approximations by Algorithms* 1–4 *(which use the CHIM grid): green big circles represent the individual minima, red small circles depict Pareto points, yellow diamonds depict weeded-out dominated points, green triangles depict dominated points that could not be weeded out, and blue dotted curves in* (a)–(d) *are the actual inner and outer boundaries of the Pareto front.*



(a) Algorithm 5.

(b) Algorithm 6.

(c) Algorithm 7 (SBG method).

(d) Algorithm 8.

FIG. 6. *The three-objective test problem—Pareto front approximations by Algorithms* 5–8 *(which use the SBG grid): green big circles represent the individual minima, red small circles depict boundary Pareto points, blue stars depict interior Pareto points, yellow diamonds depict weeded-out dominated points, green triangles depict dominated points that could not be weeded out, and magenta dotted curves depict the actual inner and outer boundaries of the Pareto front.*

points. Figure 6 shows performance in terms of how well the Pareto front is approximated by each algorithm. The outer boundary of the Pareto front has been well approximated by all algorithms, owing to the fact that (i) the three segments of the outer boundary (approximated by the SBG grid) are two-objective connected Pareto fronts and (ii) each of the four scalarization techniques works well for two objectives and connected Pareto fronts (see [4] for the performances of the three existing scalarization techniques for two-objective problems). The inner boundary, however, as well as the part between the inner and outer boundaries, has been best approximated by Algorithm 5, which demonstrates the capability of the new scalarization technique.

Overall, Algorithms 1 and 5, both using the new scalarization, one employing the CHIM grid and the other the SBG grid, manage to produce the best approximations of the Pareto front. When the lengths of the CPU times are taken into account Algorithm 1 clearly emerges as the most efficient for this test problem.

6.2. An application to rocket injector design. A multiobjective rocket injector design problem was previously studied in [18, 30]. The design problem has two primary considerations: (i) improvement of the performance and (ii) lengthening of the lifetime. The performance is mainly influenced by the axial length of the thrust chamber, while the lifetime is associated with the thermal field inside the thrust chamber. For a diagram of the injector illustrating these design "parameters," we refer the reader to [18]. As expected, there is a conflicting interplay: High thermal stresses on the injector and the thrust chamber are induced by high temperatures and thus reduce the life of the components. For the rocket injector design, four design variables are defined in [18], which are listed below:

x_1: hydrogen flow angle,
x_2: oxidizer post tip thickness,
x_3: decrement with respect to the baseline cross-section area of the tube carrying oxygen,
x_4: increment with respect to the baseline cross-section area of the tube carrying hydrogen.

Four objective functions are identified and listed in [18] as follows:

$f_1(x)$: face temperature, which is the maximum temperature of the injector face,
$f_2(x)$: tip temperature, which is the maximum temperature on the post tip of the injector,
$f_3(x)$: combustion length, which is the distance from the inlet where 99% of the combustion is complete,
$f_4(x)$: wall temperature, which is the wall temperature at three inches (fourth probe) from the injector face.

The rocket injector design problem is formulated mathematically as follows [18]:

$$\min \quad [f_1, f_2, f_3, f_4],$$

where

$$\begin{aligned} f_1 = {} & 0.692 + 0.477x_1 - 0.687x_4 - 0.08x_3 - 0.065x_2 - 0.167x_1^2 - 0.0129x_1x_4 \\ & + 0.0796x_4^2 - 0.0634x_1x_3 - 0.0257x_3x_4 + 0.0877x_3^2 - 0.0521x_1x_2 \\ & + 0.00156x_2x_4 + 0.00198x_2x_3 + 0.0184x_2^2\,, \end{aligned}$$

$$\begin{aligned} f_2 = {} & 0.37 - 0.205x_1 + 0.0307x_4 + 0.108x_3 + 1.019x_2 - 0.135x_1^2 + 0.0141x_1x_4 \\ & + 0.0998x_4^2 + 0.208x_1x_3 - 0.0301x_3x_4 - 0.226x_3^2 + 0.353x_1x_2 - 0.0497x_2x_3 \\ & - 0.423x_2^2 + 0.202x_1^2x_4 - 0.281x_1^2x_3 - 0.342x_1x_4^2 - 0.245x_3x_4^2 + 0.281x_3^2x_4 \\ & - 0.184x_1x_2^2 + 0.281x_1x_3x_4, \end{aligned}$$

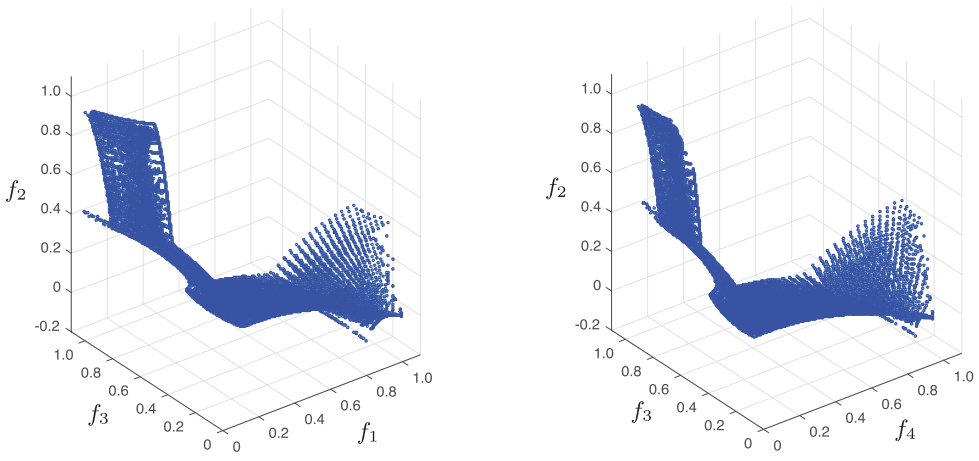
$$\begin{aligned} f_3 = {} & 0.153 - 0.322x_1 + 0.396x_4 + 0.424x_3 + 0.0226x_2 + 0.175x_1^2 + 0.0185x_1x_4 \\ & - 0.0701x_4^2 - 0.251x_1x_3 + 0.179x_3x_4 + 0.015x_3^2 + 0.0134x_1x_2 + 0.0296x_2x_4 \\ & + 0.0752x_2x_3 + 0.0192x_2^2, \end{aligned}$$

$$\begin{aligned} f_4 = {} & 0.758 + 0.358x_1 - 0.807x_4 + 0.0925x_3 - 0.0468x_2 - 0.172x_1^2 + 0.0106x_1x_4 \\ & + 0.0697x_4^2 - 0.146x_1x_3 - 0.0416x_3x_4 + 0.102x_3^2 - 0.0694x_1x_2 \\ & - 0.00503x_2x_4 + 0.0151x_2x_3 + 0.0173x_2^2\,, \end{aligned}$$

subject to

$$0 \le x_1, x_2, x_3, x_4 \le 1\,.$$

Note that the variables of the problem have been normalized.



(a) Projection of the Pareto Front to the $f_1 f_3 f_2$-space.

(b) Projection of the Pareto Front to the $f_4 f_3 f_2$-space.

FIG. 7. *Projected Pareto fronts for the rocket injector design problem.*

We have used Algorithm 9, given in Appendix A, which pairs up the feasible-value-constraint scalarization and the SBG grid, to obtain an approximation of the Pareto front in the four-dimensional space. In view of not being able to visualize the Pareto front in four dimensions, and for comparisons with [18], we choose to display the projections $(f_1, f_2, f_3, f_4) \mapsto (f_1, f_2, f_3, 0)$ and $(f_1, f_2, f_3, f_4) \mapsto (0, f_2, f_3, f_4)$ in the $f_1 f_3 f_2$- and $f_4 f_3 f_2$-spaces in Figures 7(a) and (b), respectively.

In [18], the minimization is carried out only with three objectives, namely, with f_1, f_2, and f_3, where a (heuristic) genetic algorithm is used. This reduced problem of minimization (with f_4 dropped out) is justified by a claim in [18] that f_1 and f_4 are correlated after a correlation analysis. The projected approximation we get in Figure 7(a) with about 70,000 Pareto points looks (from an angle) somewhat similar to that in Figure 7 in [18] with about 90,000 points, although the projection of a Pareto front in four dimensions to a three-dimensional space is in general not a surface. We do not have the means to compare these two approximations in their entirety. However, Figure 7(b) shows that the projection of the Pareto front to the $f_4 f_3 f_2$-space clearly differs from that in Figure 7(a): the "middle" part of the front, i.e., the points facing the $f_3 f_2$-plane, changes more rapidly with f_2 and with f_3. This implies that incorporation of f_4 in this multiobjective optimization problem is essential, or necessary, as opposed to its exclusion in [18].

We have also implemented an algorithm using the feasible-value-constraint scalarization and the CHIM grid, which we do not provide here because of its length. This algorithm did not work well, presumably because of the complicated boundary of the Pareto front in the current example.

The grid size for Algorithm 9 was taken to be $(100\times6)+(1176\times4)+18424 = 23728$. The algorithm generated $79,140$ points of which $69,705$ were Pareto points. We have chosen to generate roughly as many points as those generated for Figure 7 in [18]. The elapsed CPU time was nearly 2 hours. However, it is worthwhile to note that we also ran Algorithm 9 with a much coarser grid, resulting in a Pareto front looking not too different from those in Figure 7, in less than 10 minutes of CPU time.

7. Conclusion. We have proposed a new scalarization approach and new algorithms for generating the Pareto fronts of nonlinear multiobjective optimization problems. We established the theoretical properties of the proposed scalarization technique. We formed three-objective optimization algorithms, using combinations of one of four scalarization techniques (one new and the other three existing) with one of the CHIM and SBG grids, giving rise to eight algorithms, numbered from 1 to 8. Of these, Algorithms 4 and 7, respectively, are the popular NBI and SBG methods. The other six are new. We compared the performance of all eight algorithms via numerical experiments involving a problem with a Pareto front which has a hole in the interior, and concluded that Algorithms 1 and 5, which use the new scalarization, appear to be the most efficient in approximating the Pareto front.

We also introduced Algorithm 9, as an extension of Algorithm 5 using the SBG grid, for four-objective optimization problems. We illustrated the working of the algorithm on a rocket injector design problem involving four objective functions to minimize, and constructed a faithful approximation of the Pareto front. Our numerical experiments with this example also illustrated the importance of not dropping out an objective function before carrying out optimization involving all of the objective functions. Another algorithm using the NBI grid (an extension of Algorithm 1 to four objectives, which is not listed in this article) was not successful in revealing all parts of the Pareto front of this example, conceivably owing to the complicated looking boundary.

Although we have used MATLAB in all of the numerical experiments reported in this paper, one can use other modeling languages, such as AMPL [16], linked with an optimization solver, for example, Algencan [1] and Ipopt [29]. In fact, we have coded the algorithms in this paper in AMPL and have observed a similar performance behavior to that we have already reported with MATLAB.

It would be interesting to study more specific applications of the new approach, for example, to optimal control [3, 20]. Since the new approach is particularly suitable for problems with disconnected feasible sets, it will also be interesting to study applications of our new technique and algorithms to multiobjective mixed-integer programming problems.

Our new scalarization technique can be loosely viewed as a special case of the epsilon-constraint scalarization by Chankong and Haimes [5]. In the latter, the objective function values are constrained by a parameter ε. However, it is well-known that there exists no systematic way to choose ε and careless choices of ε lead to poor performance of the scalarization. So our approach furnishes, in some sense, an informed choice of ε using the structure of the problem (also see sections 2.4.2 and 4.2.2 of [28]) leading to a satisfactory behavior.

It was pointed out to us by a reviewer that problem $(\mathrm{P}^k_{\hat{x}})$ can be expressed in terms of a function φ_A discussed in section 5.2.1 of [22]. In view of this connection, the results of Theorems 4.4 and 4.6 may alternatively be obtained by using the monotonicity properties of the function φ_A and associated results in [22].

Appendix A. A four-objective algorithm. In this appendix, we provide Algorithm 9 for solving four-objective optimization problems. Algorithm 9 implements over an SBG grid the new feasible-value-constraint scalarization method as given in $(\mathrm{P}^k_{\hat{x}})$. The algorithm is an extension of Algorithm 5.

In Algorithm 9, the Pareto front of $\{f_1, f_2, f_3, f_4\}$ is approximated systematically in four main successive steps. First, in Step 2, the individual minima of all four objectives are obtained. Then, in Step 3, the discretized (approximate) boundaries of the Pareto fronts of the two-objective minimization (with all possible combinations) are

constructed. In Step 5, the boundaries of the three-objective combinations are generated. These combinations are $\{f_1, f_2, f_3\}$, $\{f_2, f_3, f_4\}$, $\{f_3, f_4, f_1\}$, and $\{f_4, f_1, f_2\}$. Finally, in Step 7, an approximation of the interior of the Pareto front of the problem is generated.

Let e_k be the kth standard basis vector of $\mathbb{R}^4$, e.g., $e_2 = [0, 1, 0, 0]$.

Algorithm 9 (SBG grid + ($\mathbf{P}^k_{\hat{x}}$)).

Step 1 (*Input*)
Set the number of partition points N.

Step 2 (*Determine the individual minima*)
Find $\bar{x}_{f_k}$ that solves Problem (P_k):= $\min_{x \in X} f_k(x)$ for $k = 1, 2, 3, 4$.
Set $F(k) := [f_1(\bar{x}_{f_k}), f_2(\bar{x}_{f_k}), f_3(\bar{x}_{f_k}), f_4(\bar{x}_{f_k})]$ and $W(k) := e_k$ for $k = 1, 2, 3, 4$. Set $s = k + 1$.

Step 3 (*Generate Pareto fronts of the objective pairs*)
Set the direction vector $v^{(i,j)}$ in the $f_i f_j$-space, $i = 1, 2, 3$, $j = 2, 3, 4$, $i < j$.
Do the same as in the rest of Step 3 of Algorithm 5.

Step 4 (*Form base points in the spaces of objective triplets)*
Given the Pareto points obtained in Step 3 for the objective pairs $\{f_i, f_j\}$, $\{f_i, f_k\}$ and $\{f_j, f_k\}$, construct base points by solving and LP problem in the $f_i f_j f_k$-space as in [25, section 7.4], $i = 1, 2, 3, 4$, $j = 2, 3, 4$, $k = 3, 4$, $i < j < k$.

Step 5 (*Generate Pareto fronts of the objective triplets*)
Do the same as in Step 5 of Algorithm 5 for $\{f_i, f_j, f_k\}$, $i = 1, 2, 3, 4$, $j = 2, 3, 4$, $k = 3, 4$, $i < j < k$.

Step 6 (*Form base points in the $f_1 f_2 f_3 f_4$-space*)
Given the Pareto points obtained in Steps 3 and 5 solve an LP problem to construct base points in the $f_1 f_2 f_3 f_4$-space.

Step 7 (*Generate interior Pareto points for $\{f_1, f_2, f_3, f_4\}$*)
Follow a procedure which is an extension of that in Step 5 of Algorithm 5 for $\{f_1, f_2, f_3, f_4\}$ to obtain Pareto points.

Step 8 (*Output*)
The array of Pareto points, F, is an approximation of the Pareto front.

Acknowledgments. The authors would like to thank Jörg Fliege for making them aware of the challenging multiobjective rocket injector design problem. They also acknowledge useful discussions with Gabriele Eichfelder on epsilon-constraint scalarization. They are indebted to the anonymous reviewers for their careful reading and suggestions which improved the manuscript.

REFERENCES

[1] R. Andreani, E. G. Birgin, J. M. Martínez, and M. L. Schuverdt, *On augmented Lagrangian methods with general lower-level constraints*, SIAM J. Optim., 18 (2007), pp. 1286–1309.

[2] C. Audet, G. Savard, and W. Zghal, *A mesh adaptive direct search algorithm for multiobjective optimization*, European J. Oper. Res., 204 (2010), pp. 545–556.

[3] H. Bonnel and C. Y. Kaya, *Optimization over the efficient set of multi-objective convex optimal control problems*, J. Optim. Theory Appl., 147 (2010), pp. 93–112.

[4] R. S. Burachik, C. Y. Kaya, and M. M. Rizvi, *A new scalarization technique to approximate Pareto fronts of problems with disconnected feasible sets*, J. Optim. Theory Appl., 162 (2014), pp. 428–446.

[5] V. Chankong and Y. Y. Haimes, *Multiobjective Decision Making: Theory and Methodology*, North-Holland, Amsterdam, 1983.

[6] G. Chen, X. Huang, and X. Yang, *Vector Optimization: Set-Valued and Variational Analysis*, Springer, Berlin, 2005.
[7] Y. Collette and P. Siarry, *Multiobjective Optimization*, Springer, Berlin, 2004.
[8] I. Das and J. E. Dennis, *Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems*, SIAM J. Optim., 8 (1998), pp. 631–657.
[9] J. Dutta and C. Y. Kaya, *A new scalarization and numerical method for constructing weak Pareto front of multi-objective optimization problems*, Optimization, 60 (2011), pp. 1091–1104.
[10] M. Ehrgott, *Multicriteria Optimization*, 2nd ed., Springer, Berlin, 2005.
[11] M. Ehrgott and S. Ruzika, *Improved ϵ-constraint method for multiobjective programming*, J. Optim. Theory Appl., 138 (2008), pp. 375–396.
[12] G. Eichfelder, *Adaptive Scalarization Methods in Multiobjective Optimization*, Springer, Berlin, 2008.
[13] G. Eichfelder, *An adaptive scalarization method in multi-objective optimization*, SIAM J. Optim., 19 (2009), pp. 1694–1718.
[14] G. Eichfelder, *Scalarizations for adaptively solving multi-objective optimization problems*, Comput. Optim. Appl., 44 (2009), pp. 249–273.
[15] A. Engau and M. M. Wiecek, *Generating ε-efficient solutions in multiobjective programming*, European J. Oper. Res., 177 (2007), pp. 1566–1579.
[16] R. Fourer, D. M. Gay, and B. W. Kernighan, *AMPL: A Modeling Language for Mathematical Programming*, 2nd ed., Brooks/Cole, Pacific Grove, CA, 2003.
[17] S. Gass and T. Saaty, *The computational algorithm for the parametric objective function*, Naval Res. Logist. Quart., 2 (1955), pp. 39–45.
[18] T. Goel, R. Vaidyanathan, R. T. Haftka, W. Shyy, N. V. Queipo, and K. Tucker, *Response surface approximation of Pareto optimal front in multi-objective optimization*, Comput. Methods Appl. Mech. Engrg., 196 (2007), pp. 879–893.
[19] J. Jahn, *Vector Optimization: Theory, Applications, and Extensions*, Springer-Verlag, Berlin, 2011
[20] C. Y. Kaya and H. Maurer, *A numerical method for nonconvex multi-objective optimal control problems*, Comput. Optim. Appl., 57 (2014), pp. 685–702.
[21] K. Khaledian and M. Soleimani-damaneh, *A new approach to approximate the bounded Pareto front*, Mathods Methods. Oper. Res., 82 (2015), pp. 211–228.
[22] A. A. Khan, C. Tammer, and C. Zălinescu, *Set-Valued Optimization: An Introduction with Applications*, Springer, Berlin, 2015
[23] R. T. Marler and J. S. Arora, *Survey of multi-objective optimization methods for engineering*, Struct. Multidiscip. Optim., 26 (2004), pp. 369–395.
[24] K. M. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer Academic, Boston, 1999
[25] D. Mueller-Gritschneder, H. Graeb, and U. Schlichtmann, *A successive approach to compute the bounded Pareto front of practical multiobjective optimization problems*, SIAM J. Optim., 20 (2009), pp. 915–934.
[26] A. Pascoletti and P. Serafini, *Scalarizing vector optimization problems*, J. Optim. Theory Appl., 42 (1984), pp. 499–524.
[27] D. Quagliarella and A. Vicini, *Sub-population policies for a parallel multiobjective genetic algorithm with applications to wing design*, in Proceeding of IEEE SMC, 1998, pp. 3142–3147.
[28] M. M. Rizvi, *New Optimality Conditions for Non-Linear Multiobjective Optimization Problems and New Scalarization Techniques for Constructing Pathological Pareto fronts*, Ph.D. thesis, University of South Australia, 2013.
[29] A. Wächter and L. T. Biegler, *On the implementation of a primal-dual interior point filter line search algorithm for large-scale nonlinear programming*, Math. Progam., 106 (2006), pp. 25–57.
[30] R. Vaidyanathan, P. K. Tucker, N. Papila, and W. Shyy, *CFD-based design optimization for a single element rocket injector*, J. Propul. Power, 20 (2004), pp. 705–717.
[31] P. L. Yu, *Multicriteria Decision Making: Concepts, Techniques, and Extensions*, Plenum Press, New York, 1985.
[32] L. Zadeh, *Optimality and non-scalar-valued performance criteria*, IEEE Trans. Automat. Control, 8 (1963), pp. 59–60.