Hindawi
Advances in Operations Research
Volume 2021, Article ID 1857030, 11 pages
https://doi.org/10.1155/2021/1857030

Research Article

A Comparison of Benson's Outer Approximation Algorithm with an Extended Version of Multiobjective Simplex Algorithm

Paschal B. Nyiam and Abdellah Salhi

Department of Mathematical Sciences, University of Essex, Colchester, UK

Correspondence should be addressed to Paschal B. Nyiam; nyiampaschal@unical.edu.ng

Received 1 August 2020; Revised 2 February 2021; Accepted 22 April 2021; Published 5 July 2021

Academic Editor: Panagiotis P. Repoussis

The multiple objective simplex algorithm and its variants work in the decision variable space to find the set of all efficient extreme points of multiple objective linear programming (MOLP). Other approaches to the problem find either the entire set of all efficient solutions or a subset of them and also return the corresponding objective values (nondominated points). This paper presents an extension of the multiobjective simplex algorithm (MSA) to generate the set of all nondominated points and no redundant ones. This extended version is compared to Benson's outer approximation (BOA) algorithm that also computes the set of all nondominated points of the problem. Numerical results on nontrivial MOLP problems show that the total number of nondominated points returned by the extended MSA is the same as that returned by BOA for most of the problems considered.

1. Introduction

Multiobjective linear programming seeks to optimize two or more linear objective functions subject to a set of linear constraints with a view of obtaining either all the efficient solutions or nondominated points or a subset of them, or a most preferred solution depending on the approach adopted. MOLP has been studied over the years because of its relevance in practice.

Indeed, many decision-making problems that arise in the real world involve more than one objective function. Consequently, it has been widely applied in many fields and has become a useful tool in decision-making.

Formally, it can be written as

$$\begin{aligned} &c_1^T x = f_1 \\ \min\ &\vdots \\ &c_q^T x = f_q \end{aligned} \tag{1}$$

$$\text{subject to} x \in X = \{x \in \mathbb{R}^n : Ax = b, b \in \mathbb{R}^m, x \geq 0\}.$$

We noted in [1] that "in practice, MOLP is typically solved by the Decision-Maker (DM) in conjunction with the analyst who looks for a most preferred solution in the feasible region X. This is because optimizing all the objective functions at the same time is not possible due to their conflicting nature. Consequently, the concept of optimality is replaced with that of efficiency. Therefore, the purpose of MOLP is to obtain either all the efficient extreme points or nondominated extreme points or a subset of them, or a most preferred point depending on the purpose for which it is needed."

Many algorithms have been suggested for the problem. Most of them are based on the simplex method for linear programming. Prominent among them is the multiobjective simplex algorithm (MSA) and its variants. According to Eiselt and Sandblom [2], Evans and Steuer [3], Philip [4], and Zeleny [5] all derived generalized versions of the simplex method known as MSA. This algorithm works in the decision variable space to find the entire set of all efficient solutions. However, it was noted in [6] that finding the nondominated points instead of the efficient set is more important for the DM.

The aim of this paper is to extend the MSA of Evans and Steuer [3] whose explicit form can be found in [7] to generate the whole set of nondominated points. We shall

then compare this extended version with the original one and with the primal variant of Benson's outer approximation (BOA) algorithm [8] which is an objective space based method that also computes the set of all nondominated points of the problem.

This paper is organized as follows: Section 2 is the motivation. Section 3 introduces MOLP and basic notation. Section 4 is a brief review of the relevant literature. We present MSA and its extended version in Sections 5 and 6, respectively. Section 7 discusses two scalarization techniques. BOA is presented in Section 8. Section 9 presents experimental results obtained with the different algorithms. Finally, a conclusion is presented in Section 10.

2. Motivation

From the outset, MSA and BOA are not comparable since one is decision space based while the other is objective space based; one computes efficient solutions and the other nondominated points. However, if one can generate nondominated points from MSA, then this can be performed.

It is well known [6–16] that, in practice, decision-makers prefer to base their choice of a most preferred point on the objective values (nondominated points) rather than on the efficient solutions. This means that algorithms such as MSA which return only the set of efficient solutions are less favourable to the DM, say, compared to BOA which works in the objective space and returns the nondominated set. It is therefore desirable to generate the nondominated set from the efficient set returned by MSA. In other words, we extend it. This extended variant of MSA becomes as desirable as those computing the nondominated set for the DM. However, this can only be decided after a comparison with such an algorithm. Here, we suggest comparing with BOA on a number of nontrivial MOLP instances. Clearly, extended MSA becomes very attractive given that it gives all of the efficient solutions and nondominated points.

3. Notation and Definitions

An alternative and compact formulation of (1) is as follows:

$$\begin{aligned} &\min \quad Cx \\ &\text{subject to} \quad \begin{aligned} Ax &= b \\ x &\geq 0, \end{aligned} \end{aligned} \tag{2}$$

where C is a $q \times n$ criterion matrix consisting of the rows c_k, $k = 1, 2, \ldots, q$, A is an $m \times n$ constraint matrix, and $b \in \mathbb{R}^m$ is the right-hand side vector. The feasible set in the decision space is $X = \{x \in \mathbb{R}^n : Ax = b, x \geq 0\}$ and, in the objective space, it is $Y = \{Cx : x \in X\}$. The set Y is also referred to as the image of X [1].

A nondominated point in the objective space is the image of an efficient solution in the decision space, and the set of all nondominated points forms the nondominated set [6].

An efficient solution to the problem is a solution that cannot improve any of the objective functions without reducing at least one of the other objectives. A weakly efficient solution is the one that cannot improve all the objective functions simultaneously, [17]. Let $\widehat{x} \in X$ be a feasible solution of (2) and let $\widehat{y} = C\widehat{x}$:

(i) $\widehat{x}$ is called efficient if there is no $x \in X$ such that $Cx \leq C\widehat{x}$ and $Cx \neq C\widehat{x}$; correspondingly, $\widehat{y} = C\widehat{x}$ is called nondominated

(ii) $\widehat{x}$ is called weakly efficient if there is no $x \in X$ such that $Cx < C\widehat{x}$; and $\widehat{y} = C\widehat{x}$ is called weakly nondominated [7]

The set of all efficient solutions and the set of all weakly efficient solutions of (2) are denoted by X_E and X_{WE}, respectively [10]. $Y_N = \{Cx : x \in X_E\}$ and $Y_{\mathrm{WN}} = \{Cx : x \in X_{\mathrm{WE}}\}$ are the nondominated and weakly nondominated sets in the objective space of (2), respectively.

The nondominated faces in the objective space of the problem constitute the nondominated frontier and the efficient faces in the decision space of the problem constitute the efficient frontier [1].

4. Literature Review

As stated earlier, Eiselt and Sandblom [2] note that Evans and Steuer [3], Philip [4], and Zeleny [5] all derived generalized versions of the simplex method known as MSA for generating the entire efficient decision set X_E of the problem. That of Philip [4] first determines if an extreme point is efficient and subsequently checks if it is the only one that exists. If not, the algorithm finds them all. This MSA approach, however, may fail at a degenerate vertex. In [18], Philip modified it to overcome this difficulty.

The MSA of Evans and Steuer [3] also generates the set of all efficient solutions and unbounded efficient edges of the problem; see also Algorithm 7.1, page 178 of [7]. The algorithm first confirms that the problem is feasible and has efficient extreme points. Thereafter, it computes all of them by moving from one efficient extreme point to an adjacent efficient extreme point, until all of the efficient extreme points have been computed. An LP test problem is solved to determine the pivots that lead to efficient extreme points. The algorithm is implemented as software called ADBASE [19].

The MSA variant of Zeleny [5] also uses an LP test problem to determine the efficiency of extreme points. But here, vertices are tested for efficiency after they have been obtained unlike in [3] where the test problem determines pivots leading to efficient vertices.

Yu and Zeleny [20, 21] used the approach in [5] to generate the set of all efficient solutions and presented a formal procedure for testing the efficiency of extreme points.

The efficient solutions are derived from the efficient faces, in a top-to-bottom search strategy. Numerical illustrations with three objectives were used to demonstrate the effectiveness of the method. In a similar paper, Yu and Zeleny [22] applied their approach expanded in [21] to parametric linear programming. Two basic forms of the problem and two computational approaches for generating the entire efficient set were presented: the direct decomposition approach that decomposes the parametric space

into subspaces associated with extreme points and the indirect algebraic approach. From a numerical experience point of view, the indirect algebraic approach was superior to the direct decomposition method.

In [23], Isermann proposed a variant of the MSA in [3] that solves fewer LPs when determining the entering variables. The algorithm first establishes whether an efficient solution for the problem exists and solves a test problem to determine pivots leading to efficient vertices. It was implemented as a software called EFFACET in Isermann and Naujoks [24].

The MSA of Gal [25] generates the set of all higher-dimensional faces and all efficient vertices of the problem. This approach is meant to address the problem of determining efficient faces and higher-dimensional faces that were not resolved in [3, 4]. Here, efficient solutions are computed using a test problem. The algorithm also determines higher-dimensional efficient faces for degenerate problems which were only discussed in [5, 23] but were not solved. The efficient faces are computed in a bottom-to-top search strategy unlike what was suggested in [20, 21].

Steuer [26] used the MSA of Evans and Steuer [3] to solve parametric and nonparametric problems. Different approaches for determining an initial efficient extreme point as well as different LP test problems were also considered. Efficient solutions were computed through the direct decomposition of the weight space into finite subsets that provided optimal weights corresponding to efficient solutions.

In [7], Ehrgott also used the MSA of Evans and Steuer [3] to solve MOLP problem instances with two and three objective functions. Ecker and Kouada [27] also proposed a variation on the MSA of Evans and Steuer [3]. They noted that algorithms usually started from an initial efficient extreme point and moved to an adjacent one following the solution of an LP problem. The proposed method does not require the solution of any LP problem to test for the efficiency of extreme points and the feasible region need not be bounded. The algorithm enumerates all efficient extreme points and appears to have a computational advantage over other methods.

In a different paper, Ecker et al. [28] presented yet another variant of MSA. The algorithm first determines the maximal efficient faces incident to a given efficient vertex (i.e., containing the efficient vertex) and ensures that previously generated efficient faces are not regenerated. This is done following a bottom-to-top search strategy as in [25], which dramatically improves computation time. The proposed approach was illustrated with a degenerate example given in [21], to demonstrate its applicability. It was computationally more efficient than the method in [21].

The MSA of Armand and Malivert [29] determines the set of efficient extreme points even for degenerate MOLPs. The approach follows a bottom-to-top search strategy and utilizes a lexicographic selection rule to choose the leaving variables which proves effective when solving degenerate problems. It was tested successfully on a number of degenerate problems. A numerical example with five objectives and eight constraints which was solved in [21] was also used to demonstrate its effectiveness. The proposed MSA was superior to that in [21].

Rudloff et al. [30] suggested a MSA which works in the decision variable space but does not generate all the efficient extreme points unlike the algorithm in [3]. Instead, it finds a subset of efficient extreme points based on the idea of Löhne [31]. That is, a subset of efficient extreme points and directions that allows computing the whole efficient frontier. The algorithm was compared with BOA [8] which also provides a solution based on the idea in [31] and with Evans and Steuer's MSA [3]. Numerical experiments show that the proposed method is superior to Benson's algorithm for nondegenerate problems. However, that of Benson's outperforms it for highly degenerate ones.

In [1], we presented the results of a more detailed computational investigation of the MSA in [30] and BOA [8] using existing small, medium, and realistic MOLP instances to evaluate the robustness and quality of the most preferred nondominated point (MPNP) returned by these two algorithms which was not discussed or considered in [30]. Also presented in [1] was a formal procedure for the computation of a MPNP of the problem. Numerical results on the robustness, efficiency, and quality of a MPNP show that BOA outperforms PSA in terms of the quality of a MPNP it returns and robustness, as well as confirming what was reported in [30] that BOA is computationally more efficient than PSA on highly degenerate problems, while PSA is superior to BOA computationally on nondegenerate MOLP problems.

Of all these variants, it was noted in [32] that, that of Evans and Steuer [3] is the most popular and successful for computing all efficient extreme points of the problem.

Apart from MSA and its variants that work in the decision variable space to find the entire set of all efficient solutions, there are algorithms that work in the objective space to find the set of nondominated points of the problem. Prominent among them is Benson's outer approximation algorithm [8]. As was noted in [1], the author who presented an account of decision space based methods proposed an algorithm for computing the nondominated set in the objective space of the problem. According to him, his method is the first of its kind. It was motivated by the observation that many efficient extreme points map onto the same nondominated point in the objective space; Decision-Makers prefer to base their choice of a most preferred point on the nondominated set rather than the efficient set; and moreover, the dimension of the objective space is much smaller than that of the decision space, [12]. Therefore, finding the nondominated set instead of the efficient set is also more important for the DM [6]. The algorithm was compared with the MSA of Evans and Steuer [3]. Results show that the average number of nondominated solutions returned by BOA is less than the average number of efficient solutions returned by MSA in all the problems considered. In a similar paper, a further analysis of the objective space based methods was presented in Benson [9]. Here, it was shown that the algorithm in [8] also computes the weakly nondominated points, thereby enhancing the usefulness of the method as a decision aid [1].

Before Benson's proposal, Dauer and Liu [12] suggested a procedure for obtaining the nondominated points and edges in the objective space. It was noted that not all efficient extreme points necessarily map onto the nondominated points and the procedure analyzes a simpler structure.

In [10], Benson suggested a hybrid method for solving an MOLP in the objective space. The method involves partitioning the objective space into simplices that lie in each face so as to compute the nondominated set. This idea was earlier presented in [33]. The method is quite similar to his algorithm in [8]. The difference between them is in the way in which the nondominated extreme points are generated. While a vertex enumeration approach is utilized in [8], a simplicial partitioning method is used in the latter [1].

We also noted in [1] that a modification of the algorithm of Benson [8] was presented in [15]. While in [8], a bisection approach that requires the solution of more than one LP is required in one step; here, solving only one LP gives the desired effect and in the process improves computation time. Shao and Ehrgott [16] suggested an approximate dual variant of the algorithm of Benson [8] for generating approximate nondominated points of the problem. The proposed method was tested on the beam intensity optimization problem of radiotherapy treatment planning for which approximate nondominated points were generated. Numerical results show that the method is faster than solving the primal problem directly.

Shao and Ehrgott [15] modified the algorithm of Benson [8] and presented it in explicit form in [31]. This modified version solves two LPs in each iteration during the process of computing the nondominated set. In [34], Löhne introduced the MATLAB implementation of this modified version called BENSOLVE-1.2, for generating all the nondominated extreme points and directions of the problem.

Csirmaz [35] also presented an improved version of the algorithm of Benson [8] where only one LP and a vertex enumeration problem is solved in each iteration while in [8], two steps, as well as two LPs, are required to be solved in order to determine a unique boundary point and supporting hyperplane of the image; here, the two steps are merged into one and solving one LP does both tasks and dramatically improves computation time. The algorithm was used to compute all the nondominated points of the polytope defined by a set of Shannon inequalities on four random variables so as to map their entropy region [1]. Numerical results show the applicability of the algorithm to medium and large instances.

Similarly, Hamel et al. [36] introduced new variants of the algorithm in [8] that solves only one LP problem in each iteration. Numerical experiments reveal a reduction in computation time.

We noted in [1] that Löhne et al. [14] presented an extension of the primal and dual variants of the algorithm of Benson [8] to solve convex vector optimization problems approximately in the objective space.

5. The Multiobjective Simplex Algorithm

The MSA of Evans and Steuer [3] described in this section can be found in [7]. We consider this algorithm because of its popularity (see [32]), and because most of the MSA algorithms discussed earlier are either based on or are variants of it. It works in the decision variable space to find the entire set of all efficient solutions.

In an MOLP problem, only one of the following situations can occur: the problem can be infeasible, meaning that the feasible set X is empty ($X = \emptyset$); the problem may be feasible, that is ($X \neq \emptyset$) but may not have efficient solutions, that is, ($X_E = \emptyset$); or it is feasible and has efficient solutions, that is $X_E \neq \emptyset$. This algorithm handles these situations in three phases: in the first phase, it finds an initial basic feasible solution or stop with the conclusion that $X = \emptyset$; in the second phase, it finds an initial efficient basis or stop with the conclusion that $X_E = \emptyset$; and, in the final phase, it pivots among efficient bases to determine all efficient extreme points of the problem [7].

The algorithm starts by solving two auxiliary LPs to determine whether the problem is feasible and to verify that it has efficient solutions. If the feasible region X and the efficient set X_E are not empty, a weighted sum LP is solved to determine an initial efficient basis B. Its implementation stores a list of efficient bases L_1 to be processed, a list L_2 of efficient bases for output, and a list of efficient nonbasic variables N_E. An LP test problem is solved to determine pivots that lead to efficient bases. The algorithm pivots from an initial efficient basis to an adjacent efficient basis until the list L_1 of efficient bases to be processed is empty. The algorithm terminates and returns a list L_2 of efficient bases from where all efficient extreme points are computed.

Before we present the pseudocode of MSA, we first explain the notation used.

A, b, C: the problem data

L_1: list of efficient bases to be processed

L_2: list of efficient bases for output

$e^T = (1, \ldots, 1) \in \mathbb{R}^q$

I: the identity matrix of proper order

X: the feasible set

X_E: the set of efficient solutions

B: the efficient basis

N_E: list of efficient nonbasic variables

N: the set of nonbasic variables

B': the new basis

A and b: the updated constraint matrix and RHS vector, respectively

R: the nonbasic part of the reduced cost matrix

r^j: a column of R corresponding to the nonbasic variable being tested for efficiency

(0) **Input:** A, b, C: Problem data
(1) **Initialize:** Set $L_1 \longleftarrow \varnothing, L_2 \longleftarrow \varnothing$;
Phase I : solve the LP $\min\{e^T z\colon\ Ax + Iz = b, x, z \geq 0\}$. If the optimal value of this LP is nonzero, STOP, $X = \varnothing$;
Otherwise x^0 is a basic feasible solution of MOLP
Phase II: solve the LP $\min\{u^T b + w^T C x^0\colon\ u^T A + w^T C \geq 0, w \geq e\}$. If it is infeasible,
STOP, $X_E = \varnothing$; Otherwise $(\widehat{u}, \widehat{w})$ is an optimal solution; Find an optimal basis B of the LP $\min\{\widehat{w}^T C x\colon\ Ax = b, x \geq 0\}$;
Set $L_1 \longleftarrow \{B\}, L_2 \longleftarrow \varnothing$.;
(2) while $L_1 \neq \varnothing$
(3) Choose $B \in L_1, L_1 \longleftarrow L_1 \backslash \{B\}, L_2 \longleftarrow L_2 \cup \{B\}$;
(4) Compute $\bar{A}, \bar{b}$, and R according to B;
(5) $N_E \longleftarrow N$;
(6) **for** all $j \in N$
(7) Solve the LP $\max\{e^T v\colon\ Ry - r^j \sigma + Iv = 0;\ y, \sigma, v \geq 0\}$.
(8) If this LP is unbounded $N_E \longleftarrow N_E \backslash \{j\}$;
(9) **for** all $j \in N_E$
(10) **for** all $i \in B$
(11) **if** $B' \longleftarrow (B \backslash \{i\}) \cup \{j\}$ is feasible, $B' \notin L_1 \cup L_2$ then;
(12) $L_1 \longleftarrow L_1 \cup B'$;
(13) **endif**
(14) **endfor**
(15) **endfor**
(16) **endfor**
(17) **endwhile**
(18) **Output**: L_2: List of efficient bases.

Algorithm 1: Multiobjective Simplex Algorithm [7].

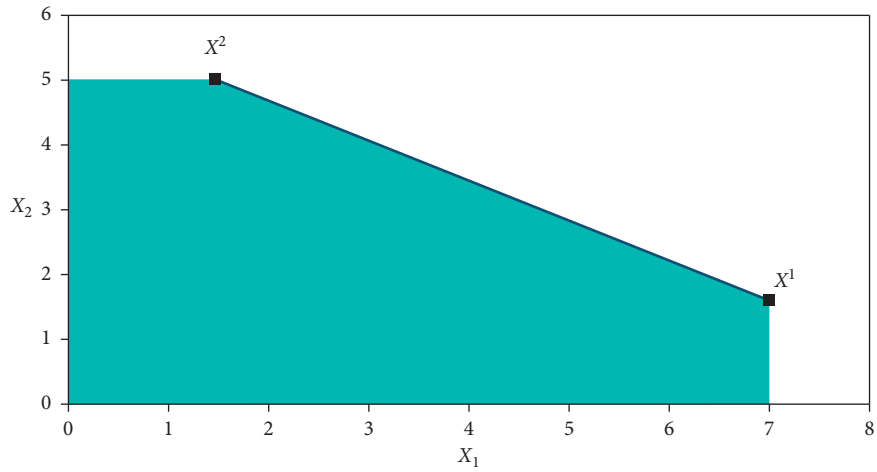

Figure 1: Edge connecting the two efficient points in the decision variable space.

5.1. Illustration of MSA. Consider the following MOLP adapted from [37]:

$$\begin{aligned} \min \quad & f_1 = -x_1 \\ \min \quad & f_2 = -x_2 \\ \text{subject to } & 6x_1 + 10x_2 \leq 60, \\ & x_1 \leq 7, \\ & x_2 \leq 5, \\ & x_1, x_2 \geq 0. \end{aligned} \tag{3}$$

The efficient solutions found using a MATLAB implementation of Algorithm 1 are $x^1 = (7.0, 1.8)^T$, $x^2 = (1.6, 5.0)^T$, $x^3 = (1.6, 5.0)^T$, and $x^4 = (7.0, 1.8)^T$, where $x^1 = (x_1^1, x_2^1)^T, \ldots, x^4 = (x_1^4, x_2^4)^T \in X_E$. The algorithm is prone to generating more efficient solutions due to the way it operates and due to the fact that MSA may find the same efficient solutions in more than one iteration, as in this case; $x^1 = x^4$ and $x^2 = x^3$ are repetitive of what has already been found. Solutions x^3 and x^4 are redundant and would be of little or no use to the DM. The feasible region in the decision variable space is shown in Figure 1.

(0) **Input**: A, b, C (data of MOLP problem)
(1) **Initialize**: set $L_1 \longleftarrow \varnothing, L_2 \longleftarrow \varnothing, X_E \longleftarrow \varnothing, Y_N \longleftarrow \varnothing$;
Phase I: solve the LP min $\{e^T z\colon Ax + Iz = b, x, z \geq 0\}$. If the optimal value of LP is not zero; STOP; $X = \varnothing$. Otherwise, x^0 is a basic feasible solution of MOLP.
Phase II : solve the LP min $\{u^T b + w^T C x^0\colon u^T A + w^T C \geq 0, w \geq e\}$. If it is infeasible STOP, $X_E = \varnothing$. Otherwise, $(\widehat{u}, \widehat{w})$ is an optimal solution. Find optimal basis B and basic feasible solution $\overline{x}$ of LP min $\left\{\widehat{w}^T Cx\colon Ax = b, x \geq 0\right\}$;
Set $L_1 \longleftarrow \{B\}, L_2 \longleftarrow \varnothing, X_E \longleftarrow \{\overline{x}\}, Y_N \longleftarrow \{C^T\overline{x}\}$.
(2) **while** $L_1 \neq \varnothing$ **do**
(3) Choose $B \in L_1, L_1 \longleftarrow L_1 \backslash \{B\}, L_2 \longleftarrow L_2 \cup \{B\}$;
(4) Compute $\bar{A}, \bar{b}$, and R according to B;
(5) $N_E \longleftarrow N$;
(6) **for** all $j \in N$
(7) Solve the LP $\max\{e^T v\colon Rz - r^j\sigma + Iv = 0; z, \sigma, v \geq 0\}$.
(8) If this LP is unbounded $N_E \longleftarrow N_E \backslash \{j\}$;
(9) **for** all $j \in N_E$
(10) **for** all $i \in B$
(11) **if** $B' \longleftarrow (B \backslash \{i\}) \cup \{j\}$ is feasible, $B' \notin L_1 \cup L_2$, let $\overline{x}'$ be its basic solution; then
(12) $L_1 \longleftarrow L_1 \cup B'$;
(13) $X_E \longleftarrow X_E \cup \{\overline{x}'\}$;
(14) $Y_N \longleftarrow Y_N \cup \{C^T\overline{x}'\}$;
(15) **endif**
(16) **endfor**
(17) **endfor**
(18) **endfor**
(19) **endwhile**
(20) **Output:** X_E: the efficient set
Y_N: the non dominated set.

Algorithm 2: Extended Multiobjective Simplex Algorithm.

6. The Extended Multiobjective Simplex Algorithm

As part of the initialization step (line 1 of Algorithm 2), we have included the set of efficient extreme points X_E and that of nondominated points Y_N. In the second phase, as the algorithm finds an initial efficient basis B by solving a weighted sum LP, the algorithm also finds a corresponding efficient basic feasible solution and appends it to the set of efficient solutions ($X_E \leftarrow \{\overline{x}\}$). The first nondominated point is also computed from $C^T\overline{x}$ and appended to the nondominated set ($Y_N \leftarrow \{C^T\overline{x}\}$).

As the algorithm iterates, a new efficient basis B' is obtained after each pivot and the corresponding efficient basic feasible solution $\overline{x}'$ (line 11 of Algorithm 2) is found and added to the set of efficient solutions X_E (line 13). Likewise, the corresponding nondominated points are also found at each iteration and added to the nondominated set Y_N (line 14). This continues until the set of efficient bases L_1 to be processed is empty. The algorithm returns the set of all efficient extreme points and the corresponding nondominated points (line 20).

Before we present Algorithm 2 as the extended MSA in pseudocode form, we first state here that the structure of the algorithm and the used notation remain the same as those in Algorithm 1. The additional components are $\overline{x}$, $\overline{x}'$, X_E, and Y_N which stand for the efficient basic feasible solution, the new efficient basic feasible solution, the set of efficient solutions, and the corresponding set of nondominated points for output.

6.1. Illustration of the Extended MSA. We modified and extended the MATLAB implementation of Algorithm 1 and used it to solve problem 3 of Section 5.1.

The efficient solutions found are $x^1 = (7.0, 1.8)^T$, $x^2 = (1.6, 5.0)^T$, and the corresponding nondominated points are $f^1 = (-7.0, -1.8)^T$ and $f^2 = (-1.6, -5.0)^T$, respectively, where $x^1 = (x_1^1, x_2^1)^T$, $x^2 = (x_1^2, x_2^2)^T \in X_E$, and $f^1 = (f_1^1, f_2^1)^T$, $f^2 = (f_1^2, f_2^2)^T \in Y_N$. Note here that, the efficient extreme points x^1 and x^2 and the corresponding nondominated points f^1 and f^2 returned are devoid of redundant points. The algorithm is designed to avoid returning redundant nondominated points unlike the original version. The feasible region in the decision space is the same as in Figure 1.

6.2. Limitations of Extended MSA. As it is already known that the efficiency of the simplex-type algorithm is better when the problem is nondegenerate [30]. Like other simplex-type algorithms, the extended MSA or EMSA is not an exception. The extended MSA may exhibit one or more redundancies whenever the problem is degenerate. The first is that EMSA may find the same efficient solutions in more than one iteration or find different efficient solutions that leads to the same nondominated point. This can be seen clearly in Table 1, as the number of efficient solutions returned by EMSA is the same as that returned by the original MSA, though this is corrected in the computation of the corresponding nondominated points as the

Table 1: Comparative results for individual problem.

Prob.	Algorithm Origin	n	m	q	MSA NES	EMSA NES	EMSA NNP	BOA NNP
1	Ehrgott, 2006	3	3	3	5	5	3	3
2	Zeleny, 1982	2	2	2	5	5	3	3
3	”	2	4	2	16	16	2	2
4	”	2	4	3	36	36	3	3
5	”	2	6	2	64	64	3	3
6	”	3	3	3	15	15	5	5
7	”	5	3	3	6	6	4	4
8	”	5	2	2	1	1	1	1
9	”	6	4	2	36	36	1	1
10	”	7	4	3	36	36	5	5
11	iMOLPe	2	3	2	8	8	4	3
12	”	3	3	4	12	12	3	3
13	”	3	5	3	88	88	10	10
14	”	3	3	3	19	19	7	7
15	”	4	3	3	18	18	8	8
16	”	4	2	3	8	8	6	6
17	”	4	4	3	44	44	11	11
18	”	3	3	3	37	37	5	5
19	”	15	10	2	98	98	11	11
20	”	15	10	3	254	254	28	37
21	Steuer, 1986	10	15	3	50	50	15	14
22	”	5	5	2	28	28	5	5
23	”	4	4	3	10	10	3	3
24	”	5	5	4	154	154	14	14
25	”	10	8	4	2096	2096	51	63
26	”	5	4	3	26	26	9	9
27	”	6	8	4	560	560	13	13
28	”	7	6	4	48	48	12	36
29	”	7	6	4	152	152	9	9
30	”	8	8	6	1080	1080	56	56
31	”	8	8	3	208	208	5	5
32	”	8	8	3	64	64	1	1
33	”	5	5	4	74	74	12	12
34	”	6	6	3	304	304	17	17
35	”	5	5	4	202	202	9	9
36	”	10	10	4	3,072	3,072	6	6
37	”	8	8	3	608	608	13	13
38	”	6	7	4	440	440	25	21
39	”	12	16	4	*	—	—	601
40	”	10	14	5	*	—	—	132
41	Steuer, 1986	7	6	3	40	40	3	3
42	”	7	7	3	56	56	7	7
43	”	6	6	4	128	128	5	5
44	”	6	6	4	168	168	10	10
45	”	10	14	5	*	—	—	471
46	”	10	14	5	*	—	—	128
47	”	7	7	3	60	60	6	6
48	Bensolve-1.2	100	101	2	*	—	—	32
49	Bensolve-2.0	5	31	5	*	—	—	22
50	”	36	36	2	82	82	31	8
51	”	64	64	2	292	292	57	14
52	”	100	100	2	1102	1102	99	20
53	”	343	343	3	x	—	—	1,368
54	MOPLIB	30	21	12	*	—	—	1
55	”	53	226	3	561	561	552	552
56	”	53	221	3	*	—	—	2552

(∗) Aborted after 3 days of running time. (x) Out of memory.

nondominated points are sorted after they are generated, leading to the same number of nondominated points returned by EMSA with that returned by BOA. In other words, EMSA may find redundant efficient solutions but the nondominated points found are devoid of redundant ones. Secondly, EMSA may also be sensitive to the number of nondominated points in a particular problem as can also be seen in the result table, as larger instances with more than four objective functions tend to take more than 3 days to return the required nondominated points. We note here that some of these problems most especially those from Bensolve and MOPLIB [38] are numerically ill-posed and highly challenging MOLP instances with difficult structures.

7. Scalarization Techniques

We now present two basic scalarization approaches that play an important part in the implementation of BOA as was discussed in [1]. These approaches are weighted sum scalarization and scalarization by a reference variable. As contained in [31], scalarization is one of the most important methods used in MOLP.

In the weighted sum method, a new objective function based on the q-linear objectives is obtained by assigning nonnegative weights $w_i \in \mathbb{R}^q$ to each of the objectives. The weighted sum of the objectives is $\sum_{i=1}^{q} w_i c_i x = w^T C x$. For each vector $w \in \mathbb{R}^q$, $w \geq 0$, we obtain a scalar linear program

$$\begin{aligned} &\min && w^T C x \\ &\text{subject to} && Ax \geq b \end{aligned} \tag{4}$$

The weights are usually normalized so that $e^T w = 1$, with $e^T = (1, \ldots, 1)$. The dual of (4) is

$$\begin{aligned} &\max && b^T u \\ &\text{subject to} && \begin{cases} A^T u = C^T w, \\ u \geq 0. \end{cases} \end{aligned} \tag{5}$$

In the method of scalarization by a reference variable, the q objectives are associated with a common reference variable z and the i-th objective is restrained from being larger than the reference variable and a fixed real number y_i, that is, $c_1 x \leq y_1 + z, c_2 x \leq y_2 + z, \ldots, c_q x \leq y_q + z$.

The reference variable z is the objective function that has to be minimized. By setting $e = (1, \ldots, 1)^T$, we obtain for each vector $y \in \mathbb{R}^q$ the scalar linear program:

$$\begin{aligned} &\min && z \\ &\text{subject to} && \begin{cases} Ax \geq b \\ Cx - z.e \leq y \end{cases} \end{aligned} \tag{6}$$

The dual program is

$$\begin{aligned} &\max && b^T u - y^T w \\ &\text{subject to} && \begin{cases} A^T u - C^T w = 0 \\ e^T w = 1 \\ (u, w) \geq 0 \end{cases} \end{aligned} \tag{7}$$

as in [31]. The above two scalarization techniques are fundamental for the implementation of BOA which is discussed in Section 8 [1].

8. Benson's Outer Approximation Algorithm

We hereby present a description of BOA as we did earlier in [1]. This version of BOA is due to [15]. It can be found in [31]. It works in the objective space to compute the nondominated set and directions of the problem. The algorithm is regarded as a primal-dual method as it also solves the dual problem. But here, our interest is only in the solution of the primal problem. The algorithm first constructs an initial polyhedron Y_0 (outer approximation) containing the upper image Y in the objective space and an interior point $\widehat{p}$ of the image is determined by solving equation (4). The inequality representation of Y_0 is also determined by solving equation (5). The algorithm then constructs a sequence of decreasing polytopes $Y_0 \supseteq Y_1 \supseteq \cdots \supseteq Y_k = Y$. The vertices of each polytope Y_k and their inequality representation are stored in each iteration. Then, for each vertex v of the polytope, the algorithm confirms if it is on the boundary of Y. If the vertices are on the boundary of Y, the problem is solved. The outer vertices of Y are among the vertices of Y_k. Otherwise, for any vertex v of Y_k that is not on the boundary of Y, the algorithm connects this vertex to the interior point $\widehat{p}$ and finds the intersection y of this line with the boundary of Y by solving equation (6). Then, a supporting hyperplane adjacent to y is constructed by solving equation (7). This hyperplane is added to Y_k to provide a smaller approximation. The algorithm is repeated in the same manner until the vertices of Y_k coincide with the boundary of Y. The algorithm returns the set of vertices on the boundary of Y as the nondominated points $\overline{Y}$ and directions $\overline{Y^h}$ of the problem.

The used notation in the pseudocode of BOA is presented below.

A, b, C: problem data

P^h: the homogeneous problem

D^{*h}: the homogeneous dual problem T

h: the solution of the homogeneous dual problem

$\widehat{p}$: an interior point

$\overline{T}$: a set of solutions of the dual problem

Y_k^d: the inequality representation of the current polytope

k: the iteration counter

Y_k^p: the representation by vertices

$(\widehat{y}, z)$: an optimal solution to $P_2(y)$

$\delta(0 < \delta < 1)$: a unique value that determines the intersection or boundary point y

$R(v)$: the LP that finds the unique value δ

The command solve: solves an LP

vert(): function that returns the vertices of a polytope Yk

$\overline{Y}$: the set of nondominated vertices

$(\overline{Y}^h)$: the set of extreme directions [1]

(0) **Input:** A, b, C: Problem data
a solution $(\{0\}, \overline{Y}^h)$ to P^h;
a solution $\overline{T}^h$ to D^{*h};
(1) **Initialize:** $\widehat{p} \longleftarrow P(\text{solve}(P_1(0))) + e$;
(2) $\overline{T} \longleftarrow \left\{(\text{solve}(D_1(w)), w) | (u, w) \in \overline{T}^h\right\}$;
(3) **while** $z = 0$ **do**
(4) $Y_k^d \longleftarrow \{D^*(u, w) | (u, w) \in \overline{T}\}$;
(5) $Y_k^p \longleftarrow \text{vert}(Y^d)$;
(6) $\overline{Y} \longleftarrow \varnothing$;
(7) **for** $i = 1$ to $|Y^p|$ **do**
(8) $v \longleftarrow Y_k^p[i]$;
(9) $(\widehat{y}, z) \longleftarrow \text{solve}(P_2(y))$;
(10) $\overline{Y} \longleftarrow \overline{Y} \cup \{\widehat{y}\}$;
(11) **if** $z \neq 0$ **then**
(12) $(x, \delta) \longleftarrow \text{solve}(R(v))$, $(0 < \delta < 1)$;
(13) $y \longleftarrow \delta v + (1 - \delta)\widehat{p}$;
(14) $(u, w) \longleftarrow \text{solve}(D_2(y))$;
(15) $\overline{T} \longleftarrow \overline{T} \cup \{(u, w)\}$;
(16) **endif**;
(17) **endfor**;
(18) **endwhile**
(19) **Output:** $(\overline{Y}, \overline{Y^h})$: nondominated set and directions;
$\overline{T}$: a solution to dual.

Algorithm 3: Benson's Outer Approximation Algorithm [31].

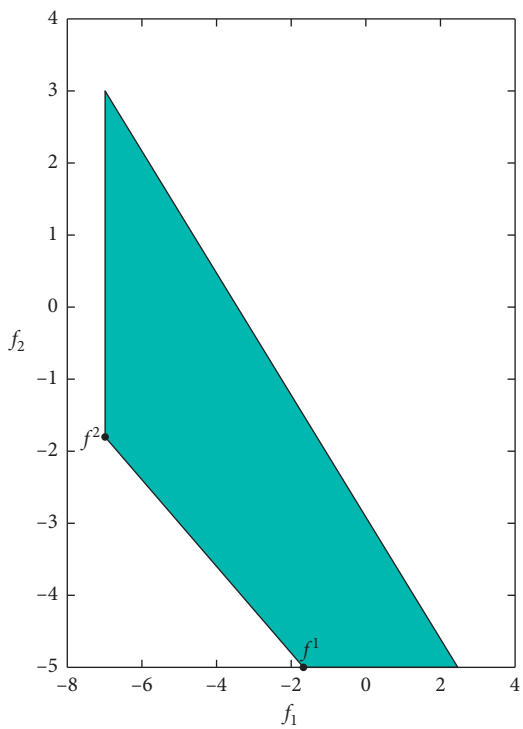

Figure 2: The edge joining the two nondominated points in the objective space [1].

8.1. Illustration of BOA. We consider again problem (3) of Section 5.1. The nondominated points found using a MATLAB implementation of Algorithm 3 are $f^1 = (-7.0, -1.8)^T$ and $f^2 = (-1.6, -5.0)^T$ where f^1 and $f^2 \in Y_N$. These points are shown in Figure 2.

9. Experimental Results

In this section, we provide numerical results to study the number of efficient solutions (NES) and to compare the number of nondominated points (NNP) returned by Algorithms 2 and 3. Table 1 shows the numerical results for a collection of 56 problems from the literature; these problems range from small to moderate size MOLP instances and a few large instances. Problem 1 is taken from Ehrgott [7]. Problems 2 to 10 were taken from Zeleny [39]. Problems 11 to 20 are test problems from the interactive MOLP explorer (iMOLPe) of Alves et al. [40]. Problems 21 to 47 are taken from Steuer [26]. Problem 48 is a test problem in Bensolve-1.2 of Löhne [34], while problems 49 and 53 are test problems in Bensolve-2.0 of Löhne and Weißing [41]. Problems 50 to 52 are obtained using a script in Bensolve-2.0 of Löhne and Weißing [41] that is used to generate problem 53 with the same number of variables and constraints. Finally, problems 54 to 56 are test problems in MOPLIB [38] which stands for multiobjective problem library.

Problem 48 is such that the constraint matrix is sparse while the objective matrix is dense. All the components of the RHS vector are ones except for 200 at the end as the largest entry. Problem 49 has a dense constraint matrix with an identity matrix of order n as its objective matrix where n is the number of variables in the problem.All the components of the RHS vectors are zeros except for the first entry which is one (1) as the only nonzero component it is a degenerate problem. Problems 50 to 53 have dense objective matrices with identity matrices of order n as their constraint matrices where n is also the number of variables in the respective problem. All the components of the RHS vectors are ones.

Note that problem 54 is also highly degenerate. Its structure is such that the constraint and objective matrices are sparse while all the components of the RHS vector are zeros except for a one (1) as the only nonzero entry. Finally, problems 55 and 56 have sparse constraints and objective matrices with dense RHS vectors.

Results for Algorithms 1 were obtained using a MATLAB implementation of the algorithm provided by Rudloff et al. [30]. We modified and extended Algorithm 1 of Evans and Steuer [3] into Algorithm 2 or EMSA, the Extended Multiobjective Simplex Algorithm introduced here. We have implemented it in # in the same way as in [30] and experimented with it on a set of test problems. We also used a MATLAB implementation of Algorithm 3 (BOA), known as Bensolve-1.2 [34]. The current version, Bensolve-2.0 of Löhne and Weißing [41] is implemented in the *C* programming language. We employed Bensolve-1.2 of Löhne [34] which is implemented in MATLAB to test the algorithms with the same tools and for a meaningful comparison. Note that the current version returns the same number of nondominated points as Bensolve-1.2 but has improved running time as noted in [42]. All algorithms were executed on an Intel Core *i*5–2500 CPU at 3.30 GHz with 16.0 GB RAM. In all tests, n is the number of variables, m the number of constraints, and q the number of objectives. Algorithm 1 is MSA of Evans and Steuer [3], Algorithm 2 its extended version, and Algorithm 3 is BOA as presented in [15]. We recorded the number of efficient solutions (NES) returned by MSA, the number of nondominated points (NNP) returned by EMSA, and the NNP returned by BOA for each problem.

As can be seen in Table 1, the NNP returned by EMSA is the same as that returned by BOA for most of the problems considered. This is due to the fact that EMSA is designed to avoid returning redundant nondominated points. This feature is also reported in [8] that BOA avoids redundant calculations of points that would be of little or no use to the DM. This makes EMSA compare favourably in terms of the NNP it returns. Though, we noticed a few differences in the NNP returned for some of the problems considered. These differences occur when some of the nondominated extreme points computed by BOA are repeated. We also noticed a significant reduction in the NNP returned by EMSA compared to the NES returned by MSA. This is due to the fact that MSA returns different efficient extreme points that yield the same nondominated points.

It was also observed that the simplex-type algorithms could not produce results for problems 39, 40, 45, 46, 48, and 56 despite the long running time allowed (3 days); they were aborted. As we noted in [1], the fact that some problems were aborted after 3 days of running time does not necessarily mean that the algorithms cannot solve these problems; if allowed to run further, they could potentially return a huge number of efficient extreme points or nondominated points, or run out of memory which would indicate that the total number of efficient extreme points or nondominated points has exceeded the MATLAB storage capacity of the machine used.

It was also found that for these problems where EMSA could not return a solution after running for three (3) days, these problems may have a huge number of efficient solutions or nondominated points as can be seen from the nondominated extreme points computed by BOA for these problems (see problems 53 and 56) which shows the sensitivity of EMSA to the number of nondominated extreme points in a given problem. EMSA may also find it difficult to return a solution for the degenerate problems (problems 49 and 54) as already mentioned in Section 6.2 due to cycling; that is, one may remain at the same vertex of the feasible region for many iterations or return the same efficient extreme points in more than one iteration. Though this is not the case when the algorithm is computing the nondominated points, as the nondominated points are sorted after computation. Thus, leading to a nondominated set that is devoid of redundant ones.

10. Conclusion

We have reviewed the relevant literature on MSA and BOA [8]. We have also extended the MSA of Evans and Steuer [3] to compute the entire set of all nondominated extreme points and illustrated the algorithms on a small MOLP instance. We then proceeded to compare the total number of nondominated extreme points computed by BOA and EMSA. It was observed that the total number of nondominated extreme points computed by EMSA is the same as that returned by BOA for most of the problems considered.

Data Availability

The authors did not use any secondary data of any kind; they have used a collection of existing test problems in their manuscript whose origins have been clearly stated in Section 9 for the purpose of reproducibility. The sources of all the test instances used are shown in the Experimental Results section.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The authors are grateful to ESRC (Grant ES/L011859/1) for partially funding this research.

References

[1] P. B. Nyiam and A. Salhi, "A comparative study of two key algorithms in multiple objective linear programming," *Journal of Algorithms & Computational Technology*, vol. 13, 2019.

[2] H. A. Eiselt and C.-L. Sandblom, *Linear Programming and its Applications*, Springer Science & Business Media, Berlin, Germany, 2007.

[3] J. P. Evans and R. E. Steuer, "A revised simplex method for linear multiple objective programs," *Mathematical Programming*, vol. 5, no. 1, pp. 54–72, 1973.

[4] J. Philip, "Algorithms for the vector maximization problem," *Mathematical Programming*, vol. 2, no. 1, pp. 207–229, 1972.

[5] M. Zeleny, *Linear Multiobjective Programming*, Springer-Verlag, Berlin, Germany, 1974.

[6] M. Ehrgott, L. Shao, and A. Schöbel, "An approximation algorithm for convex multi-objective programming problems," *Journal of Global Optimization*, vol. 50, no. 3, pp. 397–416, 2011.

[7] M. Ehrgott, *Multicriteria Optimization*, Springer Science & Business Media, Berlin, Germany, 2006.

[8] H. P. Benson, "An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem," *Journal of Global Optimization*, vol. 13, no. 1, pp. 1–24, 1998.

[9] H. P. Benson, "Further analysis of an outcome set-based algorithm for multiple objective linear programming," *Journal of Optimization Theory and Applications*, vol. 97, no. 1, p. 10, 1998.

[10] H. P. Benson, "Hybrid approach for solving multiple-objective linear programs in outcome space," *Journal of Optimization Theory and Applications*, vol. 98, no. 1, pp. 17–35, 1998.

[11] J. P. Dauer, "Analysis of the objective space in multiple objective linear programming," *Journal of Mathematical Analysis and Applications*, vol. 126, no. 2, pp. 579–593, 1987.

[12] J. P. Dauer and Y.-H. Liu, "Solving multiple objective linear programs in objective space," *European Journal of Operational Research*, vol. 46, no. 3, pp. 350–357, 1990.

[13] M. Ehrgott, A. Löhne, and L. Shao, "A dual variant of Benson's "outer approximation algorithm" for multiple objective linear programming," *Journal of Global Optimization*, vol. 52, no. 4, pp. 757–778, 2012.

[14] A. Löhne, B. Rudloff, and F. Ulus, "Primal and dual approximation algorithms for convex vector optimization problems," *Journal of Global Optimization*, vol. 60, no. 4, pp. 713–736, 2014.

[15] L. Shao and M. Ehrgott, "Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning," *Mathematical Methods of Operations Research*, vol. 68, no. 2, pp. 257–276, 2008.

[16] L. Shao and M. Ehrgott, "Approximating the nondominated set of an molp by approximately solving its dual problem," *Mathematical Methods of Operations Research*, vol. 68, no. 3, pp. 469–492, 2008.

[17] L. Pourkarimi, M. A. Yaghoobi, and M. Mashinchi, "Determining maximal efficient faces in multiobjective linear programming problem," *Journal of Mathematical Analysis and Applications*, vol. 354, no. 1, pp. 234–248, 2009.

[18] J. Philip, "Vector maximization at a degenerate vertex," *Mathematical Programming*, vol. 13, no. 1, pp. 357–359, 1977.

[19] R. E. Steuer, *Adbase: A Multiple Objective Linear Programming Solver for All Efficient Extreme Points and All Unbounded Efficient Edges*, Terry college of Business, University of Georgia, Athens, Georgia, 2003.

[20] P. L. Yu and M. Zeleny, "The techniques of linear multiobjective programming. Revue française d'Automatique, d'Informatique et de Recherche Opérationnelle," *Recherche Opérationnelle*, vol. 8, no. 3, pp. 51–71, 1974.

[21] P. L. Yu and M. Zeleny, "The set of all nondominated solutions in linear cases and a multicriteria simplex method," *Journal of Mathematical Analysis and Applications*, vol. 49, no. 2, pp. 430–468, 1975.

[22] P. L. Yu and M. Zeleny, "Linear multiparametric programming by multicriteria simplex method," *Management Science*, vol. 23, no. 2, pp. 159–170, 1976.

[23] I. Heinz, "The enumeration of the set of all efficient solutions for a linear multiple objective program," *Journal of the Operational Research Society*, vol. 28, no. 3, pp. 711–725, 1977.

[24] H. Isermann and G. Naujoks, *Operating Manual for the EFFACET Multiple Objective Linear Programming Package*, Fakultaet fuer Wirtschaftswissenschaften, University of Bielefeld, Bielefeld, Germany, 1984.

[25] T. Gal, "A general method for determining the set of all efficient solutions to a linear vectormaximum problem," *European Journal of Operational Research*, vol. 1, no. 5, pp. 307–322, 1977.

[26] R. E. Steuer, *Multiple Criteria Optimization: Theory, Computation, and Applications*, Wiley, Hoboken, NJ, USA, 1986.

[27] J. G. Ecker and I. A. Kouada, "Finding all efficient extreme points for multiple objective linear programs," *Mathematical Programming*, vol. 14, no. 1, pp. 249–261, 1978.

[28] J. G. Ecker, N. S. Hegner, and I. A. Kouada, "Generating all maximal efficient faces for multiple objective linear programs," *Journal of Optimization Theory and Applications*, vol. 30, no. 3, pp. 353–381, 1980.

[29] P. Armand and C. Malivert, "Determination of the efficient set in multiobjective linear programming," *Journal of Optimization Theory and Applications*, vol. 70, no. 3, pp. 467–489, 1991.

[30] B. Rudloff, F. Ulus, and R. Vanderbei, "A parametric simplex algorithm for linear vector optimization problems," *Mathematical Programming*, vol. 163, pp. 213–242, 2017.

[31] A. Löhne, *Vector Optimization with Infimum and Supremum*, Springer Science & Business Media, Berlin, Germany, 2011.

[32] S. Murray and R. E. Steuer, "A correction to the connectedness of the Evans-Steuer algorithm of multiple objective linear programming," *Foundations of Computing and Decision Sciences*, vol. 30, no. 4, pp. 351–360, 2005.

[33] T. B. Vu, "A finite algorithm for minimizing a concave function under linear constraints and its applications," in *Proceedings of IFIP Working Conference on Recent Advances in System Modelling and Optimization*, Hanoi, Vietnam, 1983.

[34] A. Löhne, "Bensolve: VLP solver," 2012, http://www.bensolve.org.

[35] L. Csirmaz, "Using multiobjective optimization to map the entropy region," *Computational Optimization and Applications*, vol. 63, no. 1, pp. 45–67, 2016.

[36] A. H. Hamel, A. Löhne, and B. Rudloff, "Benson type algorithms for linear vector optimization and applications," *Journal of Global Optimization*, vol. 59, no. 4, pp. 811–836, 2014.

[37] H. V. Junior and M. P. E. Lins, "A win-win approach to multiple objective linear programming problems," *Journal of the Operational Research Society*, vol. 60, no. 5, pp. 728–733, 2009.

[38] A. Löhne and S. Schenker, "MOPLIB: multi-objective problem library," 2015, http://moplib.uni-jena.de.

[39] M. Zeleny, *Multiple Criteria Decision Making*, McGraw-Hill, New York, NY, USA, 1982.

[40] M. J. Alves, C. H. Antunes, and J. Clímaco, "Interactive MOLP explorer-A graphical-based computational tool for teaching and decision support in multi-objective linear programming models," *Computer Applications in Engineering Education*, vol. 23, no. 2, pp. 314–326, 2015.

[41] A. Löhne and W. Benjamin, "Bensolve: VLP solver, version 2.0.x," 2015, http://www.bensolve.org.

[42] A. Löhne and B. Weißing, "The vector linear program solver Bensolve - notes on theoretical background," *European Journal of Operational Research*, vol. 260, no. 3, pp. 807–813, 2017.