


# On the simplex, interior-point and objective space approaches to multiobjective linear programming

Paschal B Nyiam  and Abdellah Salhi

Journal of Algorithms & Computational  
Technology  
Volume 15: 1–20  
© The Author(s) 2021  
DOI: 10.1177/17483026211008414  
journals.sagepub.com/home/act  


## Abstract

Most Multiple Objective Linear Programming (MOLP) algorithms working in the decision variable space, are based on the simplex algorithm or interior-point method of Linear Programming. However, objective space based methods are becoming more and more prominent. This paper investigates three algorithms namely the Extended Multiobjective Simplex Algorithm (EMSA), Arbel's Affine Scaling Interior-point (ASIMOLP) algorithm and Benson's objective space Outer Approximation (BOA) algorithm. An extensive review of these algorithms is also included. Numerical results on non-trivial MOLP problems show that EMSA and BOA are at par and superior in terms of the quality of a most preferred nondominated point to ASIMOLP. However, ASIMOLP more than holds its own in terms of computing efficiency.

## Keywords

Multiple objective linear programming, multiobjective simplex algorithm, affine scaling interior MOLP algorithm, outer-approximation algorithm, most preferred nondominated point

Received 23 July 2020; revised 23 July 2020; accepted 12 February 2021

## Introduction

MOLP is a branch of Multiple Criteria Decision Making (MCDM) that seeks to optimize two or more linear objective functions subject to a set of linear constraints. MOLP has been an active area of research since the 1960s 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

$$\begin{aligned} \min \quad & c_1^T x = f_1 \\ & \vdots \\ & c_q^T x = f_q \\ \text{subject to } & x \in X = \{x \in \mathbb{R}^n : Ax = b, b \in \mathbb{R}^m, x \geq 0\} \end{aligned} \quad (1)$$

where  $c_1, \dots, c_q$  are  $n$ -vectors containing the coefficients of the multiple objective functions,  $A$  is an

$m \times n$  constraint matrix and  $b$  is the right hand side vector.

In practice, MOLP is typically solved by the Decision Maker (DM) with the support of the analyst looking for a most preferred (best) solution in the feasible region  $X$ . This is because optimizing all the objective functions simultaneously is not possible due to their conflicting nature. Consequently, the concept of optimality is replaced with that of efficiency. The purpose of MOLP is to obtain either all the efficient or nondominated extreme points or a subset of either, or a most preferred point depending on the purpose for which it is needed.

In the last few decades a number of algorithms have been suggested for MOLP. Most are based on the simplex and interior-point algorithms for Linear

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

### Corresponding author:

Paschal B Nyiam, Department of Mathematical Sciences, University of Essex, Colchester, UK.  
Email: pbnyia@essex.ac.uk



Programming which work in the decision space. However, Benson<sup>1</sup> argued that since the number of objectives in an MOLP is often much smaller than the number of decision variables and typically many efficient extreme points in the decision space map to a single point in the objective space, generating the set of nondominated points in the objective space would require less computation. He then suggested an outer-approximation algorithm for computing all the nondominated points in the objective space of the problem.

In this paper, we will extend the Multi-objective Simplex Algorithm (MSA) of Evans and Steuer<sup>2</sup> and compare it with the primal variant of BOA<sup>1</sup> as well as with a variant of the Interior-Point Method (IPM) developed by Arbel<sup>3</sup> known as ASIMOLP. These algorithms will be compared comprehensively on a series of existing test problems and the results will be reported and discussed.

This comparison may sound not possible given that the three algorithms are based on three different philosophies and compute different things: MSA works in the decision space and finds the set of all efficient extreme points; ASIMOLP also works in the decision space but finds a most preferred efficient point and also returns the corresponding most preferred nondominated point; BOA, on the other hand, works in the objective space to find the set of all nondominated points of the problem.

To achieve this comparison, we will extend the MSA of Evans and Steuer<sup>2</sup> whose explicit form is given in Ehrgott<sup>4</sup> and which computes all efficient extreme points, to also generate the set of all nondominated points of the problem. It has been shown that, in practice, the DM prefers basing his or her choice of a most preferred (best) solution in the nondominated points, Benson.<sup>1</sup> We shall then act as the DM and choose a Most Preferred Nondominated Point (MPNP) whose components are as close as possible to an unattainable ideal objective point from the nondominated set returned by the extended MSA and BOA to compare with a MPNP returned by ASIMOLP.

To the best of our knowledge, no comparison of the computing efficiency and the quality of MPNP chosen from the nondominated set returned by BOA and EMSA with that returned by ASIMOLP has been carried out before. We intend to fill this gap here.

This paper is organized as follows. The next section introduces MOLP and basic notation. The literature review section is a brief review of the relevant literature. We present MSA, its extended version EMSA, ASIMOLP and BOA in the Multiobjective simplex algorithm, The extended multiobjective simplex algorithm, The affine scaling interior point algorithm and Benson's outer approximation algorithm sections, respectively. The Selection of the most preferred

nondominated point section discusses the selection of a MPNP and the Experimental results section presents numerical results obtained with the different algorithms. Finally, a conclusion is presented in the last section.

## Notation and definitions

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

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

where  $C$  is a  $q \times n$  criterion matrix consisting of the rows  $c_k^T$ ,  $k = 1, 2, \dots, q$ ,  $A$  and  $b$  are as described earlier. 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  $X$ .

A nondominated point in the objective space is the image of an efficient solution in the decision space; nondominated points form the nondominated set.

An efficient solution of an MOLP problem is a solution that cannot improve any of the objective functions without deteriorating at least one of the other objectives. A weakly efficient solution is one that cannot improve all the objective functions simultaneously. Mathematically, let  $\hat{x} \in X$  be a feasible solution of (2) and let  $\hat{y} = C\hat{x}$ :

- $\hat{x}$  is called efficient if there is no  $x \in X$  such that  $Cx \leq C\hat{x}$  and  $Cx \neq C\hat{x}$ ; correspondingly,  $\hat{y} = C\hat{x}$  is called nondominated.
- $\hat{x}$  is called weakly efficient if there is no  $x \in X$  such that  $Cx < C\hat{x}$ ; and  $\hat{y} = C\hat{x}$  is called weakly nondominated, Ehrgott.<sup>4</sup>

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, Benson.<sup>5</sup>  $Y_N = \{Cx : x \in X_E\}$  and  $Y_{WN} = \{Cx : x \in X_{WE}\}$  are the nondominated and weakly nondominated sets in the objective space of (2), respectively.

The nondominated faces in the objective space of a given problem constitutes the nondominated frontier and the efficient faces in the decision space of the problem constitutes the efficient frontier.

The ideal objective point  $y^*$  is the minimum criterion values over the efficient set  $X_E$ . The ideal objective values are easy to obtain by simply minimizing each objective function individually over the feasible region  $X$ , Alves and Costa.<sup>6</sup>

### Illustration

We consider the following MOLP adapted from Junior and Lins.<sup>7</sup>

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

The feasible region in the decision space is shown in Figure 1, where  $x^1 = (7.0, 1.8)^T$ ,  $x^2 = (1.6, 5.0)^T \in X_E$ .

The nondominated points in the objective space are shown in Figure 2, where  $f^1 = (-1.6, -5.0)^T$ , and  $f^2 = (-7.0, -1.8)^T \in Y_N$ .

### Literature review

#### Decision space methods

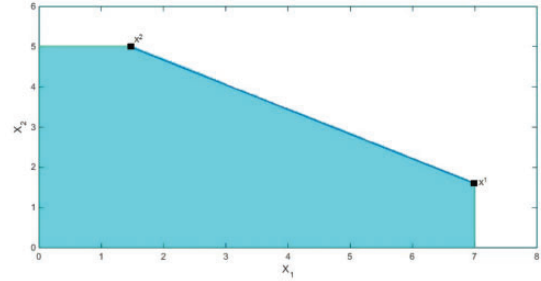
As stated earlier, a number of approaches have been suggested for generating either the entire efficient decision set  $X_E$  or the nondominated set  $Y_N$  or a subset thereof, or a most preferred solution to the problem.

Eiselt and Sandblom<sup>8</sup> note that, Evans and Steuer,<sup>2</sup> Philip<sup>9</sup> and Zeleny<sup>10</sup> derived generalized versions of the simplex method known as MSA. That of Philip<sup>9</sup> 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 Philip,<sup>11</sup> it was modified to overcome this difficulty.

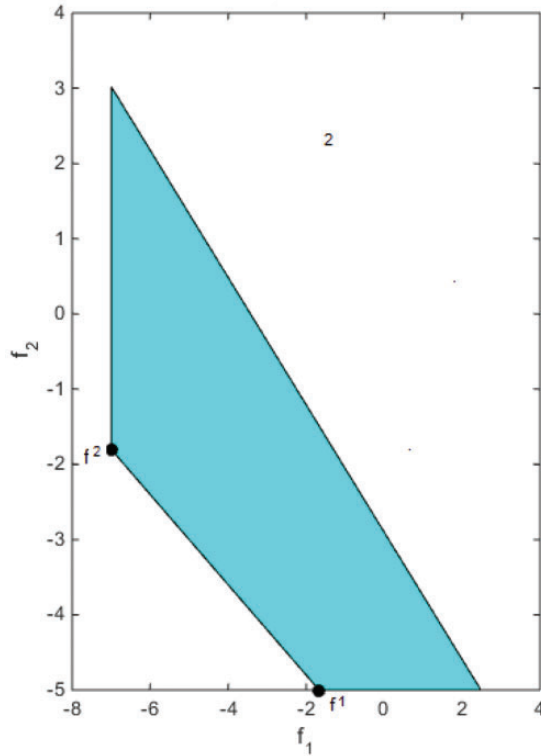
The MSA of Evans and Steuer<sup>2</sup> also generates all the efficient extreme points and unbounded efficient edges of MOLPs; see also Algorithm 7.1, on page 178 of Ehrgott.<sup>4</sup> The algorithm first establishes that the problem is feasible and has efficient solutions. Thereafter, it generates them all. An LP test problem is solved to determine the pivots that lead to efficient vertices. The algorithm is implemented as a software called ADBASE in Steuer.<sup>12</sup>

The MSA variant of Zeleny<sup>10</sup> 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 Evans and Steuer<sup>2</sup> where the test problem determines pivots leading to efficient vertices.

Yu and Zeleny<sup>13,14</sup> used the approach in Zeleny<sup>10</sup> to generate the set of all efficient solutions and presented a formal procedure for testing the efficiency of extreme



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



**Figure 2.** The edge joining the two nondominated points in the objective space.

points. The efficient extreme points are derived from the efficient faces, in a top-to-down search strategy. Numerical illustrations with three objectives were used to demonstrate the effectiveness of the method. In a similar paper, Yu and Zeleny<sup>15</sup> applied their approach expanded in Yu and Zeleny<sup>14</sup> to parametric linear programming. Two basic forms of the problem and two computational procedures for computing the efficient set were presented: the direct decomposition of the parametric space into subspaces associated with extreme points and the indirect algebraic method. From a numerical experience point of view, the indirect

algebraic method outperforms the direct decomposition.

Isermann<sup>16</sup> proposed a variant of the MSA of Evans and Steuer<sup>2</sup> 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.<sup>17</sup>

The MSA of Gal<sup>18</sup> generates the set of all efficient vertices and higher-dimensional faces. This approach is meant to address the problem of determining efficient faces and higher dimensional faces not resolved in Evans and Steuer<sup>2</sup> and Philip.<sup>9</sup> Here, efficient extreme points are generated using a test problem. The algorithm also determines higher-dimensional efficient faces for degenerate problems which were only discussed in Isermann<sup>16</sup> and Zeleny<sup>10</sup> but not solved. The efficient faces are generated in a bottom-to-top search strategy unlike what was suggested in Yu and Zeleny.<sup>13,14</sup>

Steuer<sup>19</sup> applied the MSA of Evans and Steuer<sup>2</sup> to parametric and non-parametric MOLP. Different methods for obtaining an initial efficient extreme point as well as different LP test problems were also presented. Efficient extreme points are generated through the decomposition of the weight space into finite subsets that provide optimal weights corresponding to extreme point solutions.

Ehrgott<sup>4</sup> applied the MSA of Evans and Steuer<sup>2</sup> to solve MOLP problem instances with two and three objective functions. Ecker and Kouada<sup>20</sup> also proposed a variation on the MSA of Evans and Steuer.<sup>2</sup> 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 needs not be bounded. The algorithm enumerates all efficient extreme points and appears to have computational advantage over other methods.

In a different paper, Ecker, Hegner and Kouada<sup>21</sup> 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 Gal,<sup>18</sup> which dramatically improves computation time. The proposed approach was illustrated with a degenerate example given in Yu and Zeleny,<sup>14</sup> to demonstrate its applicability. It was computationally more efficient than the method in Yu and Zeleny.<sup>14</sup>

The MSA of Arman and Malivert<sup>22</sup> 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 Yu and Zeleny<sup>14</sup> was also used to demonstrate its effectiveness. The proposed MSA was superior to that in Yu and Zeleny.<sup>14</sup>

Rudloff, Ulus and Vanderbei<sup>23</sup> suggested a parametric MSA which works for bounded and unbounded problems, but does not find all the efficient solutions unlike the algorithm of Evans and Steuer.<sup>2</sup> Instead, it finds a subset of efficient solutions based on the idea of Löhne.<sup>24</sup> That is, a subset of efficient extreme points and directions that allows to generate the whole efficient frontier. The algorithm performs pivoting for only one leaving variable among the set of all possible leaving variables. It was compared with that in Benson<sup>1</sup> which is an objective space based algorithm, and that of Evans and Steuer.<sup>2</sup> Numerical experiments show that the proposed algorithm outperforms Benson's algorithm for non-degenerate problems. However, Benson's algorithm is better for highly degenerate ones. The parametric MSA was also found to be computationally more efficient than that of Evans and Steuer.<sup>2</sup> Of all these variants, it was noted in Schechter and Evans<sup>25</sup> that, the algorithm of Evans and Steuer<sup>2</sup> is the most popular and successful for computing all efficient extreme points of the problem.

MSA and its variants make explicit use of the vertices of the feasible region. Interior-point approaches, however, generate iterates in the interior of the feasible region. Various such approaches have been suggested. The difference between them depends on the methodology employed to assess the suitability of points used to derive a combined search direction along which one heads towards the next iterate.

The first to adapt a variant of Karmarkar<sup>26</sup> interior-point algorithm, to solve MOLP appears to be Abhyankar, Morin and Trafalis.<sup>27</sup> It relies on the method of centers. It uses a parameterization of ellipsoids in the  $n$ -dimensional space to approximate the efficient frontier of the problem in polynomial time.

Arbel<sup>28</sup> also modified and adapted a variant of Karmarkar<sup>26</sup> algorithm resulting in the so called Affine Scaling Interior MOLP (ASIMOLP) algorithm. He used the convex combination of individual directions to derive a combined direction along which to step toward the next iterate. Specifically, the algorithm generates step direction vectors based on the objectives of the problem. The relative preference of these directions is then assessed using a utility (or preference) function to obtain the points used in combining them into a single direction vector that moves the current

iterate to a new one. The process is repeated until the algorithm converges to a most preferred efficient solution after meeting some termination conditions.

Arbel<sup>3</sup> proposed another ASIMOLP algorithm. This approach offers another means of assessing preference information to establish a combined search direction rather than using the DM's utility function. The Analytic Hierarchy Process (AHP) developed in Saaty<sup>29</sup> was applied to obtain the relative preference of points used to derive a combined direction along which the next step is taken. It is based on the assessed preferences to weigh the step direction vectors for each of the objectives in order to derive a combined step direction vector. This process continues to generate search directions and new feasible points at each iteration, until the algorithm converges to a most preferred point on the efficient frontier.

In Arbel,<sup>30</sup> another ASIMOLP algorithm based on the AHP has been suggested. The derived preference information is applied to the projected gradients in order to obtain anchoring points and cones used in searching for a most preferred solution. The boundaries of the constraints polytope are constantly probed to make more directions available, which enables one to arrive at a most preferred solution.

Wen and Weng<sup>31</sup> modified the ASIMOLP algorithm in Arbel<sup>28</sup> to resolve zigzagging issues. The modified algorithm, however, may not yield a most preferred efficient solution.

Lin, Chen and Chen<sup>32</sup> also proposed a modification of the ASIMOLP of Arbel.<sup>28</sup> They adopted the utility function trade-off method to weigh the objective functions involved and compared the modified algorithm with that in Wen and Weng<sup>31</sup> and the simplex method. Numerical experiments show that their algorithm is superior. On computing efficiency, the interior point based algorithms outperform the simplex-based ones on large scale problems.

Weng and Wen<sup>33</sup> presented an ASIMOLP based algorithm. It computes a weighted sum of the different search directions involved using a utility function. These search directions are then normalized with the weights to obtain a combined direction that moves the current solution to an anchor point. Computational experiments show that the proposed algorithm is suitable for solving large scale instances.

### Objective space methods

Due to the various difficulties arising from solving MOLP problems in the decision space (such as having different efficient solutions that map onto the same point in the objective space), efforts were made to look at the possibility of solving them in the objective space.

Benson,<sup>1</sup> who presented a detailed account of decision space approaches, proposed an algorithm for generating the set of all nondominated points in the objective space. This is the so called BOA. According to him, this algorithm is the first of its kind. Computational results suggest that the objective space based approach is better than the decision space based one. A further analysis of the objective space based algorithm for the problem was presented in Benson.<sup>34</sup> This outer approximation algorithm also generates the set of all weakly nondominated points, thereby enhancing the usefulness of the algorithm as a decision aid.

Another of Benson's<sup>5</sup> suggestions is a hybrid approach for solving the problem in the objective space. The approach partitions the objective space into simplices that lie in each face so as to generate the set of nondominated points. This idea was earlier presented in Ban.<sup>35</sup> The algorithm is quite similar to that in Benson.<sup>1</sup> The difference between them is in the manner in which the nondominated vertices are found. While a vertex enumeration procedure is employed in Benson,<sup>1</sup> a simplicial partitioning technique is used in the latter.

In Shao and Ehrgott<sup>36</sup> a modification of the algorithm of Benson<sup>1</sup> was presented. While in Benson,<sup>1</sup> a bisection method that requires the solution of many LPs in one step is required, here, solving one LP achieves the desired effect and in the process improves computation time. In Shao and Ehrgott<sup>37</sup> was proposed an approximate dual variant of the algorithm of Benson<sup>1</sup> for obtaining approximate nondominated points to the problem. The proposed algorithm was applied to the beam intensity optimization problem of radio therapy treatment planning for which approximate nondominated points were obtained. Numerical testing shows that the approach is faster than solving the primal directly.

The explicit form of the algorithm of Benson<sup>1</sup> as modified by Shao and Ehrgott<sup>36</sup> is presented in Löhne.<sup>24</sup> This version solves two LPs in each iteration during the process of obtaining the nondominated extreme points. Löhne<sup>38</sup> presented a Matlab implementation of this algorithm called BENSOLVE-1.2, for computing all the nondominated points and directions (unbounded nondominated edges) of the problem.

Csirmaz<sup>39</sup> presented an improved version of the algorithm in Benson<sup>1</sup> that solves one LP and a vertex enumeration problem in each iteration. While in Benson,<sup>1</sup> solving two LPs to determine a unique boundary point and a supporting hyperplane of the image is required in two steps, here, the two steps are merged and solving only one LP does both tasks and improves computation time. The algorithm was used to generate all the nondominated vertices of the polytope defined by a set of Shannon inequalities on four

random variables so as to map their entropy region. Numerical testing shows the applicability of the approach to medium and large instances.

Hamel and Löhne<sup>40</sup> introduced new versions and extensions of the algorithm in Benson.<sup>1</sup> The primal and dual variants of the algorithm solve only one LP problem in each iteration. Tests reveal a reduction in computation time.

Similarly, Löhne, Rudloff and Ulus<sup>41</sup> extended the primal and dual variants of the algorithm in Benson<sup>1</sup> to approximately solve convex vector optimization problems in the objective space.

Based on our extensive review of the topic, it was observed that no comparison of the computing efficiency and quality of a MPNP chosen from the nondominated set returned by BOA and extended MSA with the MPNP returned by ASIMOLP has been carried out. We intend to fill this gap here.

## Multiobjective simplex algorithm

A typical multiple objective simplex algorithm is that of Evans and Steuer.<sup>2</sup> The version described here can be found in Ehrgott,<sup>4</sup> page 178. We consider this algorithm because of its popularity (see Schechter and Steuer<sup>25</sup>) and because most of the MSA algorithms discussed earlier are either based on or are variants of

it. It works in the decision space and finds the set of all efficient extreme points.

The algorithm is initialized by solving two auxiliary LPs to determine whether the problem is feasible and to verify that it has efficient solutions. If the feasible region  $X$  is not empty and the set of efficient extreme points  $X_E$  exists, 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  $L_1$  to be processed is empty. The algorithm stops and returns list  $L_2$  from where all efficient extreme points are computed. Before we describe MSA in pseudo-code form, we first explain the used notation.

**Notation:**  $A, b, C$  form the problem data;  $L_1$  and  $L_2$  as above;  $e^T = (1, \dots, 1) \in \mathbb{R}^q$ ;  $I$  is the identity matrix of proper order;  $X$  is the feasible set;  $X_E$ , the set of efficient solutions;  $B$ , the efficient basis;  $N_E$ , a list of efficient nonbasic variables;  $N$ , the set of nonbasic variables;  $B'$ , the new basis;  $\bar{A}$ , and  $\bar{b}$  are updated constraint matrix and RHS vector;  $R$  is the nonbasic part of the reduced cost matrix and  $r^j$  is a column of  $R$  corresponding to a nonbasic variable being tested for efficiency.

---

### Algorithm 1: Multiobjective Simplex Algorithm, Ehrgott<sup>4</sup>

0: **Input:**  $A, b, C$  : Problem data

1: **Initialize:** Set  $L_1 \leftarrow \emptyset$ ,  $L_2 \leftarrow \emptyset$ ; Phase I : Solve the LP  $\min\{e^T z : Ax + Iz = b, x, z \geq 0\}$ . **If the optimal value of this LP is nonzero, STOP,  $X = \emptyset$ ;**

Otherwise  $x^0$  is a basic feasible solution of MOLP

Phase II : Solve the LP  $\min\{u^T b + w^T Cx^0 : u^T A + w^T C \geq 0, w \geq e\}$ . **If it is infeasible, STOP,  $X_E = \emptyset$ ;**

Otherwise  $(\hat{u}, \hat{w})$  is an optimal solution;

Find an optimal basis  $B$  of the LP  $\min\{\hat{w}^T Cx : Ax = b, x \geq 0\}$ ;

Set  $L_1 \leftarrow \{B\}$ ,  $L_2 \leftarrow \emptyset$ ;

2: **while**  $L_1 \neq \emptyset$

3:     Choose  $B \in L_1$ ,  $L_1 \leftarrow L_1 \setminus \{B\}$ ,  $L_2 \leftarrow L_2 \cup \{B\}$ ;

4:     Compute  $\bar{A}$ ,  $\bar{b}$ , and  $R$  according to  $B$ ;

5:      $N_E \leftarrow N$ ;

6:     **for all**  $j \in N$

7:         Solve the LP  $\max\{e^T v : Ry - r^j \sigma + Iv = 0; y, \sigma, v \geq 0\}$ .

8:         **If this LP is unbounded**  $N_E \leftarrow N_E \setminus \{j\}$ ;

9:     **for all**  $j \in N_E$

10:         **for all**  $i \in B$

11:             **if**  $B' \leftarrow (B \setminus \{i\}) \cup \{j\}$  is feasible,  $B' \notin L_1 \cup L_2$  **then;**

12:                  $L_1 \leftarrow L_1 \cup B'$ ;

13:             **endif;**

14:         **endfor;**

15:     **endfor;**

16:     **endfor;**

17: **endwhile.**

18: **Output:**  $L_2$  : List of efficient bases.

---

### Illustration of MSA

Consider MOLP (3) of the Illustration section. We solve this problem using a Matlab implementation of Algorithm 1 provided by Rudloff, Ulus and Vanderbei.<sup>23</sup> The efficient extreme points found are  $x^1 = (7.0, 1.8)^T$ ,  $x^2 = (1.6, 5.0)^T$ ,  $x^3 = (1.6, 5.0)^T$ ,  $x^4 = (7.0, 1.8)^T$ . Where  $x^1 = (x_1^1, x_2^1)^T$ , her  $x^4 = (x_1^4, x_2^4)^T \in X_E$ . The algorithm is prone to generating more efficient extreme points due to the way it operates and due to the fact that it may find the same efficient extreme point 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. The feasible region in the decision space is the same as in Figure 1.

### 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 extreme points ( $X_E \leftarrow \{\bar{x}\}$ ). The first nondominated point is also computed from  $C^T \bar{x}$  and appended to the nondominated set ( $Y_N \leftarrow \{C^T \bar{x}\}$ ).

As the algorithm iterates, a new efficient basis  $B'$  is obtained after each pivot and the corresponding efficient basic feasible solution  $\bar{x}'$  (line 11 of Algorithm 2) is found and added to the set of efficient extreme points  $X_E$  (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 pseudo-code form, we first state here that the structure of the algorithm and the used notation remain the same as that in Algorithm 1. The additional components are  $\bar{x}$ ,  $\bar{x}'$ ,  $X_E$  and  $Y_N$  which stand for the efficient basic feasible solution, the new efficient basic feasible solution, the set of efficient extreme points and the corresponding set of nondominated points for output.

---

#### Algorithm 2: Extended Multiobjective Simplex Algorithm

```

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

```

---

### Illustration of the extended MSA

We modified and extended the Matlab implementation of Algorithm 1 provided by Rudloff, Ulus and Vanderbei<sup>23</sup> and used it to solve problem (3) of the Illustration section. The efficient extreme points 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  $f^1 = (f_1^1, f_2^1)^T$ ,  $f^2 = (f_1^2, f_2^2)^T \in Y_N$ . Notice here that, the efficient extreme points  $x^1$ ,  $x^2$  and the corresponding nondominated points  $f^1$  and  $f^2$  returned are devoid of redundant points. The algorithm is designed to avoid returning redundant efficient and nondominated points unlike the original version. The feasible region in the decision space is also the same as in Figure 1.

### The affine scaling interior point algorithm

ASIMOLP whose general form can be found in Arbel,<sup>3</sup> works in the decision space and returns only one efficient extreme point of the problem, or at most, an efficient face of the feasible region. It also returns the corresponding nondominated point. The algorithm is initialized with a feasible and interior starting solution vector  $x^0$  and generates  $q$  interior step direction vectors  $dx_i$  ( $1 \leq i \leq q$ ). AHP is then used to derive the relative priority or preference vector  $p$  for these directions by filling a pairwise comparison matrix, which is then normalized and the rows are averaged to obtain the priority vector. The components of the derived priority vector  $p$  are then used as coefficients of a convex combination of the  $q$  interior step directions that yields a combined step direction vector  $dx$  that moves toward a new feasible point. This process continues until the algorithm converges to a most preferred efficient extreme point after meeting some termination conditions. Before we present the pseudo-code form of ASIMOLP, the used notation is described.

**Notation:**  $A, b, C$  form the problem data;  $x^0$  is the initial interior feasible solution vector;  $\alpha_i$  is the step size ( $1 \leq i \leq q$ );  $\sigma$  is a stopping tolerance;  $\rho$  is the step size factor;  $D$  is the diagonal and scaling matrix;  $y_i$  is an estimate of the dual vector ( $1 \leq i \leq q$ );  $dx_i$  is the  $i$ th interior step direction vector ( $1 \leq i \leq q$ );  $p_i$  is the derived priority vector ( $1 \leq i \leq q$ );  $x_{new}$  is the new feasible point;  $f_{new}$  is a vector containing the new objective values;  $x_{end}$  is the most preferred efficient extreme point at the boundary of the feasible region at termination and  $f_{end}$  is a vector containing the corresponding objective values at termination.

**Algorithm 3:** Affine Scaling Interior MOLP Algorithm

```

0: Input:  $A, b, C$ : Problem data
1: Initialize: Choose  $x^0 > 0$ , Stopping tolerance  $\sigma$  ( $0 < \sigma < 1$ ), Step size factor  $\rho$  ( $0 < \rho < 1$ ),  $Converged = 0, k \leftarrow 0$ ;
2: while  $Converged \neq 1$  do
3:    $k \leftarrow k + 1$ 
4:    $D \leftarrow \text{diag}(x_0)$ 
5:    $y_i(k) \leftarrow (AD^2A^T)^{-1}AD^2c_i, 1 \leq i \leq q$ 
6:    $dx_i(k) \leftarrow D^2(c_i^T - A^T y_i(k)), 1 \leq i \leq q$ 
7:   if  $dx_i(k) \geq 0, 1 \leq i \leq q$ , stop
8:   else
9:      $\alpha_i \leftarrow \min[\frac{-x_i}{dx_i(k)}, \forall dx_i(k) < 0], 1 \leq i \leq q$ 
10:     $x_i(k+1) \leftarrow x(k) + \rho \alpha_i dx_i(k), 1 \leq i \leq q$ 
11:     $dx \leftarrow \sum_{i=1}^q p_i \alpha_i dx_i$ 
12:     $x(k+1) \leftarrow x(k) + \rho dx(k)$ 
13:     $x_{new} \leftarrow x(k) + dx(k)$ 
14:     $f_{new} \leftarrow C^T x_{new}$ 
15:     $dx_{end} \leftarrow x_{end} - x(k)$ 
16:    if  $k > 1$ , do
17:       $dx \leftarrow \sum_{i=1}^q p_i \alpha_i dx_i + p_{end} dx_{end}$ 
18:       $x_{new} \leftarrow x(k) + \rho dx(k)$ 
19:       $f_{new} \leftarrow C^T x_{new}$ 
20:       $x_{end} \leftarrow x(k) + dx(k)$ 
21:       $f_{end} \leftarrow C^T x_{end}$ 
22:       $dx_{end} \leftarrow x_{end} - x_{new}$ 
23:    else
24:      if  $\|dx_{end}\| \leq \sigma$ , stop
25:      else
26:         $x^0 \leftarrow x_{new}$ 
27:        Go to step 4
28:      endif
29:    endif
30:  endif
31: endwhile
32: Output:  $x_{end}$ : Most preferred efficient extreme point
     $f_{end}$ : Values of the objective functions

```

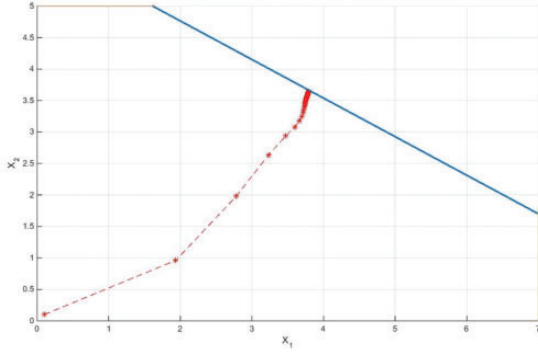
### Illustration of ASIMOLP

We developed the pseudo-code of this algorithm, implemented it in Matlab and used it to solve Problem (3) of the Illustration section. The most preferred efficient extreme point found using our Matlab implementation is  $x^1 = (3.6418, 3.7913)^T$ , and the values of the objective functions are  $f^1 = (-3.6418, -3.7913)^T$ . The search path as generated by Algorithm 3 is shown in Figure 3.

### Determination of the priority vector used in ASIMOLP

From Arbel,<sup>28</sup> one can either use a utility function if it is available (as was done in Arbel<sup>42</sup>) to assess preference information needed to establish a combined step





**Figure 3.** ASIMOLP search path showing convergence to the efficient frontier.

direction instead of interacting with the DM or use the AHP methodology, but in most cases, the utility function is not known, as stated in Arbel.<sup>3</sup> In this paper, we have used AHP as was done in Arbel<sup>3,30</sup> to derive the relative preference or priority vector  $p$  whose components are used as coefficients of a convex combination of the  $q$  interior step directions that yields a combined step direction that moves the current iterate to a new one.

The procedure involves a pairwise comparison of the  $q$  interior step directions and construction of a  $q \times q$  comparison matrix for comparing the interior step directions. A complete pairwise comparison matrix  $A$  can be expressed as

$$A = \begin{matrix} & d_1 & d_2 & \dots & d_q \\ \begin{matrix} d_1 \\ d_2 \\ \vdots \\ d_q \end{matrix} & \begin{pmatrix} \frac{w_1}{w_1} & \frac{w_1}{w_2} & \dots & \frac{w_1}{w_q} \\ \frac{w_2}{w_1} & \frac{w_2}{w_2} & \dots & \frac{w_2}{w_q} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{w_q}{w_1} & \frac{w_q}{w_2} & \dots & \frac{w_q}{w_q} \end{pmatrix} & = & \begin{pmatrix} 1 & a_{12} & \dots & a_{1q} \\ a_{21} & 1 & \dots & a_{2q} \\ \vdots & \vdots & \ddots & \vdots \\ a_{q1} & a_{q2} & \dots & 1 \end{pmatrix} \end{matrix}$$

where the entry  $a_{ij}$  indicates the strength of the step direction  $d_i$  when compared with the step direction  $d_j$ . These entries are obtained from the well-established comparison scale used by AHP in Saaty,<sup>29</sup> Saaty and Vargas.<sup>43</sup> Such a scale is shown in Table 1.

We note that the above comparison matrix is a reciprocal matrix, where  $a_{ij} = 1/a_{ji}$ ,  $a_{ij} > 0$  and  $a_{ij} = 1$  for  $i = j$ . After filling out and obtaining the comparison matrix, the matrix is then normalized. The normalized principal eigen-vector herein referred to as the preference or priority vector,  $p$ , is obtained by averaging across the rows of the matrix. The components of the priority vector are then used as coefficients of a convex combination of the  $q$  interior directions that yields a

**Table 1.** Graduation scale for comparing alternatives.

Numerical value	Interpretation
1	requirement $i$ and $j$ are of equal value
3	requirement $i$ has a slightly higher value than $j$
5	requirement $i$ has a strongly higher value than $j$
7	requirement $i$ has a very strongly higher value than $j$
9	requirement $i$ has a absolutely higher value than $j$
2,4,6,8	intermediate values between two adjacent judgments

combined direction that enables one to move from the current iterate to the next, (more details can be seen in Saaty,<sup>29</sup> Saaty and Vargas.<sup>43</sup>)

### Benson's outer approximation algorithm

We present here the version of BOA due to Shao and Ehrgott.<sup>36</sup> This version can be found in Löhne.<sup>24</sup> It works in the objective space of the problem and returns the set of all nondominated points and extreme directions. The algorithm can be regarded as a primal-dual method because it also solves the dual problem. But here, we are only concerned with the solution of the primal. The algorithm first constructs an initial surrounding polytope  $Y_0$  containing the image  $Y$  in the objective space and an interior point  $\hat{p}$  of the image is determined. At each iteration, a bounding polytope  $Y_k$  is maintained and its vertices as well as inequality representation (facets) are stored. Then for each vertex  $v$  of the polytope, the algorithm checks if the vertex is on the boundary of  $Y$ . If all the vertices are on the boundary of  $Y$ , the problem is solved. The external 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  $\hat{p}$  and finds the intersection  $y$  of this line with the boundary of  $Y$  by solving LP  $P_2(v)$ . Then a supporting hyperplane adjacent to  $y$  is constructed by solving LP  $D_2(y)$ . This hyperplane is added to  $Y_k$  to provide a smaller polytope. The algorithm is repeated in the same way 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 set  $\bar{Y}$  and directions  $\bar{Y}^h$  of the problem. Consider the notation used in the pseudo-code of Benson's outer Approximation Algorithm.

**Notation:**  $A$ ,  $b$ ,  $C$  are the problem data;  $P^h$  is the homogeneous problem;  $D^{*h}$  is the homogeneous dual

problem;  $\bar{T}^h$  is the solution of the homogeneous dual problem;  $\hat{p}$  is an interior point;  $\bar{T}$  is a set of solutions of the dual problem;  $Y_k^d$  is the inequality representation of the current polytope;  $k$  is the iteration counter;  $Y_k^p$  is the representation by vertices;  $(\hat{y}, z)$  is an optimal solution to  $P_2(v)$ ;  $\delta(0 < \delta < 1)$  is a unique value that determines the intersection or boundary point  $y$ ;  $R(v)$  is the LP that finds the unique value  $\delta$ ; the command `solve()` solves an LP; `vert()` returns the vertices of a polytope  $Y$ ;  $\bar{Y}$  is the set of nondominated vertices;  $(\bar{Y}^h)$  is the set of extreme directions.

**Algorithm 4:** Benson's Outer Approximation Algorithm, Löhne<sup>24</sup>

0: **Input:**  $A, b, C$  : Problem data a solution  $(\{0\}, \bar{Y}^h)$  to  $P^h$ ;

a solution  $\bar{T}^h$  to  $D^{*h}$ ;

1: **Initialize:**  $\hat{p} \leftarrow P(\text{solve}(P_1(0))) + e$ ;

2:  $\bar{T} \leftarrow \{(\text{solve}(D_1(w)), w) \mid (u, w) \in \bar{T}^h\}$ ;

3: **while**  $z = 0$  **do**

4:  $Y_k^d \leftarrow \{D^*(u, w) \mid (u, w) \in \bar{T}\}$ ;

5:  $Y_k^p \leftarrow \text{vert}(Y_k^d)$ ;

6:  $\bar{Y} \leftarrow \emptyset$ ;

7: **for**  $i = 1$  to  $|Y^p|$  **do**

8:  $v \leftarrow Y_k^p[i]$ ;

9:  $(\hat{y}, z) \leftarrow \text{solve}(P_2(v))$ ;

10:  $\bar{Y} \leftarrow \bar{Y} \cup \{\hat{y}\}$ ;

11: **if**  $z \neq 0$  **then**

12:  $(x, \delta) \leftarrow \text{solve}(R(v))$ ,  $(0 < \delta < 1)$ ;

13:  $y \leftarrow \delta v + (1 - \delta)\hat{p}$ ;

14:  $(u, w) \leftarrow \text{solve}(D_2(y))$ ;

15:  $\bar{T} \leftarrow \bar{T} \cup \{(u, w)\}$ ;

16: **endif**;

17: **endfor**;

18: **endwhile**

19 **Output:**  $(\bar{Y}, \bar{Y}^h)$  : Nondominated set and directions;  
 $\bar{T}$  : a solution to dual.

### Illustration of BOA

For continuity, we consider again problem (3) of the Illustration section. The nondominated points found using an existing Matlab implementation of Algorithm 4, namely `Bensolve-1.2` of Löhne,<sup>38</sup> 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 nondominated points are as shown in Figure 2.

### Selection of the most preferred nondominated point

To determine the Most Preferred Nondominated Point (MPNP), we employ the technique of compromise programming and compute the ideal objective point which

would serve as a reference point in each case. Compromise programming is a mathematical programming method that is based on the notion of distance of a most preferred solution from the ideal point  $y^*$ , Zeleny.<sup>44</sup> Ehrgott<sup>45</sup> note that the ideal point is an essential component of compromise programming, and the idea is to find a nondominated point which is as close as possible to it. This is a point in the objective space whose components are the optimal values of the objective functions when they are individually optimized Alves, Antunes and Climaco.<sup>46</sup> It was also noted in Zeleny<sup>44</sup> that the ideal point serves as a rationale directing and facilitating human choice and decision making. To find the ideal point, we simply solve  $q$  single objective problems

$$\begin{aligned} \min \quad & c_k^T x, \quad k = 1, 2, \dots, q \\ \text{subject to} \quad & x \in X. \end{aligned} \quad (4)$$

We note here that, the ideal point itself is not an element of the nondominated set ( $y^* \notin Y_N$ ). Otherwise, this would mean that the objective functions are not conflicting, but it always exists in the objective space. Its corresponding point in the decision space may not exist Alves, Antunes and Climaco.<sup>46</sup>

For our numerical illustration above (problem 3 of the Illustration section), solving each of the objective function individually over the feasible region  $X$  yields the ideal objective point  $y^* = (-7.0, -5.0)^T$ . Clearly  $y^* \notin Y_N$  where  $Y_N = \{(-7.0, -1.8)^T, (-1.6, -5.0)^T\}$ .

Having computed the ideal objective point  $y^*$ , we now determine the minimum distance of each nondominated point  $\hat{y}$  from it by finding

$$\min \{ \|\hat{y}_1 - y^*\|, \|\hat{y}_2 - y^*\|, \dots, \|\hat{y}_n - y^*\| \}$$

where  $\hat{y}_i \in Y_N$  has already been found either by BOA or EMSA,  $\|\cdot\|$  is the Euclidean norm on  $\mathbb{R}^q$  and  $y^*$  is the ideal objective point. Using the nondominated points  $f^1$  and  $f^2$  returned by BOA and EMSA for problem (3) yields

$$\|f^1 - y^*\| = 3.2 \quad \text{and} \quad \|f^2 - y^*\| = 5.4.$$

Since, the relative distance of  $f^1$  from the ideal point  $y^*$  is 3.2 which is the smallest of the two, it therefore means that  $f^1 = (-7.0, -1.80)^T$  is the closest of the two nondominated points to the ideal point  $y^* = (-7.0, -5.0)^T$ . Hence,  $f^1$  is selected as the DM's most preferred nondominated point.

Next, we measure the distance of the nondominated point  $f^1 = (-3.6418 - 3.7913)^T$  returned by ASIMOLP in the Illustration of ASIMOLP section for the same

numerical illustration (Problem 3 of the Illustration section) from the ideal point  $y^* = (-7.0, -5.0)^T$ , as was done with those returned by BOA and EMSA for the same example. It turned out that, the distance

$$\|f^1 - y^*\| = 3.5691$$

is bigger than 3.2 which was the closest when measuring the points returned by BOA and EMSA, thereby making the nondominated points returned by BOA and EMSA closer to the ideal point and of higher quality.

The following more substantial illustrative MOLP adapted from Zeleny<sup>44</sup> with three objectives makes the point.

$$\begin{aligned} \min f_1 &= -x_1 - 2x_2 + x_3 - 3x_4 - 2x_5 && -x_7 \\ \min f_2 &= && -x_2 - x_3 - 2x_4 - 3x_5 - x_6 \\ \min f_3 &= -x_1 && -x_3 + x_4 + x_6 + x_7 \end{aligned}$$

Subject to

$$\begin{aligned} x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 + 2x_7 &\leq 16 \\ -2x_1 - x_2 + x_4 + 2x_5 + x_7 &\leq 16 \\ -x_1 + x_3 + 2x_5 - 2x_7 &\leq 16 \\ x_2 + 2x_3 - x_4 + x_5 - 2x_6 - x_7 &\leq 16 \\ x_1, x_2, x_3, x_4, x_5, x_6, x_7 &\geq 0. \end{aligned} \quad (5)$$

Again, optimizing each of the objective functions individually over the feasible region yields the ideal objective point  $y^* = (-48.0, -32.0, -16.0)^T$ . Solving (5) with BOA and EMSA, the set of nondominated points found is  $Y_N = \{(-48.0, -32.0, 16.0)^T, \setminus\setminus(-16.0, 0.0, -16.0)^T, (0.0, -8.0, -16.0)^T, (-5.33, -21.33, -5.33)^T, (-16.0, -24.0, 0.0)^T\}$  with  $y^* \notin Y_N$ . By determining the minimum distance of each of these nondominated points from the ideal point  $y^*$ , it was found that the point  $(-48.0, -32.0, 16.0)^T$  is the closest. Its distance from it is 32. It is selected as the DMoi MPNP.

For problem (5), the MPNP returned by ASIMOLP is  $f^1 = (-7.65 - 13.80 - 7.75)^T$  as shown in Table 2, Problem 10. Again, we measure its distance from the ideal point  $y^* = (-48.0, -32.0, -16.0)^T$ . It was found that the distance of the point  $(-7.65 - 13.80 - 7.75)^T$  from  $y^* = (-48.0, -32.0, -16.0)^T$  is 45.0269, which is also larger than the corresponding values of BOA and EMSA, thereby making the MPNPs returned by BOA and EMSA to be of higher quality.

We have used this method to choose the MPNP from the nondominated sets returned by BOA and EMSA for comparison. There is no selection of a MPNP in ASIMOLP as the algorithm computes a most preferred efficient solution and also returns the corresponding most preferred nondominated point.

To determine the quality of the MPNP returned by ASIMOLP, we simply measure its distance from the ideal point in each case and compare with the distances of those returned by BOA and EMSA in order to determine that which is the closest to the ideal point and of higher quality. The way the preference or priority vector  $p$  needed in AHP is derived, is explained in the Determination of the priority vector used in ASIMOLP section.

## Experimental results

In this section, we provide numerical results to compare the quality of a Most Preferred Nondominated Point (MPNP) and the efficiency of Algorithms 2, 3 and 4. Table 2 shows the numerical results for a collection of 60 existing problems ranging from small to medium and realistic MOLP instances. Problem 1 is taken from Ehrgott,<sup>4</sup> and Problems 2 to 10 are from Zeleny.<sup>44</sup> Problems 11 to 20 are test problems from the interactive MOLP explorer (iMOLPe) of Alves, Antunes and Climaco.<sup>46</sup> Problems 21 to 46 are taken from Steuer.<sup>19</sup> Problem 47 is a test problem in Bensolve-1.2 of Löhne,<sup>38</sup> while problems 48 and 52 are test problems in Bensolve-2.0 of Löhne and Weißing.<sup>47</sup> Problems 49 to 51 are obtained using a script in Bensolve-2.0 of Löhne and Weißing<sup>47</sup> that was also used to generate problem 52 with the same number of variables and constraints. Finally, problems 53 to 60 are from MOPLIB which stands for Multi-Objective Problem Library and is maintained by Löhne and Schenker.<sup>48</sup>

Problem 47 is such that the constraint matrix is sparse while the criterion matrix is dense. The RHS vector is such that all the components are ones except for 200 at the end as the largest entry. Problem 48 has a dense constraint matrix with an identity matrix of order  $n$  as its criterion matrix where  $n$  is the number of variables in the problem. The RHS vector is such that all the components are zeros except for a one (1) at the beginning as the only none zero element. Problems 49 to 52 have dense criterion 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 elements in the RHS vectors are ones. Problem 53 is highly degenerate, its structure is such that, the constraint and criterion matrices are sparse while all the components of the RHS vector are zeros except for a one (1) as the only non-zero entry. Problem 54 has dense RHS vector while the constraint and criterion matrices are sparse. In Problems 55 and 58, the constraint matrices are sparse, the criterion matrices are dense and all the elements in the RHS vectors are ones. Problems 56 and 57 have sparse constraints and criterion matrices with dense RHS vectors. Problem 59

Table 2. Comparative results for individual problem.

Algorithm	EMSA				ASIMOLP			BOA			
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)
1	Ehrgott 2006	3	3	3	3	f1 = -2.00 f2 = 10.00 f3 = -5.00	0.169	f1 = -1.74 f2 = 5.56 f3 = -2.75	0.036	f1 = -2.00 f2 = 10.00 f3 = -5.00	0.038
2	Zeleny 1982	2	2	2	2	f1 = -25000.00 f2 = -66667.00	0.027	f1 = -30626.00 f2 = -64132.00	0.111	f1 = -25000.00 f2 = -66667.00	0.021
3	"	2	4	2	2	f1 = -9.00 f2 = -15.00	0.072	f1 = 4.00 f2 = -18.42	0.031	f1 = -9.00 f2 = -15.00	0.026
4	"	2	4	3	3	f1 = -3.00 f2 = -7.50 f3 = 4.00	0.182	f1 = -3.50 f2 = -2.74 f3 = 4.89	0.027	f1 = -3.00 f2 = -7.50 f3 = 4.00	0.161
5	"	2	6	2	2	f1 = -24.00 f2 = -16.00	0.399	f1 = -21.29 f2 = -17.29	0.032	f1 = -24.00 f2 = -16.00	0.212
6	"	3	3	3	3	f1 = 3.00 f2 = -6.00 f3 = -12.00	0.075	f1 = 1.33 f2 = -6.20 f3 = -9.68	0.028	f1 = 3.00 f2 = -6.00 f3 = -12.00	0.046
7	"	5	3	3	3	f1 = 0.00 f2 = -4.00 f3 = -24.00	0.041	f1 = -1.38 f2 = -2.77 f3 = -10.04	0.034	f1 = 0.00 f2 = -4.00 f3 = -24.00	0.043
8	"	5	2	2	2	f1 = -52.0 f2 = -52.0	0.055	f1 = -4.11 f2 = -29.30	0.019	f1 = -52.0 f2 = -52.0	0.016
9	"	6	4	2	2	f1 = 0.00 f2 = 0.00	0.229	f1 = -0.02 f2 = -0.00	0.043	f1 = 0.00 f2 = 0.00	0.017
10	"0	7	4	3	3	f1 = -48.00 f2 = -32.00 f3 = 16.00	0.317	f1 = -7.65 f2 = -13.80 f3 = -7.75	0.035	f1 = -48.00 f2 = -32.00 f3 = 16.00	0.163
11	iMOLPe	2	3	2	2	f1 = -21.00 f2 = -7.00	0.052	f1 = -11.87 f2 = -10.22	0.032	f1 = -21.00 f2 = -7.00	0.047
12	"2	3	3	4	4	f1 = -10.00 f2 = -20.00 f3 = -100.00	0.074	f1 = -5.59 f2 = -18.62 f3 = -34.83	0.037	f1 = -10.00 f2 = -20.00 f3 = -100.00	0.033
13	"3	3	5	3	3	f4 = -10.00 f1 = -25.00 f2 = 0.00	0.551	f4 = -42.23 f1 = -10.48 f2 = -3.62	0.035	f4 = -10.00 f1 = -25.00 f2 = 0.00	0.042
14	"4	3	3	3	3	f3 = -5.00 f1 = -2.66 f2 = -2.00	0.089	f3 = -2.14 f1 = -1.10 f2 = -1.22	0.041	f3 = -5.00 f1 = -2.66 f2 = -2.00	0.035
						f3 = -0.33		f3 = -1.57		f3 = -0.33	

(continued)

Table 2. Continued.

Algorithm	EMSA				ASIMOLP				BOA				
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)
15	"5		4	3	3	f1 = -48.50 f2 = -19.50 f3 = -37.00	0.097	f1 = -35.80 f2 = -43.97 f3 = -29.82	0.051	f1 = -48.50 f2 = -19.50 f3 = -37.00	0.038		
16	"6		4	2	3	f1 = -20.00 f2 = -80.00 f3 = -40.00	0.041	f1 = -31.71 f2 = -49.12 f3 = -38.69	0.046	f1 = -20.00 f2 = -80.00 f3 = -40.00	0.036		
17	"		4	4	3	f1 = -40.00 f2 = -50.00 f3 = -10.00	0.243	f1 = -32.22 f2 = 32.50 f3 = -36.27	0.041	f1 = -40.00 f2 = -50.00 f3 = -10.00	0.186		
18	"8		3	3	3	f1 = 0.00 f2 = -2.00 f3 = -4.00	0.178	f1 = -1.12 f2 = -2.14 f3 = 2.63	0.042	f1 = 0.00 f2 = -2.00 f3 = -4.00	0.033		
19	"9		15	10	2	f1 = -363.82 f2 = -33.70	1.581	f1 = -137.09 f2 = -198.96	0.142	f1 = -363.82 f2 = -33.70	0.195		
20	"0		15	10	3	f1 = -363.82 f2 = -33.70 f3 = -136.71	4.852	f1 = -107.15 f2 = -169.94 f3 = -16.26	0.223	f1 = -363.82 f2 = -33.70 f3 = -136.71	0.476		
21	Steuer 1986		5	5	2	f1 = -10.00 f2 = -3.00	0.197	f1 = -6.30 f2 = -6.90	0.033	f1 = -10.00 f2 = -3.00	0.036		
22	"2		4	4	3	f1 = 3.42 f2 = -10.28 f3 = -3.42	0.065	f1 = -3.79 f2 = 11.38 f3 = -2.96	0.035	f1 = 3.42 f2 = -10.28 f3 = -3.42	0.015		
23	"3		5	5	4	f1 = 1.02 f2 = -25.46 f3 = 24.44	1.432	f1 = 2.28 f2 = -22.58 f3 = 25.30	0.037	f1 = 1.02 f2 = -25.46 f3 = 24.44	0.098		
24	"4		10	8	4	f1 = 106.29 f2 = -462.13 f3 = 175.57	125.406	f1 = 80.00 f2 = -54.36 f3 = -163.73	0.048	f1 = 106.29 f2 = -462.13 f3 = 175.57	1.973		
25	"5		5	4	3	f1 = -52.07 f2 = 31.50 f3 = -17.35	0.196	f1 = -4.44 f2 = -13.17 f3 = -14.37	0.041	f1 = -52.07 f2 = 31.50 f3 = -17.35	0.054		
26	"6		6	8	4	f1 = -6.94 f2 = -5.38 f3 = 6.83	14.846	f1 = -6.69 f2 = -2.25 f3 = 6.77	0.045	f1 = -6.94 f2 = -5.38 f3 = 6.83	0.065		
27	"7		7	6	4	f1 = -31.53 f2 = -26.48	0.451	f1 = -25.90 f2 = -23.94	0.032	f1 = -31.53 f2 = -26.48	0.286		

(continued)

Table 2. Continued.

Algorithm	EMSA				ASIMOLP				BOA					
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)	
28	"8	7	6	4	f3 = -26.57	1.601	f3 = -19.06	0.033	f3 = -26.57	0.192	f3 = -26.57	0.192	f4 = -0.34	f4 = -8.62
					f4 = -0.34		f4 = -8.62		f4 = -0.34		f4 = -8.62			
					f1 = 26.80		f1 = 4.03		f1 = 26.80		f1 = 4.03		f1 = 26.80	f1 = 4.03
					f2 = -37.73		f2 = -29.03		f2 = -37.73		f2 = -29.03		f2 = -37.73	f2 = -29.03
29	"9	8	8	6	f3 = -24.33	36.125	f3 = -18.07	0.084	f3 = -24.33	73.963	f3 = -24.33	73.963	f4 = -59.60	f4 = -28.17
					f4 = -59.60		f4 = -28.17		f4 = -59.60		f4 = -28.17			
					f1 = -74.00		f1 = -15.46		f1 = -74.00		f1 = -15.46		f1 = -74.00	f1 = -15.46
					f2 = -107.50		f2 = -38.73		f2 = -107.50		f2 = -38.73		f2 = -107.50	f2 = -38.73
30	"0	8	8	3	f3 = -41.25	2.726	f3 = -43.30	0.036	f3 = -41.25	0.168	f3 = -41.25	0.168	f4 = -27.25	f4 = -30.95
					f4 = -27.25		f4 = -30.95		f4 = -27.25		f4 = -30.95			
					f5 = -9.00		f5 = -8.30		f5 = -9.00		f5 = -8.30		f5 = -9.00	f5 = -8.30
					f6 = -30.75		f6 = -26.72		f6 = -30.75		f6 = -26.72		f6 = -30.75	f6 = -26.72
31	"1	8	8	3	f1 = -36.57	0.706	f1 = -32.03	0.036	f1 = -36.57	0.135	f1 = -36.57	0.135	f2 = -22.28	f2 = -20.03
					f2 = -22.28		f2 = -20.03		f2 = -22.28		f2 = -20.03			
					f3 = -14.00		f3 = -17.73		f3 = -14.00		f3 = -17.73		f3 = -14.00	f3 = -17.73
					f1 = -14.03		f1 = -8.77		f1 = -14.03		f1 = -8.77		f1 = -14.03	f1 = -8.77
32	"2	5	5	4	f2 = -18.00	0.618	f2 = -10.56	0.049	f2 = -18.00	0.277	f2 = -18.00	0.277	f3 = -4.93	f3 = -5.13
					f3 = -4.93		f3 = -5.13		f3 = -4.93		f3 = -5.13			
					f1 = -21.50		f1 = -20.83		f1 = -21.50		f1 = -20.83		f1 = -21.50	f1 = -20.83
					f2 = -39.25		f2 = -21.78		f2 = -39.25		f2 = -21.78		f2 = -39.25	f2 = -21.78
33	"3	6	6	3	f3 = -16.25	3.907	f3 = -16.05	0.046	f3 = -16.25	0.212	f3 = -16.25	0.212	f4 = 27.00	f4 = 14.45
					f4 = 27.00		f4 = 14.45		f4 = 27.00		f4 = 14.45			
					f1 = -12.65		f1 = 12.69		f1 = -12.65		f1 = 12.69		f1 = -12.65	f1 = 12.69
					f2 = 0.00		f2 = -3.21		f2 = 0.00		f2 = -3.21		f2 = 0.00	f2 = -3.21
34	"4	5	5	4	f3 = -30.15	2.016	f3 = -28.39	0.033	f3 = -30.15	0.183	f3 = -30.15	0.183	f1 = -14.66	f1 = -6.33
					f1 = -14.66		f1 = -6.33		f1 = -14.66		f1 = -6.33			
					f2 = -21.06		f2 = -14.44		f2 = -21.06		f2 = -14.44		f2 = -21.06	f2 = -14.44
					f3 = 35.73		f3 = 20.77		f3 = 35.73		f3 = 20.77		f3 = 35.73	f3 = 20.77
35	"5	10	10	4	f4 = -16.00	325.555	f4 = -14.63	0.057	f4 = -16.00	0.333	f4 = -16.00	0.333	f1 = 46.50	f1 = 50.69
					f1 = 46.50		f1 = 50.69		f1 = 46.50		f1 = 50.69			
					f2 = 19.21		f2 = 18.98		f2 = 19.21		f2 = 18.98		f2 = 19.21	f2 = 18.98
					f3 = -27.07		f3 = -23.38		f3 = -27.07		f3 = -23.38		f3 = -27.07	f3 = -23.38
36	"6	8	8	3	f4 = -27.07	13.251	f4 = -23.85	0.042	f4 = -27.07	0.217	f4 = -27.07	0.217	f1 = -14.48	f1 = -2.46
					f1 = -14.48		f1 = -2.46		f1 = -14.48		f1 = -2.46			
					f2 = -4.74		f2 = -3.22		f2 = -4.74		f2 = -3.22		f2 = -4.74	f2 = -3.22
					f3 = 6.93		f3 = -1.93		f3 = 6.93		f3 = -1.93		f3 = 6.93	f3 = -1.93

(continued)

Table 2. Continued.

Algorithm	EMSA				ASIMOLP				BOA				
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)
37	"7		6	7	4	f1 = -2.61 f2 = -12.63 f3 = 9.70 f4 = -2.37	6.051	f1 = -1.80 f2 = -4.00 f3 = 2.78 f4 = -2.07	0.039	f1 = -2.61 f2 = -12.63 f3 = 9.70 f4 = -2.37	0.386		
38	"8		12	16	4	Aborted after 3 days*	-	f1 = -5.09 f2 = -9.83 f3 = -9.53 f4 = -6.18	0.062	f1 = -5.25 f2 = -14.25 f3 = -8.25 f4 = -1.00	31.034		
39	"9		10	14	5	*	-	f1 = -1.07 f2 = -3.83 f3 = -5.53 f4 = -16.87	0.051	f1 = -5.16 f2 = -2.79 f3 = -4.38 f4 = -18.70	102.952		
40	"0		7	6	3	f1 = -29.40 f2 = -65.30 f3 = -39.30	0.447	f1 = -10.74 f2 = -32.20 f3 = -24.39	0.044	f1 = -29.40 f2 = -65.30 f3 = -39.30	0.165		
41	"1		7	7	3	f1 = -62.18 f2 = -93.50 f3 = -52.00	0.583	f1 = -47.39 f2 = -86.99 f3 = -54.78	0.039	f1 = -62.18 f2 = -93.50 f3 = -52.00	0.036		
42	"2		6	6	4	f1 = -37.50 f2 = -11.25 f3 = -7.50 f4 = -20.25	1.311	f1 = -10.40 f2 = -6.52 f3 = -5.91 f4 = -0.34	0.032	f1 = -37.50 f2 = -11.25 f3 = -7.50 f4 = -20.25	0.158		
43	"3		6	6	4	f1 = 34.50 f2 = -7.50 f3 = -56.00 f4 = -31.50	1.856	f1 = 28.39 f2 = -6.38 f3 = -45.43 f4 = -25.83	0.029	f1 = 34.50 f2 = -7.50 f3 = -56.00 f4 = -31.50	0.211		
44	"4		10	14	5	*	-	f1 = 3.35 f2 = -3.18 f3 = -2.48 f4 = -2.50 f5 = 2.10	0.043	f1 = 1.03 f2 = -2.19 f3 = 2.01 f4 = -8.13 f5 = -7.22	307.611		
45	"5		10	14	5	*	-	f1 = 2.35 f2 = 0.73 f3 = -11.72 f4 = -1.90 f5 = -10.33	0.057	f1 = -4.95 f2 = -3.42 f3 = -4.38 f4 = -18.91 f5 = -9.27	105.344		
46	"6		7	7	3	f1 = -3.83 f2 = -76.46	0.601	f1 = -6.82 f2 = -68.93	0.038	f1 = -3.83 f2 = -76.46	0.045		

(continued)

Table 2. Continued.

Algorithm	EMSA				ASIMOLP				BOA				
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)
47	Bensolve 1.2	100	101	2	2	f3 = -49.57 *	-	f3 = -25.08 f1 = -4.89 f2 = -106.50	0.037	f3 = -49.57 f1 = -8.42 f2 = -116.65	0.503		
48	Bensolve 2.0	5	31	5	5	*	-	f1 = 0.00 f2 = 0.00 f3 = 0.00 f4 = 0.00 f5 = 0.01	0.021	f1 = 0.00 f2 = -1.00 f3 = 0.00 f4 = 0.00 f5 = -2.00	2.877		
49	Script of Prob. 52	36	36	2	2	f1 = -5.00 f2 = -26.00	1.921	f1 = -8.41 f2 = -20.12	0.018	f1 = -5.00 f2 = -26.00	0.211		
50	"0	64	64	2	2	f1 = -63.00 f2 = -7.00	14.455	f1 = -52.24 f2 = -10.56	0.028	f1 = -63.00 f2 = -7.00	0.403		
51	"1	100	100	2	2	f1 = -124.00 f2 = -9.00	119.821	f1 = -119.51 f2 = -11.72	0.031	f1 = -124.00 f2 = -9.00	0.621		
52	Bensolve 2.0	343	343	3	3	Out of memory <sup>x</sup>	-	f1 = -35.32 f2 = -101.41 f3 = -9.63	0.651	f1 = -42.00 f2 = -294.00 f3 = -6.00	55.302		
53	MOPLIB	30	21	12	12	f1 = 5.05X10-12, f2 = 5.02X10-12, f3 = 5.03X10-12, f4 = 5.05X10-12, f5 = 5.05X10-12, f6 = 5.05X10-12, f7 = 5.05X10-12, f8 = 5.04X10-12, f9 = 5.06X10-12, f10 = 5.04X10-12, f11 = 5.04X10-12, f12 = -5.54X10-11	7.235	No initial starting solution	-	f1 = 5.05X10-12, f2 = 5.02X10-12, f3 = 5.03X10-12, f4 = 5.05X10-12, f5 = 5.05X10-12, f6 = 5.05X10-12, f7 = 5.05X10-12, f8 = 5.04X10-12, f9 = 5.06X10-12, f10 = 5.04X10-12, f11 = 5.04X10-12, f12 = -5.54X10-11	0.598		
54	"4	4492	1003	4	4	x	-	f1 = 42.70 f2 = 83.10 f3 = 0.00 f4 = -725.90	18.281	x	-		
55	"5	100	20	3	3	f1 = -168.00 f2 = -124.00 f3 = -143.00	5.281	f1 = -61.18 f2 = -73.93 f3 = -96.38	0.011	f1 = -168.00 f2 = -124.00 f3 = -143.00	4.291		
56	"6	53	221	3	3	*	-	f1 = -19.23 f2 = -54.56 f3 = 0.00	0.725	f1 = 0.00 f2 = -2.00 f3 = -13959.00	1663.803		

(continued)



Table 2. Continued.

Algorithm	EMSA				ASIMOLP			BOA			
	Prob.	Origin	n	m	q	MPNP	CPU (s)	MPNP	CPU (s)	MPNP	CPU (s)
57	"7	53	226	226	3	f1 = -188.00 f2 = -123.00 f3 = 16842.00	28.116	f1 = -50.22 f2 = -47.01 f3 = 0.00	-	f1 = -188.00 f2 = -123.00 f3 = 16842.00	6.551
58	"8	900	60	60	4	*	-	f1 = -283.50 f2 = -352.74 f3 = -268.85 f4 = -372.16	0.262	*	-
59	"9	1143	1211	1211	3	x	-	f1 = 2.11 f2 = 51.82 f3 = 0.61	0.725	x	-
60	"0	218	28	28	27	*	-	f1 = 0.52, f2 = 0.01, f3 = 6.95 f4 = 2.41, f5 = 4.94, f6 = -5.53 f7 = -9.71, f8 = 2.23, f9 = -0.31 f10 = 2.67, f11 = -3.19, f12 = -5.55 f13 = -5.42, f14 = -4.46, f15 = -4.35 f16 = 6.01, f17 = -3.36, f18 = 1.71 f19 = -8.01, f20 = 8.90, f21 = 8.01 f22 = -5.35, f23 = 5.35, f24 = 5.35 f25 = 5.35, f26 = -5.37, f27 = -4.35	0.481	*	-

\*Aborted after 3 days of running time.

xOut of memory.

has dense RHS vector while the constraint and criterion matrices are sparse. Finally, Problem 60 is such that the constraint and criterion matrices are sparse while the components of the RHS vector are all zeros except for a ninety (90) at the end as the only non-zero entry.

We modified and extended Algorithm 1 of Evans and Steuer<sup>2</sup> into Algorithm 2 or EMSA the Extended Multiobjective Simplex Algorithm. We have implemented it in Matlab in the same way as in Rudloff, Ulus and Vanderbei<sup>23</sup> and experimented with it on a set of MOLP in Matlab in the same way as Algorithm 3 in Matlab and used an existing Matlab implementation of Algorithm 4, known as Bensolve-1.2 of Löhne.<sup>38</sup> The current version, Bensolve-2.0 of Löhne and Weißing<sup>47</sup> is implemented in the C programming language. We employed Bensolve-1.2 of Löhne<sup>38</sup> which is implemented in Matlab to test the algorithms with the same tools and for more meaningful comparisons. In all tests,  $m$  is the number of constraints,  $n$  the number of variables and  $q$  the number of objectives. Algorithm 3 is ASIMOLP of Arbel<sup>3</sup> and Algorithm 4 is BOA of Benson<sup>1</sup> as presented in Shao and Ehrgott.<sup>36</sup> All algorithms were executed on an Intel Core i5-2500 CPU at 3.30 GHz with 16.0GB RAM.

We recorded the CPU times (in seconds) for each problem and acted as the DM by choosing a most preferred (best) nondominated point (whose components are as close as possible to the ideal objective point as explained in the Selection of the most preferred nondominated point section) from the nondominated set  $Y_N = \{Cx : x \in X_E\}$  returned by BOA to compare with the MPNP returned by EMSA and with that returned by ASIMOLP.

As can be seen from Table 2, the CPU times for all algorithms increase as the problem sizes increase. It was observed that ASIMOLP returns a CPU time of less than a second for most of the test problems it solves, thereby making it computationally more efficient than BOA and EMSA. However, BOA was found to be computationally more efficient than EMSA for all the test problems considered. We noticed that ASIMOLP did not solve problems 53 as there exists no initial and strictly positive starting solution ( $x^0 > 0$  such that  $Ax^0 = b$ ) due to singularity issues which indicates that either the initial solution does not exist or it is not unique. We suspect that the difficulty this problem pose to ASIMOLP is due to its matrix structure and the way Interior Point methods work. A diagonal matrix whose diagonal elements are the elements of an initial positive starting solution is required. Once the diagonal elements are not strictly positive ( $x^0 > 0$ ), ASIMOLP exhibits this difficulty. We even employed a decomposition approach which did not yield the required initial positive starting solution.

In terms of the quality of a MPNP returned by the algorithms, it was observed that EMSA and BOA return the same MPNPs for all test problems considered. This makes these two algorithm comparable and the nondominated points they returned are of higher quality than those returned by ASIMOLP in all cases. We also observed in Table 2 that EMSA and BOA could not produce results for some of the test problems considered despite the long running time allowed (3 days); they were aborted. 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 would potentially return a huge number of nondominated points or run out of memory which would indicate that the total number of nondominated points has exceeded the Matlab solution capacity of the machine used. We note here that, some of these problems most especially from problem 47 to 60 are numerically ill-posed and highly challenging MOLP instances with difficult structures.

## Conclusion

We have reviewed the extensive literature on three iconic algorithms namely MSA, ASIMOLP and BOA. We have extended MSA to find the set of nondominated points and presented the algorithms as well as illustrated them on small MOLP instances. We then proceeded to investigate their computational efficiency and compare the quality of a most preferred nondominated point they returned on a collection of 60 existing problems ranging from small to moderate and large MOLP instances. It was observed that Benson's BOA and EMSA, introduced here, are superior to ASIMOLP in terms of the quality of the most preferred nondominated point they returned. The measure of quality used is the distance to the ideal point as explained in the Selection of the most preferred nondominated point section. ASIMOLP, however, outperforms them in terms of computing efficiency. Moreover, on the test problems considered, BOA was also found to be computationally superior to EMSA.


## Declaration of Conflicting Interests

The author(s) declared no potential conflicts of interest with respect to the research, authorship, and/or publication of this article.

## Funding

The author(s) disclosed receipt of the following financial support for the research, authorship, and/or publication of this article: We are grateful to ESRC, Grant ES/L011859/1, for partially funding this research.

**ORCID iD**

Paschal B Nyiam  <https://orcid.org/0000-0003-2535-3526>

**References**

- Benson HP. An outer approximation algorithm for generating all efficient extreme points in the outcome set of a multiple objective linear programming problem. *J Glob Optim* 1998; 13: 1–24.
- Evans JP and Steuer R. A revised simplex method for linear multiple objective programs. *Math Program* 1973; 5: 54–72.
- Arbel A. A weighted-gradient approach to multi-objective linear programming problems using the analytic hierarchy process. *Math Comput Modell* 1993; 17: 27–39.
- Ehrgott M. *Multicriteria optimization*. Berlin: Springer Science & Business Media, 2006.
- Benson HP. Hybrid approach for solving multiple-objective linear programs in outcome space. *J Optim Theory Appl* 1998; 98: 17–35.
- Alves MJ and Costa JP. An exact method for computing the nadir values in multiple objective linear programming. *Eur J Oper Res* 2009; 198: 637–646.
- Junior H and Lins MPE. A WIN-WIN approach to multiple objective linear programming problems. *J Oper Res Soc* 2009; 60: 728–733.
- Eiselt HA and Sandblom CL. *Linear programming and its applications*. Berlin: Springer Science & Business Media, 2007.
- Philip J. Algorithms for the vector maximization problem. *Math Program* 1972; 2: 207–229.
- Zeleny M. *Linear multiobjective programming*. vol. 95. Berlin: Springer-Verlag, 1974.
- Philip J. Vector maximization at a degenerate vertex. *Math Program* 1977; 13: 357–359.
- Steuer RE. *ADBASE: a multiple objective linear programming solver for all efficient extreme points and all unbounded efficient edges*. Athens: Terry College of Business, University of Georgia, 2003.
- Yu P and Zeleny M. The techniques of linear multiobjective programming. *Rairo Recherche Opérationnelle* 1974; 8: 51–71.
- Yu P and Zeleny M. The set of all nondominated solutions in linear cases and a multicriteria simplex method. *J Math Anal Appl* 1975; 49: 430–468.
- Yu P and Zeleny M. Linear multiparametric programming by multicriteria simplex method. *Manage Sci* 1976; 23: 159–170.
- Isermann H. The enumeration of the set of all efficient solutions for a linear multiple objective program. *J Oper Res Soc* 1977; 28: 711–725.
- Isermann H and Naujoks G. *Operating manual for the EFFACET multiple objective linear programming package*. Bielefeld, Germany: Fakultät fuer Wirtschaftswissenschaften, University of Bielefeld, 1984.
- Gal T. A general method for determining the set of all efficient solutions to a linear vectormaximum problem. *Eur J Oper Res* 1977; 1: 307–322.
- Steuer RE. *Multiple criteria optimization: theory, computation, and applications*. Hoboken: Wiley; 1986.
- Ecker J and Kouada I. Finding all efficient extreme points for multiple objective linear programs. *Math Program* 1978; 14: 249–261.
- Ecker J, Hegner NS and Kouada I. Generating all maximal efficient faces for multiple objective linear programs. *J Optim Theory Appl* 1980; 30: 353–381.
- Armand P and Malivert C. Determination of the efficient set in multiobjective linear programming. *J Optim Theory Appl* 1991; 70: 467–489.
- Rudloff B, Ulus F and Vanderbei R. A parametric simplex algorithm for linear vector optimization problems. *Math Program* 2017; 163: 213–230.
- Löhne A. *Vector optimization with infimum and supremum*. Berlin: Springer Science & Business Media, 2011.
- Schechter M and Steuer RE. A correction to the connectedness of the Evans-Steuer algorithm of multiple objective linear programming. *Found Comput Decis Sci* 2005; 30: 351–360.
- Karmarkar N. A new polynomial-time algorithm for linear programming. In: *Proceedings of the sixteenth annual ACM symposium on theory of computing*, 1984, pp.3024ompu New York: ACM.
- Abhyankar S, Morin T and Trafalis T. Efficient faces of polytopes: interior point algorithms, parametrization of algebraic varieties, and multiple objective optimization. *Contemp Math* 1990; 114: 319–341.
- Arbel A. An interior multiobjective linear programming algorithm. *Comput Oper Res* 1993; 20: 723–735.
- Saaty TL. *The analytic hierarchy process: planning, priority setting, resources allocation*. New York: McGraw, 1980.
- Arbel A. Anchoring points and cones of opportunities in interior multiobjective linear programming. *J Oper Res Soc* 1994; 45: 83–96.
- Wen UP and Weng WT. An interior algorithm for solving multiobjective linear programming problem. In: *Institute for operations research and the management sciences international meeting*, Tel Aviv, Israel, 1998.
- Lin C, Chen C and Chen P. On the modified interior point algorithm for solving multi-objective linear programming problems. *Int J Inform Manag Sci* 2006; 17: 107.
- Weng WT and Wen UP. An interior point algorithm for solving linear optimization over the efficient set problems. *J Chin Inst Ind Eng* 2001; 18: 21–30.
- Benson HP. Further analysis of an outcome set-based algorithm for multiple objective linear programming. *J Optim Theory Appl* 1998; 97: 1–10.
- Ban VT. 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*, 1983.
- Shao L and Ehrgott M. Approximately solving multiobjective linear programmes in objective space and an application in radiotherapy treatment planning. *Math Meth Oper Res* 2008; 68: 257–276.
- Shao L and Ehrgott M. Approximating the nondominated set of an MOLP by approximately solving its

- dual problem. *Math Methods Oper Res* 2008; 68: 469–492.
38. Löhne A. Bensolve: VLP solver, version 1.2, [www.bensolve.org](http://www.bensolve.org) (2012, accessed 14 May 2021).
39. Csirmaz L. Using multiobjective optimization to map the entropy region. *Comput Optim Appl* 2016; 63: 45–67.
40. Hamel AH, Löhne A and Rudloff B. Benson type algorithms for linear vector optimization and applications. *J Glob Optim* 2014; 59: 811–836.
41. Löhne A, Rudloff B and Ulus F. Primal and dual approximation algorithms for convex vector optimization problems. *J Glob Optim* 2014; 60: 713–736.
42. Arbel A. Using efficient anchoring points for generating search directions in interior multiobjective linear programming. *J Oper Res Soc* 1994; 45: 330–344.
43. Saaty T and Vargas LG. *The logic of priorities: application in business, energy, health, and transportation*. Boston: Nijhoff, 1982.
44. Zeleny M. *Multiple criteria decision making*. New York: McGraw-Hill, 1982.
45. Ehrgott M and Tenfelde-Podehl D. Computation of ideal and nadir values and implications for their use in MCDM methods. *Eur J Oper Res* 2003; 151: 119–139.
46. Alves MJ, Antunes CH and Clímaco J. Interactive MOLP explorer: a graphical-based computational tool for teaching and decision support in multi-objective linear programming models. *Comput Appl Eng Educ* 2015; 23: 314–326.
47. Löhne A and Weißing B. Bensolve: VLP solver, version 2.0.x, [www.bensolve.org](http://www.bensolve.org) (2015, accessed 14 May 2021).
48. Löhne A and Schenker S. MOPLIB: multi-objective problem library, <http://moplib.uni-jena.de> (2015, accessed 13 March 2017).