A comparative study of two key algorithms in multiple objective linear programming

Paschal B Nyiam and Abdellah Salhi

Abstract
Multiple objective linear programming problems are solved with a variety of algorithms. While these algorithms vary in philosophy and outlook, most of them fall into two broad categories: those that are decision space-based and those that are objective space-based. This paper reports the outcome of a computational investigation of two key representative algorithms, one of each category, namely the parametric simplex algorithm which is a prominent representative of the former and the primal variant of Bensons Outer-approximation algorithm which is a prominent representative of the latter. The paper includes a procedure to compute the most preferred nondominated point which is an important feature in the implementation of these algorithms and their comparison. Computational and comparative results on problem instances ranging from small to medium and large are provided.

Keywords
Multiple objective linear programming, parametric simplex algorithm, outer-approximation algorithm, most preferred nondominated point, multiple criteria decision making

Received 8 May 2019; accepted 26 March 2019

Introduction
Multiple objective linear programming (MOLP) is a branch of multiple criteria decision making (MCDM)\(^{32,33}\) that seeks to optimize two or more linear objective functions subject to linear constraints. Indeed, many real-world decision-making problems involve more than one objective function and can be formulated as MOLP problems. MOLP models have been widely applied in many fields of human endeavour such as science, engineering and management and have become a useful tool in decision making. An MOLP problem can be expressed as:

\[
\text{min } c^T x = f_1 \\
\vdots \\
\text{subject to } x \in X = \{ x \in \mathbb{R}^n : Ax = b, b \in \mathbb{R}^m, x \geq 0 \} \\
\text{or alternatively as a linear vector optimization problem}
\]

\[
\begin{align*}
\min & \quad Cx \\
\text{subject to } & \quad Ax = b \\
& \quad x \geq 0
\end{align*}
\]

where \(C\) is a \(q \times n\) criterion matrix consisting of the rows \(c_k, k = 1, \ldots, q\), \(A\) 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 \(X\). The upper image is defined as \(Y + \mathbb{R}^q_+\).

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

Corresponding author:
Paschal B Nyiam, Department of Mathematical Sciences, University of Essex, Wivenhoe Park, Colchester CO4 3SQ, UK. Email: pbnyia@essex.ac.uk

Creative Commons CC BY: This article is distributed under the terms of the Creative Commons Attribution 4.0 License (http://www.creativecommons.org/licenses/by/4.0/) which permits any use, reproduction and distribution of the work without further permission provided the original work is attributed as specified on the SAGE and Open Access pages (https://us.sagepub.com/en-us/nam/open-access-at-sage).
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 points or a subset of either or a most preferred point depending on the purpose for which it is needed.

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 no other solution can improve all the objective functions simultaneously.

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. Let $\tilde{x} \in X$ be a feasible solution of equation (2) and let $\tilde{y} = C\tilde{x}$:

- $\tilde{x}$ is called efficient if there is no $x \in X$, such that $Cx \leq C\tilde{x}$ and $Cx \neq C\tilde{x}$; correspondingly, $\tilde{y} = C\tilde{x}$ is called nondominated.
- $\tilde{x}$ is called weakly efficient if there is no $x \in X$, such that $Cx < C\tilde{x}$; and $\tilde{y} = C\tilde{x}$ is called weakly nondominated.1

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

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

Robustness of method can be defined in different ways, in terms of computing efficiency or the ability of a method to solve problems depending on the researcher. In this paper, we consider it as the ability of a method to solve all problems (both simple and difficult).

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 $X$.2

MOLP has been an active area of research since the 1960s. During this period, various algorithms have been developed for generating either the entire efficient or nondominated set, or a subset of it, or a most preferred efficient or nondominated point for the problem. Most of these approaches are decision space-based. However, objective space-based methods are becoming more and more prominent.

In this paper, we are concerned with the state-of-the-art technology for MOLP. We are interested in a detailed comparison of the recently introduced parametric simplex algorithm (PSA) of Rudloff et al.3 with the primal variant of Benson’s4 outer-approximation algorithm (BOA) which is an objective space-based method. It has been noted by Benson4 that, in practice, the DM prefers to base his or her choice of a most preferred (best) solution on the nondominated points. To achieve this comparison, we shall act here 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 PSA to compare with a MPNP returned by BOA. These two algorithms are based on the same solution concept introduced by Löhne.5 The algorithms will be compared comprehensively on a series of existing test problems.

One of the key issues in MOLP is computing the MPNP’s. We give a detailed procedure for the purpose here which allows us to carry out the comparison.

This paper is organized as follows: ‘Motivation’ section is the motivation. ‘Literature review’ section is a review of the related literature which centres on the parametric simplex and the objective space-based method that is BOA. We present PSA in ‘The parametric simplex algorithm’ section. ‘Scalarization techniques’ section discusses two scalarization techniques. BOA is presented in ‘Benson’s outer-approximation algorithm’ section. ‘Selection of the MPNP’ section discusses the selection of an MPNP. Detailed numerical experiments are presented in ‘Experimental results’ section to compare the quality of a MPNP, robustness and computing efficiency of the two algorithms. The results are summarized in the ‘Summary of results’ section. Finally, a conclusion is drawn in ‘Conclusion’ section.

Motivation

These two algorithms are similar in output but different in philosophy since one of them, BOA, is an objective space-based search algorithm, while PSA is a decision space-based algorithm. They are prominent examples of MOLP algorithms. Unfortunately, there is no empirical evidence in the literature, as far as we can tell, that separates them in terms of robustness and the quality of a MPNP they returned. This paper intends to fill this gap. It considers all available test problems ranging from small size to large size and
efficient extreme points are obtained using a test problem. All the efficient extreme points have been found. The algorithm moves from one efficient extreme point to adjacent efficient extreme points until all the efficient extreme points have been found. The vertices are tested for efficiency as soon as they are generated by solving an LP that determines the efficiency of such vertices.

Yu and Zeleny\textsuperscript{10} presented two basic forms of parametric linear programming approaches and their computational procedures for computing the efficient set; the direct decomposition of the parametric space that was earlier introduced in Zeleny\textsuperscript{9} and the indirect algebraic method. Based on a numerical experiment, the indirect algebraic method outperforms the direct decomposition method.

A parametric linear programming algorithm for generating the set of efficient vertices and higher-dimensional faces of the problem was presented by Gal\textsuperscript{11}. In this procedure, efficient extreme points are generated through the use of an auxiliary problem which itself is an ordinary LP. The algorithm also determines higher-dimensional efficient faces for degenerate problems which were only discussed in Zeleny\textsuperscript{9} but not solved. The efficient faces are generated following a bottom-up search strategy, that is, they are generated based on the information provided by the efficient extreme points.

Steuer\textsuperscript{12} applied the MSA of Evans and Steuer\textsuperscript{8} to parametric MOLP problems. Different methods for obtaining an initial efficient basis as well as different LP test problems were also presented. Similarly, Ehrgott\textsuperscript{1} used this MSA variant to solve parametric MOLP problems.

A modification of the PSA for single objective LP to solve bounded bicriterion LP problems was presented by Ruszczyński and Vanderbei\textsuperscript{13}. The approach was applied to a large mean-risk portfolio optimization problem for which the nondominated portfolios were generated.

Ehrgott et al.\textsuperscript{14} introduced a primal-dual simplex algorithm for bounded problems. This algorithm finds a subset of efficient solutions that are enough to generate the whole efficient frontier. The algorithm starts with a coarse partitioning of the weight space which continues in each iteration as well as solves a costly LP in each iteration. A vertex enumeration is then performed in the last step to obtain efficient solutions. Numerical illustrations show the applicability of the algorithm.

Recently, Rudloff et al.\textsuperscript{3} presented a PSA for the problem. The algorithm is a generalization of the algorithm of Ruszczyński and Vanderbei\textsuperscript{14} and is similar to that of Ehrgott et al.\textsuperscript{14} It works for any dimension, solves bounded and unbounded problems (unlike that of Ehrgott et al.\textsuperscript{14} and Ruszczyński and Vanderbei\textsuperscript{13}) and does not find all the efficient solutions just like that.
of Ehrrog et al.\textsuperscript{14} Instead, it finds a solution based on the idea of Löhne,\textsuperscript{5} i.e. a subset of efficient extreme points and directions that allows to generate the whole efficient frontier. This is the so-called PSA. It was compared with a version of BOA in Hamel et al.\textsuperscript{15} and MSA of Evans and Steuer\textsuperscript{8} using small MOLP instances which were randomly generated with three and four objectives and up to 50 variables and constraints. The numerical results show that the proposed algorithm outperforms Benson’s algorithm for non-degenerate problems. However, Benson’s algorithm is better for highly degenerate problems. PSA was also found to be computationally more efficient than the algorithm of Evans and Steuer.\textsuperscript{15} This comparison only focused on computing efficiency. Here, apart from computing efficiency, we will also compare the robustness and quality of MPNP’s returned by these two methods using existing and realistic MOLP instances.

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,\textsuperscript{4} 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 was presented in Benson.\textsuperscript{16} 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\textsuperscript{17} 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.\textsuperscript{18} The algorithm is quite similar to that in Benson.\textsuperscript{4} The difference is in the manner in which the nondominated vertices are found. While a vertex enumeration procedure is employed in Benson,\textsuperscript{4} a simplicial partitioning technique is used in the latter.

In Shao and Ehrrog,\textsuperscript{19} a modification of the algorithm of Benson\textsuperscript{4} was presented. While in Benson,\textsuperscript{4} 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. Shao and Ehrrog\textsuperscript{20} proposed an approximate dual variant of the algorithm of Benson\textsuperscript{4} for obtaining approximate nondominated points of 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\textsuperscript{4} as modified by Shao and Ehrrog\textsuperscript{19} is presented in Löhne.\textsuperscript{5} This version solves two LPs in each iteration during the process of obtaining nondominated points and is extended to unbounded problems. Löhne\textsuperscript{21} developed 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\textsuperscript{22} presented an improved version of the algorithm of Benson\textsuperscript{4} that solves one LP and a vertex enumeration problem in each iteration. While in Benson,\textsuperscript{4} 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 with 3 and 10 objectives and up to 5772 variables and 635 constraints.

Hamel et al.\textsuperscript{15} introduced new versions and extensions of the algorithm of Benson.\textsuperscript{3} The primal and dual variants of the algorithm solve only one LP problem in each iteration and is extended to pointed solid polyhedral ordering cones. Tests reveal a reduction in computation time. Similarly, Löhne et al.\textsuperscript{23} extended the primal and dual variants of this algorithm to solve convex vector optimization problems approximately in the objective space.

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

The parametric simplex algorithm

The PSA of Rudloff et al.\textsuperscript{3} is one of the current solution approaches for MOLP. It can be viewed as a variant of the algorithm of Evans and Steuer,\textsuperscript{9} with a similar structure. It is different in the sense that it does not find all the efficient extreme points and unbounded efficient edges (extreme rays) as is being
done in Evans and Steuer. As mentioned earlier, the algorithm works in the decision space and finds a solution based on the idea of Löhne; i.e., it finds a finite subset of efficient extreme points and directions that allows to generate the whole efficient frontier. The algorithm is initialized by solving an LP to find a weight vector, such that the weighted sum problem using this weight vector yields an optimal solution. The corresponding optimal dictionary (containing basic and nonbasic variables) is used to construct an initial dictionary $D^0$ and an index set of entering variables $J^D_0$. The optimal solution is then used as an initial efficient basic feasible solution $\bar{x}_0$. Its implementation stores a set of Boundary Dictionaries ($BD$) containing dictionaries that are not yet visited and a set of Visited Dictionaries ($VD$) that contains dictionaries that are already visited. At each iteration, the algorithm moves from one dictionary to another, collecting their basic solutions into a set of efficient solutions $\bar{X}$. When all the dictionaries are visited, the algorithm stops and returns the set of efficient extreme points and directions that would be eliminated which also improves the computational time. The indices $i, j$ correspond to basic variable $x_i \in B$ and nonbasic variable $x_j \in N$, respectively

- $B^{-1}$ the inverse of the basic matrix
- $D$ the new dictionary
- $E^0$ the set of explored pivots for the current dictionary
- $\bar{D}$ the set of all explored pivots of the new dictionary $D$
- $P^T[\bar{x}_h]$ the image of the direction
- $VD$ the set of visited dictionaries
- $\bar{x}$ the basic feasible solution for the new dictionary $D$
- $\bar{x}_h$ a direction in the decision space
- $\bar{X}$ the set of efficient extreme points in the decision space
- $\bar{Y}^h$ the set of extreme directions in the decision space
- $-Z^T_N e^j$ a direction in the objective space

### Notation

| Symbol | Description |
|--------|-------------|
| $A, b, C$ | problem data |
| $B$ | the set of basic variables |
| $BD$ | the set of boundary dictionaries |
| $D^0$ | the initial dictionary |
| $E^0$ | the set of explored pivots for the initial dictionary |
| $J^D$ | the index set of entering variables |
| $J^D_0$ | an initial index set of entering variables |
| $N$ | the set of nonbasic variables |
| $R$ | the recession cone of the image |
| $\bar{x}_0$ | an initial efficient basic feasible solution corresponding to the initial dictionary |

### Algorithm 1 Parametric simplex algorithm

0: **Input:** $A, b, C$
1. **Initialize:** Find $D^0$ and the index set of entering variables $J^D_0$; $BD \leftarrow \{D^0\}$, $\bar{X} \leftarrow \{\bar{x}_0\}$, $\bar{Y} \leftarrow \emptyset$, $VS \leftarrow \emptyset$, $\bar{X}^h \leftarrow \emptyset$, $ED^0 \leftarrow \emptyset$, $R \leftarrow \emptyset$.
2. **while** $BD \neq \emptyset$ **do**
3. **Let** $D \in BD$ with nonbasic variables $N$ and index set of entering variables $J^D$;
4. **for** $j \in J^D$ **do**
5. **Let** $x_j$ be the entering variable;
6. **if** $B^{-1} Ne^j \leq 0$ **then**
7. **Let** $\bar{x}_h$ be such that $\bar{x}_h = -B^{-1} Ne^j$ and $\bar{x}_h = e^j$;
8. $\bar{X}^h \leftarrow \bar{X}^h \cup \{\bar{x}_h\}$
9. $\bar{Y}^h \leftarrow P^T[\bar{x}_h] \cup \{-Z^T_N e^j\}$
10. **else**
11. **Pick** $i \in \arg \min_{i \in B \setminus (B^{-1}N)} (B^{-1}b)^T e^j$.
12. **if** $(j, i) \notin E^0$ **then**
13. **Perform** the pivot with entering variable $x_j$ and leaving variable $x_i$;
14. **Call** the new dictionary $D$ with nonbasic variables $N = N \cup \{i\} \setminus \{j\}$;


Illustration of PSA

We consider the following MOLP adapted from Alves et al., which we solved using a Matlab implementation of PSA

\[
\begin{align*}
\min f_1 &= -3x_1 - x_2 \\
\min f_2 &= -x_1 - 4x_2 \\
\text{Subject to} & \\
-x_1 + x_2 & \leq 2 \\
x_1 + x_2 & \leq 7 \\
x_1 + 2x_2 & \leq 10 \\
x_1, & \quad x_2 \geq 0
\end{align*}
\]

The efficient extreme points found are \( x^1 = (2.0, 4.0)^T, x^2 = (4.0, 3.0)^T \) and \( x^3 = (7.0, 0.0)^T \). The corresponding nondominated points are \( f^1 = (-10.0, -18.0)^T, f^2 = (-15.0, -16.0)^T \) and \( f^3 = (-21.0, -7.0)^T \), respectively, where \( x^1 = (x_1^1, x_2^1)^T, x^2 = (x_3^1, x_2^1)^T, x^3 = (x_3^1, x_2^1)^T \in X_E \) and \( f^1 = (f_1^1, f_2^1)^T, f^2 = (f_1^2, f_2^2)^T, f^3 = (f_1^3, f_2^3)^T \in Y_E \). The feasible region in the decision space is shown in Figure 1.

Scalization techniques

Before presenting BOA, we first present two basic scalization methods that play an important role in its implementation. These methods are weighted sum scalization and translative or scalization by a reference variable. As noted in Löhne, scalization is one of the most important techniques used in MOLP.

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

\[
\min w^T C x \quad \text{subject to } A x \geq b \quad P_1(w)
\]

The weights are usually normalized, so that \( e^T w = 1 \), with \( e^T = (1, \ldots, 1) \). The dual of \( P_1(w) \) is

\[
\max b^T u \quad \text{subject to } \begin{cases}
A^T u = C^T w \\
u \geq 0
\end{cases} \quad D_1(w)
\]

In the method of scalization by a reference variable, the \( q \) objectives are associated to 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

\[
\min z \quad \text{subject to } \begin{cases}
A x \geq b \\
C x - ze \leq y \quad P_2(y)
\end{cases}
\]

The dual program is

\[
\max b^T u - y^T w \quad \text{subject to } \begin{cases}
A^T u - C^T w = 0 \\
ee^T w = 1 \\
(u, w) \geq 0
\end{cases} \quad D_2(y)
\]
The above two scalarization techniques are fundamental for the implementation of BOA which is discussed in the next section.\textsuperscript{5}

\textbf{Benson’s outer-approximation algorithm}

This version of BOA is due to Shao and Ehrigott.\textsuperscript{19} It can be found in Löhne.\textsuperscript{5} 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 polyhedron \(Y_0\) (outer approximation) containing the upper image \(Y\) in the objective space and an interior point \(\bar{p}\) of the image is determined by solving \(P_1(w)\). The inequality representation of the outer approximation is also determined by solving \(D_1(w)\). The algorithm constructs a sequence of decreasing polytope \(Y_0 \supseteq Y_1 \supseteq \ldots \supseteq Y_k = Y\). The vertices of each polytope \(Y_k\) as well as inequality representation (facets) are stored in each iteration. Then, for each vertex \(v\) of the polytope, the algorithm checks if the vertex is on the boundary of \(Y\). If the vertices are on it, 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 \(\bar{p}\) and finds the intersection \(y\) of this line with the boundary of \(Y\) by solving \(P_2(y)\). Then a supporting hyperplane adjacent to \(y\) is constructed by solving \(D_2(y)\). This hyperplane is added to \(Y_k\) to provide a smaller approximation. 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.

The notation used in the pseudo-code of BOA is as follows.

\textbf{Notation}

\begin{align*}
A, b, C & \quad \text{problem data} \\
D^h & \quad \text{the homogeneous dual problem} \\
k & \quad \text{the iteration counter} \\
\bar{p} & \quad \text{an interior point} \\
P_h & \quad \text{the homogeneous problem} \\
R(v) & \quad \text{the LP that finds the unique value } \delta \\
\bar{T} & \quad \text{a set of solutions of the dual problem} \\
\bar{T}^h & \quad \text{the solution of the homogeneous dual problem} \\
Y^h_k & \quad \text{the inequality representation of the current polytope} \\
Y^p_k & \quad \text{the representation by vertices} \\
(y, z) & \quad \text{an optimal solution to } P_2(y)
\end{align*}

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

The command \texttt{solve()} solves an LP

\begin{align*}
\text{vert()} & \quad \text{returns the vertices of a polytope } Y_k \\
\bar{Y} & \quad \text{the set of nondominated vertices} \\
\bar{y}_h & \quad \text{the set of extreme directions}
\end{align*}

\begin{algorithm}
\caption{Benson’s outer-approximation algorithm\textsuperscript{5}}
0: \textbf{Input:} \(A, b, C\); Problem data
\hspace{1em} a solution \(\{0\}, \bar{y}^h\) to \(P^0\); \\
\hspace{2em} a solution \(\bar{T}^h\) to \(D^h\);
1. \textbf{Initialize:} \(\bar{p} \gets P_1(0) + \epsilon\); \\
\hspace{2em} \bar{T} \gets \{\text{solve}(D_1(w)), (u, w) \in T^h\};
2. \textbf{while} \(z = 0\) \textbf{do}
3. \hspace{2em} \(Y^h_k \gets \{D^h(u, w), (u, w) \in T^h\};
4. \hspace{2em} Y^h_k \gets \text{vert}(Y^h_k);
5. \hspace{2em} \bar{Y} \gets \emptyset;
6. \hspace{2em} \text{for } i = 1 \text{ to } |Y^h| \text{ do}
7. \hspace{3em} v \gets Y^h_k[i];
8. \hspace{3em} (y, z) \gets \text{solve}(P_2(y));
9. \hspace{3em} \bar{Y} \gets \bar{Y} \cup \{y\};
10. \hspace{3em} \text{if } z \neq 0 \text{ then}
11. \hspace{4em} (x, \delta) \gets \text{solve}(R(v)), (0 < \delta < 1); \\
12. \hspace{4em} y \gets \delta v + (1 - \delta)\bar{p}; \\
13. \hspace{4em} (u, w) \gets \text{solve}(D_2(y));
14. \hspace{3em} \bar{T} \gets \bar{T} \cup \{(u, w)\};
15. \hspace{2em} \text{endfor;}
16. \hspace{2em} \text{endwhile}
17. \textbf{end;}
18. \textbf{Output:} \(\bar{Y}, \bar{y}_h\): Nondominated set and directions; \\
\hspace{1em} \bar{T} : \text{a solution to dual.}
\end{algorithm}

\textbf{Illustration of BOA}

Consider again Problem 3 of ‘Illustration of PSA’ section. The nondominated points found by Algorithm 2 are \(f^* = (-21.0, -7.0)^T\), \(f^* = (-15.0, -16.0)^T\) and \(f^* = (-10.0, -18.0)^T\), respectively. These nondominated points are shown in Figure 2.

\textbf{Selection of the MPNP}

This issue has been alluded to in the introduction. To determine the MPNP, we employ the technique of Compromise Programming (CP) introduced by Zeleny\textsuperscript{25} and compute the ideal objective point which would serve as a reference point in each case. CP is a
Having computed the ideal objective point \( y^* \), we now determine the minimum distance of each nondominated point \( y \) from it by finding

\[
\min \{ \| y_1 - y^* \|, \| y_2 - y^* \|, \ldots, \| y_n - y^* \| \}
\]

where \( y_i \in Y_N \) has already been found either by PSA or BOA, \( \| \cdot \| \) is the Euclidean norm on \( \mathbb{R}^q \) and \( y^* \) is the ideal objective point. Using the nondominated points \( f^1, f^2 \) and \( f^3 \) for Problem 3 yield

\[
\| f^1 - y^* \| = 11.0, \| f^2 - y^* \| = 6.3 \quad \text{and} \quad \| f^3 - y^* \| = 11.0
\]

Since the relative distance of \( f^3 \) from the ideal point \( y^* \) is 6.3 which is the smallest of the three, it therefore means that \( f^3 = (-15.0, -16.0)^T \) is the closest of the three nondominated points to the ideal point \( y^* = (-21.0, -18.0)^T \). Hence, \( f^3 \) is selected as the DM’s MPNP.

The following more substantial illustrative MOLP adapted from Zeleny\(^9\) with three objectives makes the point

\[
\begin{align*}
\min f_1 &= -x_1 - 2x_2 + x_3 - 3x_4 - 2x_5 - x_7 \\
\min f_2 &= -x_2 - x_3 - 2x_4 - 3x_5 - x_6 \\
\min f_3 &= -x_1 - x_3 + x_4 + x_5 + x_6 + x_7
\end{align*}
\]

Subject to

\[
\begin{align*}
x_1 + 2x_2 + x_3 + x_4 + 2x_5 + x_6 + 2x_7 &\leq 16 \\
\quad -2x_1 - x_2 + x_4 + 2x_5 + x_6 &\leq 16 \\
\quad -x_1 + x_3 + 2x_5 - 2x_7 &\leq 16 \\
\quad x_2 + 2x_3 - x_4 + x_5 - 2x_6 - x_7 &\leq 16 \\
\quad x_1, x_2, x_3, x_4, x_5, x_6, x_7 &\geq 0
\end{align*}
\]

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 Problem 5 with BOA, the set of nondominated points found is \( Y_N = \{ (-48.0, -32.0, 16.0)^T, (-16.0, 0.0, -16.0)^T, (0.0, -8.0, -16.0)^T, (-5.33, -21.33, -5.33)^T, (-16.0, -24.0, 0.0)^T \} \) 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 DM’s MPNP.

For PSA, the set of nondominated points found is \( Y_N = \{ (-8.0, -4.0, 12.0)^T, (-16.0, 0.0, -16.0)^T, (0.0, -8.0, 8.0)^T, (-8.0, 0.0, 8.0)^T \} \) also with \( y^* \notin Y_N \).

Next, we measure the distances of each of these points from the ideal point \( y^* = (-48.0, -32.0, -16.0)^T \) as was done with those returned by BOA. It turned out that the nondominated
point \((-16.0, 0.0, -16.0)^T\) is the closest to the ideal point \(y^\ast\) and is selected as the DM’s MPNP as shown in Table 1, Problem 9. Its distance from it is 55.42 which is bigger than 32 which was the closest when measuring the points returned by BOA, thereby making the MPNP returned by BOA closer to the ideal point and of higher quality for this problem.

We have used this method to choose the MPNP from the nondominated sets returned by PSA and BOA for comparison. The measure of the quality of solutions used is the distance to the ideal point as explained above.

**Experimental results**

In this section, we provide numerical results to compare the computing efficiency, robustness and the quality of a MPNP returned by Algorithms 1 and 2.

Table 2 shows the numerical results for a collection of 53 problems, from the existing literature. Problem 1 is taken from Ehrgott, Problems 2–10 were taken from Zeleny. Problems 11–21 are test problems from the interactive MOLP explorer (iMOLPe) of Alves et al. Problems 22–47 are taken from Steuer. Problems 48 and 53 are test problem in Bensolve-2.0 of Löhne and Weißing, while Problem 52 is a test problem in Bensolve-1.2 of Löhne. Finally, Problems 49–51 are obtained using a script in Bensolve-2.0 of Löhne and Weißing that was used to generate problem 53 with the same number of variables and constraints. 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 49–51 and 53 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. Finally, Problem 52 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.

Both Algorithms 1 and 2 were implemented in Matlab. In all tests, \(m\) is the number of constraints, \(n\) the number of variables, \(q\) the number of objectives and \(NPN\) the number of nondominated points returned by the algorithms. All problems were executed on an Intel Core i5-2500 CPU at 3.30 GHz with 16.0GB RAM. We recorded the CPU times (in seconds) returned by the algorithms for each problem and also acted as the DM by choosing a MPNP (whose components are as close as possible to the ideal objective point as explained in ‘Selection of the most preferred nondominated point’ section) from the nondominated set \(Y_N = \{Cx : x \in X_F\}\) returned by PSA to compare with a MPNP returned by BOA.

As can be seen in Table 2, the CPU times increase as the problem dimension increases. We can also infer from Tables 2 and 3 that the CPU times also depend to some extent on the total number of nondominated points returned by the algorithm for a given problem. That is to say, the more the number of nondominated points in a given problem, the more computational effort would be required to obtain them. We note here that most of the problems in Table 2 are non-degenerate. For these problems, PSA appears to have computational advantage over BOA, most especially for those problems with more nondominated points as it returns only a subset of them; see problems 20, 25, 30, 39, 40 and 46. We noticed that for those problems where both algorithms return the same number of nondominated points, there is a slight difference in CPU time which is in favour of PSA. We also observed that PSA returns more nondominated points for some of the problems than BOA; this is not supposed to happen as it is meant to return a subset of these points. Some of the nondominated points returned
Table 2. Comparative results for small to medium instances.

| Prob. | Algorithms | n | m | q | BOA NNP | BOA MPNP | BOA CPU (s) | PSA NNP | PSA MPNP | PSA CPU (s) |
|-------|------------|---|---|---|---------|----------|---------|---------|---------|---------|
|      |            |   |   |   | f1 = -2.00 | f2 = 10.00 | 0.038 | f1 = -2.00 | f2 = 10.00 | 0.031 |
| 1    | Ehrgotz   | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 2    | Zeleny    | 2 | 2 | 2 | 3 | 3 | 3 | 3 | 3 |
| 3    | Zeleny    | 2 | 4 | 2 | 2 | 2 | 2 | 2 | 2 |
| 4    | Zeleny    | 2 | 4 | 3 | 3 | 3 | 3 | 3 | 3 |
| 5    | Zeleny    | 2 | 6 | 2 | 3 | 3 | 3 | 3 | 3 |
| 6    | Zeleny    | 3 | 3 | 3 | 5 | 5 | 5 | 5 | 5 |
| 7    | Zeleny    | 5 | 3 | 3 | 4 | 4 | 4 | 4 | 4 |
| 8    | Zeleny    | 5 | 2 | 2 | 1 | 1 | 1 | 1 | 1 |
| 9    | Zeleny    | 6 | 4 | 2 | 1 | 1 | 1 | 1 | 1 |
| 10   | Zeleny    | 7 | 4 | 3 | 5 | 5 | 5 | 5 | 5 |
| 11   | iMOLPe    | 2 | 3 | 2 | 3 | 3 | 3 | 3 | 3 |
| 12   | iMOLPe    | 3 | 3 | 4 | 3 | 3 | 3 | 3 | 3 |
| 13   | iMOLPe    | 3 | 5 | 3 | 10 | 10 | 10 | 10 | 10 |
| 14   | iMOLPe    | 3 | 3 | 3 | 7 | 7 | 7 | 7 | 7 |
| 15   | iMOLPe    | 4 | 3 | 3 | 8 | 8 | 8 | 8 | 8 |
| 16   | iMOLPe    | 4 | 2 | 3 | 6 | 6 | 6 | 6 | 6 |
| 17   | iMOLPe    | 4 | 4 | 3 | 11 | 11 | 11 | 11 | 11 |
| 18   | iMOLPe    | 3 | 3 | 3 | 5 | 5 | 5 | 5 | 5 |
| 19   | iMOLPe    | 15 | 10 | 2 | 11 | 7 | 7 | 7 | 7 |
| 20   | iMOLPe    | 15 | 10 | 3 | 37 | 7 | 7 | 7 | 7 |

(continued)
| Prob. | Origin | n   | m   | q   | BOA      | PSA   | CPU (s) | BOA      | PSA   | CPU (s) |
|-------|--------|-----|-----|-----|----------|-------|---------|----------|-------|---------|
| 21    | iMOLPe | 10  | 5   | 3   | 14       | f1 = 226.40 | 0.623   | 14       | f1 = 223.09 | 0.589   |
|       |        |     |     |     | f2 = 501.86 | f2 = 496.23 | f3 = 246.64 |
|       |        |     |     |     | f3 = 351.14 |       |         | f3 = 351.14 |       |         |
| 22    | Steuer | 5   | 5   | 2   | 5       | f1 = 10.00  | 0.036   | 5        | f1 = 10.00  | 0.034   |
|       |        |     |     |     | f2 = 3.00   | f2 = 3.00   | f3 = 3.00   |
| 23    | Steuer | 4   | 4   | 3   | 3       | f1 = 3.42   | 0.015   | 3        | f1 = 3.42   | 0.012   |
|       |        |     |     |     | f2 = 10.28  | f2 = 10.28  | f3 = 3.42   |
| 24    | Steuer | 5   | 5   | 4   | 14      | f1 = 1.02   | 0.098   | 14       | f1 = 1.02   | 0.081   |
|       |        |     |     |     | f2 = 25.46  | f2 = 25.46  | f3 = 25.46  |
| 25    | Steuer | 10  | 8   | 4   | 72      | f1 = 106.29 | 1.973   | 65       | f1 = 183.36 | 0.921   |
|       |        |     |     |     | f2 = 462.13 | f2 = 424.26 | f3 = 117.29 |
|       |        |     |     |     | f3 = 175.57 | f3 = 175.57 | f4 = 33.41   |
| 26    | Steuer | 5   | 4   | 3   | 9       | f1 = 52.07  | 0.054   | 8        | f1 = 52.07  | 0.045   |
|       |        |     |     |     | f2 = 31.50  | f2 = 31.50  | f3 = 17.35   |
| 27    | Steuer | 6   | 8   | 4   | 14      | f1 = 6.94   | 0.065   | 6        | f1 = 6.94   | 0.053   |
|       |        |     |     |     | f2 = 5.38   | f2 = 5.38   | f3 = 6.83    |
| 28    | Steuer | 7   | 6   | 4   | 15      | f1 = 31.53  | 0.286   | 12       | f1 = 31.53  | 0.555   |
|       |        |     |     |     | f2 = 26.48  | f2 = 26.48  | f3 = 26.57   |
|       |        |     |     |     | f3 = 26.57  | f3 = 26.57  | f4 = 0.34    |
| 29    | Steuer | 7   | 6   | 4   | 9       | f1 = 26.80  | 0.192   | 9        | f1 = 26.80  | 0.142   |
|       |        |     |     |     | f2 = 37.73  | f2 = 37.73  | f3 = 24.33   |
|       |        |     |     |     | f3 = 24.33  | f3 = 24.33  | f4 = 59.60   |
| 30    | Steuer | 8   | 8   | 6   | 286     | f1 = 74.00  | 73.963 | 40       | f1 = 77.00  | 0.699   |
|       |        |     |     |     | f2 = 107.50 | f2 = 52.00  | f3 = 16.00   |
|       |        |     |     |     | f3 = 41.25  | f3 = 41.25  | f4 = 27.25   |
|       |        |     |     |     | f4 = 27.25  | f4 = 27.25  | f5 = 9.00    |
|       |        |     |     |     | f5 = 9.00   | f5 = 9.00   | f6 = 30.75   |
|       |        |     |     |     | f6 = 30.75  | f6 = 30.75  | f7 = 20.00   |
| 31    | Steuer | 8   | 8   | 3   | 5       | f1 = 36.57  | 0.168   | 5        | f1 = 36.57  | 0.156   |
|       |        |     |     |     | f2 = 22.28  | f2 = 22.28  | f3 = 14.00   |
|       |        |     |     |     | f3 = 14.00  | f3 = 14.00  | f4 = 4.93    |
| 32    | Steuer | 8   | 8   | 3   | 12      | f1 = 14.03  | 0.135   | 1        | f1 = 6.50   | 0.121   |
|       |        |     |     |     | f2 = 18.00  | f2 = 11.00  | f3 = 7.50    |
|       |        |     |     |     | f3 = 4.93   | f3 = 4.93   | f4 = 39.25   |
| 33    | Steuer | 5   | 5   | 4   | 12      | f1 = 21.50  | 0.277   | 8        | f1 = 8.00   | 0.216   |
|       |        |     |     |     | f2 = 39.25  | f2 = 23.87  | f3 = 7.62    |
|       |        |     |     |     | f3 = 16.25  | f3 = 16.25  | f4 = 27.00   |
| 34    | Steuer | 6   | 6   | 3   | 17      | f1 = 12.65  | 0.212   | 17       | f1 = 13.62 | 0.210   |
|       |        |     |     |     | f2 = 0.00   | f2 = 9.75   | f3 = 26.25   |
|       |        |     |     |     | f3 = 30.15  | f3 = 30.15  | f4 = 27.00   |
| 35    | Steuer | 5   | 5   | 4   | 9       | f1 = 14.66  | 0.462   | 2        | f1 = 14.00  | 0.345   |
|       |        |     |     |     | f2 = 21.06  | f2 = 0.00   | f3 = 27.00   |

(continued)
Table 2. Continued.

| Prob. | Origin | BOA | PSA |
|-------|--------|-----|-----|
|       | n      | m  | q  | CPU (s) | NNP | MPNP | CPU (s) | NNP | MPNP | CPU (s) |
| 36    | Steuer | 10 | 10 | 4    | 6   | f4 = 16.00 | 0.333 | 6   | f4 = 0.00 | 0.241 |
|       |        |    |    |      | f2 = 19.21 | f4 = 27.07 | f4 = 27.07 | f2 = 19.21 | f4 = 27.07 | f4 = 27.07 |
| 37    | Steuer | 8  | 8  | 3    | 13  | f4 = 14.48 | 0.217 | 13  | f4 = 0.00 | 0.201 |
|       |        |    |    |      | f2 = 19.21 | f4 = 27.07 | f4 = 27.07 | f2 = 19.21 | f4 = 27.07 | f4 = 27.07 |
| 38    | Steuer | 6  | 7  | 4    | 21  | f4 = 2.61 | 0.386 | a  | – | – |
|       |        |    |    |      | f2 = 12.63 | f3 = 9.70 | f4 = 2.37 | f2 = 12.63 | f3 = 9.70 | f4 = 2.37 |
| 39    | Steuer | 12 | 16 | 4    | 601 | f4 = 5.25 | 31.034 | 23  | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 14.25 | f3 = 8.25 | f4 = 1.00 | f2 = 14.25 | f3 = 8.25 | f4 = 1.00 |
| 40    | Steuer | 10 | 14 | 5    | 132 | f4 = 5.16 | 102.952 | 9   | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 2.79 | f3 = 4.38 | f4 = 18.70 | f2 = 2.79 | f3 = 4.38 | f4 = 18.70 |
| 41    | Steuer | 7  | 6  | 3    | 3   | f4 = 29.40 | 0.165 | 3   | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 65.30 | f3 = 39.30 | f5 = 9.69 | f2 = 65.30 | f3 = 39.30 | f5 = 9.69 |
| 42    | Steuer | 7  | 7  | 3    | 7   | f4 = 62.18 | 0.036 | 7   | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 93.50 | f3 = 52.00 | f5 = 9.69 | f2 = 93.50 | f3 = 52.00 | f5 = 9.69 |
| 43    | Steuer | 6  | 6  | 4    | 5   | f4 = 37.50 | 0.158 | 5   | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 11.25 | f3 = 7.50 | f5 = 9.69 | f2 = 11.25 | f3 = 7.50 | f5 = 9.69 |
| 44    | Steuer | 6  | 6  | 4    | 10  | f4 = 34.50 | 0.211 | 10  | f1 = 18.00 | 1.395 |
|       |        |    |    |      | f2 = 7.50 | f3 = 56.00 | f4 = 31.50 | f2 = 7.50 | f3 = 56.00 | f4 = 31.50 |
| 45    | Steuer | 10 | 14 | 5    | 471 | f4 = 1.03 | 307.611 | a  | – | – |
|       |        |    |    |      | f2 = 2.19 | f3 = 2.01 | f4 = 8.13 | f2 = 2.19 | f3 = 2.01 | f4 = 8.13 |
| 46    | Steuer | 10 | 14 | 5    | 128 | f4 = 4.95 | 105.344 | 1   | f1 = 4.93 | 0.291 |
|       |        |    |    |      | f2 = 3.42 | f3 = 4.38 | f4 = 18.91 | f2 = 3.42 | f3 = 4.38 | f4 = 18.91 |
| 47    | Steuer | 7  | 7  | 3    | 6   | f4 = 3.83 | 0.045 | 9   | f1 = 3.83 | 0.031 |
|       |        |    |    |      | f2 = 7.646 | f3 = 49.57 | f4 = 9.727 | f2 = 7.646 | f3 = 49.57 | f4 = 9.727 |
| 48    | Bensolve-2.0 | 5 | 31 | 5    | 22  | f4 = 0.00 | 2.877 | 1   | f1 = 0.00 | 0.125 |
|       |        |    |    |      | f2 = 1.00 | f3 = 0.00 | f4 = 0.00 | f2 = 1.00 | f3 = 0.00 | f4 = 0.00 |
are repeated. In terms of the quality of a MPNP returned by these algorithms, we observed in Table 2 that both algorithms returned the same MPNP points for most of the problems considered. However, for a few of these problems where the MPNP are not the same, BOA returned higher quality MPNP than PSA as illustrated in pages 15 and 16 (second numerical illustration of Problem 5). This observation may be largely due to the fact that PSA is not meant to return all nondominated points, as it returns a subset of them, whereas BOA returns all the nondominated points of the problem.

Next, we use practical size MOLP instances from Csirmaz\textsuperscript{30} which is an MOLP solver called Inner and MOPLIB\textsuperscript{31} which stands for Multi-Objective Problem Library. These test problems were also executed on the same machine, and the results are reported in Table 3. Problems 54–72 are from Csirmaz\textsuperscript{30} while Problems 73–86 are from MOPLIB. Note that Problems 54–73 are highly degenerate. Their structure is such that their constraint and criterion matrices are sparse while all the components of the RHS vectors are zeros except for a one (1) at the beginning as the only non-zero element. Problem 85 is such that the constraint and criterion matrices are sparse while the components of the RHS vector are all zeros except for a 90 at the end as the only non-zero entry. Finally, Problem 86 is such that the constraint and criterion matrices as well as the RHS vector are all sparse. For the highly degenerate problems, it was observed that BOA is computationally superior to PSA which confirms what was reported by Rudloff et al.\textsuperscript{3} that BOA outperforms PSA on highly degenerate problems. Even the nondominated points returned by PSA for these problems are also of lower quality than those returned by BOA.

In terms of robustness of methods, we noticed in Tables 2 and 3 that PSA could not solve problems 38, 45, 52, 68 and 81. It returns the image which is the whole region indicating that none of the vertices in the image is nondominated, meaning that no solution is returned thereby making BOA more robust. However, we also observed in Table 3 that BOA could not produce results for some of the test problems despite the long running time allowed (three days); it was aborted. The fact that some problems were aborted after three 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 nondominated points or run out of memory which would indicate that the total number of nondominated points has exceeded the Matlab storage capacity on the machine used.

For those problems which were solved by BOA in Table 3, it was also observed that the MPNPs returned are of higher quality than those returned by PSA. However, for the non-degenerate problems, PSA was found to be computationally superior to BOA.

| Algorithms | BOA | PSA |
|------------|-----|-----|
| Prob. | Origin | n | m | q | NNP | MPNP | CPU (s) | NNP | MPNP | CPU (s) |
| 49 | Bensolve-2.0 | 36 | 36 | 2 | 8 | fl = -5.00 | 0.211 | 82 | fl = -5.00 | 0.772 |
| | | | | | | f2 = -26.00 | | | | |
| 50 | Bensolve-2.0 | 64 | 64 | 2 | 14 | fl = -63.00 | 0.403 | 292 | fl = -34.50 | 5.167 |
| | | | | | | f2 = -7.00 | | | | |
| 51 | Bensolve-2.0 | 100 | 100 | 2 | 20 | fl = -124.00 | 0.621 | 1102 | fl = -123.50 | 36.323 |
| | | | | | | f2 = -9.00 | | | | |
| 52 | Bensolve-1.2 | 100 | 101 | 2 | 32 | fl = -8.42 | 0.503 | a | – | – |
| | | | | | | f2 = -116.65 | | | | |
| 53 | Bensolve-2.0 | 343 | 343 | 3 | 1368 | fl = -42.00 | 55.302 | b | – | – |
| | | | | | | f2 = -294.00 | | | | |
| | | | | | | f3 = -6.00 | | | | |

*The image is the whole region, implying that the problem has no solution.

*aOut of memory.
Table 3. Comparative results for large instances (NNP stands for Number of Nondominated Points).

| Prob. Origin | n  | m  | q  | NNP MPNP | BOA        | CPU (s) | PSA        | NNP MPNP | CPU (s) |
|--------------|----|----|----|----------|------------|---------|------------|----------|---------|
| 54 Inner     | 844| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.871   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 2.835    |
| 55 Inner     | 853| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.892   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 2.888    |
| 56 Inner     | 857| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.871   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 2.922    |
| 57 Inner     | 873| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.884   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.041    |
| 58 Inner     | 877| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.935   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.071    |
| 59 Inner     | 880| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 0.968   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.113    |
| 60 Inner     | 882| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 1.009   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.115    |
| 61 Inner     | 886| 12 | 10 | 2        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 1.341   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.118    |
| 62 Inner     | 888| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 1.104   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.119    |
| 63 Inner     | 1009| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 1.281   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 3.911    |
| 64 Inner     | 1956| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 4.288   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 13.374   |
| 65 Inner     | 1983| 12 | 10 | 1        | $f_1 = -1.00, f_2 = -1.00, f_3 = 0.00$ | 4.418   | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 13.605   |
| 66 Inner     | 3722| 338| 10 | 55       | $f_1 = -0.25, f_2 = -0.50, f_3 = -2.75, 22.167$ | 56.198  | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 56.458   |
| 67 Inner     | 3725| 338| 10 | 61       | $f_1 = -0.20, f_2 = -0.40, f_3 = -2.40$ | 24.605  | $f_1 = 0.00, f_2 = 0.00, f_3 = 0.00$ | 56.458   |

(continued)
Table 3. Continued.

| Algorithms | BOA | PSA |
|------------|-----|-----|
| Prob. Origin | n | m | q | NNP | MPNP | CPU (s) | NNP | MPNP | CPU (s) |
| 68 Inner | 3897 | 362 | 10 | b | – | – | a | – | – |
| 69 Inner | 5646 | 492 | 10 | 1575 | f1 = -0.38, f2 = 0.00, f3 = -2.55 | 125.488 | l | f1 = 0.00, f2 = 0.00, f3 = 0.00 | 130.159 |
| 70 Inner | 8891 | 707 | 10 | 13 | f1 = -0.20, f2 = 0.00, f3 = -2.20, f4 = -2.20, f5 = -0.20, f6 = -0.20 | 228.312 | l | f1 = 0.00, f2 = 0.00, f3 = 0.00 | 329.701 |
| 71 Inner | 9472 | 707 | 10 | b | – | – | l | f1 = 0.00, f2 = 0.00, f3 = 0.00 | 362.449 |
| 72 Inner | 10017 | 779 | 10 | 31 | f1 = -0.11, f2 = 0.00, f3 = -2.44 | 260.494 | l | f1 = 0.00, f2 = 0.00, f3 = 0.00 | 412.929 |
| 73 MOPLIB | 30 | 21 | 12 | l | f1 = -5.0E-12, f2 = -5.0E-12, f3 = -5.0E-12, f4 = -5.0E-12, f5 = -5.0E-12, f6 = -5.0E-12 | 0.598 | l | f1 = 0, f2 = 0 | 0.167 |
| 74 MOPLIB | 100 | 20 | 3 | 291 | f1 = -168.00 | 4.291 | l | f1 = -168.00 | 0.122 |
| 75 MOPLIB | 53 | 221 | 3 | 2552 | f1 = 0.00 | 1663.803 | l | f1 = 0.00 | 0.682 |
| 76 MOPLIB | 53 | 226 | 3 | 552 | f1 = -180.00 | 6.551 | 74 | f1 = -144.00 | 1.628 |
| 77 MOPLIB | 1143 | 1211 | 3 | c | – | – | l | f1 = -85.00, f2 = 0, f3 = 0 | 16.248 |
| 78 MOPLIB | 36939 | 4608 | 3 | c | – | – | l | f1 = 0, f2 = 0, f3 = 0 | 18927.102 |
| 79 MOPLIB | 900 | 60 | 4 | b | – | – | l | f1 = -434.00, f2 = -452.00 | 3.005 |
| 80 MOPLIB | 729 | 729 | 4 | b | – | – | c | – | – |
| 81 MOPLIB | 4492 | 1003 | 4 | c | – | – | l | – | – |
| 82 MOPLIB | 900 | 60 | 10 | c | – | – | l | f1 = -394.00, f2 = -429.00 | 4.306 |
| 83 MOPLIB | 779 | 10174 | 10 | b | – | – | l | f1 = 0.00, f2 = 0.00 | 424.58 |
| 84 MOPLIB | 376 | 1917 | 19 | c | – | – | l | f1 = 2, f2 = -1, f3 = -1 | 69.765 |

(continued)
Summary of results

In this section, we present the summary of experimental results discussed in the previous section in Table 1. We have also presented the CPU time of BOA and PSA for 70 out of the 86 instances (which represent 81.40%) of the total problems solved by both methods in Figure 3.

Conclusion

We have reviewed the existing literature on the parametric simplex and Benson’s BOAs. We have also presented these algorithms, explained and illustrated them on small MOLP instance. Crucially, we have explained how MPNP’s are obtained. A detailed computational
experience to compare the efficiency, robustness as well as the quality of MPNP’s is provided. The CPU times and quality of MPNP’s returned by these algorithms for a collection of 86 existing MOLP problems, ranging from small to medium and practical size instances is reported. It was observed that BOA is superior to PSA in terms of robustness, quality of the MPNP’s it returns and is also computationally more efficient than PSA on highly degenerate problems. The measure of quality used is the distance to the ideal point as explained in ‘Experimental results’ section. However, PSA outperforms BOA on non-degenerate problems.

Declaration of Conflicting Interests

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

Funding

The research work leading to this paper and its publication were partially financially supported by the UK ESRC grant ES/L011859/1.

References

1. Ehrgott M. Multicriteria optimization. Berlin, Germany: Springer Science & Business Media.
2. 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.
3. Rudloff B, Ulus F and Vanderbei R. A parametric simplex algorithm for linear vector optimization problems. *Math Prog* 2017; 163(1–2): 213–242.
4. 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.
5. Löhne A. Vector optimization with infimum and supremum. Berlin, Germany: Springer Science & Business Media, 2011.
6. Schönfeld KP. Effizienz und dualität in der aktivitätsanalyse. PhD Dissertation, Freie Universität Berlin, Germany, 1964.
7. Geoffrion AM. Solving bicriterion mathematical programs. *Oper Res* 1967; 15: 39–54.
8. Evans JP and Steuer R. A revised simplex method for linear multiple objective programs. *Math Prog* 1973; 5: 54–72.
9. Zeleny M. Linear multiobjective programming, vol. 95. Heidelberg, Berlin: Springer-Verlag, 1974.
10. Yu P and Zeleny M. Linear multiparametric programming by multicriteria simplex method. *Manag Sci* 1976; 23: 159–170.
11. Gal T. A general method for determining the set of all efficient solutions to a linear vector maximum problem. *Eur J Oper Res* 1977; 1: 307–322.
12. Steuer RE. Multiple criteria optimization: theory, computation, and applications. Hoboken, NJ: Wiley, 1986.
13. Ruszczynski A and Vanderbei RJ. Frontiers of stochastically nondominated portfolios. *Econometrica* 2003; 71(4): 1287–1297.
14. Ehrgott M, Puerto J and Rodriguez-Chia A. Primal-dual-simplicial method for multiobjective linear programming. *J Optim Theory Appl* 2007; 134: 483–497.
15. Hamel AH, Löhne A and Rudloff B. Benson type algorithms for linear vector optimization and applications. *J Glob Optim* 2014; 59: 811–836.
16. Benson HP. Further analysis of an outcome set-based algorithm for multiple objective linear programming. *J Optim Theory Appl* 1998; 97: 1–10.
17. Benson HP. Hybrid approach for solving multiobjective linear programs in outcome space. *J Optim Theory Appl* 1998; 98: 17–35.
18. 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.
19. 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.
20. Shao L and Ehrgott M. Approximating the nondominated set of an MOLP by approximately solving its dual problem. *Math Meth Oper Res* 2008; 68: 469–492.
21. Löhne A. Bensolve: VLP solver, version 1.2, www.bensolve.org (2012, accessed 13 March, 2017).
22. Csirmaz L. Using multiobjective optimization to map the entropy region. *Comput Optim Appl* 2016; 63: 45–67.
23. Löhne A, Rudloff B and Ulus F. Primal and dual approximation algorithms for convex vector optimization problems. *J Glob Optim* 2014; 60: 713–736.
24. Alves MJ, Antunes CH and Climaco 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.
25. Zeleny M. Compromise programming. In: JL Cochrane and M Zeleny (eds) *Multiple criteria decision making*. Columbia, SC: University of South Carolina Press, pp.262–301, 1973.
26. Zeleny M. *Multiple criteria decision making*. New York: McGraw-Hill, 1982.
27. Zhang W. A compromise programming method using multibounds formulation and dual approach for multi-criteria structural optimization. *Int J Numer Meth Eng* 2003; 58: 661–678.
28. 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.
29. Löhne A and Weißing B. Bensolve: VLP solver, version 2.0.x, www.bensolve.org (2015, (accessed 13 March, 2017).
30. Csirmaz L. Inner: MOLP solver, https://github.com/lcsir/inner (2016, accessed 14 March 2017).
31. Löhne A and Schenker S. MOPLIB: multi-objective problem library, http://moplib.uni-jena.de (2015, accessed 13 March 2017).
32. Proll LG, Rios-Insua D, and Salhi A. Mathematical programming and the sensitivity of multi-criteria decisions. *Annals of Operations Research* 1993; 43: 109–122.
33. Proll LG, Salhi A, and Rios-Insua D: Improving an optimisation-based framework for sensitivity analysis in Multi-Criteria Decision Making. *Journal of Multi-Criteria Decision Analysis* 2001; 10: 1–9.