Computing Tchebychev weight space decomposition for multiobjective discrete optimization problems

Tyler Perini
Georgia Institute of Technology, H. Milton Stewart School of Industrial & Systems Engineering, Atlanta, Georgia, USA,
Correspondence: perinita@gatech.edu, ORCID: 0000-0002-8502-8111

Stephan Helfrich
Technische Universität Kaiserslautern, Department of Mathematics, Kaiserslautern, Germany,
helfrich@mathematik.uni-kl.de, ORCID: 0000-0001-6588-5266

Pascal Halffmann
Technische Universität Kaiserslautern, Department of Mathematics, Kaiserslautern, Germany,
halffmann@mathematik.uni-kl.de, ORCID: 0000-0002-3462-4941

Natashia Boland
Georgia Institute of Technology, H. Milton Stewart School of Industrial & Systems Engineering, Atlanta, Georgia, USA,
natashia.boland@isye.gatech.edu, ORCID: 0000-0002-6867-6125

Stefan Ruzika
Technische Universität Kaiserslautern, Department of Mathematics, Kaiserslautern, Germany,
ruzika@mathematik.uni-kl.de, ORCID: 0000-0002-3230-0900

Multiobjective discrete optimization (MODO) techniques, including weight space decomposition, have received increasing attention in the last decade. The primary weight space decomposition technique in the literature is defined for the weighted sum utility function, through which sets of weights are assigned to a subset of the nondominated set. Recent work has begun to study the decomposition defined for weighted Tchebychev utility function, which provides the benefit of including all nondominated images but at the cost of "nice" convexity geometric properties. The current work presents the computational details for this weight space decomposition, which contributes an essential visualization technique for MODOs with three objectives that includes the entire nondominated set and overlapping weight set components. A thorough evaluation of the added value of the new weight set decomposition (in contrast to weighted sum) is conducted. Existing box-based MODO algorithms are shown to return insufficient information to compute the weight set decomposition, then the necessary modifications to the algorithm are proven. We further provide a computational study to illustrate the decomposition's use in evaluating and improving design choices within box-based MODO algorithms.

Key words: multiobjective optimization, Tchebychev scalarization, weight space decomposition, triobjective discrete optimization

History:

Acknowledgments

The work of Tyler Perini is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1650044. The research of Stephan Helfrich is supported by Deutsche Forschungsgemeinschaft (DFG) grant RU 1524/6-1. Pascal Halffmann acknowledges DFG grant RU 1524/4-1.

Declarations

Funding

The work of Tyler Perini is supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-1650044. The research of Stephan Helfrich is supported by Deutsche Forschungsgemeinschaft (DFG) grant RU 1524/6-1. Pascal Halffmann acknowledges DFG grant RU 1524/4-1.

Conflicts of Interest

The authors report no conflicts of interest.

Code Availability

Code for computing and plotting weight space decompositions were generated in R and are available at https://github.com/perinita/TchebychevDecomposition. Code is included to run the simulated experiments. Additionally, the following dashboard illustrates the examples included in this manuscript: https://perinigraphics.shinyapps.io/tchebdashboard/.

1. Introduction

Multiobjective optimization problems (MOPs) have gained substantial and increasing attention in the optimization literature over recent decades. This is in part due to the prevalence of multiple, conflicting objectives in practical applications, e.g., cancer radiation therapy Romeijn and Dempsey (2008), Ehrgott et al. (2010), Chan et al. (2014), energy systems and storage Cui et al. (2017), Dai and Charkhgard (2018), Musselman et al. (2019), wildfire prevention and response van der Merwe et al. (2017), and more Ehrgott (2005). In general, a MOP does not usually have a feasible solution which optimizes all objectives simultaneously. Hence, a decision maker is interested in the set of efficient solutions, which are solutions for which one objective cannot be improved without worsening another. The vector of objective values associated with an efficient solution is referred to as a *nondominated (ND) image*, and a MOP generally has multiple ND images. Hence the aim for a MOP is to *compute and represent* the set of all ND images, often called the *ND frontier*.

Compared to multiobjective linear programs, it is more computationally burdensome to compute the ND frontier for multiobjective discrete optimization problems (MODOs).

Many *exact* algorithms, which generally compute ND images iteratively, have been developed over the last twenty years; see recent survey by Halffmann et al. (2022). In particular, the class of *criterion space search* methods include many approaches that decompose the space of objective vectors (called criterion space) into hyperrectangular regions. These "box-based" algorithms (e.g., Boland et al. (2015), Dächert and Klamroth (2015), Dächert et al. (2017), Boland et al. (2017), Perini et al. (2020), Tamby and Vanderpooten (2020)) have been proven to efficiently compute the ND frontier, exactly, as well as to quickly approximate it.

Once the ND frontier is available, choosing a single efficient solution (equivalently, ND image) remains an unclear and nuanced challenge for decision makers. *Weighted utility functions*–including weighted sum and Tchebychev (weighted 0-norm and ∞-norm, respectively)–are commonly used to transform a vector of objectives into a single-objective problem. These functions benefit from interpretability as the vector of weights directly represent a decision maker's prioritization of each objective. If these weights are certain a priori, then they may be used in one single-objective optimization program to provide a preferred solution. A posteriori, a weighted utility function is useful for evaluating the set of ND images, and for a fixed weight, the ND images may be ranked. These functions are commonly integrated within exact MODO algorithms for computing ND images.

Practically, however, it remains unclear what is the best practice for determining the weight vector for a weighted utility function. Decision makers posing MOPs may find it difficult to prescribe their preferred weight vectors when little is known about the MOP beforehand (Xin et al. 2018). Also, as the weights are modified, then the utility value of each ND image should be recomputed, and so the image of "greatest value" is also prone to change. Sensitivity analysis has been studied through the technique of *weight space decomposition (WSD)*. The weight space is represented by the set of convex multipliers–nonnegative weights which sum to one, which we refer to as the *weight set*–and is then separated into components, each of which represent weights when one ND image has greatest utility (Przybylski et al. 2010b, Alves and Costa 2016, Karakaya and Köksalan 2021). See Figure 1 as an example of the triangular weight set and the resulting decompositions for both weighted sum and weighted Tchebychev utility functions.

WSD offers is most advantageous for MODOs with three objectives. First, visualizing a triobjective ND frontier and understanding the relationships between ND images can

be challenging with 3-dimensional representations. Improved graphical visualizations can promote the decision maker's interpretation and analysis of the ND frontier (Xin et al. 2018, Lotov et al. 2013, Berezkin et al. 2006). Miettinen (2014) surveys common visualization techniques, which are generally preferred for large numbers of objectives, but they lack the two features discussed next.

First, WSD provides a *complete sensitivity analysis* about the ND frontier. Intuitively, if an ND image optimizes the utility function for only weights in a small subset of the WSD, then a small change in the weight would result in lower utility compared to an alternative ND image. An ND image which is highly sensitive to the choice of weight vector may be undesirable for decision makers. On the other hand, ND images which are more robust to changes in the weight vector be more attractive to decision makers. Thus, understanding the sensitivity of efficient solutions with respect to weights is useful.

Second, WSD yields information on the *adjacency structure* of images in the ND set. For instance, in Figure 1, according to weighted sum (left), images 5 and 13 are adjacent; however, for weighted Tchebychev , they are not adjacent, and image 15 is relatively "in between." This notion of adjacency is particularly valuable for compromising between a few weight vectors, say for a set of decision makers. Furthermore, we argue that weighted Tchebychev offers a *complete adjacency structure* that is not captured by weighted sum.

This paper makes the following contributions to the MODO literature for WSD with weighted Tchebychev utility function:

1. We provide a detailed procedure for computing and visualizing the WSD for weighted Tchebychev utility function. Unlike Karakaya and Köksalan (2021), our presentation includes thorough discussion of visualizing overlapping weight set components as well as a triangulation scheme to easily compute the area of components.

2. Our procedure generalizes concepts and methods of Klamroth et al. (2015), thus integrating the computation of the WSD with an exact MODO algorithm that computes the ND frontier.

3. We present *inner* and *outer approximations* for the weight set components of this WSD. Therefore, WSD is not only an option for post-processing, but it is also available *on-the-fly*, i.e., during runtime of an exact MODO algorithm.

4. As a sample computationaly study, we use the WSD approximation to evaluate box-selecting policies for an exact MODO algorithm. We compare a naïve policy, a pre-existing

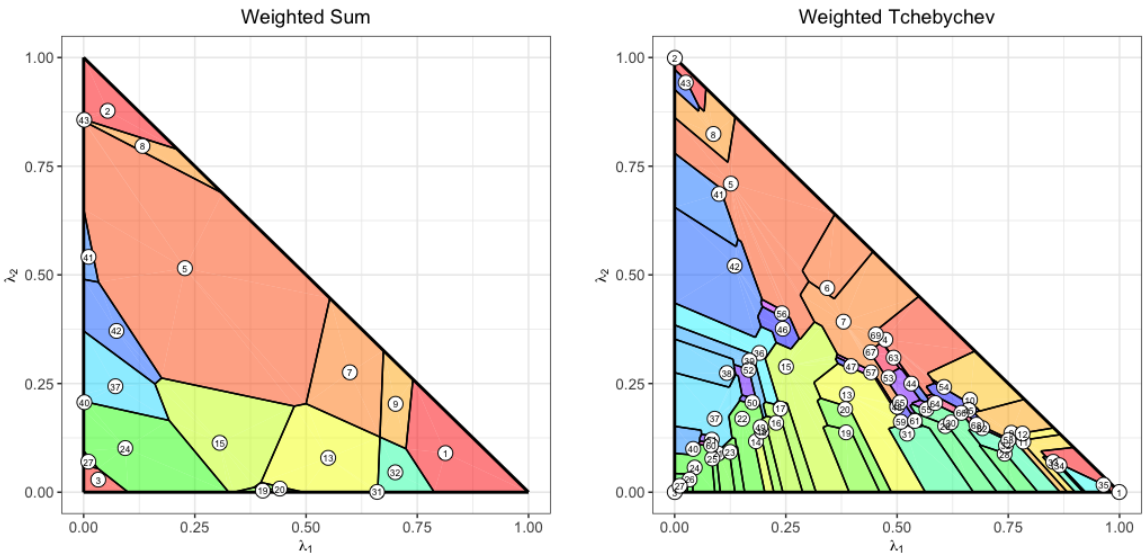


Figure 1 **WSD for two different utility functions on the same MOIP instance with 3 objectives, 25 variables and 69 ND images. When using weighted sum (left), only 20 (extreme supported) ND images are represented by the decomposition. When using weighted Tchebychev (right), all 69 ND images are represented. The numbered labels and colors used to identify images and are consistent between figures.**

criterion space search policy, and three novel policies based on the approximated WSD. On average, one of the novel policies dominates all others according to three different metrics.

Section 1.1 summarizes prior work, including weight space approaches and criterion space search methods. Preliminaries are presented in Section 2. In Section 3, we generalize concepts from prior work to compute the WSD, after which the visualization is detailed in Section 4. Section 5 includes details about inner and outer approximations of the WSD as well as a computational study on the box-selection policy of a box-based algorithm. We conclude in Section 6 with closing remarks.

1.1. Related Work

We refer readers to Ehrgott (2005) for a proper introduction to multiobjective optimization, to Dächert et al. (2021) for a recent overview on solution methods for MODOs, and to Helfrich et al. (2020) for an overview of results, adaptions, and modifications of the weighted Tchebychev function. Next, we review relevant algorithms which systematically decompose the weight space or criterion space.

Weight space approaches primarily appear in the form of decomposition for weighted sum and date back to the work by Yu and Zeleny (1975). For biobjective problems, the well-known dichotomic search approach (Aneja and Nair 1979, Cohon 2003) in fact calculates

(a subset[1] of) ND images based on the fundamental idea of WSD. Benson and Sun (2000, 2002) extend this idea and establish a link between these images and a partitioning of the weight set for multiobjective linear programs.

Przybylski et al. (2010a) provided the first geometric analysis of WSD for weighted sum applied to MODOs, which included fundamental properties such as: weight set components are convex and the implied adjacency structure is symmetric. The weakness of the weighted sum utility function is that it is restricted to a particular subset of the ND frontier (defined in the next section). Przybylski et al. (2010a) further proved that the weight set components for this subset of images is sufficient to cover the weight set. Additionally, they present an algorithm for computing such images for three objectives; before terminating, this algorithm provides *outer approximations*, or supersets, of the weight set components. Later, Alves and Costa (2016) and Halffmann et al. (2020) developed algorithms which iteratively augment *inner approximations*, or subsets of the weight set components, based on the convexity of the components.

Early work on WSD for the weighted Tchebychev utility function included Eswaran et al. (1989) and Ralphs et al. (2006) for biobjective problems. Helfrich et al. (2020) provided the first geometric analysis of the WSD for an arbitrary number of objectives, which included fundamental properties such as: weight set components are star-shaped and can have full-dimensional intersections. Recently, Karakaya and Köksalan (2021) presents a method for computing the more common type of intersection between weight set components (for 2, 3, and 4 objectives) and applies the WSD to the integrated preference function.

To our knowledge, no prior work discusses (i) how to compute and visualize the weight set components with *full-dimensional intersections (i.e., overlapping)* between weight set components; (ii) a triangulation method to compute the areas of weight set components which are nonconvex but star-shaped; and (iii) how to integrate WSD computation into an exact method such that the decomposition is available *on-the-fly*, (i.e., during run-time), including inner and outer approximations. These topics are the focus of the present work.

As an additional note of contrast: the term *approximation* for MODOs tends to have two connotations of representing the ND frontier "well." First, in the case of heuristic methods, the concept is to provide a set of "good" images, each of which may or may not

[1] For readability, we formally define this subset, i.e., *extreme supported* ND images, later in Section 2. In short, they are the ND images which are extreme points of the convex hull of the ND set.

be dominated, close to the ND frontier. Any of these images, if they are dominated, may be replaced by a "better" image as the heuristic method proceeds. Second, in the case of exact methods, is the concept of providing a subset of the ND frontier, and this subset grows as more ND images are found. Whereas Karakaya and Köksalan (2021) focuses on the former sense of approximating ND frontiers, our work focuses on the latter.

Criterion space search approaches are primarily characterized by two features: First is the representation of the unexplored portion of the criterion space, called the search region, which may contain new ND images. Second is the mathematical optimization subproblem that returns a new ND image from (a subset of) the search region, if one exists. Solving the subproblem is generally the more expensive procedure; however, intelligently updating the representation of the search region can reduce the number of subproblems to be solved. We now focus on *box-based algorithms*; see, for instance, Dächert and Klamroth (2015), Klamroth et al. (2015), and Dächert et al. (2017) which study such algorithms and are intricately related to our work.

Dächert and Klamroth (2015) present an algorithm tailored to three objectives that decomposes the search region into boxes; to maintain a nonredundant decomposition of the search region, a *neighborhood* concept of the boxes is defined. Klamroth et al. (2015) establish a representation of this search region for general multiobjective problems by *upper bound points* which sufficiently represent the search region. Dächert et al. (2017) derive a neighborhood relation of upper bounds, which yields significant practical improvements in running time. The concepts of boxed regions, upper bounds, and neighborhoods are fundamental to efficiently calculating the weighted Tchebychev decomposition; however, we have generalize and improve upon them in this work.

2. Preliminaries

Mathematically, a *multiobjective discrete optimization problem (MODO)* with $p \geq 2$ objectives can be stated as

$$\begin{aligned} \min \quad & f(x) = (f_1(x), \ldots, f_p(x)) \\ \text{s.t.} \quad & x \in X, \end{aligned} \tag{MODO}$$

where $X \subseteq \mathbb{Z}^n$, for $n \in \mathbb{N}$, is called the *feasible set*, and $f = (f_1, \ldots, f_p) : \mathbb{R}^n \to \mathbb{R}^p$ is the (vector-valued) *objective function*. We call $Y = f(X) = \{y \in \mathbb{R}^p : y = f(x),\ x \in X\}$ the *image set*. Sets $\mathbb{R}^n$ and $\mathbb{R}^p$ the *decision space* and the *criterion space*, respectively.

For $y, \bar{y} \in \mathbb{R}^p$, define the following *component-wise orderings*:

- $y \leqq \bar{y}$ if and only if $y_i \leq \bar{y}_i$ for all $i = 1, \ldots, p$,
- $y \leq \bar{y}$ if and only if $y \leqq \bar{y}$ and $y \neq \bar{y}$, and
- $y < \bar{y}$ if and only if $y_i < \bar{y}_i$ for all $i = 1, \ldots, p$.

Further, the *nonnegative orthant* is defined by $\mathbb{R}^p_{\geqq} := \{t \in \mathbb{R}^p : t \geqq 0\}$. The sets $\mathbb{R}^p_{\geq}$ and $\mathbb{R}^p_{>}$ are defined analogously. Feasible solution x *dominates (strictly dominates)* solution x' if and only if $f(x) \leq f(x')$ $(f(x) < f(x'))$. A feasible solution $x^* \in X$ is *efficient* (*weakly efficient*) if there does not exist another feasible solution $x \in X$ such that $f(x) \leq f(x^*)$ $(f(x) < f(x^*))$. We call an image $y = f(x)$ *(weakly) nondominated* (ND) if x is (weakly) efficient. The set of (weakly) ND images is denoted by Y_N (Y_{wN}), and Y_N is often called the *ND frontier*.

The *weighted sum utility function*, defined using weight vector $\lambda \in \mathbb{R}^p_{\geq}$, is given by $\min_{x \in X}\{\lambda^\top f(x)\}$. The image of every optimal solution is a (weakly) ND image of (MODO) if $\lambda \in \mathbb{R}^p_{>}$ ($\lambda \in \mathbb{R}^p_{\geq}$) (Geoffrion 1968). Efficient solutions which are optimal for weighted sum utility function for some $\lambda \in \mathbb{R}^p_{\geq}$ are called *supported*; their associated images are also called supported and lie on the boundary of the convex hull of Y. Supported ND images that are also extreme points of the convex hull of Y are called *extreme supported ND (ESND)* images. However, the ND frontier of a MODO may include *unsupported* ND images. Moreover, it is not difficult to construct an instance of (MODO) in which a large portion of the ND frontier is unsupported.

The *weighted Tchebychev utility function* (Bowman 1976) is defined with a *reference point* $s \in \mathbb{R}^p$ and weight vector $\lambda \in \mathbb{R}^p_{\geq}$ as

$$\min \{\|f(x) - s\|^\lambda_\infty : x \in X\}, \tag{$\Pi^{TS}(\lambda)$}$$

where $\|y\|^\lambda_\infty := \max_{i=1,\ldots,p}\{|\lambda_i\ y_i|\}$. In practice, the single-objective program in $(\Pi^{TS}(\lambda))$ is modeled by introducing an auxiliary variable. Usually, the reference point is chosen to be the *ideal point* (or *utopia point*), defined as $y^I_i := \min_{x \in X}\{f_i(x)\}$ for $i = 1, \ldots, p$, $(y^U < y^I)$. In the remainder of this work, we assume the following.

Assumption 1. *Without loss of generality, assume the image set is shifted such that $Y \subset \mathbb{R}^p_{>}$. Then the origin is a utopia point, which we fix as the reference point, i.e., $s = \vec{0}$.*

Assumption 1 allows s to be omitted from the objective of $(\Pi^{TS}(\lambda))$ and all equations moving forward.

For weights $\lambda \in \mathbb{R}^p_>$ and reference points $s \leqq y^I$, every optimal solution to $\Pi^{TS}(\lambda)$ is at least weakly efficient for (MODO). If the optimal solution is unique, its image is ND. Conversely, there exists a positive weight vector for each ND image y' such that an optimal solution x' for $(\Pi^{TS}(\lambda))$ satisfies $y' = f(x')$ (Steuer and Choo 1983).

For both weighted utility functions, the set of optimal solutions does not change when the weight vector is multiplied by a positive scalar. Hence, the space of nonnegative weight vectors, $\mathbb{R}^p_\geq$, can be represented entirely by the $(p-1)$-dimensional *weight set* containing weights normalized to have unit sum, denoted by

$$\Lambda := \{\lambda \in \mathbb{R}^p_\geq : \sum_{i=1}^{p} \lambda_i = 1\}.$$

This closed set includes the boundaries. We refer to the interior of Λ by $\text{int}(\Lambda) := \{\lambda \in \mathbb{R}^p_> : \sum_{i=1}^p \lambda_i = 1\}$.

For each ND image, there exists a set of weights (possibly empty) such that the image is optimal for a weighted utility function. Hence, for $y \in Y_N$, we define *weight set components* as $\Lambda(y) := \left\{\lambda \in \Lambda : \|y\|^\lambda_\infty \leq \|\bar{y}\|^\lambda_\infty \text{ for all } \bar{y} \in Y\right\}$ for weighted Tchebychev and $\Lambda^{WS}(y) := \left\{\lambda \in \Lambda : \lambda^\top y \leq \lambda^\top \bar{y} \text{ for all } \bar{y} \in Y\right\}$ for weighted sum. Then $\{\Lambda(y)\}_{y \in Y_N}$ is referred to as the *weighted Tchebychev decomposition*; the weighted sum decomposition is analogous.

2.1. Geometry of Weight Set Components

We summarize from Helfrich et al. (2020) what is known about the geometry of the weighted Tchebychev decomposition. For a given (weakly) ND image, one particular weight is central to the geometric analysis of its weight set component.

Definition 1. For $y \in Y_{wN}$, the *kernel weight*[2] of y, denoted by $\lambda(y)$, is defined as

$$\lambda_i(y) := \frac{1}{y_i} \frac{1}{\sum_{j=1}^p \frac{1}{y_j}} \text{ for } i = 1, \ldots, p.$$

Observe that the level set for the objective function of $(\Pi^{TS}(\lambda))$ is the surface of a hyperrectangle that is centered at the origin with proportions determined by λ. Geometrically, $\lambda(y)$ is the unique weight in Λ such that image y exists on the (positive) vertex of such a hyperrectangle.

[2] Also known as *T-vertex* (Luque et al. 2010) or break-even weight (Karakaya and Köksalan 2021).

Observe from Figure 1 that the weight set components with respect to weighted sum (left) are convex, while the weight set components with respect to weighted Tchebychev (right) are nonconvex. However, the nonconvex polygons in fact share a relaxed form of convexity.

DEFINITION 2. A set $S \subseteq \mathbb{R}^p$ is *star-shaped* if there exists $\bar{y} \in S$ such that $\theta\bar{y} + (1-\theta)y \in S$ for all $y \in S$ and all $\theta \in (0,1)$. Such a point $\bar{y}$ is called a *kernel*.[3]

The following fundamental result motivates the name for Definition 1 and many geometric techniques used in this work. The theorem is stated here without proof.

THEOREM 1 **(Helfrich et al. (2020))**. *Let $y \in Y_{wN}$. Then, $\Lambda(y)$ is a star-shaped set with $\lambda(y)$ as a kernel.*

In Figure 1 and onward, each numbered circle indicates the location of a kernel weight.

2.2. Intersection Sets

The intersections of weight set components are used to define an adjacency structure among ND images, and they are essential to visualizing the boundaries of weight set components. Recall $\dim(\Lambda) = p-1$.

DEFINITION 3. Two ND images, $y^1, y^2 \in Y_N$, are *adjacent* if $\dim(\Lambda(y^1) \cap \Lambda(y^2)) \geq p-2$.[4]

In the weighted sum decomposition, intersections between weight set components have several nice properties, including that they are convex and $(p-2)$-dimensional (Przybylski et al. 2010a). However, in the weighted Tchebychev decomposition, the intersection set is rarely convex and may be full-dimensional, which is properly proven in Helfrich et al. (2020) but here simply means that it contains a $(p-1)$-dimensional ball with positive radius.

In the most common case of intersection sets, a simple structure frequently appears for $p = 3$. Between two adjacent ND images with *distinct* components, the intersection set is a union of two line segments of the form $[\lambda^1, \lambda^2] \cup [\lambda^2, \lambda^3]$ where $\lambda^1, \lambda^2, \lambda^3 \in \Lambda$ and $[a,b] := \{\mu a + (1-\mu)b : 0 \leq \mu \leq 1\}$. All intersection sets in Figure 1(b) are of this simple form, and this case of intersection is explored in Karakaya and Köksalan (2021).

More complex intersections, including full-dimensional overlaps of weight set components, often occur when adjacent ND images coincide in one objective value. Previously,

[3] Definition of star-shaped is simplified from Preparata and Shamos (1988), which allows for a *kernel set*; however we have taken this set to be restricted to a singleton.

[4] This definition aligns with those in Przybylski et al. (2010a).

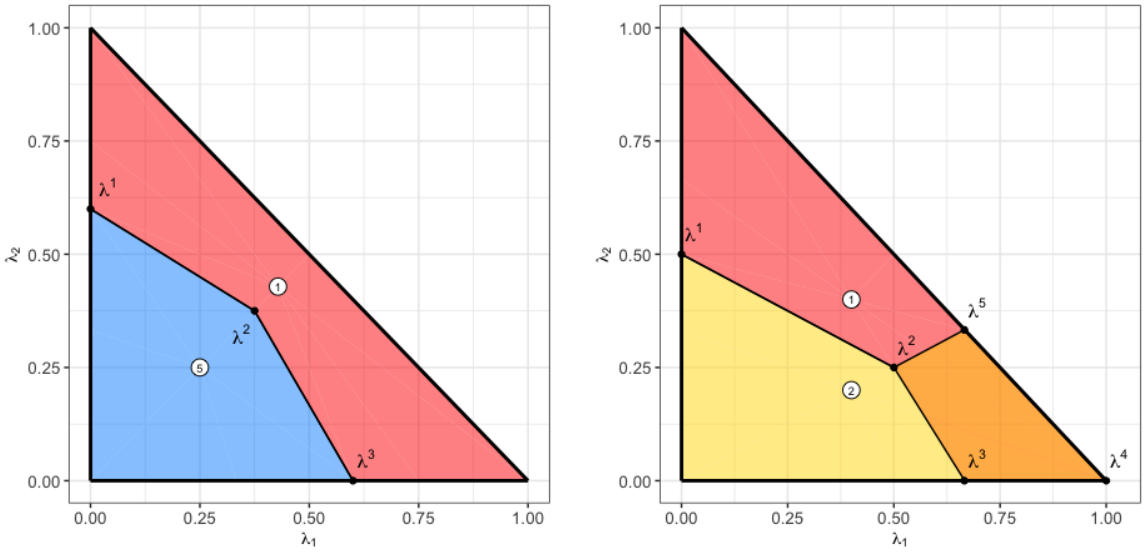


Figure 2 **Weighted Tchebychev decompositions for Y' (Example 1, left) and Y'' (Example ??, right). On left, the two images are distinct in all components, and both images are optimal for $\lambda \in [\lambda^1, \lambda^2] \cup [\lambda^2, \lambda^3]$. On right, images are equal in the first component, and both images are optimal for $\lambda \in [\lambda^1, \lambda^2] \cup$ conv$(\lambda^2, \lambda^3, \lambda^4, \lambda^5)$. The orange polytope illustrates the overlap of the red and yellow weight set components, i.e., $\Lambda(y^1) \cap \Lambda(y^2) =$ conv$(\lambda^2, \lambda^3, \lambda^4, \lambda^5)$.**

Dächert and Klamroth (2015) also documented complications for defining upper bound sets for MODOs with this phenomenon. To the best of our knowledge, these more complex intersections have not been studied before, algorithmically, but they are critical to accurate visualization and evaluation of weighted Tchebychev decompositions.

Example 1. For $Y' = \{y^1, y^5\} = \{(1,2,3),(2,3,1)\}$, observe that images are distinct in every objective value. The weighted Tchebychev decomposition for Y' is illustrated in Figure 2(a). A minimal description of the intersection set is given by the interior weight $\lambda^2 = (0.375, 0.375, 0.25)$ and two weights on the boundary of Λ, $\lambda^1 = (0.0, 0.6, 0.4)$ and $\lambda^3 = (0.6, 0.0, 0.4)$. Hence, $\Lambda(y^1) \cap \Lambda(y^5) = [\lambda^1, \lambda^2] \cup [\lambda^2, \lambda^3]$.

For $Y'' = \{y^1, y^2\} = \{(1,2,3),(1,3,2)\}$, observe that images are equal in the first objective value. The weighted Tchebychev decomposition for Y'' is illustrated in Figure 2(b). Now, the intersection contains a full-dimensional set (orange in lower right). We have $\Lambda(y^1) \cap \Lambda(y^2) = [\lambda^1, \lambda^2] \cup \text{conv}(\{\lambda^2, \lambda^3, \lambda^4, \lambda^5\})$, as illustrated. □

To circumscribe the intersection sets, we use local nadir points and weights. The former has been studied by many different names; our definition generalizes the definition for "upper bound points" by Klamroth et al. (2015), Dächert and Klamroth (2015), and Dächert et al. (2017).

DEFINITION 4. Given a nonempty set of ND images, $\bar{Y} = \{y^1, \ldots, y^{\bar{R}}\} \subseteq Y_{wN}$, we define the *local nadir point (LNP)* with respect to $\bar{Y}$, $y^N(\bar{Y})$, by

$$y_i^N(\bar{Y}) = \max_{r=1,\ldots,\bar{R}} y_i^r \text{ for } i = 1, \ldots, p.$$

We say an image y^r for $r \in \{1, \ldots, \bar{R}\}$ *contributes (in objective i)* to the LNP $n := y^N(\bar{Y})$ if $y_i^r = n_i$ for some $i \in \{1, \ldots, p\}$. For $i \in \{1, \ldots, p\}$, the *ith dimensional contributing set of* n, denoted $C^i(n)$, is the subset of $\bar{Y}$ that contributes to LNP n in objective i. An image y^r *uniquely contributes* to $y^N(\bar{Y})$ in objective i, if $C^i(n) = \{y^r\}$. The *contributing set* is denoted by $C(n) = \cup_{i=1}^p C^i(n)$.

We often make explicit how many images are included in $\bar{Y}$, e.g., if $\bar{R} = 2$, 3, or 4, then we call $y^N(\bar{Y})$ a 2-way, 3-way, or 4-way LNP, respectively. In the case that $\bar{R} = 1$, then $y^N(\{y^0\}) = y^0$. We now define the projection of a LNP to Λ.

DEFINITION 5. We call the kernel weight of a LNP the *local nadir weight* and denote it as $\lambda^N(\bar{Y}) := \lambda(y^N(\bar{Y}))$.

Two LNPs, or their corresponding local nadir weights, are said to *coincide* if they are equal in all p components. It can easily be shown that given contributing set $\bar{Y}$, the weighted Tchebychev utility function has equal value for the local nadir weight $\lambda^N(\bar{Y})$ and for any image in $\bar{Y}$.

Surprisingly, the properties of star-shapedness apply to all (nonempty) intersections of weight set components. (The result of an intersection between star-shaped sets being star-shaped, when the kernel weights do not belong to the intersection, is nontrivial.) See Appendix Section 7.3 (Example 7) for an illustration of this property for Example 1. This geometric property is given succinctly as the following theorem.

THEOREM 2 **(Helfrich et al. (2020))**. *Let $\bar{Y} \subseteq Y_{wN}$. Then,*

$$\bigcap_{y \in \bar{Y}} \Lambda(y) = \Lambda(y^N(\bar{Y})).$$

The intersection $\bigcap_{y \in \bar{Y}} \Lambda(y)$ is star-shaped with kernel $\lambda^N(\bar{Y})$.

Furthermore, Theorem 2 asserts that an intersection of weight set components is fundamentally the same as a weight set component and that its kernel is readily available.

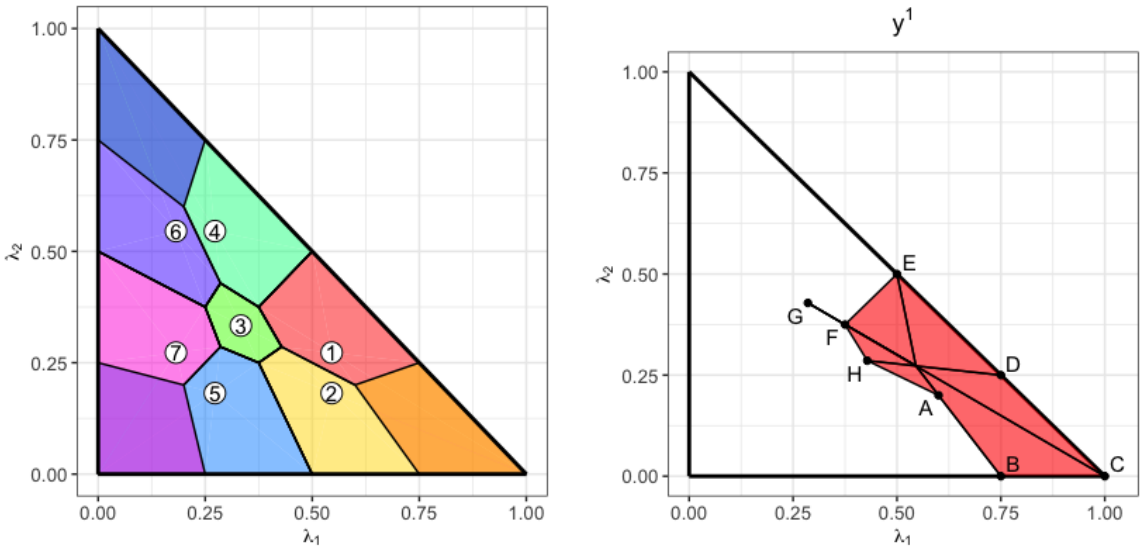


Figure 3 **The weighted Tchebychev decomposition for the running example (Example 2). (a) The following full-dimensional intersections between weight set components can be observed: $\Lambda(y^1) \cap \Lambda(y^2)$ in orange (right corner), $\Lambda(y^4) \cap \Lambda(y^6)$ in navy (top corner), and $\Lambda(y^5) \cap \Lambda(y^7)$ in violet (lower left). (b) Each unique local nadir weight for $\Lambda(y^1)$ is labeled.**

2.3. Running Example

The following example exhibits common complications of computing and representing the weighted Tchebychev decomposition.

EXAMPLE 2. Consider the following (MODO):

$$\min\{(x_1, x_2, x_3): \quad x_1 + x_2 + x_3 = 6, \quad 1 \leq x_i \leq 3 \ \forall i = 1, 2, 3, \quad x \in \mathbb{Z}^3\}.$$

The following sets coincide: $X = Y = Y_N$. There are seven ND images, labeled: $y^1 = (1, 2, 3)$, $y^2 = (1, 3, 2)$, $y^3 = (2, 2, 2)$, $y^4 = (2, 1, 3)$, $y^5 = (2, 3, 1)$, $y^6 = (3, 1, 2)$, and $y^7 = (3, 2, 1)$. The complete weighted Tchebychev decomposition is illustrated in Figure 3(a) as well as a single weight set component, $\Lambda(y^1)$, in Figure 3(b). The remaining weight set components are illustrated in Figure 10 of Appendix Section 7.1.

The first complication is visually representing when weight set components overlap, meaning a pairwise intersection is *full-dimensional*. To make this visibly apparent, we choose to plot each colored component as *translucent* (half-transparent) so that the overlapping components are indicated by mixing of colors. For example, the orange corner of Λ in Figure 3(a) is an intersection between components $\Lambda(y^1)$ in red and $\Lambda(y^2)$ in yellow. Regions without numbered labels (e.g., all three corners of Λ) represent overlapping components.

The second complication is that *lower-dimensional* regions also occur. For three objectives, this appears as a 1-dimensional line segment protruding from a component, which we call a *whisker*. In Figure 3(b), the weight set component $\Lambda(y^1)$ contains one whisker, i.e., the line segment $[F, G]$. Every weight set component in the running example, $\Lambda(y^1), \ldots, \Lambda(y^7)$ (individually illustrated in Figure 10 of the Appendix Section 7.1) includes at least one whisker. $\square$

2.4. Boundary Weights

So far we have focused on LNPs for $\bar{Y} \subset Y_{wN}$ which yield a weight $\lambda^N(\bar{Y})$ in the interior of Λ. However, weight set components (and their intersections) must also include weights on the boundary of Λ. Properly representing weights on the boundary of Λ requires the inclusion of p standard dummy images (with all zero objective values except one large value). We let $Y_B := \{y^{b(1)}, \ldots, y^{b(p)}\}$ denote the set of dummy images and hereon assume that $Y_B \subset Y$. Details are included in Section 7.2 of the Appendix.

3. Maximal LNPs and Complete Contributing Sets

LNPs necessary for the weighted Tchebychev decomposition require two features. First, what is most commonly studied, is the concept of a *maximal* LNP (Klamroth et al. 2015). Let $e_j \in \mathbb{R}^p$ be a vector of zeros except for a one in the jth component.

Definition 6. A given LNP $y \in \mathbb{R}^p$, where $y = y^N(\bar{Y})$ for some $\bar{Y} \subseteq Y_N$, is *maximal* with respect to Y_N if y is weakly nondominated (with respect to Y_N) and $y + e_j$ is strictly dominated by some $y^j \in Y_N$ for all $j = 1, \ldots, p$.

Note that our definition of a maximal LNP fundamentally aligns with the definition for an upper bound in Klamroth et al. (2015) (cf. Proposition 2.6). While their definition only accounts for k-way LNPs where $k \geq p$, LNPs where $k < p$ (we call "lower order LNPs") are essential in the weighted Tchebychev decomposition.

For the second property, observe that due to ND images being equal in objective values, a 2-way (k-way) LNP may coincide with a 3-way ($k+1$-way) LNP. Therefore, there is not a one-to-one relationship between contributing sets and LNPs, which motivates the following definition.

Definition 7. For a given LNP, its contributing set $\bar{Y}$ is *complete* with respect to Y_N if for all $y' \in Y_N \setminus \bar{Y}$, $y^N(\bar{Y}) \neq y^N(\bar{Y} \cup \{y'\})$.

In general, k-way LNPs for $k > p+1$ (we call "higher-order LNPs") are prevalent where adjacent ND images are equal in one or more objectives. Hence, an algorithm that omits higher-order points will also omit adjacency information.

EXAMPLE 3. Consider the following list of LNPs defined with images from Example 2:

$$y^N(\{y^1, y^2\}) = (1,3,3), \tag{1}$$
$$y^N(\{y^1, y^2, y^3\}) = (2,3,3), \tag{2}$$
$$y^N(\{y^1, y^2, y^3, y^4\}) = (2,3,3), \tag{3}$$
$$y^N(\{y^1, y^2, y^3, y^4, y^5\}) = (2,3,3), \tag{4}$$
$$y^N(\{y^1, y^2, y^3, y^4, y^5, y^6\}) = (3,3,3), \text{ and,} \tag{5}$$
$$y^N(\{y^1, y^2, y^3, y^4, y^5, y^6, y^7\}) = (3,3,3). \tag{6}$$

First, note that LNPs in (5) and (6) are strictly dominated by $y^2 = (2,2,2)$, but LNPs in (1)-(4) are weakly ND with respect to Y_N. Since $(2,3,3) + e_1$ results in a strictly dominated image, $(2,3,3)$ is a *maximal* LNP. Second, since LNPs coincide in (2)-(4), we say that the contributing set $\{y^1, y^2, y^3, y^4, y^5\}$ is *complete* for the LNP. (Contributing sets in (1) and (6) are also complete.) Hence, (4) both gives a maximal LNP and its complete contributing set, which will be essential for the WSD. □

As ND images are added to a contributing set, the coordinates of the LNP may increase or remain the same, and the resulting LNP may or may not be weakly ND. Completeness is concerned with the former condition, and maximality is concerned with the latter condition. In general, the weighted Tchebychev decomposition requires *all maximal LNPs*, each with its *complete contributing set*. The remainder of this section presents how to compute these maximal LNPs and complete contributing sets efficiently. Section 4 presents how this combined information is used to plot the weight set components and compute their area.

3.1. Box-Based Algorithm

In Section 1.1, we introduced the class of box-based algorithms for MODO. We now review an existing box-based approach for computing the sets of ND images and maximal LNPs, and then Section 3.2 extends it to additionally compute the complete contributing sets.

Pseudocode is given in Algorithm 1 (modified from Dächert and Klamroth (2015)). The algorithm takes an image set Y and returns the ND set Y_N; it is understood that Y is not known explicitly but can be represented implicitly by a MODO formulation. Each box is

Algorithm 1 Box-Based Algorithm for Generating ND Set

Require: Image set $Y \in \mathbb{R}^p_>$ (represented as a MODO formulation), optimization oracle $opt(Y, B)$.

Ensure: Nondominated set $Y_N \subseteq Y$.

1: $Y_N \leftarrow \emptyset$
2: $\mathcal{B}_1 \leftarrow$ {InitStartingBox(Y)}
3: Step $s \leftarrow 1$
4: **while** $\mathcal{B}_s \neq \emptyset$ **do**
5: $\quad B \leftarrow$ BoxSelectionPolicy($\mathcal{B}_s$)
6: $\quad y^s := opt(Y, B)$
7: $\quad$ **if** $y^s = \emptyset$ **then**
8: $\quad\quad \mathcal{B}_{s+1} \leftarrow \mathcal{B}_s \setminus B$
9: $\quad$ **else**
10: $\quad\quad Y_N \leftarrow Y_N \cup \{y^s\}$
11: $\quad\quad \mathcal{B}_{s+1} \leftarrow$ UPDATE($\mathcal{B}_s, B, y^s$) [Algorithm 2]
12: $\quad$ **end if**
13: $\quad s \leftarrow s + 1$
14: **end while**
15: **Return** Y_N

defined with respect to its uppermost (for minimization problems) corner, which in fact is a LNP defined with found ND images. In practice, an integer program (IP) is solved to find a new ND image in a given box, if one exists; the precise form of the IP differs depending on the design of the box-based algorithm. Represent the IP as an input oracle, denoted by $opt(Y, B)$ for box B, which returns $y \in Y_N \cap B$ if $Y_N \cap B \neq \emptyset$ or $y = \emptyset$ otherwise.

Initialization includes defining an empty ND set (line 1) and the initial search region as a box (line 2). Boxes are contained within a queue, called $\mathcal{B}_s$ for step s, from which a box is selected to process (line 5) and to which new boxes are added for future processing (lines 8 and 11). The algorithm continues until an empty queue is reached, after which the set of ND images is returned (line 15).

For every step s, the algorithm proceeds as follows: First, a box B is selected from queue $\mathcal{B}_s$ for processing based on some box selection policy (line 5). Then, the oracle *opt* returns a new ND image from the interior of the box, if one exists (line 6). If no ND image is found (equivalently, the IP is infeasible, the interior of the box is empty, and the LNP is weakly ND), then the box is removed from the queue (lines 7-8). Otherwise, if a new ND image is found, denoted by y^s, then y^s is added to Y_N (line 10), and the queue of boxes is updated by UPDATE (line 11) for the next step, $\mathcal{B}_{s+1}$.

Three crucial design elements of Algorithm 1 impact its efficiency: (i) the box selection policy, (ii) the IP used in place of the oracle, and (iii) the method to update the queue. Elements (ii) and (iii) have received great attention; see (Boland et al. 2016, Tamby and Vanderpooten 2020) and (Klamroth et al. 2015, Dächert and Klamroth 2015, Dächert et al. 2017), respectively, for discussions and strategies. In Section 5, we study element (i) and how the (approximate) WSD can better inform its design.

The notable difference between the criterion space and weight space paradigms is the elimination of an "empty" box in line 8, along with its LNP. From the perspective of the criterion space, the LNPs used to define boxes are a necessary by-product of the algorithm: LNPs are computed throughout the procedure but then are usually deleted. However, for the weighted Tchebychev decomposition, the LNPs are essential, in particular the maximal ones. When no new ND image is found in a box, then its associated LNP is weakly ND and therefore *maximal* with respect to Y_N and should therefore be returned for use in the weighted Tchebychev decomposition. The queue of boxes should then be understood as the current list of LNPs that are maximal with respect to the known ND images but are not yet *certified* as maximal with the complete ND set. In line 8, when a box is removed, its maximal LNP should be added to a list of LNPs which are certified to be maximal with respect to Y_N. In summary, at each step of Algorithm 1, either a new ND image is discovered or a LNP is certified as maximal, as demonstrated by Example 10 in Appendix Section 7.3.

3.2. Extended Algorithm

The Klamroth et al. (2015) method is an UPDATE subroutine which updates a set of maximal LNPs when given a new ND image; Algorithm 2 includes extension of this procedure (in red) such that the contributing sets are complete. The original algorithm guarantees at least one image per dimensional contributing set, however our contribution is to guarantee completion. Input is denoted by horizontal bars, e.g., the input set of maximal LNPs as $\bar{N}$ and the input image as $\bar{y}$. The algorithm takes as input the dimensional contributing sets of each LNP, denoted by $\bar{C}^j(n)$ for all $j \in \{1,\dots,p\}$ and $n \in \bar{N}$. The output includes a set of LNPs, N, and two "versions" of the contributing sets, $\{D^j(n)\}_{j=1..p,n\in N}$ and $\{C^j(n)\}_{j=1..p,n\in N}$. The former set, $D^j(n)$, represents the contributing sets as defined in the original Klamroth et al. (2015) algorithm. The latter set, $C^j(n)$, represents our addition to the procedure to capture the complete contributing sets. The lines associated with our

Algorithm 2 Extended UPDATE Subroutine: Generate Maximal LNP Set.

Require: Set of maximal LNPs $\bar{N} \subset \mathbb{R}^p$, contributing sets $\{\bar{C}^j(n)\}_{j=1..p,n\in\bar{N}}$, and ND image $\bar{y}$.

Ensure: Updated set of maximal LNPs N, contributing sets $\{D^j(n)\}_{j=1..p,n\in N}$, and complete contributing sets $\{C^j(n)\}_{j=1..p,n\in N}$.

1: $D^j(n), C^j(n) \leftarrow \bar{C}^j(n) \quad \forall j = 1..p, n \in \bar{N}$
2: $A \leftarrow \{n \in \bar{N} : \bar{y} < n\}$
3: $P \leftarrow \emptyset$
4: **for** $n \in \bar{N}$ and $j \in \{1,\dots,p\}$ such that $\bar{y}_j = n_j$ **do**
5: **if** $(\bar{y}_{-j} < n_{-j})$ **then** $D^j(n) \leftarrow D^j(n) \cup \{\bar{y}\}$
6: **if** $(\bar{y}_{-j} \leq n_{-j})$ **then** $C^j(n) \leftarrow C^j(n) \cup \{\bar{y}\} \quad (*)$
7: **end for**
8: **for** $n \in A$ **do**
9: **for** $j = 1,\dots,p$ **do**
10: $m_j(n) \leftarrow \max_{k\neq j} \min\{y_j : y \in \bar{C}^k(n)\}$
11: **if** $\bar{y}_j > m_j(n)$ **then**
12: $n' \leftarrow y^N(\{\bar{C}^k(n)\}_{k\neq j} \cup \{\bar{y}\}) = [\bar{y}_j, n_{-j}]$
13: $P \leftarrow P \cup \{n'\}$
14: $D^j(n') \leftarrow \{\bar{y}\}$
15: $C^j(n') \leftarrow \{\bar{y}\} \cup \{y \in C^k(n) : y \leq n', y_j = \bar{y}_j,\ k = 1,\dots,p\} \quad (*)$
16: **for** $k \in \{1,\dots,p\} \setminus \{j\}$ **do**
17: $D^k(n') \leftarrow \{y \in D^k(n) : y_j < \bar{y}_j\}$
18: $C^k(n') \leftarrow \{y \in C^k(n) : y_j \leq \bar{y}_j\} \quad (*)$
19: **end for**
20: **end if**
21: **end for**
22: **end for**
23: $N \leftarrow (\bar{N} \setminus A) \cup P$
24: **return** N, $\{D^j(n)\}_{j=1..p,n\in N}$, $\{C^j(n)\}_{j=1..p,n\in N}$

contribution are by red and $(*)$.[5] Throughout the algorithm, we use as shorthand $[y^1_j, y^2_{-j}]$ to represent the vector where the jth component is given by y^1 and all other components are given by y^2.

The extended update method begins with new contributing sets initialized as equal to input contributing sets (line 1). The set A represents the LNPs that are strictly dominated by the new ND image and must therefore be replaced (defined in line 2). The set P is initialized as empty (line 3) and will contain the new maximal LNPs as they are constructed. After operating over for loops (lines 4-22), the method will conclude by removing A from

[5] In practice, only one set is of interest, so either the lines with $D^j(n)$ or the lines with $C^j(n)$ will be omitted.

the input set of LNPs, adding the new LNPs in P (line 23), and returning the maximal LNPs along with contributing sets (line 24).

Here, we highlight the rationale for the modifications which provide the complete contributing sets (red in Algorithm 2):

(Line 6.) The for loop over $n \in \bar{N}$ (lines 4-7) of the original algorithm is intended to add to contributing sets $D^j(n)$ only for the particular case that for some $n \in \bar{N}$, $\bar{y}_j = n_j$ for exactly one index j. However, if this is true for two indices (or more for $p > 3$), then line 6 accounts for this image's contribution to the LNP n by adding to $C^j(n)$.

(Line 15.) The for loop over $n \in \bar{A}$ (lines 8-22) of the original algorithm generates the new LNPs that replace the removed LNPs. A straight-forward p-split approach would replace n with p new LNPs each iteration, some of which may not be maximal. However, the condition in line 11 reduces the overall effort required by determining which of the p LNPs are necessary to add; see Klamroth et al. (2015) for details. The new LNPs are defined in line 12 and added to P in line 13. The original approach for defining the new contributing set is to give a *minimal* set of contributing images per component. Therefore, the new ND image alone is sufficient for defining $D^j(n')$ (line 14). However, for complete contributing sets $C^j(n')$, it is necessary to consider all other images that contribute to the LNP that are also equal in the jth component (line 15).

(Line 18.) This addition follows the same rationale as for line 6.

The following illustrates an example where the extensions yield contributing sets $C^j(n)$ that are complete whereas sets $D^j(n)$ are *not complete* according to the original procedure.

Example 4. Note that Algorithm 2 is sensitive to the order of images used to update. Let $\bar{Y} = \{y^3, y^4, y^{b(1)}, y^{b(2)}, y^{b(3)}\}$ represent the set of images used to update before $y^1 = (1,2,3)$ is input to update. Let $\bar{N}$ contain all maximal LNPs with respect to $\bar{Y}$, and contributing sets $\bar{C}^j(n)$ be complete with respect to $\bar{Y}$ for all $j = 1, \ldots, p$ and $n \in \bar{N}$. In particular, consider $n := y^N(y^{b(1)}, y^3, y^4) = (M, 2, 3) \in \bar{N}$ with $\bar{C}^1(n) = \{y^{b(1)}\}$, $\bar{C}^2(n) = \{y^3\}$, and $C^3(n) = \{y^4\}$. Now with y^1 as the input ND image, it satisfies $y^1_j = n_j$ for both $j = 2, 3$, but it does not satisfy $y^1_{-j} < n_{-j}$ for either $j = 2, 3$. Hence, the condition of line 5 is not satisfied, and so y^1 is not added to $D^j(n)$ for $j = 2, 3$. On the other hand, $y^1_{-j} \leq n_{-j}$ is satisfied for $j = 2, 3$, so y^1 is added to $C^j(n)$ for $j = 2, 3$. The LNP n is returned within set N, regardless; however, the contributing sets $D^2(n), D^3(n)$ will be missing y^1 whereas the *complete* contributing sets $C^2(n), C^3(n)$ will include y^1.

Consider the next iteration where $\bar{Y} = \{y^1, y^3, y^4, y^{b(1)}, y^{b(2)}, y^{b(3)}\}$, and suppose *incomplete* contributing sets are input from the previous step, i.e., $\bar{C}^2(n) = \{y^3\}$ and $\bar{C}^3(n) = \{y^4\}$. Let $y^6 = (3, 1, 2)$ be input next. Note that $y^6 < n$, and so $n \in A$ by line 2. For $j = 1$, the condition in line 11 is satisfied, so $n' := y^N(\bar{C}^2(n) \cup \bar{C}^3(n) \cup \{y^6\}) = (3, 2, 3)$ is defined. By line 14, $D^1(n') = \{y^6\}$, and by lines 16-17, $D^2(n') = D^2(n)$ and $D^3(n') = D^3(n)$. Note that the local nadir weight for n' is equivalent to G (Table 2), which is the endpoint of whisker $[F, G]$ in Figure 3.

Therefore, the original UPDATE procedure (Klamroth et al. 2015) would not indicate that G belongs to $\Lambda(y^1)$, and as a result it *misses the whisker* $[F, G]$. However, if input contributing sets, $\bar{C}^j(n)$, are complete, then output contributing sets $C^j(n)$ will be complete. Hence the extended procedure will capture this whisker. □

As Algorithm 2 is an update procedure, we must clarify the initial state for the sake of proving results. The following assumption applies to all subsequent proofs.

Assumption 2. *We assume the initial set of ND images is* $\bar{Y}^0 := \{y^0, y^{b(1)}, \ldots, y^{b(p)}\}$, *i.e., all dummy images with one ND image; the initial set of maximal LNPs is* $\bar{N} = \{n^1, \ldots, n^p\}$ *where* $n^j := \{y^N(Y^0 \setminus \{y^{b(j)}\})\}$; *and for all* $j = 1..p$, *the complete, dimensional contributing sets for* n^j *are* $\bar{C}^j(n^j) := \{y^0\}$ *and* $\bar{C}^k(n^j) := \{y^{b(k)}\}$ *for all* $k \neq j$.

Note that by this initialization, each local nadir weight associated with n^i corresponds to a vertex of Λ, so essentially the weighted Tchebychev decomposition is initialized as $\Lambda(y^0) = \Lambda$.

For the following claims, let $\bar{Y} := \{y^1, \ldots, y^n\} \subset Y_N \setminus Y_B$ be the set of first $n \geq 1$ ND images input into Algorithm 2, which provides maximal LNPs $\bar{N}$ and complete contributing sets $\{\bar{C}^j(n)\}_{j=1..p, n \in \bar{N}}$ such that $\bar{C}^j(n) \subseteq \bar{Y}$ for all j, n. (For sake of readability, we do not explicitly denote the dependence on the image set $\bar{Y}$.) For update number $n + 1$, image $\bar{y} \in Y_N \setminus \bar{Y}$ is input along with $\bar{N}$ and $\{\bar{C}^j(n)\}_{j=1..p, n \in \bar{N}}$.

Theorem 3. *If input set* $\bar{N}$ *is the set of all maximal LNPs with respect to* $\bar{Y}$, $\{\bar{C}^j(n)\}_{j=1..p, n \in \bar{N}}$ *are nonempty, and* $\bar{y}$ *is ND, then the output set* N *is the set of maximal LNPs with respect to* $\bar{Y} \cup \{\bar{y}\}$.

Proven in Klamroth et al. (2015). □

It remains to be proven that the output contributing sets C^j are complete. Given image $\bar{y}$ and LNP n, it is impossible to simultaneously satisfy $\bar{y} < n$ (in line 2) and $\bar{y}_j = n_j$ (in

line 4). Hence, during one call of Algorithm 2, at most one of the following two operations will occur per $n \in \bar{N}$. First, when $\bar{y}$ is added to a contributing set of n (lines 5-6), the components of n remain unchanged. Second, when $n \in A$, there exists some $n' \in P$ (i.e., n is removed from $\bar{N}$ and new n' is added) such that $n' \leq n$ and for some j, $\bar{y}_j = n'_j < n_j$ and $D^j(n') = \{\bar{y}\}$. The following theorem shows that the relaxed conditions used for including images in $C^j(n)$ result in complete contributing sets.

THEOREM 4. *Let $\bar{y}$, $\bar{N}$, and $\{\bar{C}^j(n)\}_{j=1..p,n\in\bar{N}}$ denote the input to Algorithm 2, and let N and $\{C^j(n)\}_{j=1..p,n\in N}$ denote the output. If $\bar{N}$ contains all maximal LNPs and the contributing sets $\{\bar{C}^j(n)\}_{j=1..p,n\in\bar{N}}$ are complete, then the output contributing sets, $C^j(n)$, are complete for all $j = 1,\ldots,p$ and $n \in N$.*

Proof. By definition of output set N in line 23, we have three cases for each $n \in N$: (1) $n \in \bar{N}$ and $\bar{y}$ does not dominate n, (2) $n \in \bar{N}$ and $\bar{y} \leq n$, or (3) $n \notin \bar{N}$. Case (1) is trivial: Lines 4-5 are irrelevant since the condition in line 4 does not hold for any $j \in \{1,\ldots,p\}$, and lines 7-19 are irrelevant since $n \notin A$. Since n is not dominated by $\bar{y}$, the image cannot contribute to any of the LNP's components. Hence if the input contributing sets $\{\bar{C}^j(n)\}_{j=1..p}$ are complete, then the output contributing sets $\{C^j(n)\}_{j=1..p}$ are also complete.

Observe that conditions in lines 4 and 8 are mutually exclusive. Therefore, the remaining cases essentially refer to either lines 4-6 or lines 8-22. Case (2) occurs when $\bar{y}$ dominates but not strictly dominates $n \in \bar{N}$, and therefore the contributing sets are updated without removing n. However, case (3) occurs when $\bar{y}$ strictly dominates some $n \in \bar{N}$, and thus new $n' \in P$ is defined with its contributing sets.

Case (2): Suppose $n \in \bar{N}$ and $\bar{y} \leq n$ with at least one equal component. Let $J := \{j = 1,\ldots,p : \bar{y}_j = n_j\}$, which is nonempty. By definition in line 2, $J \neq \emptyset$ implies $n \notin A$, so n is included in the output set N. Given the inequality in line 6, all $j \in J$ satisfy $\bar{y}_{-j} \leq n_{-j}$, so $\bar{y}$ is added to $C^j(n)$ for all j. Lines 8-22 are irrelevant. We must only guarantee that the image $\bar{y}$ is added to all necessary contributing sets. This follows from the for loop in line 4 iterating over all $j = 1,\ldots,p$. The result therefore holds.

Case (3): Suppose $\bar{y} < n$. By definition in line 2, $n \in A$. If $P = \emptyset$, then there is nothing to show (although this case never happens). Otherwise, for at least one $j \in \{1,\ldots,p\}$, $\bar{y}_j > m_j(n)$. Consider one such index j, and let n' be the corresponding LNP defined in

line 12. Note that $n' \leq n$ and $\bar{y}_j = n'_j < n_j$. It remains to prove that the contributing sets, $\{C^k(n')\}_{k=1..p}$, are maximal.

First, observe that $C^j(n') = \{\bar{y}\} \cup \{y \in C^k(n) : y \leq n', y_j = \bar{y}_j, k = 1, \ldots, p\}$ (by line 15) is incomplete only if the initialization of $C^k(n)$ for some k is incomplete. Since $C^k(n)$ are initialized to be $\bar{C}^k(n)$ for all $k = 1, \ldots, p$, this leads to a contradiction.

Second, suppose for $k \neq j$ that $C^k(n')$ is not complete. By definition, $\bar{y} \in C^k(n')$. Then, there exists some $y' \in \bar{Y}$ such that $y'_k \leq n'$ and $y'_k = \bar{n}'_k$. Since $n'_k = n_k$, then $y' \leq n$ and $y'_k = n_k$, so y' contributes to n in the kth component. By assumption of completeness, $y' \in \bar{C}^k(n)$, and so we have a contradiction since $C^k(n)$ is initialized by $\bar{C}^k(n)$.

By exhausting all possibilities, we have that all contributing sets $\{C^k(n')\}_{k=1..p}$ are complete for all $n' \in P$. Thus, the claim is proven in all cases. □

Assumption 3. *Since we have now proven how to compute complete contributing sets, hereon we let $C^j(n)$ represent the complete, jth-dimensional contributing set for LNP n.*

4. Plotting and Computing Area

This section is dedicated to plotting and computing area of the nonconvex weight set components for the weighted Tchebychev decomposition. While there exist common subroutines to determine the convex hull of a set of points, to our knowledge, there are no analogous routines for a "star-shaped hull." However, the highly structured geometry of the weight set components is amenable to *triangulation*, i.e., covering with a set of triangles without intersection between interiors, as illustrated by Figure 3(b). Once triangulated, it is straightforward to plot the component and compute its total area. Previously, Karakaya and Köksalan (2021) relied on a complex integration approach for computing area. For this section, we restrict attention to the $p = 3$, and so we aim to plot a 2-dimensional representation of weight set components and compute their area.

The following is a tailored procedure for computing a minimal representation of the weight set components by triangulation when the maximal LNPs and complete contributing sets are available. Let N represent the set of maximal LNPs for Y_N, and let the complete contributing sets for LNPs be represented by $\{C(n)\}_{n \in N}$. (Section 5 presents how a subset of Y_N may be used to compute an approximate WSD on-the-fly.)

First, consider a maximal LNP, n, which is weakly ND by definition. Given its complete contributing set, $C(n)$, if $\bar{Y} \subset C(n)$, then $y^N(\bar{Y})$ is a LNP that is weakly ND and bounded

Algorithm 3 ImpliedLNPs (Recursive)

Require: Contributing set $C := C(n)$

Ensure: Set S of pairs (n, C) where n is the LNP with contributing set C

```
S ← {(y^N(C), C)}
for y ∈ C do
  C⁻ ← C \ {y}
  S ← S ∪ (y^N(C⁻), C⁻)
  if |C⁻| > 2 then
    S ← S∪ImpliedLNPs(C⁻)
  end if
end for
return unique(S)
```

above by n, i.e., $y^N(\bar{Y}) \leqq n$. Furthermore, the set of complete contributing sets *imply all weakly ND LNPs.* The set of LNPs implied from one complete contributing set is generated by a recursive-style function, *ImpliedLNPs* (Algorithm 3), where the recursion is performed by iteratively removing one image from the contributing set.

The pseudocode for the plotting procedure is given in Algorithm 4. For readability, we discuss LNPs and local nadir weights as interchangeable. For each $y \in Y_N$, the perimeter of $\Lambda(y)$ is circumscribed by (weakly ND) LNPs to which y contributes. Therefore, the set of maximal LNPs to which y contributes (M in line 2) is used to generate all the implied LNPs to which y also contributes. In line 3, this set of LNPs is generated by a recursive-style function, *ImpliedLNPs*, and is denoted by L; the associated complete contributing sets C are then partitioned into dimensional contributing sets in line 4 as $\{C^1, C^2, C^3\}$.

The perimeter of $\Lambda(y)$ is then computed counter-clockwise by ordering three subsets defined by the dimensional contributing sets. The fundamental approach for each contributing set follows the same three steps. Let $i \in \{1, 2, 3\}$. First, the subset of L for which y belongs to C^i is determined (e.g., line 6). Since all LNPs in this set are equal in the ith component, then reduce dimension (omitting i) to a biobjective perspective. Second, the *upper envelope* of the LNPs is computed. This requires removing images that strictly dominate others (e.g., in line 7). Then the LNPs are ordered in the natural ordering for biobjective problems: decreasing in one objective and increasing in the other (e.g., line 8 where $-y_2$ indicates descending order for second dimension and y_3 indicates ascending order for third dimension). In effect, this upper envelope provides the outer-most descrip-

Algorithm 4 Plot weighted Tchebychev decomposition

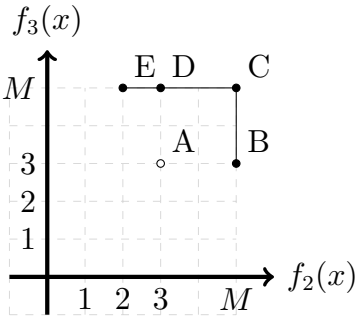
Require: ND set Y_N, maximal LNPs $N \subset \mathbb{R}^3$ with complete contributing sets $\{C(n)\}_{n\in N}$.

Ensure: Triangulations of weight set components $\{\Lambda(y)\}_{y\in Y_N}$ along with area and borders given by $\{\text{area}(y)\}_{y\in Y_N}$ and $\{\text{border}(y)\}_{y\in Y_N}$, respectively.

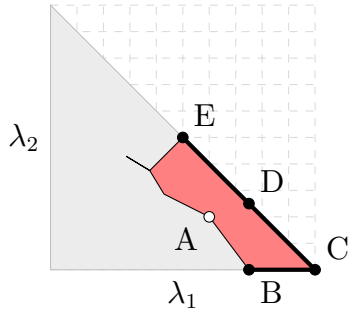
1: **for** $y \in Y'$ **do**
2: $M \leftarrow \{n \in N : y \in C(n)\}$ (set of maximal LNPs to which y contributes)
3: $L, C \leftarrow$ImpliedLNPs(M)
4: $C^1, C^2, C^3 \leftarrow$partitionByDimension(C)
5: # Perimeter defined by C^1:
6: $L^1 \leftarrow \{n \in L : y \in C^1(n)\}$
7: $L^1 \leftarrow$RemoveDominating$(L^1, (y_2, y_3))$
8: $L^1 \leftarrow$Order$(L^1, (-y_2, y_3))$
9: # Perimeter defined by C^2:
10: $L^2 \leftarrow \{n \in L : y \in C^2(n)\}$
11: $L^2 \leftarrow$RemoveDominating$(L^2, (y_3, y_1))$
12: $L^2 \leftarrow$Order$(L^2, (-y_3, y_1))$
13: # Perimeter defined by C^3:
14: $L^3 \leftarrow \{n \in L : y \in C^3(n)\}$
15: $L^3 \leftarrow$RemoveDominating$(L^3, (y_1, y_2))$
16: $L^3 \leftarrow$Order$(L^3, (-y_1, y_2))$
17: # Join perimeters and triangulate:
18: $P \leftarrow$ join(L^1, L^2, L^3)
19: $\Lambda(y) \leftarrow \emptyset$
20: **for** $(n^i, n^{i+1}) \in P$ **do**
21: $T \leftarrow \text{conv}\{\lambda(n^i), \lambda(n^{i+1}), \lambda(y)\}$
22: **if** $T \in \Lambda(y)$ (duplicate) **continue**
23: $\Lambda(y) \leftarrow \Lambda(y) \cup T$
24: area$(y) \leftarrow$area$(y)+$area(T)
25: border$(y) \leftarrow \{[\lambda(n^i), \lambda(n^{i+1}]\}$
26: **end for**
27: **end for**
28: **Return** $\Lambda(y)$, area(y), border(y) for all $y \in Y'$

tion for triangulation of the weight set component. Figure 4 illustrates the procedure and results from the biobjective perspective.

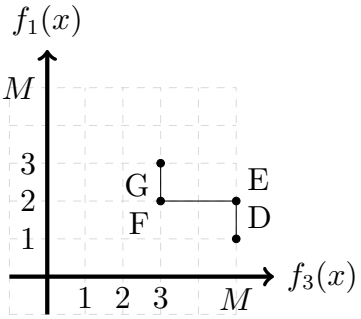
Once L^1, L^2, and L^3 are computed, they are joined and denoted as P (line 18) which is sufficient to describe the full perimeter. The remaining loop (lines 20-27) uses local nadir weights associated with every adjacent pair of LNPs in P to define a triangle (the kernel weight is the third vertex). Line 22 checks if this triangle is a duplicate (i.e., has already been added to this weight set component). If it is a duplicate, then this triangle is discarded



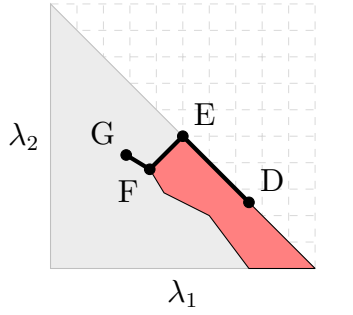
(a) Initial set L^1 projected onto (f_2, f_3). LNP A is strictly dominates C and is removed.



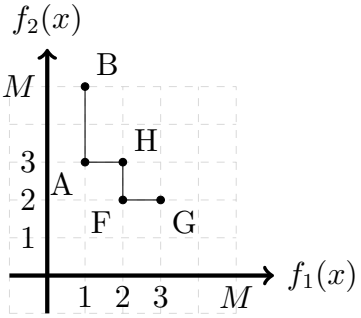
(b) Perimeter of $\Lambda(y^1)$ defined by local nadir weights from the final L^1.



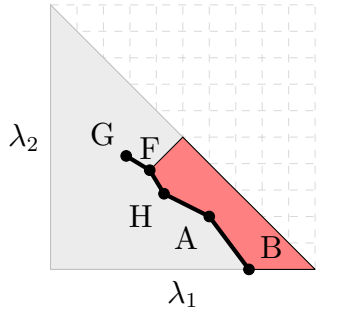
(c) Set L^2 projected onto (f_3, f_1) and ordered.



(d) Perimeter of $\Lambda(y^1)$ defined by local nadir weights from L^2.



(e) Set L^3 projected onto (f_1, f_2) and ordered.



(f) Perimeter of $\Lambda(y^1)$ defined by local nadir weights from L^3.

Figure 4 **Computing the perimeter of weight set component $\Lambda(y^1)$ using biobjective projections of sets L^1, L^2, and L^3.**

and the remaining steps of the loop are skipped; otherwise, the triangle is added to the description of $\Lambda(y)$ (line 23), and the component's area and border are updated (lines 24 and 25, respectively). Note that the loop in line 20 should include the triangle between $P.last$ and $P.first$. Once the for loop is complete, the algorithm returns the weight set

components (each as a set of triangles) along with computed area and border for each (line 28).

Consider a common case where, for triangle with exterior vertices $\lambda(n^i)$ and $\lambda(n^{i+1})$, that these vertices are colinear with the kernel weight. (A simple case to consider is when a 3-way LNP coincides with a 2-way LNP, so $\lambda(n^i) = \lambda(n^{i+1})$.) Then the triangle is in fact a line segment with zero area. However, as long as $\lambda(n^i) \neq \lambda(n^{i+1})$, then the line segment $[\lambda(n^i), \lambda(n^{i+1})]$ does contribute to the *border* of $\Lambda(y)$; this is in fact how whiskers are captured. Hence, in the colinear case, updating the area may be skipped, but updating the border representation should not.

EXAMPLE 5. Consider computing the weight set component $\Lambda(y^1)$ in Example 2. The four maximal LNPs are labeled C, E, G, and H (detailed in Table 2 in Appendix 7.3), and we use the labels for the LNPs and local nadir weights interchangeably. In Algorithm 4 step 2, $M = \{C, E, G, H\}$. Step 3 generates the set of implied LNPs $L = \{A, B, C, \ldots, G, H\}$, as labeled in Figure 3(b). The partitioning and ordering of L^1, L^2, L^3 sets are illustrated in Figure 4.

First, consider $L^1 = \{A, B, C, D, E\}$ since these LNPs contribute in the first component. These LNPs are projected onto the (f_2, f_3) biobjective criterion space in Figure 4(a). LNP C strictly dominates LNP A, so the latter is removed by step 7. Then the LNPs (and local nadir weights) are ordered as B, C, D, E. Figure 4(b) highlights the resulting portion of the perimeter of $\Lambda(y^1)$. Figure 4(c)-(e) illustrates the procedure for L^2 and L^3.

Observe the following:

- The sequence of LNPs from L^2 includes D and E again meaning that, when joined, the sequence (D,E,D,E) occurs. Duplicated triangles detected by step 19 prevent from "double counting" these triangles and their areas/borders.
- The whisker $[F, G]$ is captured by the terminal sequence $(F, G) \in L^2$ followed by the initial sequence $(G, F) \in L^3$.
- While LNP A is removed from L^1, since it is dominated by C in the (f_2, f_3) perspective, it is not dominated in L^3 and therefore contributes to the overall perimeter. □

4.1. Proof of Coverage

In Appendix Section 7.4, Theorem 5 states that the construction of a weight set component $\Lambda(y)$ requires the knowledge of all maximal LNPs to which the image y contributes. Consequently, the contributing sets of all maximal LNPs need to be complete in order to fully

construct all weight set components properly for the weighted Tchebychev decomposition. The proof is included in the Appendix.

4.2. Comparison to Weighted Sum Decomposition

Five instances of ND sets are considered, each defined as a (MODO) instance with 25 variables. WSDs for the first instance have already been presented in Figure 1; WSDs for the remaining instances are illustrated in the Appendix Section 7.5 (Figures 11 and 12) as well as the full quantitative comparison. Here, we summarize the results with three observations.

First, the proportion of ND images that are ESND is low: on average, only 28% of the ND images are extreme supported (range 23%-36%). Therefore, on average, the weighted sum decomposition *excludes the majority* of ND images.

Second, while the weight set component with maximum area in each decomposition often corresponds to the same image, the *magnitude* of the largest component is significantly larger for weighted sum. That is, the weighted sum decomposition inflates the area or the relative value of its largest component compared to weighted Tchebychev. This observation can be partially explained by simply having fewer ND images included in the decomposition, such that on average each component must be larger; however, this alone does not fully explain the difference due to the final observation.

Third, the ranked orders of area per weight set component do not match between decompositions, including some extreme differences. Some images comprise a small portion of the weight set according to weighted Tchebychev while occupying a much larger area under weighted sum. We argue this is yet another major weakness of decomposing the weight set with respect to a strict subset of the ND frontier: as ND images are omitted, the crucial adjacency structure is significantly altered, which leads to disproportionate gains in some of the remaining images. With the WSD for weighted Tchebychev now available, this disproportionate reallocation becomes clear.

5. Approximations and Box-Based Algorithms

Prior to termination of Algorithm 1, which provides sufficient and necessary information for the weighted Tchebychev decomposition, the partial information maintained by the algorithm can be used for approximation on-the-fly. Approximation of the weight set components during run time can be useful in two ways. First, box-based algorithms can be

evaluated against one another by metrics which evaluate the approximation of the weight set components. For example, the quality of an approximated weighted sum decomposition has been computed by Halffmann (2021) using both outer and inner approximations, simultaneously. Second, the order of boxes chosen during runtime (step 5 in Algorithm 1) can be informed by the area of approximated weight set components.

To approximate weight set components, we provide both an outer and an inner approximation scheme, analogous to the approaches of Przybylski et al. (2010a) and Alves and Costa (2016), respectively, for weighted sum decomposition. An *outer approximation* of a WSD $\{\Lambda(y)\}_{y\in Y_N}$ for a proper subset of the images $Y' \subset Y_N$ is a set of approximated weight set components, $\{\Lambda^+(y)\}_{y\in Y'}$, for each NDP in Y', where $\Lambda(y) \subseteq \Lambda^+(y)$ for all $y \in Y'$. On the other hand, an *inner approximation*, $\{\Lambda^-(y)\}_{y\in Y'}$ is such that the approximated weight set components have positive area and $\Lambda^-(y) \subseteq \Lambda(y)$ for all $y \in Y'$.

Outer approximation. Outer approximation can be achieved trivially by following the standard steps for computing the weighted Tchebychev decomposition but for a proper subset of the ND set. Given $Y' \subset Y_N$ computed by Algorithm 1, along with the set of maximal LNPs with complete contributing sets, computed by Algorithm 2, then the plotting algorithm (Algorithm 4) will triangulate and plot the outer approximation by default. When a new ND image is found at each step of the box-based algorithm, then the set of maximal LNPs and complete contributing sets are updated, and hence the outer approximation of each component either reduces in area or remains the same. Figure 5 illustrates the outer approximation of weight set components for the instance illustrated in Figure 11.

Inner approximation The inner approximation is less straightforward to compute. Given $Y' \subset Y_N$ from Algorithm 1 and the updated set of maximal LNPs computed by Algorithm 2, then at least one maximal LNP must be certified as weakly ND. When integrated into the box-based algorithm, the inner approximation is computed with *only the maximal LNPs certified as weakly ND* at the current step. Note that these LNPs will be weakly ND with respect to the full ND set even if their contributing sets are not complete with respect to the full ND set.

With the subset of maximal LNPs that are certified as weakly ND, the plotting algorithm (Algorithm 4) must be modified. Often, an inner approximation will not contain the full perimeter of the component but rather disconnected subsets of the perimeter.

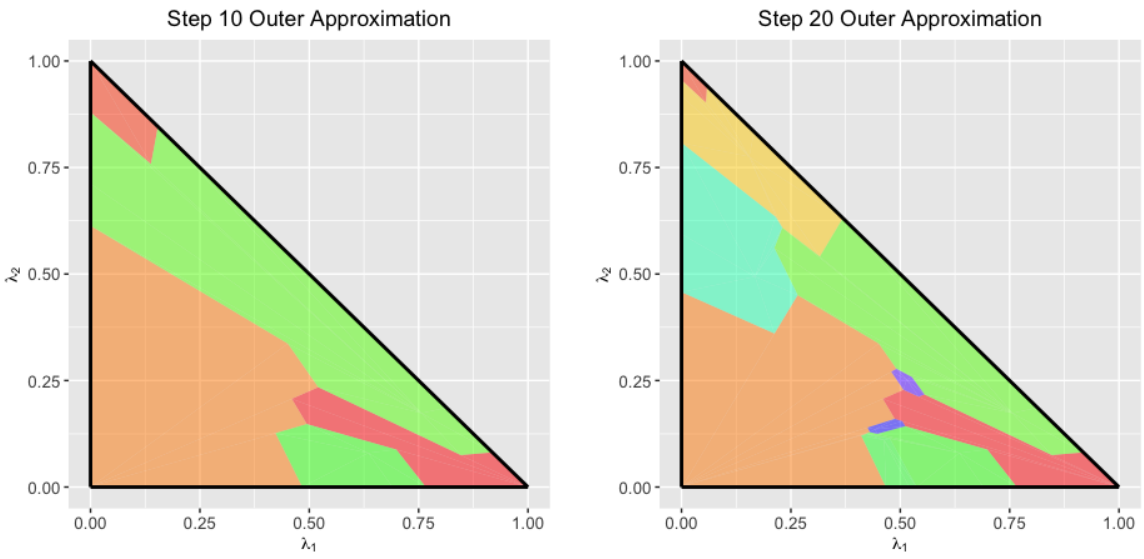


Figure 5 **Outer approximations for weighted Tchebychev decomposition are illustrated for the example in Figure 11(b), which has 22 ND images. On the left, the outer approximation is computed at step 10 of Algorithm 1, which includes 5 ND images. On the right, it is computed at step 20, which includes 10 ND images.**

Therefore, triangularization must occur independently for each dimensional contribution set, C^1, C^2, C^3; hence, in Algorithm 4, the sets L^1, L^2, L^3 are not joined (step 17), and the for loop (steps 18-23) is performed over each L^i set, independently.

At each step of the box-based algorithm, when an empty box is found, then the number of certified LNPs increases by one, and the inner approximation of the contributing images' weight set components may increase (or stay the same). Furthermore, since the box-based algorithm using Algorithm 2 computes maximal LNPs with *complete* contributing sets, then it is possible to check when a single image's weight set component can be fully computed: if all maximal LNPs to which image y' contributes are certified as weakly ND, then $\Lambda(y')$ can be fully represented. Otherwise, if y' contributes to some LNPs that are yet to be certified, then $\Lambda(y')$ is not yet fully represented.

EXAMPLE 6. Figure 6 illustrates the inner approximation of weight set components for the instance illustrated in Figure 11. Note that at step 20, the inner approximation $\Lambda^-(y^8)$ (in green) is nearly equal to the outer approximation $\Lambda^+(y^8)$ in Figure 5. However, since y^8 contributes to one of the remaining uncertified LNPs, then $\Lambda^-(y^8)$ is not yet complete. □

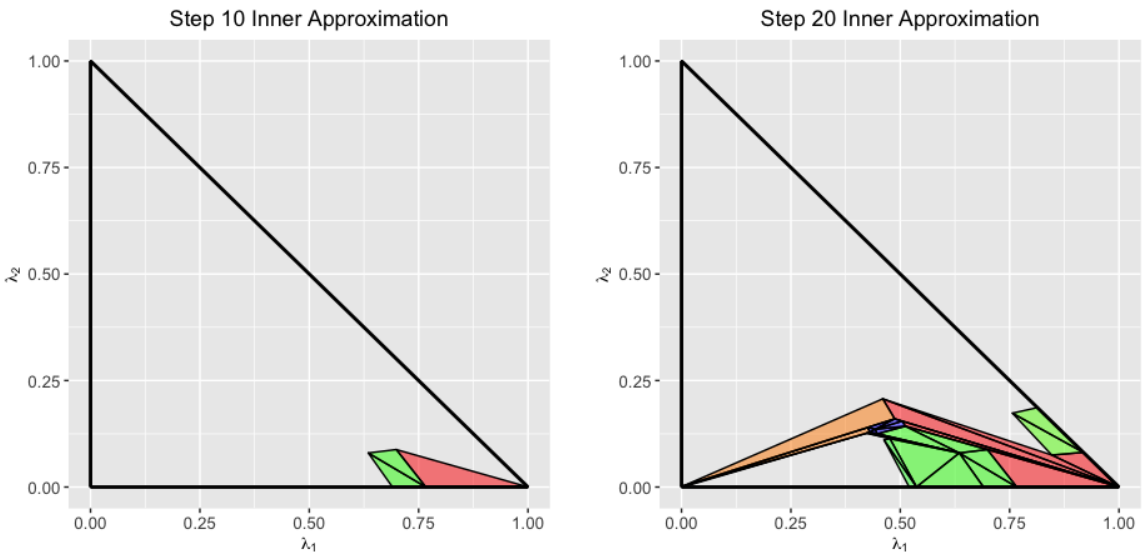


Figure 6 **Inner approximations for weighted Tchebychev decomposition are illustrated for the example in Figure 11(a). On the left, the approximated decomposition is computed at step 10 of the box-based algorithm, which includes 5 ND images and only 4 certified weakly ND LNPs. One LNP implies lower-order LNPs that define all three visible triangles. The other three LNPs are near the vertices of Λ, so the resulting triangles are miniscule and therefore not visible. On the right, the approximated decomposition is computed at step 20 of the box-based algorithm, which includes 10 ND images and 9 certified weakly ND LNPs. The colors assigned to each image's weight set component are consistent.**

Approximation time series. At each step of Algorithm 1, the inner and outer approximation for each image's weight set component can be computed along with its area. This area can be plotted for each step of the algorithm as a time series. An example is given in Figure 7 for the algorithm run on the instance in Figure 11. The outer approximated and inner approximated areas are indicated by lines (dashed and solid, respectively), and the x-axis represents the step s in Algorithm 1. The outer approximated area is monotonically decreasing and acts as an upper bound for the true area of the weight set component; the inner approximated area acts in the opposite manner. When Algorithm 1 terminates, the gap is zero for all weight set components.

The time series analysis in Figure 7 should be viewed as a *simulated* computational study in which all IP solves are normalized to a single unit of time. Valuable data are easily accessible by this analysis: the ranked order of weight set components based on outer- or inner-approximated area as well as the gap between these two estimates. Performance metrics may be designed around these data in order to evaluate the power of an box-based algorithm to approximate the solution, if the algorithm is terminated early. For instance,

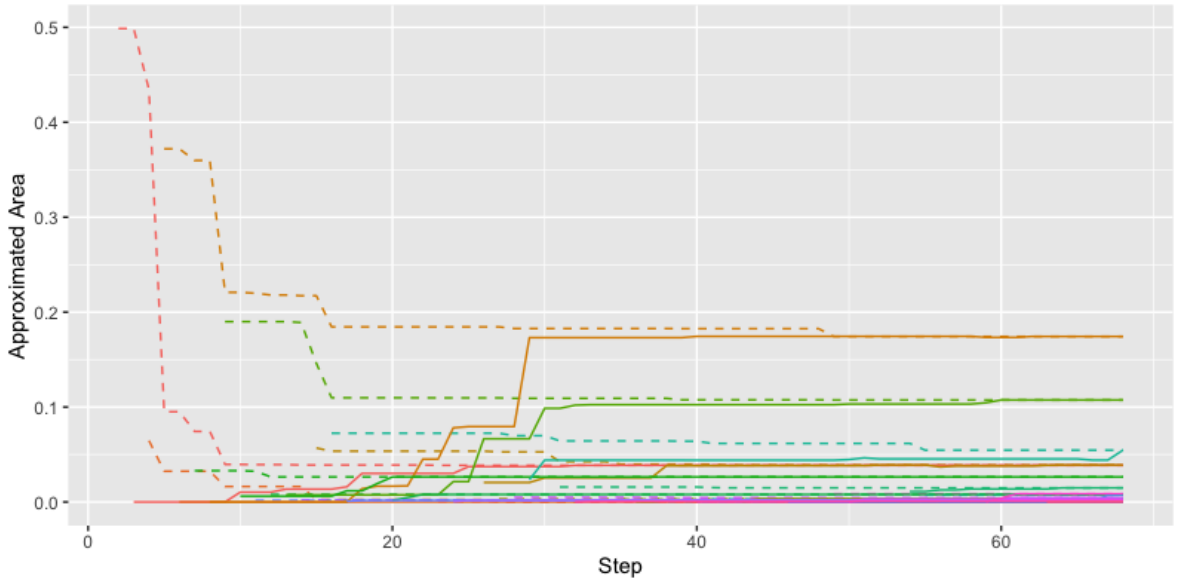


Figure 7 **Area of the approximated weight set components for each image are plotted as a time series. The x-axis represents the step of the box-based algorithm. Outer approximated areas are indicated by dashed lines and are monotonically decreasing. Inner approximated areas are indicated by solid lines and are monotonically increasing.**

the gap between outer and inner approximations would ideally be small for algorithms that perform well in this way. This analysis also highlights the impact of the box selection policy, which is discussed next.

5.1. Box Selection Policy

The selection of boxes from the queue (line 5 of Algorithm 1) is crucial to the performance of box-based algorithms. Although this policy does not effect the correctness of the algorithm, it is critical to the quality of the approximation prior to termination. A commonly used policy is to choose the box with largest hypervolume. Recently, the algorithm of Tamby and Vanderpooten (2020) includes a heuristic designed to improve the selection based on the particular form of the optimization subproblem. Now, the information given by the approximated WSD offers an alternative basis for policy design.

We compare the following five policies.

- *Policy 1* is the naïve first-in-first-out policy of choosing the box which was added to the queue most recently.

- *Policy 2* is a simple volumetric policy that chooses the LNP which defines the box with greatest hypervolume, i.e., $\arg\max_{n\in N}\{n_1 \times n_2 \times n_3\}$. The ideal point is taken to be the lower bound on the box. [6]
- *Policy 3* is a novel strategy that prioritizes boxes whose LNPs are associated with local nadir weights that are *central* to the weight set. This is achieved by choosing the local nadir weight with maximum minimal weight, i.e., $\arg\max_{n\in N}\min_{i=1,2,3}\lambda_i(n)$.
- *Policy 4* is a novel strategy that prioritizes boxes whose LNPs are associated with local nadir weights that are *close to the boundary* of the weight set. This is achieved by choosing the local nadir weight with maximum maximal weight, i.e., $\arg\max_{n\in N}\max_{i=1,2,3}\lambda_i(n)$.
- *Policy 5* is a novel strategy driven by the outer approximation for the weight set components. In particular, the policy chooses the LNP whose contributing images have the largest sum area in outer approximations, i.e., $\arg\max_{n\in N}\sum_{y\in C(n)}\text{area}(\Lambda^+(y))$.

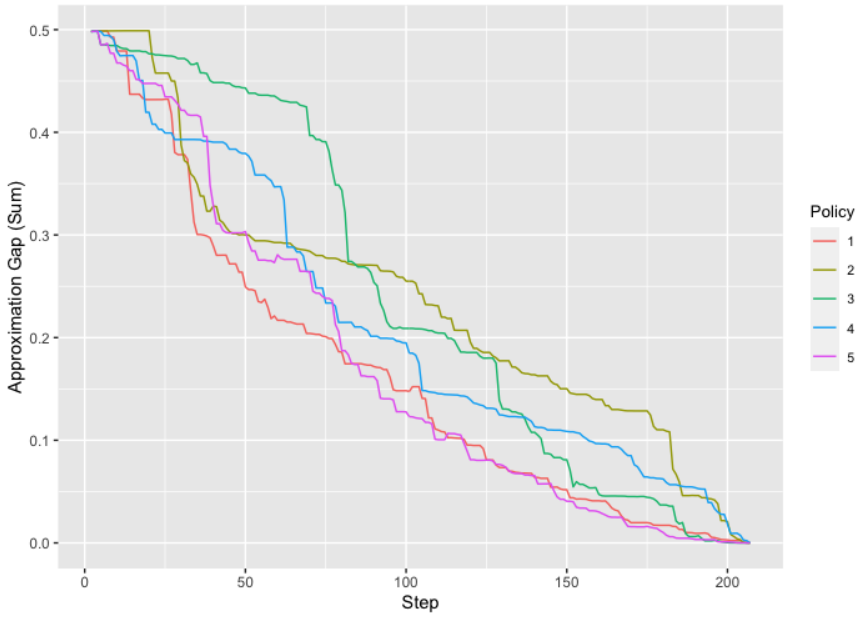
As the first performance metric, consider the quality of approximated weight set components. We represent the quality of this approximation as the sum of all "gaps" between outer and inner approximated areas, i.e., $\sum_{y\in Y_N}[\text{area}(\Lambda^+(y)) - \text{area}(\Lambda^-(y))]$.[7] Note that for a given image y, from its *initial discovery*, the area of $\Lambda^+(y)$ is monotonically decreasing and the area of $\Lambda^-(y)$ is monotonically increasing; however, the gap metric is not monotonically decreasing because as new images are discovered, the sum of gaps may increase between step s and $s+1$. As the sum of gaps reduces to zero, the confidence in each approximated weight set component increases.

Figure 8(a) illustrates this gap measured at every step of the box-based algorithm for each policy on the instance in Figure 1. Observe that the naïve Policy 1 performs the best for steps 30-70, and then is competitive with the novel Policy 5 as best for the remainder of run time.

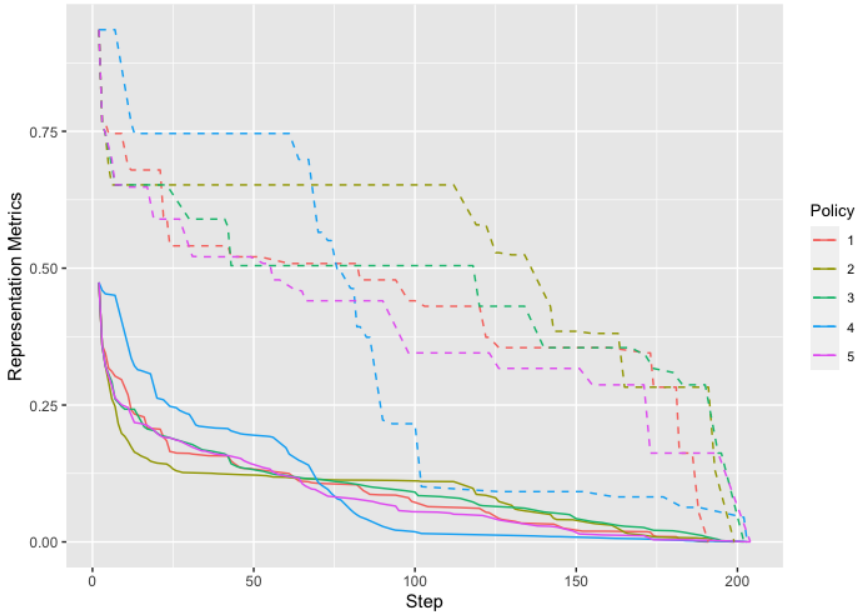
Second, consider the quality of representing the ND set. Generating a "good" subset of the discrete ND set, which is quantitatively *representative* of the entire ND set, has been studied using many different quality metrics (see Faulkenberg and Wiecek (2010) and citations within). We consider two of the simplest measures for *coverage*, which communicates

[6] For LNPs defined with dummy images, any components equal to M are excluded from the computation (equivalently, replace $M = 1$) to prevent domination.

[7] Notably, for images that have yet to be discovered, the inner and outer approximations have zero area, and so the associated gap is zero. When an image is first discovered, it immediately has an outer approximation with positive area. The inner approximation for image y has positive area once a LNP to which y contributes is certified as maximal.



(a) Approximation gap between outer and inner approximate weight set components.



(b) Representation metrics Dmean (solid line) and Dmax (dashed lines).

Figure 8 **Policy analysis over run time using three metrics; for all metrics, smaller values are better. Instance illustrated in Figure 1.**

differences between the complete ND set and a subset. The two measures, which we refer to as simply *Dmean* and *Dmax* (Czyżak and Jaszkiewicz 1997), represent the "distance between" the sets in average and in worst-case, respectively. These metrics use weighted

Tchebychev metric to measure distance. Let $w_i := 1/(\max_{y \in Y_N} y_i)$ and define $d(y, y') = \|y - y'\|_\infty^w$; the weight vector w normalizes the distances by the range of each objective function value over the ND set. For $\bar{Y} \subseteq Y_N$, the two coverage measures are defined as

$$\text{Dmean}(Y_N, \bar{Y}) = \frac{1}{|Y_N|} \sum_{y \in Y_N} \min_{\bar{y} \in \bar{Y}} d(y, \bar{y}), \text{ and} \tag{7}$$

$$\text{Dmax}(Y_N, \bar{Y}) = \max_{y \in Y_N} \min_{\bar{y} \in \bar{Y}} d(y, \bar{y}). \tag{8}$$

The representation metrics, Dmean and Dmax, measured at each step of the box-based algorithm for each policy are illustrated in Figure 8(b) (solid and dashed lines, respectively). Similar to the approximation gap, lower metrics indicate better performance, and these metrics tend towards zero over run time. Observe the opposite performance of Policy 2 and 4: Policy 4 (in blue) initially performs the worst in terms of both representation metrics, then steeply improves and becomes the best-performing policy. Policy 2 (in yellow), on the other hand, performs best early on (at least for Dmean), but then plateaus and gradually becomes one of the worst policies.

Interpreting the overall performance of these policies across multiple metrics is its own multicriteria challenge. Figure 9 gives a complete picture of the performance for each policy by *ranking policies at each step* and then counts, for each policy, the *time spent in each ranked position*. Consider a cross-section at step 100, where one views the performance of the five policies across all three metrics: according to the approximation gap (Figure 8), Policy 5 is first (best), Policy 1 is second, Policy 4 is third, Policy 3 is fourth, and Policy 2 is fifth (worst). Therefore, each policy has spent one unit of time in each of these ranked positions. By repeating this counting procedure for all steps and all metrics, we can exactly compute the frequency that a policy was in ranked positions 1-5, where we can interpret "frequency" as "time spent" in this position. The previously mentioned observations can be seen clearly in Figure 9: According to approximation gap, Policies 1 and 5 spend the most time in first or second place (see vertical columns labeled "P1-Gap" and "P5-Gap"); however, Policy 5 spends more time in first position than Policy 1. For representation metrics, Policy 4 spends most of its time in either first or last position (see vertical columns labeled "P4-Dmax" and "P4-Dmean").

Finally, the black diamonds in Figure 9 summarize the *average rank* of each policy according to each metric. This value penalizes time spent in low-ranked positions, especially

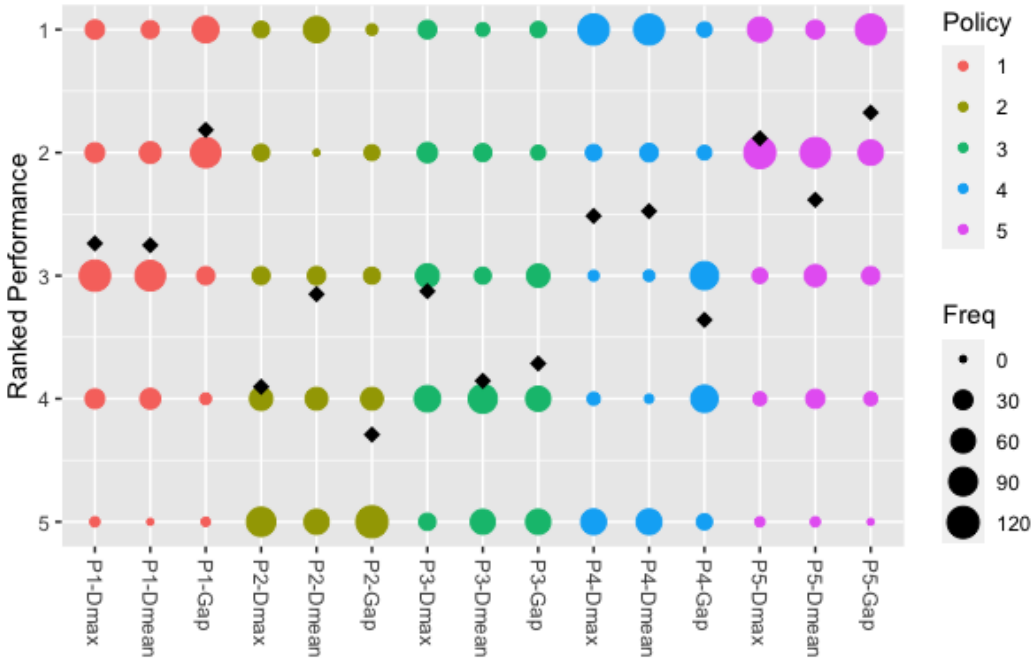


Figure 9 **Summary ranks of policy analysis using three metrics. The horizontal axis is labeled by combinations of policies (1-5) and metrics (Gap, Dmax, Dmean). The vertical axis represents the ranked positions 1-5, where position 1 represents best and position 5 represents worst at any one step. The size of a colored circle indicates how many steps (or how much time spent) that the policy was ranked in that position according to the metric. Black diamonds indicate the average rank over all of runtime. Instance illustrated in Figure 1.**

for long periods of time. Clearly, by Figure 9, Policy 5 performs very well according to this summary statistic; in fact, its average rank according to each metric dominates all other policies for this instance.

6. Conclusions

The weighted Tchebychev decomposition is a flexible technique which is particularly promising for interactive multiobjective procedures (see Xin et al. (2018) for a survey). The decomposition provides a useful a posteriori method for decision makers to evaluate the ND set and to make decisions informed by the weight space. Interactive multiobjective procedures alternate between phases of algorithmic progression and eliciting preferences of decision makers, and can overcome the weaknesses of both a priori and a posteriori methods Xin et al. (2018). One promising example of a decision maker's interaction with a box-based algorithm (Algorithm 1) could be to replace the box selection policy, where the weight space could be used to inform this choice. A second example relates to the reference

point, s in $\Pi^{TS}(\lambda)$, which we have fixed to be the utopia point in this work; however, the reference point could be dynamically chosen, which is a common practice for interactive multiobjective procedures (Xin et al. 2018).

In closing, the authors observe potential insights from this understudied WSD as a *dual* perspective on MODOs. Over decades of research, multiple distinct notions of duality have been proposed for single objective integer programs and multiobjective linear programs (see surveys of Hooker (2009) and Luc (2011), respectively). However, to our knowledge, duality for MODOs is still an open area of research. The geometric duality presented by Heyde and Löhne (2008) for multiobjective linear programs closely parallels the weighted sum decomposition by providing a dual polyhedral relationship to the image set. In doing so, their work set a *precedent* for WSDs to provide a valid dual perspective for multiobjective optimization problems. Much more research must be done to formally study these potential relationships.

7. Appendix

We have implemented a plotting tool in R using basic functions and the *ggplot2* package.[8] The triangles are plotted as translucent (alpha parameter set to 0.5) so that full-dimensional intersections between weight set components are shown visibly as overlapping regions, e.g., Figure 2(b). In addition to computing each triangle, the outline of the weight set component is represented as line segments of the form $[\lambda(n^i), \lambda(n^{i+1})]$ for each $n^i, n^{i+1} \in P$.

7.1. Running Example

[8] All R code and instances are accessible at https://github.com/perinita/TchebychevDecomposition.

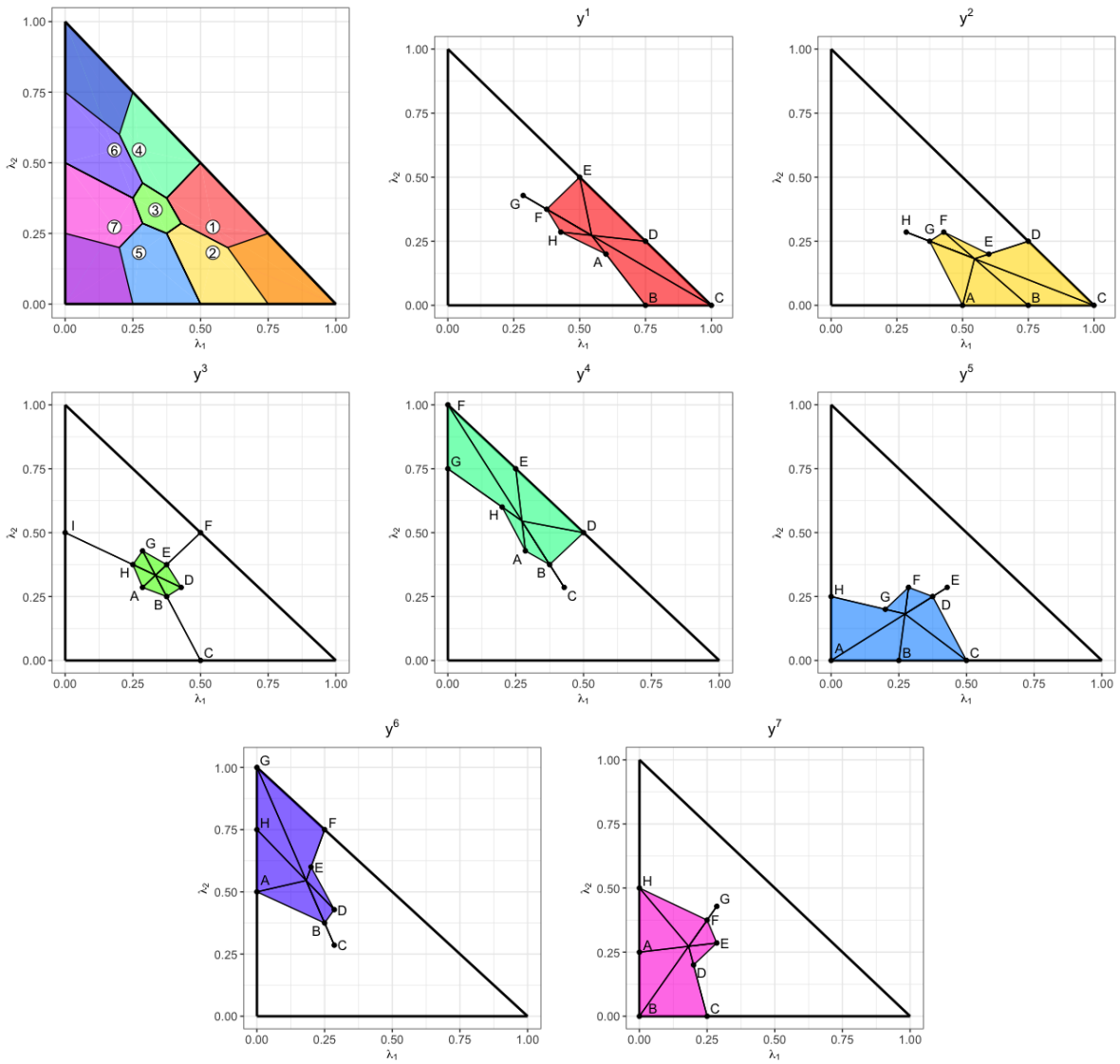


Figure 10 **All weight set components for Example 2. Labeled local nadir weights correspond to Table 1, which gives the contributing set for each.**

7.2. Boundary Weights

To represent boundary weights, we introduce p dummy images. For $i \in \{1, \ldots, p\}$, define the ith dummy image, $y^{b(i)}$, by $y_i^{b(i)} = M$ for some large $M > y^N(Y_N)$ and $y_j^{b(i)} = y_j^U = 0$ for all $j \neq i$. Let $Y_B := \{y^{b(1)}, \ldots, y^{b(p)}\}$ be the set of dummy images for representing boundary weights.

Note that a local nadir weight $\lambda^N(\bar{Y})$ is not defined for any $\bar{Y} \subset Y_B$ due to zero in the denominator. The following assumption avoids this issue.

Table 1 Each element indicates the labels (e.g., $i, j, b(k)$ for $y^i, y^j, y^{b(k)}$, respectively) for the complete contributing set per local nadir weight. Each column is associated with one weight set component, and each row corresponds to labels given in Figure 10.

Label	y^1	y^2	y^3	y^4	y^5	y^6
A	1,2	2,3,5,b(2)	2,3,5,6,7	1,3,4,6,7	5,b(1),b(2)	3,6,7,b(1)
B	1,2,b(2)	1,2,b(2)	2,3,5	1,3,4	5,7,B(2)	3,6,7
C	1,2,b(2),b(3)	1,2,b(2),b(3)	2,3,5,b(2)	1,2,3,4,5	2,5,b(2)	2,5,6,7
D	1,2,b(3)	1,2,b(3)	1,2,3,4,5	1,3,4,b(3)	2,3,5	1,3,4,6,7
E	1,3,4,b(3)	1,2	1,3,4	4,6,b(3)	1,3,4,5	4,6
F	1,3,4	1,2,4,5	1,3,4,b(3)	4,b(1),b(3)	2,3,5,6,7	4,6,b(3)
G	1,3,4,6,7	2,3,5	1,3,4,6,7	4,6,b(1)	5,7	6,b(1),b(3)
H	1,2,4,5	2,3,5,6,7	3,6,7	4,6	5,7,b(1)	4,6,b(1)
I			3,6,7,b(1)			

Assumption 4. *We assume that the set of dummy images is included in the set of ND images, i.e.,* $Y_B \subset Y_N$.

Observe that if $y^{b(i)} \in \bar{Y} \subseteq Y_N$, then for finite M the local nadir weight $\lambda^N(\bar{Y})$ will have small, positive ith component and hence lie in int(Λ); however, by limit we have $\lim_{M\to\infty} \lambda_i^N(\bar{Y}) = 0$ and so $\lim_{M\to\infty} \lambda^N(\bar{Y})$ yields a weight on the boundary. The more practical way to compute boundary weights, as discussed in Karakaya and Köksalan (2021), is to set the ith component to zero and compute the remaining components by computing the $(p-1)$-dimensional local nadir weight while dropping the ith dimension.

7.3. Textual Examples

Example 7 (Illustration of Theorem 2). Consider Example 1. In Figure **??**(a), the kernel point of the intersection set is the local nadir weight for the 2-way LNP: $\lambda^2 = \lambda^N(\{y^1, y^5\})$. In Figure 2(b), the kernel point of the intersection set is similarly $\lambda^2 = \lambda^N(\{y^1, y^2\})$.

Example 8 (Illustration of Boundary Weights). Image y^1 contributes to four maximal LNPs. These LNPs and each of the complete (dimensional) contributing sets are given in Table 2 with labels corresponding to Figure 3. Note that C and E are 4-way LNPs but G and H are 7-way LNPs. □

Table 2 The maximal LNPs to which y^1 contributes with complete contributing sets in Example 2. For each LNP, the label used in Figure 3 is given as well as the images in each dimensional contributing set, C^1, C^2, C^3. Images $y^{b(2)}$, and $y^{b(3)}$ refer to the dummy images.

LNP	$(1, M, M)$	$(2, 2, M)$	$(3, 2, 3)$	$(2, 3, 3)$
Label	C	E	G	H
$C^1(n)$	y^1, y^2	y^3, y^4	y^6, y^7	y^3, y^4, y^5
$C^2(n)$	$y^{b(2)}$	y^1, y^3	y^1, y^3, y^7	y^2, y^5
$C^3(n)$	$y^{b(3)}$	$y^{b(3)}$	y^1, y^4	y^1, y^4

EXAMPLE 9 (ILLUSTRATION OF DEFINITION 8). For Example 2, consider maximal LNP $n := y^N(\{y^1, y^2, y^{b(2)}, y^{b(3)}\}) = (1, M, M)$. From Table 2 (label C), we have $C^1(n) = \{y^1, y^2\}$, $C^2(n) = \{y^{b(2)}\}$, and $C^3(n) = \{y^{b(3)}\}$. Only one contributing set is not a singleton, i.e., $|C^1(n)| = 2$. Consider permutations $\sigma^1 = (1, 2, 3)$ and $\sigma^2 = (1, 3, 2)$. The corresponding two polytopes belong to $\mathcal{P}(n)$:

$$P^1 := \text{conv}\left(\lambda^N(\{y^1, y^{b(2)}, y^{b(3)}\}), \lambda^N(\{y^1, y^{b(2)}\}), \lambda^N(\{y^1\})\right)$$
$$P^2 := \text{conv}\left(\lambda^N(\{y^1, y^{b(2)}, y^{b(3)}\}), \lambda^N(\{y^1, y^{b(3)}\}), \lambda^N(\{y^1\})\right)$$

Using the labeled local nadir weights given in Figure 3(b), these polytopes are equivalent to triangles: $P^1 = \text{conv}(\{C, B, \lambda(y^1)\})$ and $P^2 = \text{conv}(\{C, D, \lambda(y^1)\})$

Polytopes P^1 and P^2 are contained in $\Lambda(y^1)$ since $\sigma^1(1) = \sigma^2(1) = 1$. The remaining polytopes in $\mathcal{P}(n)$ are defined with the singleton local nadir weight of a boundary weight, i.e., $\sigma(1) = b(2)$ or $\sigma(1) = b(3)$. Such polytopes are generally ignored. □

EXAMPLE 10. The progression of Algorithm 1 on the ND set from Example 2 is illustrated in Table 3. We let the box selection policy in line 5 be naive, first-in-first-out order. The UPDATE procedure is Algorithm 2, which is discussed in the next section. The algorithm requires 16 steps before the queue is empty, which terminates the procedure.

7.4. Proofs of Coverage

LEMMA 1. *For $\bar{Y} \subset Y_N$ where $q := |\bar{Y}| \geq p$, let $y^N(\bar{Y})$ be weakly ND where all images contribute. Then,*

$$\text{conv}(\{\lambda^N(y^1), \lambda^N(y^1, y^2) \ldots, \lambda^N(y^1, y^2, \ldots, y^q)\}) \subseteq \Lambda(y^1),$$

where $\lambda^N(y^1) = \lambda(y^1)$.

Table 3 Progression of box-based algorithm (Algorithm 1) on Example 2.

Step	ND Image found	maximal LNP certified
1	y^1	
2		$(1, M, M)$
3	y^4	
4	y^2	
5		$(2, 2, M)$
6		$(M, 1, M)$
7	y^3	
8	y^5	
9		$(2, 3, 3)$
10	y^6	
11		$(2, M, 2)$
12	y^7	
13		$(M, M, 1)$
14		$(3, 2, 3)$
15		$(3, 3, 2)$
16		$(M, 2, 2)$

Proof. Let $\lambda = \sum_{r=1}^{q} \alpha_r \lambda^N(y^1, \dots, y^r)$ with $\alpha \in [0,1]^q$ and $\sum_{r=1}^{q} \alpha_r = 1$. We show $\lambda \in \Lambda(y^1)$. Without loss of generality, we can assume that $\alpha_i > 0$ for all i. If λ is equal to a local nadir weight, $\lambda \in \Lambda(y^1)$ is trivially satisfied; otherwise, with zero coefficients, the following procedure can be adapted to the subset of nadir weights with strictly positive coefficients.

First, consider

$$\begin{aligned}
&\alpha_q \lambda^N(y^1, \dots, y^q) + \alpha_{q-1} \lambda^N(y^1, \dots, y^{q-1}) \\
&= (\alpha_q + \alpha_{q-1}) \left[\frac{\alpha_q}{\alpha_q + \alpha_{q-1}} \lambda^N(y^1, \dots, y^q) + \frac{\alpha_{q-1}}{\alpha_q + \alpha_{q-1}} \lambda^N(y^1, \dots, y^{q-1}) \right] \\
&=: (\alpha_q + \alpha_{q-1}) \tilde{\lambda}^{q-1}
\end{aligned}$$

for $\tilde{\lambda}^{q-1}$ defined as the convex combination of the q-way and $(q-1)$-way local nadir weights. By star-shapedness of $\bigcap_{r=1}^{q-1} \Lambda(y^r)$, which includes both the local nadir weights, we have

$\tilde{\lambda}^{q-1} \in \bigcap_{r=1}^{q-1} \Lambda(y^r)$. This implies

$$\begin{aligned}
&\alpha_q \lambda^N(y^1,\ldots,y^q) + \alpha_{q-1}\lambda^N(y^1,\ldots,y^{q-1}) + \alpha_{q-2}\lambda^N(y^1,\ldots,y^{q-2}) \\
&= (\alpha_q + \alpha_{q-1})\tilde{\lambda}^{q-1} + \alpha_{q-2}\lambda^N(y^1,\ldots,y^{q-2}) \\
&= (\alpha_q + \alpha_{q-1} + \alpha_{q-2}) \left[\frac{\alpha_q + \alpha_{q-1}}{\alpha_q + \alpha_{q-1} + \alpha_{q-2}} \tilde{\lambda}^{q-1} + \frac{\alpha_{q-2}}{\alpha_q + \alpha_{q-1} + \alpha_{q-2}} \lambda^N(y^1,\ldots,y^{q-2}) \right] \\
&=: (\alpha_q + \alpha_{q-1} + \alpha_{q-2})\tilde{\lambda}^{q-2}
\end{aligned}$$

for $\tilde{\lambda}^{q-2}$ defined as the convex combination of the $(q-1)$-way and $(q-2)$-way local nadir weights. Again, by star-shapedness we have $\tilde{\lambda}^{q-2} \in \bigcap_{r=1}^{q-2} \Lambda(y^r)$. Inductively applying this procedure yields

$$\begin{aligned}
\lambda &= \sum_{r=1}^{q} \alpha_r \lambda^N(y^1,\ldots,y^r) = (\sum_{r=2}^{q} \alpha_r)\tilde{\lambda}^2 + \alpha_1 \lambda^N(y^1) \\
&= (\sum_{r=1}^{q} \alpha_r) \left[\frac{(\sum_{r=2}^{q} \alpha_r)}{(\sum_{r=1}^{q} \alpha_r)} \tilde{\lambda}^2 + \frac{\alpha_1}{(\sum_{r=1}^{q} \alpha_r)} \lambda^N(y^1) \right] \\
&= \frac{(\sum_{r=2}^{q} \alpha_r)}{(\sum_{r=1}^{q} \alpha_r)} \tilde{\lambda}^2 + \frac{\alpha_1}{(\sum_{r=1}^{q} \alpha_r)} \lambda^N(y^1) \in \Lambda(y^1). \quad \square
\end{aligned}$$

We define the family of polytopes presented in Lemma 1.

DEFINITION 8. For LNP $n = y^N(\bar{Y})$ with $\bar{Y} \subset Y_N$, with dimensional contributing sets $C^j(n), j = 1,\ldots,p$, let $\mathcal{P}(n)$ denote the family of simplices[9]

$$\begin{aligned}
\mathcal{P}(n) = \{ \operatorname{conv}(\{\lambda^N(y^{\sigma(1)},\ldots,y^{\sigma(p-1)},y^{\sigma(p)}), \lambda^n(y^{\sigma(1)},\ldots,y^{\sigma(p-1)}),\ldots,\lambda^N(y^{\sigma(1)})\}) : \\
y^j \in C^j(n), j = 1,\ldots,p, \text{ for some permutation } \sigma : \{1,\ldots,p\} \to \{y^1,\ldots,y^p\}\}.
\end{aligned}$$

Subsets of weight set components can be constructed by the union of simplices given by $\mathcal{P}$. Theorem 5 states that the maximal LNPs are indeed sufficient to cover weight set components. The following Lemma is essential for the proof of Theorem 5.

LEMMA 2. *Let $\lambda \in \Lambda(y)$ for some $y \in Y_{wN}$ and $I := \{1,\ldots,p : \|y\|_\infty^\lambda = \lambda_i y_i\}$ with $|I| < p$. Then there exists $\bar{y} \in Y_N$, weight $\bar{\lambda} \in \Lambda$, and index set $J \subseteq \{1,\ldots,p\}$ with $I \cap J = \emptyset$ such that the 1-6 below hold:*

1. $\bar{\lambda} \in \Lambda(y) \cap \Lambda(\bar{y})$,
2. $\lambda \in \operatorname{conv}(\{\lambda^N(y), \bar{\lambda}\})$,

[9] The reader is referred to an example in the Appendix.

3. $\|y\|_\infty^{\bar\lambda} = \bar\lambda_i y_i$ *for* $i \in I$ *and* $\|y\|_\infty^{\bar\lambda} > \bar\lambda_k y_k$ *for* $k \notin I$,
4. $\|\bar y\|_\infty^{\bar\lambda} = \bar\lambda_j \bar y_j$ *for* $j \in J$ *and* $\|\bar y\|_\infty^{\bar\lambda} > \bar\lambda_k \bar y_k$ *for* $k \notin J$,
5. $y_i > \bar y_i$ *for all* $i \in I$, *and*
6. $y_j < \bar y_j$ *for all* $j \in J$.

Proof. Since $|I| < p$, $\lambda \neq \lambda^N(y)$. Set I must be nonempty, so let

$$H(I) := \{\lambda' \in \Lambda : \lambda'_i y_i = \|y\|_\infty^{\lambda'} \ \forall i \in I,\ \lambda'_k y_k < \|y\|_\infty^{\lambda'} \ \forall k \notin I\}.$$

By assumption, $\lambda \in H(I)$ and $\lambda^N(y) + \theta(\lambda - \lambda^N(y)) \in H(I)$ for all $\theta > 0$. Since $\Lambda(y)$ is compact and star-shaped, there exists $\theta' \geq 1$ such that

$$\bar\lambda = \lambda^N(y) + \theta'(\lambda - \lambda^N(y)) \in \Lambda(y),$$
$$\bar\lambda^\varepsilon := \lambda^N(y) + (\theta' + \varepsilon)(\lambda - \lambda^N(y)) \notin \Lambda(y)$$

for all $\varepsilon > 0$. For small $\varepsilon > 0$, let $\bar y \in Y_N$ such that $\bar\lambda^\varepsilon \in \Lambda(\bar y)$ ($\bar y$ may be in Y_B). Thus, by continuity of the weighted Tchebychev norm in λ, we have $\|y\|_\infty^{\bar\lambda} = \|\bar y\|_\infty^{\bar\lambda}$ and $\|y\|_\infty^{\bar\lambda^\varepsilon} > \|\bar y\|_\infty^{\bar\lambda^\varepsilon}$. This satisfies results 1 and 2.

For $i \in I$, $\|y\|_\infty^{\bar\lambda} = \bar\lambda_i y_i$ by definition. Suppose for contradiction that i also maximizes $\|\bar y\|_\infty^{\bar\lambda}$, i.e., $\|\bar y\|_\infty^{\bar\lambda} = \lambda_i \bar y_i$. Since $\bar\lambda \in H(I)$, this implies

$$\bar\lambda_i \bar y_i = \|\bar y\|_\infty^{\bar\lambda} = \|y\|_\infty^{\bar\lambda} = \bar\lambda_i y_i$$

and thus $\bar y_i = y_i$. However, $\bar\lambda^\varepsilon \in H(I)$ also holds and so $\|\bar y\|_\infty^{\bar\lambda^\varepsilon} \geq \bar\lambda_i^\varepsilon \bar y_i = \bar\lambda_i^\varepsilon y_i = \|y\|_\infty^{\bar\lambda^\varepsilon}$, which is a contradiction to $\lambda^\varepsilon \in \Lambda(\bar y) \setminus \Lambda(y)$. Therefore, $\|\bar y\|_\infty^{\bar\lambda} > \lambda_i \bar y_i$ and $y_i > \bar y_i$ for all $i \in I$. Let $J \subseteq \{1, \ldots, p\} \setminus I$ such that $\|\bar y\|_\infty^{\bar\lambda} = \bar\lambda_j \bar y_j$ for $j \in J$ and $\|\bar y\|_\infty^{\bar\lambda} > \bar\lambda_k \bar y_k$ for $k \notin J$. Then, $y_j < \bar y_j$ for all $j \in J$ follows by $\bar\lambda_j \bar y_j = \|\bar y\|_\infty^{\bar\lambda} = \|y\|_\infty^{\bar\lambda} > \bar\lambda_j y_j$ for all $j \in J$. Index sets I and J satisfies results 3-6. □

In summary, results 1 and 2 of Lemma 2 prove that the line segment $[\lambda(y), \lambda]$ may be extended linearly to an intersection of weight set components, i.e., $[\lambda(y), \lambda] \subseteq [\lambda(y), \bar\lambda]$ where $\bar\lambda \in \Lambda(y) \cap \Lambda(\bar y)$ for some $\bar y \in Y_N$. The index sets I and J indicate which components maximize $\|y\|_\infty^{\bar\lambda}$ and $\|\bar y\|_\infty^{\bar\lambda}$, respectively, and satisfy results 4-6. Note that the statement of the Lemma intentionally uses the set of *weakly* ND images, Y_{wN}, so that the lemma can be applied to LNPs in the following theorem.

THEOREM 5. *For $\lambda \in \Lambda$, there exists a maximal LNP, $n = y^N(\bar{Y})$ for some $\bar{Y} \subseteq Y_N$, such that there are p contributing images $y^1 \in C^1(n), \ldots, y^p \in C^p(n)$ with*

$$\lambda \in \operatorname{conv}(\{\lambda^N(y^{\sigma(1)}), \lambda^N(y^{\sigma(1)}, y^{\sigma(2)}), \ldots, \lambda^N(y^1, y^2, \ldots, y^p)\}) \in \mathcal{P}(n)$$

for some permutation $\sigma : \{1, \ldots, p\} \to \{1, \ldots, p\}$.

Proof. Let $\lambda \in \Lambda$ and $y^1 \in Y_N$ such that $\lambda \in \Lambda(y^1)$. By iteratively applying Lemma 2, one can construct a (not necessarily maximal) local nadir weight which is defined by $R \leq p$ ND images $y^1, \ldots, y^R$, each with respective index set $I^1, \ldots, I^R$ satisfying the following four properties:

1. $\lambda^N(\{y^1, \ldots, y^R\}) \in \bigcap_{r=1}^R \Lambda(y^r)$.
2. $\lambda \in \operatorname{conv}(\{\lambda^N(y^1), \lambda^N(\{y^1, y^2\}), \ldots, \lambda^N(\{y^1, y^2, \ldots, y^R\})\})$.
3. $I^r \cap I^s = \emptyset$ for all $r, s \in \{1, \ldots, R\}$ where $r \neq s$, and $\cup_{r=1}^R I^r = \{1, \ldots, p\}$.
4. For all r it holds $y_i^r > y_i^s$ for all $i \in I^r$ and $r \neq s$.

The properties can follow as a corollary to Lemma 2, but for completeness it is proven in the construction section below. Here, assume we have constructed such a local nadir weight.

Note if there existed some $\bar{y} \in Y_N$ such that $\bar{y} < y^N(y^1, \ldots, y^r)$, this would contradict property (1). Also, there exists a maximal LNP n such that $y^N(y^1, \ldots, y^r) \leqq n$. For an image y^r, suppose for contradiction $y_i^r < n_i$ for all $i \in I^r$. Since $y_k^r < y_k^s \leq n_k$ for $k \notin I^r$ and s chosen such that $s \in I^s$, it follows that $y^r < n$, which is a contradiction. Therefore, for each image y^r there exists an index i such that $y_i^r = n_i$ and thus $y^r \in C_i(n)$.

Without loss of generality, we have $y^r \in C_r(n)$ for $r = 1, \ldots, R$. If $R < p$, we can choose images $\tilde{y}^{R+1} \in C^{R+1}(n), \ldots, \tilde{y}^p \in C^p(n)$ arbitrarily. Then, property (2) yields

$$\begin{aligned}
&\lambda \in \operatorname{conv}(\{\lambda^N(y^1), \ldots, \lambda^N(y^1, y^2, \ldots, y^R)\}) \\
&\quad \subseteq \operatorname{conv}(\{\lambda^N(y^1), \ldots, \lambda^N(y^1, y^2, \ldots, y^R), \\
&\qquad\qquad \lambda^N(y^1, y^2, \ldots, y^R, \tilde{y}^{R+1}), \ldots, \lambda^N(y^1, y^2, \ldots, y^R, , \tilde{y}^{R+1}, \ldots, \tilde{y}^p)\}) \in \mathcal{P}(n).
\end{aligned}$$

Construction: It remains to construct the local nadir weight and its defining images satisfying properties (1)-(4). Let $I^1 \subseteq \{1, \ldots, p\}$ with $\|y^1\|_\infty^\lambda = \lambda_i y_i^1$ for all $i \in I^1$ and $\|y^1\|_\infty^\lambda > \lambda_k y_k^1$ for $k \notin I^1$. If $I^1 = \{1, \ldots, p\}$, the construction terminates. Otherwise, we obtain by Lemma 2 an image $y^2 \in Y_N$, a weight $\lambda^2 \in \Lambda(y^1) \cap \Lambda(y^2)$, and an index set $I^2 \subseteq \{1, \ldots, p\}$ with

- $I^1 \cap I^2 = \emptyset$,
- $\lambda \in \text{conv}(\{\lambda^N(y^1), \lambda^2\})$,
- $\|y^2\|_\infty^{\lambda^2} = \lambda_j^2 y_i^2$ for $i \in I^2$ and $\|y^2\|_\infty^{\lambda^2} > \lambda_k^2 y_k^2$ for $k \notin I^2$,
- $\|y^1\|_\infty^{\lambda^2} = \lambda_i^2 y_i^1$ for $i \in I^1$ and $\|y^1\|_\infty^{\lambda^2} > \lambda_k^2 y_k$ for $k \notin I^1$,
- $y_i^1 > y_i^2$ for all $i \in I^1$,
- $y_j^1 < y_j^2$ for all $j \in I^2$.

If $I^1 \cup I^2 = \{1, \ldots, p\}$, the construction terminates. Otherwise, we can apply Lemma 2 with $y^N(y^1, y^2)$, $I = I^1 \cup I^2$, and λ^2 to obtain an image $y^3 \in Y_N$, a weight $\lambda^3 \in \Lambda(y^N(y^1, y^2)) \cap \Lambda(y^3)$ and an index set $I^3 \subseteq \{1, \ldots, p\}$ such that

- $I^2 \cap (I^1 \cup I^2) = \emptyset$,
- $\lambda^2 \in \text{conv}(\{\lambda^N(y^1, y^2), \lambda^3\})$,
- $\|y^3\|_\infty^{\lambda^3} = \lambda_j^3 y_i^3$ for $i \in I^3$ and $\|y^3\|_\infty^{\lambda^3} > \lambda_k^3 y_k^3$ for $k \notin I^3$,
- $\|y^N(y^1, y^2)\|_\infty^{\lambda^3} = \lambda_i^3 y_i^N(y^1, y^2)$ for $i \in I^1 \cup I^2$ and $\|y^N(y^1, y^2)\|_\infty^{\lambda^3} > \lambda_k^3 y_k^N(y^1, y^2)$ for $k \notin I^1 \cup I^2$,
- $y_i^N(y^1, y^2) > y_i^3$ for all $i \in I^1 \cup I^2$,
- $y_j^N(y^1, y^2) < y_j^3$ for all $j \in I^3$.

This implies

- $I^r \cap I^s = \emptyset$ for $r, s \in \{1, 2, 3\}$,
- For all $r \in \{1, 2, 3\}$ it holds $y_i^r > y_i^s$ for all $i \in I^r$ and $s \in \{1, 2, 3\} \setminus \{r\}$,
- $\lambda^N(y^1, y^2, y^3) \in \Lambda(y^2) \cap \Lambda(y^2) \cap \Lambda(y^3)$,
- $\lambda \in \text{conv}(\{\lambda^N(y^1), \lambda^N(y^1, y^2), \lambda^N(y^1, y^2, y^3)\})$.

If $I^1 \cup I^2 \cup I^3 = \{1, \ldots, p\}$, the construction terminates. Otherwise, we can, again, apply Lemma 2 with $y^N(y^1, y^2, y^3), I = I^1 \cup I^2 \cup I^3$ and λ^3.

This construction step can be repeated at most $p - 1$ times until we obtain the desired set of images $y^1, \ldots, y^R$ and index sets $I^1, \ldots, I^R$. $\square$

7.5. Comparing Decompositions

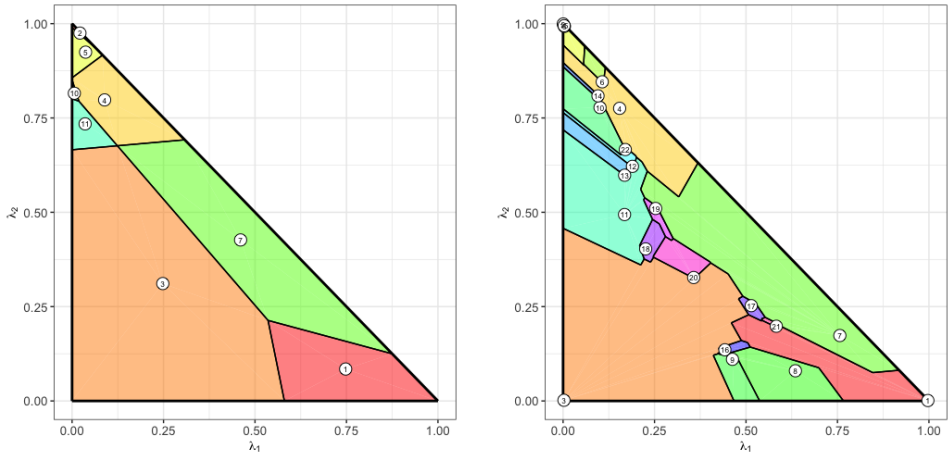
Five instances of ND sets are considered, each defined as a (MODO) instance with 25 variables. WSDs for the first instance have already been presented in Figure 1; WSDs for the remaining instances are illustrated in Figures 11 and 12.

For each of the five instances and each ND image, the area assigned to the corresponding weight set component are compared between the two decompositions in Figure 13. These paired bar graphs represent each image with *an adjacent pair of bars* where the height

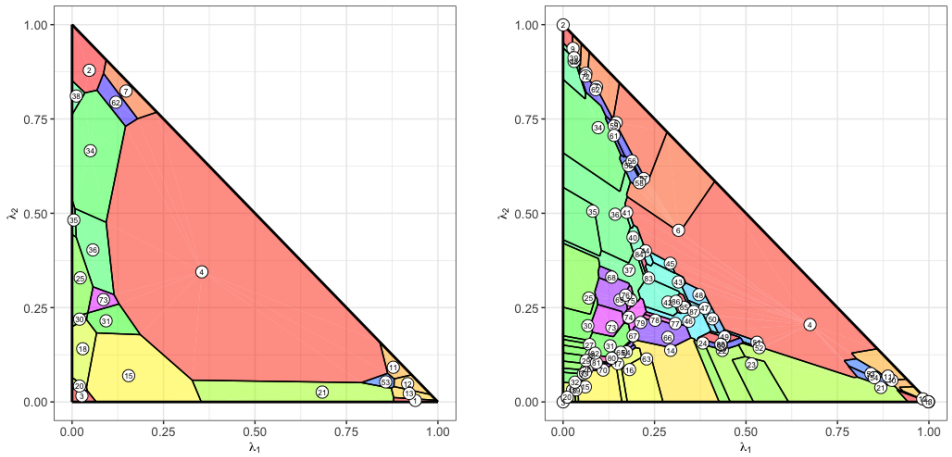
represents area of its weight set component: blue for weighted Tchebychev and red for weighted sum.

7.6. Computational study

For our computational results, we use *simulated* computational studies, i.e., the image sets are known a priori and therefore no optimization subproblem is explicitly solved. Such a simulated study is representative of measuring the performance of the algorithm when the optimization subproblems are *normalized* for length of run time. In practice, the runtime of alternative optimization subproblems is a serious factor to consider for black-box optimization software. However, as the subproblems are not the focus of our work, we use this standardization as a control method. In place of the optimization oracle (*opt* in



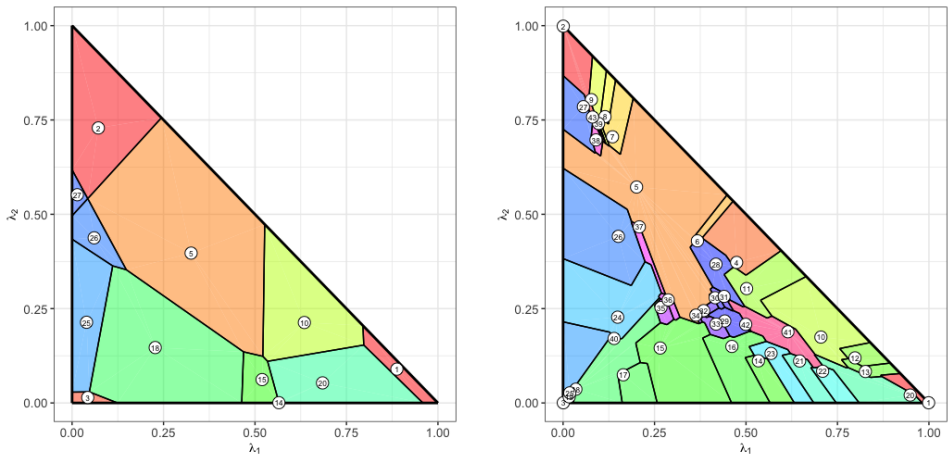
(a) Sample set with 22 ND images, including 8 ESND images.



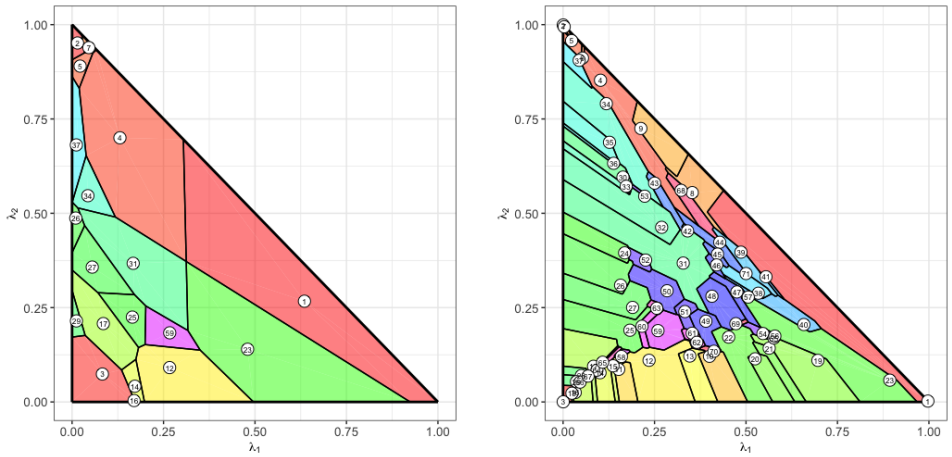
(b) Sample set with 88 ND images, including 21 ESND images.

Figure 11 **Comparing WSDs with respect to weighted sum scalarization (left) and weighted Tchebychev scalarization (right) for sample ND sets. Color key is consistent between pairs.**

line 6 of Algorithm 1), we utilize a function which returns a minimizing solution for the weighted sum scalarization with equal weights and budget constraints on the objectives, where the upper bound constraints is given by the LNP of the box.

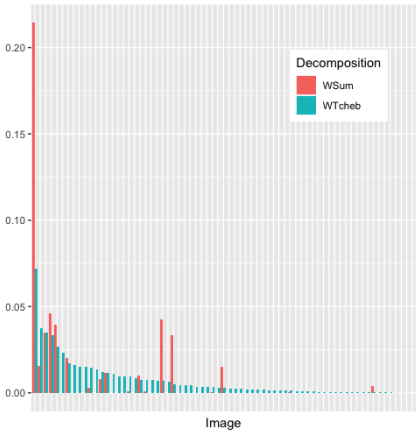


(a) Sample set with 43 ND images, including 11 ESND images.

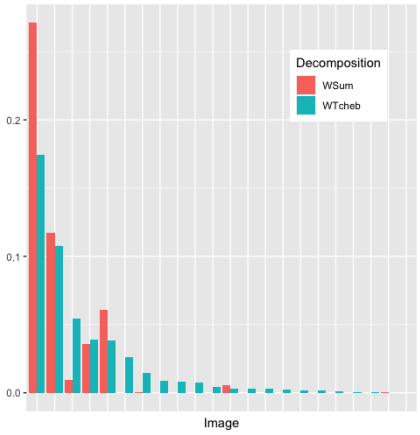


(b) Sample set with 71 ND images, including 18 ESND images.

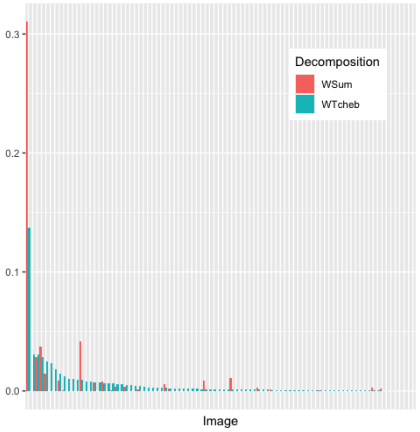
Figure 12 **Comparing WSDs with respect to weighted sum scalarization (left) and weighted Tchebychev scalarization (right) for sample ND sets. Color key is consistent between pairs.**



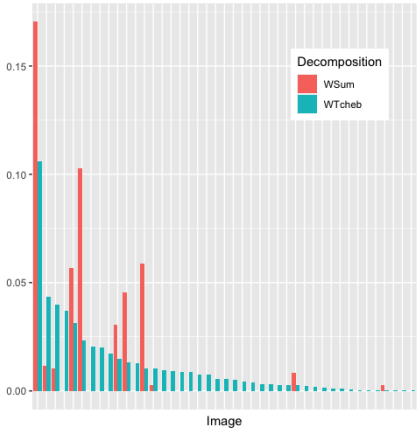
(a) Instance in Figure 1.



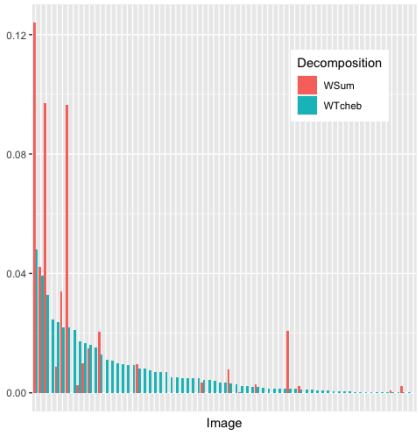
(b) Instance in Figure 11.



(c) Instance in Figure 12(a).



(d) Instance in Figure 12(b).



(e) Instance in Figure 12(c).

Figure 13 **Paired bar graphs compare weight set component area for decompositions with respect to weighted sum scalarization (red) and weighted Tchebychev scalarization (blue) for sample ND sets. The y-axis represents area, and the (pairs of) bars are ordered in descending order of area for the weighted Tchebychev decomposition.**

References

Alves MJ, Costa JP (2016) Graphical exploration of the weight space in three-objective mixed integer linear programs. *European Journal of Operational Research* 248(1):72 – 83, ISSN 0377-2217, URL http://dx.doi.org/https://doi.org/10.1016/j.ejor.2015.06.072.

Aneja Y, Nair PK (1979) Bicriteria transportation problem. *Management Science* 25:73–78, URL http://dx.doi.org/https://doi.org/10.1287/mnsc.25.1.73.

Benson HP, Sun E (2000) Outcome space partition of the weight set in multiobjective linear programming. *Journal of Optimization Theory and Applications* 105(1):17–36, ISSN 1573-2878, URL http://dx.doi.org/https://doi.org/10.1023/A:1004605810296.

Benson HP, Sun E (2002) A weight set decomposition algorithm for finding all efficient extreme points in the outcome set of a multiple objective linear program. *European Journal of Operational Research* 139:26–41, URL http://dx.doi.org/https://doi.org/10.1016/S0377-2217(01)00153-9.

Berezkin VE, Kamenev GK, Lotov AV (2006) Hybrid adaptive methods for approximating a nonconvex multidimensional pareto frontier. *Computational Mathematics and Mathematical Physics* 46(11):1918–1931.

Boland N, Charkhgard H, Savelsbergh M (2015) A criterion space search algorithm for biobjective integer programming: the balanced box method. *INFORMS Journal on Computing* 27(4):735–754.

Boland N, Charkhgard H, Savelsbergh M (2016) The l-shape search method for triobjective integer programming. *Mathematical Programming Computation* 8(2):217–251.

Boland N, Charkhgard H, Savelsbergh M (2017) The quadrant shrinking method: A simple and efficient algorithm for solving tri-objective integer programs. *European Journal of Operational Research* 260(3):873–885.

Bowman VJ (1976) On the relationship of the tchebycheff norm and the efficient frontier of multiple-criteria objectives. Thiriez H, Zionts S, eds., *Multiple Criteria Decision Making*, 76–86 (Berlin, Heidelberg: Springer Berlin Heidelberg), ISBN 978-3-642-87563-2, URL http://dx.doi.org/https://doi.org/10.1007/978-3-642-87563-2_5.

Chan TCY, Craig T, Lee T, Sharpe MB (2014) Generalized inverse multiobjective optimization with application to cancer therapy. *Operations Research* 62(3):680–695, URL http://dx.doi.org/https://doi.org/10.1287/opre.2014.1267.

Cohon JL (2003) *Multiobjective programming and planning. Reprint of the 1978 original ed.* (Mineola, NY: Dover Publications), reprint of the 1978 original ed. edition, ISBN 0-486-43263-7/pbk.

Cui Y, Geng Z, Zhu Q, Han Y (2017) Multi-objective optimization methods and application in energy saving. *Energy* 125:681–704.

Czyżak P, Jaszkiewicz A (1997) Pareto simulated annealing. *Multiple criteria decision making*, 297–307 (Springer).

Dai R, Charkhgard H (2018) Bi-objective mixed integer linear programming for managing building clusters with a shared electrical energy storage. *Computers & Operations Research* 96:173–187.

Dächert K, Halffmann P, Klamroth K, Ruzika S, Schäfer L (2021) Exact algorithms for multiobjective linear optimization problems with integer variables – a state of the art survey. Submitted.

Dächert K, Klamroth K (2015) A linear bound on the number of scalarizations needed to solve discrete tricriteria optimization problems. *Journal of Global Optimization* 61, URL http://dx.doi.org/10.1007/s10898-014-0205-z.

Dächert K, Klamroth K, Lacour R, Vanderpooten D (2017) Efficient computation of the search region in multi-objective optimization. *European Journal of Operational Research* 260(3):841 – 855, ISSN 0377-2217, URL http://dx.doi.org/https://doi.org/10.1016/j.ejor.2016.05.029.

Ehrgott M (2005) *Multicriteria optimization* (Berlin, Heidelberg: Springer-Verlag), 2nd edition, ISBN 3540213988.

Ehrgott M, Güler Ç, Hamacher HW, Shao L (2010) Mathematical optimization in intensity modulated radiation therapy. *Annals of Operations Research* 175(1):309–365.

Eswaran PK, Ravindran A, Moskowitz H (1989) Algorithms for nonlinear integer bicriterion problems. *Journal of Optimization Theory and Applications* 63(2):261–279, ISSN 1573-2878, URL http://dx.doi.org/https://doi.org/10.1007/BF00939577.

Faulkenberg SL, Wiecek MM (2010) On the quality of discrete representations in multiple objective programming. *Optimization and Engineering* 11(3):423–440.

Geoffrion AM (1968) Proper efficiency and the theory of vector maximization. *Journal of Mathematical Analysis and Applications* 22(3):618 – 630, ISSN 0022-247X, URL http://dx.doi.org/http://dx.doi.org/10.1016/0022-247X(68)90201-1.

Halffmann P (2021) *Advances in Multiobjective Optimisation - Scalarisation, Approximation, and Complexity*. Ph.D. thesis, Technische Universität Kaiserslautern.

Halffmann P, Dietz T, Przybylski A, Ruzika S (2020) An inner approximation method to compute the weight set decomposition of a triobjective mixed-integer problem. *Journal of Global Optimization* URL http://dx.doi.org/http://doi.org/10.1007/s10898-020-00898-9.

Halffmann P, Schäfer LE, Dächert K, Klamroth K, Ruzika S (2022) Exact algorithms for multiobjective linear optimization problems with integer variables: A state of the art survey. *Journal of Multi-Criteria Decision Analysis* .

Helfrich S, Perini T, Halffmann P, Boland N, Ruzika S (2020) A star among scalarizations: the weighted tchebycheff weight set decomposition for multiobjective discrete optimization, submitted.

Heyde F, Löhne A (2008) Geometric duality in multiple objective linear programming. *SIAM Journal on Optimization* 19:836–845, URL http://dx.doi.org/https://doi.org/10.1137/060674831.

Hooker JN (2009) Integer programming duality. *Encyclopedia of Optimization* 2:533–543.

Karakaya G, Köksalan M (2021) Evaluating solutions and solution sets under multiple objectives. *European Journal of Operational Research* 294(1):16–28.

Klamroth K, Lacour R, Vanderpooten D (2015) On the representation of the search region in multi-objective optimization. *European Journal of Operational Research* 245(3):767 – 778, ISSN 0377-2217, URL http://dx.doi.org/https://doi.org/10.1016/j.ejor.2015.03.031.

Lotov AV, Bushenkov VA, Kamenev GK (2013) *Interactive decision maps: Approximation and visualization of Pareto frontier*, volume 89 (Springer Science & Business Media).

Luc DT (2011) On duality in multiple objective linear programming. *European Journal of Operational Research* 210(2):158–168.

Luque M, Ruiz F, Steuer RE (2010) Modified interactive chebyshev algorithm (mica) for convex multiobjective programming. *European Journal of Operational Research* 204(3):557 – 564, ISSN 0377-2217, URL http://dx.doi.org/https://doi.org/10.1016/j.ejor.2009.11.011.

Miettinen K (2014) Survey of methods to visualize alternatives in multiple criteria decision making problems. *OR spectrum* 36(1):3–37.

Musselman A, Thomas VM, Boland N, Nazzal D (2019) Optimizing wind farm siting to reduce power system impacts of wind variability. *Wind Energy* 22(7):894–907.

Perini T, Boland N, Pecin D, Savelsbergh M (2020) A criterion space method for biobjective mixed integer programming: The boxed line method. *INFORMS Journal on Computing* 32(1):16–39.

Preparata FP, Shamos MI (1988) *Computational geometry* (New York u.a.: Springer), corr. and expanded 2. print. edition, ISBN 3-540-96131-3.

Przybylski A, Gandibleux X, Ehrgott M (2010a) A recursive algorithm for finding all nondominated extreme points in the outcome set of a multiobjective integer programme. *INFORMS Journal on Computing* 22(3):371–386, URL http://dx.doi.org/https://doi.org/10.1287/ijoc.1090.0342.

Przybylski A, Gandibleux X, Ehrgott M (2010b) A two phase method for multi-objective integer programming and its application to the assignment problem with three objectives. *Discrete Optimization* 7(3):149 – 165, ISSN 1572-5286, URL http://dx.doi.org/https://doi.org/10.1016/j.disopt.2010.03.005.

Ralphs TK, Saltzman MJ, Wiecek MM (2006) An improved algorithm for solving biobjective integer programs. *Annals of Operations Research* 147(1):43–70, ISSN 1572-9338, URL http://dx.doi.org/https://doi.org/10.1007/s10479-006-0058-z.

Romeijn HE, Dempsey JF (2008) Intensity modulated radiation therapy treatment plan optimization. *TOP* 16(2):215, ISSN 1863-8279, URL http://dx.doi.org/https://doi.org/10.1007/s11750-008-0064-1.

Steuer RE, Choo EU (1983) An interactive weighted tchebycheff procedure for multiple objective programming. *Mathematical Programming* 26(3):326–344, ISSN 1436-4646, URL http://dx.doi.org/https://doi.org/10.1007/BF02591870.

Tamby S, Vanderpooten D (2020) Enumeration of the nondominated set of multiobjective discrete optimization problems. *INFORMS Journal on Computing* .

van der Merwe M, Ozlen M, Hearne JW, Minas JP (2017) Dynamic rerouting of vehicles during cooperative wildfire response operations. *Annals of Operations Research* 254(1):467–480.

Xin B, Chen L, Chen J, Ishibuchi H, Hirota K, Liu B (2018) Interactive multiobjective optimization: A review of the state-of-the-art. *IEEE Access* 6:41256–41279.

Yu P, Zeleny M (1975) The set of all nondominated solutions in linear cases and a multicriteria simplex method. *Journal of Mathematical Analysis and Applications* 49(2):430 – 468, ISSN 0022-247X, URL http://dx.doi.org/https://doi.org/10.1016/0022-247X(75)90189-4.