Optimization Methods and Software

ISSN: 1055-6788 (Print) 1029-4937 (Online) Journal homepage: www.tandfonline.com/journals/goms20

Multiobjective optimization by using cutting angle methods and hypervolume criterion

Gleb Beliakov, Longxiang Gao, Yong Xiang & Wanlei Zhou

To cite this article: Gleb Beliakov, Longxiang Gao, Yong Xiang & Wanlei Zhou (2025) Multiobjective optimization by using cutting angle methods and hypervolume criterion, Optimization Methods and Software, 40:2, 388-405, DOI: 10.1080/10556788.2024.2405984

To link to this article: https://doi.org/10.1080/10556788.2024.2405984

Published online: 02 Oct 2024.

OPTIMIZATION METHODS & SOFTWARE
2025, VOL. 40, NO. 2, 388–405
https://doi.org/10.1080/10556788.2024.2405984

Taylor & Francis
Taylor & Francis Group

Multiobjective optimization by using cutting angle methods and hypervolume criterion

Gleb Beliakov[a], Longxiang Gao[a], Yong Xiang[a] and Wanlei Zhou[b]

[a]School of Information Technology, Deakin University, Geelong, Australia; [b]School of Computer Science, University of Technology Sydney, Sydney, Australia

ABSTRACT

We translate a multiobjective optimization problem into a single objective Lipschitz problem by using the hypervolume criterion of the Pareto set. Deterministic global optimization methods allow one to track the whole Pareto front rather than converging to a single non-dominated solution. We augmented an efficient hypervolume computation technique with hypervolume increment strategy, and established Lipschitzianity of the resulting objective function. We used two deterministic Lipschitz optimization methods together with the hypervolume objective and benchmarked them against some state-of-the-art alternative multiobjective optimization methods, establishing the competitiveness of the proposed approach.

KEYWORDS

Multiobjective optimization; global optimization; hypervolume; abstract convex functions

2020 MATHEMATICS SUBJECT CLASSIFICATIONS

90C26; 65K05

1. Introduction

We consider the multiple objective (MO) optimization problem, in which a vector function $f : X \to \mathbb{R}^k$ needs to be minimized over some feasible set $X \subset \mathbb{R}^n$. For simplicity we take X as a hyperrectangle or a simplex. In difference to single objective (SO) optimization problems, which aim at locating the local or global minimizers of a scalar objective f, MO optimization aims at determining the set of non-dominating solutions (Pareto set). The Pareto set can be presented to problem domain experts in order for them to find compromise solutions.

MO optimization problems appear in many applications [19,23,29,39,49,51]. In this work we are motivated by the modern digital watermarking methods which aim at embedding imperceptible and non-removable pieces of information into digital audio, image or video files [1,36,65]. Such embedding creates small degradations in the quality of the media, and the goal is to embed a string of watermarks of a given length while minimizing the quality degradation and maximizing other criteria such as the difficulty of watermark detection and removal by attackers. The decision variables are the parameters of the watermarking algorithm and the objective function is typically given as an oracle and is numerically expensive to evaluate.

A classical approach to MO problems is based on scalarisation of the objective [23,24,29,51,56,60]. For example, taking a linear convex combination of the objectives

CONTACT Gleb Beliakov gleb@deakin.edu.au 75 Pigdons Road, Waurn Ponds, Victoria 3216, Australia

leads to a SO problem. This approach presumes a proper articulation of preferences and tradeoffs between the objectives by the domain experts. The area of aggregation functions [9,10] provides many alternative scalarisation techniques, including assigning weights to the objectives and their combinations. Scalarisation methods imply that a single optimal solution will be found, but not the Pareto set.

Scalarisation techniques can also be applied dynamically in an approach known as active learning of Pareto front [18,33,43,44,48,61]. Here scalar objectives are determined dynamically from the available function values, such that their level sets approximate Pareto fronts at the current stage of the algorithm, and are continuously updated whenever the Pareto fronts are updated. The scalarised objective guides the SO optimization algorithm, but in contrast to the fixed scalarised objective, the method eventually determines the Pareto front over X.

Evolutionary computing (EC) is another popular strategy for MO problems [3,16,20,21, 27,29,54,66,69,70]. Such methods explore the feasible set in order to construct a population of optimal non-dominated solutions which will eventually approximate the Pareto set. Such methods converge to the Pareto set probabilistically, and as with SO, evolutionary computing requires a very large number of objective evaluations. Detailed benchmarking of such methods is provided elsewhere [29,59,69,70].

In difference to SO problems, the criterion of success in MO is not the rate of convergence to a local/global optimum, nor the value of the putative global optimum in multi-extremal problems, but a more vaguely defined quality of approximating the Pareto set [2,3,23,26,68,70]. One formal criterion is the hypervolume (HV): the volume of the set of Pareto points (relative to the nadir or its approximation). It was shown that the hypervolume is the only criterion monotonically increasing with the Pareto set [2,3] (in terms of dominance relation).

The monotonicity of the hypervolume is a very desirable feature, as its larger values indicate larger Pareto sets found by algorithms, not only in terms of their cardinality but also in terms of diversity. Significantly larger hypervolume values indicate significant Pareto points well away from others, and hence add value to the decision makers who may have otherwise overlooked such alternatives.

It is therefore meaningful to formulate the MO problem as that of maximizing its success criterion – the Pareto set hypervolume. Then one can convert the MO to SO problem without eliciting the trade-offs between the objectives from the end users (in contrast to the scalarisation methods). Of course it is the end users who make the final choice of selecting an alternative from the computed Pareto set.

Using the hypervolume as the scalar objective has two challenges. Firstly, hypervolume computation in more than two variables (objectives) is expensive. Secondly, hypervolume is a very complicated multiextremal function of the problem variables $x \in X$. In fact, hypervolume is not even a well defined function of the variables x, as its values at a given x depend on the history of the objective evaluations by the algorithm. When we pick an x whose image is in the Pareto set, the value of the hypervolume is dependent on what other values of f are known, i.e. it is a function of the whole known Pareto set.

Nevertheless there was recent interest in using hypervolume as a single ultimate objective in MO problems [2,3,13,17,33,64]. The specifics of hypervolume-based optimization restricts the suitable SO optimization methods to those that track the whole set of potentially optimal solutions, and balance the requirements of improving the objective and sampling sufficiently densely the search domain.

One recent approach involves application of the DIRECT (DIviding RECTangles) algorithm [41,42], which is a multivariate extension of Lipschitz deterministic optimization methods by Pijavsky and Shubert [52,57], to MO problems by using the hypervolume as the single objective. Besides an efficient domain partition scheme into hyperrectangles, DIRECT avoids the reliance on an estimate of the Lipschitz constant, by balancing the value of possible local minima of the objective in a given region and the size of the region. The recent papers [17,26,45–47,64] develop MO versions of the DIRECT algorithm and [30,68] look at a different extension of univariate Lipschitz optimization. In particular, deterministic algorithms based on hypervolume maximization were considered in [64]. For an overview of various most recent MO approaches based on modifications of DIRECT and other deterministic methods we refer to [47].

In this contribution we follow the above mentioned trend and use a combination of the hypervolume criterion with a different extension of the Pijavsky-Shubert method, called the Extended Cutting Angle method (ECAM) [8]. The contributions and novelty of this paper are two-fold.

Firstly we propose to use an alternative global optimization algorithm for multiobjective optimization in order to track the whole Pareto front rather than converging to a few solutions, and compare it to various alternatives. Secondly, instead of scalarising the objectives using some specified tradeoffs, we use the hypervolume criterion as the objective, which is known to be monotonically increasing with the quality of the solution (i.e. with the Pareto front). We show that such an objective can be calculated incrementally and is Lipschitz continuous, hence suitable for deterministic global optimization. To achieve computational efficiency we employ recent developments in multivariate hypervolume computations [62,63], and also design the hypervolume updating technique for its incremental computation.

In difference to the previous works which combine multivariate versions of Lipschitz optimization, such as DIRECT, with other selection criteria, we rely on the hypervolume-based objective as in [2,3,13,64], but in difference to those works, and also under the constraint of limited function evaluations, we do not employ evolutionary computing but rather focus on deterministic methods such as in [17,26,45–47,64,68]. We illustrate that within a limited functions evaluations budget ECAM outperforms DIRECT in problems with higher dimensionality but at the expense of increased CPU time.

In Section 2, we present the MO problem formulation and some standard definitions. We outline the Extended Cutting Angle Method in Section 3, which is a SO optimization method to be used in conjunction with the hypervolume criterion. In Section 4, we translate the multiobjective problem into SO problem and discuss the properties of the hypervolume objective. The combination of the deterministic global optimization with the hypervolume criterion is in Section 5, where we benchmark two methods on a set of challenging MO test problems, as well as against several state-of-the-art methods based on Evolutionary Compting. We conclude in Section 6.

2. Problem formulation

We consider the following MO optimization problem

$$\begin{array}{ll} \text{Minimise} & f(x) = (f_1(x), \ldots, f_k(x)) \\ \text{s.t.} & x \in X \subset \mathbb{R}^n, \end{array} \tag{1}$$

where X is a compact feasible set, such as a hyperrectangle or a simplex. The treatment of constraints is left apart, relying on many traditional approaches in SO optimization, such as penalty functions.

Definition 2.1: Vector $u \in \mathbb{R}^k$ dominates $v \in \mathbb{R}^k$ whenever $u_i \leq v_i$ for all $i = 1, \ldots, k$ and for at least one index $u_i < v_i$. The point $y = f(x^*) \in R^k$ is a Pareto optimum if it is not dominated by any other point $z = f(x), x \in X$. The set of the Pareto optimal function values is called the Pareto front (or Pareto set) $Par(X)$.

The goal of the MO optimization is to determine the Pareto set, although for continuous problems a representative subset is sufficient. Also note that because we focus on minimizing the objectives, the dominating relation involves the $<$ inequality.

Definition 2.2 ([25]): Assume that the Pareto set is non-empty. The vector $(\sup f_1(x), \sup f_2(x), \ldots, \sup f_k(x))$ is called the nadir point and$(\inf f_1(x), \inf f_2(x), \ldots, \inf f_k(x))$ is called the ideal point. The componentwise sup and *inf* are taken independently over $Par(X)$.

If we further assume that $Par(X)$ is compact then the sup and *inf* are attained and hence replaced with max and min, and also the determination of the ideal point can be done by solving independently k SO problems over X. The determination of the nadir point is much harder, but an upper approximation can be found similarly to the calculation of the ideal point by maximizing over X [25].

Generally the ideal point is not reached at any $x \in X$, because the objectives are conflicting. If the ideal point is reached, then it is the only Pareto optimal point.

Definition 2.3: The hypervolume of a set S relative to the nadir point is the Lebesgue measure of the set of points dominated by those in S.

The weighted hypervolume indicator was proposed in [70]; it amounts to using the weighted Lebesgue measure in the definition, where the weighting function depends on the points.

The hypervolume indicator hv is monotonically non-decreasing with respect to the dominance relation: if S dominates Q then $hv(S) \geq hv(Q)$ [2,3]. In fact it is the only indicator with this property [70], and it is one of the reasons for its popularity. Monotonicity of an indicator is a very valuable feature for MO problems which is related to the diversity of Pareto points. The computational complexity of hypervolume algorithms is exponential in k: $O(N^{k-1})$ [63], where N is the cardinality of the (discrete) Pareto set. Recent advances reduced the complexity to $O(N^{k-2} \log N)$ [28], and a specialized algorithm further reduced it to $O(N \log N + N^{k/2})$ [12,31,62]. The results in [15] prove that hypervolume computation is a #P-complete problem.

As we mentioned in the Introduction, our goal here is to convert an MO problem to an SO problem, using the hypervolume indicator as the objective function following [3,64], which is ultimately a criterion by which the success of Pareto front approximation will be measured. We are interested in a deterministic method similar to DIRECT, which would help us track the Pareto front by building its lower approximation. For this reason we recur to multivariate Lipschtz programming detailed in the next section.

3. Cutting angle methods

The field of Lipschitz programming – locating local and global optima of Lipschitz-continuous functions – has gained practical importance and applicability in the past decades due to theoretical developments, design of the algorithms and increased computing power [34,35,51,53]. Lipschitz properties of the objective function allow one to determine exact bounds on the globally optimal solutions, and in many cases find and confirm the global optima with reasonable computational effort. Overviews of the developments in deterministic global optimization are found in [32,35,51,53].

Several methods of Lipschitz optimization are based on constructing lower approximations (underestimates) to the objective function f using its known values. The global minima of the underestimates are taken as the lower bounds on the global minimum of f. Convergence of the sequence of the lower bounds to the global minimum of f is proven under very mild conditions, see [53,55]. Locating the global minimum of the underestimate is a simpler (yet still very challenging) problem compared to minimizing f. One replaces the original optimization problem with a sequence of relaxed problems, whose solutions converge to the global minimum of f.

The Cutting Angle method (CAM) [4,55] uses max-min type functions to build the underetimates to f. The CAM is based on the generalized result from *abstract convex* analysis, namely that every abstract convex function is the upper envelop of its sufficiently simple minorants [55]. It is shown in [55] that Lipschitz functions defined on the unit simplex S can be seen as restrictions of certain abstract convex functions defined on the cone $\mathbb{R}^n_+$. This gives a way to apply CAM to Lipschitz functions defined on the unit simplex.

Following, in [7,8] another type of max-min support functions was used to create the Extended CAM (ECAM)

$$h(x) = \min_{i=1,\ldots,n} (C_i x_i + b_i), \quad C_i > 0,\ x \in \mathbb{R}^n : \sum_{i=1}^{n} x_i = 1. \tag{2}$$

These functions are more flexible in building tighter underestimates of Lipschitz objective functions, and coefficients C_i can be chosen to match the Lipschitz constant of f. One-dimensional Pijavski-Shubert method (and its variations [32,52,57,58]) arises as a special case of ECAM.

A function f is abstract convex with respect to the set of functions H (or H-convex) if there exists $U \subset H$: $f(x) = \sup\{h(x) : h \in U\}, \forall\, x \in X$. The set of H-minorants of f is called the support set of f with respect to the set of functions H: $supp(f, H) = \{h \in H, h \leq f\}$. H-subgradient of f at x is a function

$$h \in H : f(y) \geq h(y) - (h(x) - f(x)), \quad \forall\, y \in X.$$

The set of all H-subgradients of f at x is called H-subdifferential

$$\partial_H f(x) = \{h \in H :\ \forall\, y \in X, f(y) \geq h(y) - (h(x) - f(x))\}.$$

CAM (and ECAM) is based on the following approach to minimizing f. It is an iterative process of building piecewise affine underestimates of f, $H^K(x)$, $K = n, n+1, \ldots$, using K known values of f at $x^k, k = 1, \ldots, K$. At each iteration K, the following relaxed problem

is solved

$$\begin{array}{ll} \min & H^K(x) \\ \text{s.t.} & x \in X. \end{array} \tag{3}$$

The global minimizer of H^K is chosen as x^{K+1}, f is evaluated at x^{K+1} and H^K is updated to H^{K+1}.

CAM is a version of the Generalized Cutting Plane algorithm, presented in [55], p.405 and reproduced below.

Algorithm 1 (Generalized Cutting Plane Algorithm): *Step 0. (Initialisation)*

0.1 *Set* $K = 1$.
0.2 *Choose an arbitrary initial point* $x^1 \in X$.
0.3 *Set* $f_{best} = f(x^1)$.

Step 1. (Calculate H-subgradient)

1.1 *Calculate* $h^K \in \partial_H f(x^K)$.
1.2 *Define* $H^K(x) = \max_{k=1,\ldots,K} h^k(x)$, *for all* $x \in X$.

Step 2. (Minimize H^K*)*

2.1 *Solve relaxed Problem (3). Let* x^* *be its solution.*
2.2 *Set* $K = K + 1, x^K = x^*$.
2.3 *Set* $f_{best} = \min\{f(x^K), f_{best}\}$.

Step 3. (Stopping criterion)

3.1 *If* $K < K_{max}$ *and* $f_{best} - H^K(x^*) > \epsilon$ *go to Step 1.*

The underestimate H^K is often called the *saw-tooth* underestimate, or saw-tooth cover of f, because of its shape. In the ECAM the support functions are

$$h^k(x) = \min_{i \in I}(f(x^k) - C_i(x_i^k - x_i)) = \min_{i \in I}(C_i x_i + b_i^k), \tag{4}$$

where $C_i \geq M$ are fixed numbers greater or equal to the Lipschitz constant of f on X, and $b_i^k = f(x^k) - C_i x_i^k$, and $I \subset \{1, s, \ldots, K\}$ has cardinality $n = m + 1$, where m is the dimension of the original problem (dimension of X). Note that a slack variable x_{m+1} is used. Then all h^k approximate f from below. The method is applicable to minimization of abstract convex functions with respect to (4), which include all Lipschitz functions on $\mathbb{R}^{n-1}$ with $n = m + 1$.

The form of the functions h^k in (4) resembles various attempts to generalize Pijavski-Shubert method [32,52,55,57,58], p.417, using underestimates

$$H^K(x) = \max_{k=1,\ldots,K}(f(x^k) - C||x - x^k||). \tag{5}$$

The difference is that instead of a norm $||\cdot||$ ECAM uses a polyhedral distance function d_P (in fact, the simplicial distance, see [8]), so that (5) becomes

$$H^K(x) = \max_{k=1,\ldots,K} h^k(x) = \max_{k=1,\ldots,K} (f(x^k) - Cd_P(x, x^k)). \tag{6}$$

For the purposes of convenience, we introduce a slack variable $x_{m+1} = 1 - \sum_{i=1}^{m} x_i$ and put $n = m+1$. With the help of the new coordinate we can write the saw-tooth underestimate H^K as

$$H^K(x) = \max_{k=1,\ldots,K} (f(x^k) - Cd_P(x, x^k)) = \max_{k=1,\ldots,K} \min_{i\in I} (f(x^k) - C_i(x_i^k - x_i)). \tag{7}$$

Notice that the values $x_1, \ldots, x_n$ are restricted only by $\sum_{i=1}^{n} x_i = 1$. For $n = m+1 = 2$ it coincides with the Pijavski-Shubert saw-tooth underestimate, but differs from other approaches which extend it to multiple variables.

The reason for choosing support functions (4), as opposed to using other distances in (6), is that the relaxed problem can be solved very efficiently by enumerating all local minima of H^K using an approach similar to the one in [6,7].

Using the values of the function f at the points $x^k, k = 1, \ldots, K$, let us define the *support vectors* l^k:

$$l_i^k = \frac{f(x^k)}{C_i} - x_i^k. \tag{8}$$

The support functions can be expressed as

$$h^k(x) = \min_{i\in I} (f(x^k) - C_i(x_i^k - x_i)) = \min_{i\in I} C_i(l_i^k + x_i). \tag{9}$$

We will enumerate all local minimizers of the function H^K in X, which after sorting will yield its global minimum.

Proposition 3.1 ([8]): *The necessary and sufficient condition for a point $x^* \in \mathrm{ri}\, X$ to be a local minimizer of $H^K = \max_{k=1,\ldots,K} h^k$ and h^k given by (9), is that there exists an index set $J = \{k_1, k_2, \ldots, k_n\}$ of cardinality n, such that*

$$d = H^K(x^*) = C_1(l_1^{k_1} + x_1^*) = C_2(l_2^{k_2} + x_2^*) = \ldots = C_n(l_n^{k_n} + x_n^*),$$

and $\forall\, i \in I$, $C_i(l_i^{k_i} + x_i^*) < C_j(l_j^{k_i} + x_j^*), j \neq i$.

Now the local minimizers enumeration procedure works by identifying the subsets of indices $L = \{l^{k_1}, l^{k_2}, \ldots, l^{k_n}\}$ which satisfy the conditions of Proposition 3.1.

Furthermore, in [5,6] it was shown that combinations L can be built incrementally, by taking initially the first n support vectors (which yields the unique combination $L = \{l^1, l^2, \ldots, l^n\}$), and then adding one new support vector at a time, as well as efficiently keeping these combinations in an n-ary tree data structure. That is, all local minima of functions $H^n, H^{n+1}, \ldots, H^k, \ldots, H^K$ can be seen as nodes of a tree, and the minima of H^K are the leaves of this tree. The root of the tree is $V^n = \{(l^1, l^2, \ldots, l^n)\}$. The parent and child nodes of this tree differ only by one support vector, and if a child node fails conditions of Proposition 3.1, its parent (and all ancestors) will also fail it. Then a tree-traversal

algorithm for incremental enumeration of all local minima of functions H^K was presented in [5,6], and it has complexity $O(\log|V^K|))$, where $|V^K|$ is the number of local minima of H^K.

We underline an important fact here: The ECAM method maintains a lower approximation to the objective (to be minimized) over the feasible domain X. In difference to stochastic methods which may focus on some of the promising parts of the domain, even though for a very large number of function evaluations they too sample X sufficiently densely, deterministic methods always maintain sampling density, as unexplored parts of the domain result in poorer approximation of the objective and hence are detected and favoured by the algorithm. Thus deterministic methods do not focus on reaching one particular optimum but rather confirming that there are no other optima, so building an accurate overall approximation of the objective is inherent in the spirit of such algorithms. In the hypervolume based MO context this comes very handy as it translates into an approximation of the whole Pareto front as a consequence, even though such an approximation is not explicit.

Another noteworthy feature of the ECAM is its efficiency in building the lower approximation, which requires only the set of previously evaluated function values put into an efficient tree structure, which tracks only the local minima of the lower approximation. These minima correspond in fact to the points of the future evaluations of the objective, and consequently one can limit the growth of that tree to a function of the available function evaluations budget B, thus limiting the complexity of the algorithm to $O(B\log(B))$ (due to logarithmic complexity of tree traversal).

4. Hypervolume objective

Let us now relate the MO problem (1) to the hypervolume-based SO problem to which we subsequently apply various SO optimization methods. Consider the set of points $S = \{y = f(x^k), k = 1, 2, \ldots, K\}$, generated by an optimization algorithm in K steps. The hypervolume is a set function $HV : S \to \mathbb{R}_+$, which takes into account all the points y seen by an algorithm. To formulate an SO problem we need a function which depends on points $x \in X$ with suitable properties. Hence we define the scalar function $Hv : X \to \mathbb{R}_+$, by using $Hv(x) = HV(S \cup \{f(x)\})$. This function Hv depends on x and the history of evaluations, but for any fixed step K of an algorithm (hence fixed S), is well defined and depends on x only. The choice of x is governed by the SO optimization algorithm, and from the algorithmic perspective Hv is viewed as any other objective function, which is Lipschitz-continuous as we will show in a moment. This way the MO problem is translated into a SO Lipschitz optimization problem to which we apply the methods of Lipschitz programming from Section 3.

Our single objective to minimize is the negative hypervolume Hv relative to an upper approximation to the nadir point, computed as the pointwise maximum of the individual objectives $N = (\max f_1(x), \ldots, \max f_k(x))$ over X.

Note two observations. Firstly, we are not computing the true nadir point (over $Par(X)$) as in [25], it is unnecessary here, as at some stages of the algorithms the hypervolume involving points whose objectives are worse than those of the nadir (e.g. the points in the top right corner in Figure 1). Such a reference point is much easier to compute than the nadir.

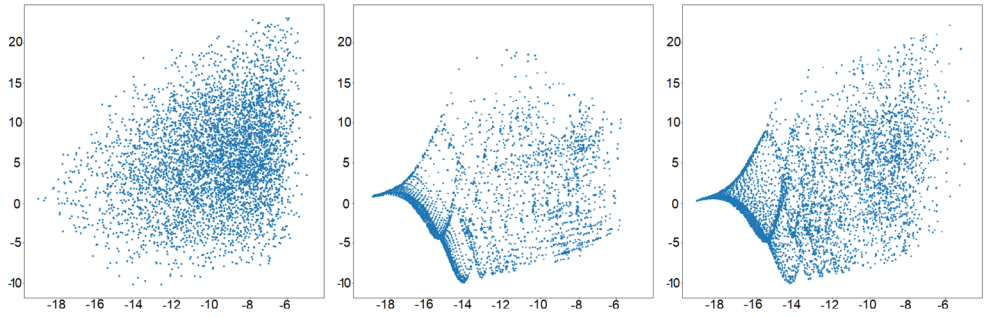

Figure 1. The first 5000 function evaluations in the objective space (Problem Kur1), from left to right: baseline random search, DIRECT, ECAM. A clear concentration of values near the Pareto front in the bottom left corner is evident for both global optimization methods.

Secondly, let us show that the hypervolume is a Lipschitz function provided that the functions f_k are Lipschitz. We need to carefully define what is meant by Lipschitzianity in this context. Recall that the hypervolume is a function of a set of points, and within the context of an optimization algorithm, it is a function of the set of already observed function values. Hence different algorithms, or realisations of the same algorithm, usually result in different values of $Hv(x)$ for the same $x \in X$. Therefore an important question for algorithm development is whether hypervolume behaves as a Lipschitz function of the coordinate vector $x \in X$ for any fixed set of previously recorded function values (for Lipschitz properties of the Pareto front see [51,68]).

So suppose that at step j of an optimization algorithm we have a discrete set of points $D_j = \{x^i, i = 1, \ldots, j\}$ and the corresponting function values $FD_j = \{f(x^i), i = 1, \ldots, j\}$, part of which constitute the current Pareto front with hypervolume H_j. Take any $x \in X$ and calculate the increment to hypervolume of the Pareto front $\Delta H_j = H_j(x) - H_j$. By Lipschitz conditions $|f_l(x) - f_l(x^i)| \leq L||x - x^i||$ for $l = 1, \ldots, k$ and all $x^i \in D_j$, hence the largest increase in the hypervolume is $H_j(x) - H_j \leq (L \max_i ||x - x^i||)^k \leq L^k diam(X)^{k-1} \max_i ||x - x^i||$, where $diam(X)$ is the diameter of the feasible domain. Therefore the hypervolume is a Lipschitz function for compact domains X, but its Lipschitz constant grows as the kth power of L.

As an example, consider the univariate k-objective linear problem on $[0, 2]$ with $f_l(x) = Lx$. Then for $x_0 = 0$ and $x_1 = 2$ the increase to the hypervolume (relative to the origin) is $\Delta H_1 = L^k 2^k$. Hence unfortunately the estimate of the Lipschitz constant (as the kth power of L) cannot be improved.

Therefore from the algorithmic perspective, the use of the hypervolume as the scalar objective can be done in the framework of Lipschitz optimization but with a larger estimate of the Lipschitz constant, derived from the Lipschitz constants of the individual objectives.

Let us now elaborate on the computation of the hypervolume. A recent overview of modern hypervolume calculation methods is given in [31]. The WFG (Walking Fish Group) algorithm [62,63] is currently one of the fastest for computing the hypervolume indicator in many dimensions, particularly for $k > 7$. WFG algorithm has time complexity $O(|P|^{k-1})$, which is polynomial in terms of the cardinality of the Pareto front $|P|$. It is mainly based on the bounding technique and on the inclusion-exclusion principle, and is further optimized by integrating ideas of dimension sweep and objective reordering.

WFG works as follows: given the point set $X \subset R^k$ and a reference point $r \in \mathbb{R}^k$, points in X are sorted and visited in ascending order of dimension k. For each point $p \in \mathbb{R}^k$ visited, the $(k-1)$- dimensional contribution of p to the set of already visited points S is then computed and multiplied by a factor. The contribution to the hypervolume is computed by as the hypervolume of S but bounded by p. The bounding technique in combination with dimension sweep allows many points to become dominated and, thus, it plays an important role in reducing the required computational effort. This places WFG among the fastest algorithms for many dimensions ($k > 4$).

The WFG algorithm is suitable for calculating the improvements to the hypervolume with the addition of new points dynamically, without recalculating the hypervolume of the whole Pareto front. It makes WFG particularly suitable for hypervolume-based SO optimization as recalculation of the objective with the addition of new Pareto points is facilitated by the algorithm.

The incremental WFG algorithm works as follows. Suppose that at step K of the optimization algorithm the hypervolume of the current Pareto front S is $Hv(S)$, and the algorithm selected point x to evaluate $y = f(x)$. The value of the objective (to minimize) is thus calculated as $-Hv(S \cup \{y\})$. Instead of recalculating the hypervolume of the whole new Pareto front, we calculate the increment $\Delta Hv = Hv(S \cup \{y\}) - Hv(S)$, by first identifying the points from S dominating y. If there are no such points (y is not on the Pareto front) the increment is 0. If there are, call that subset SY, the increment will be the hypervolume of SY relative to the 'nadir' point y. Therefore we calculate ΔHv by executing an instance of the WFG algorithm as follows

Algorithm 2 (Calculation of the hypervolume objective): Inputs: Pareto front S, current hypervolume hv, new point y. *Step 1. If y does not dominate any* $z \in S$ *then Return hv. Step 2. Assume* $SY = \{z \in S | y \prec z\}$

2.1 $SZY = \{r = z - y | z \in SY\}$.

2.2 $dh = HypervolumeWFG(SZY)$ *(relative to the origin).*

2.3 *Return* $hv + dh$.

Thus Algorithm 2 repeatedly calls the WFG calculation algorithm for a relatively small subset of points, provided the number of objectives is small to moderate ($k \leq 5$). For larger k the incremental algorithm becomes less efficient as it often happens that the subset SY of dominated points is large (sometimes over half the size of S). Nevertheless Algorithm 2 is no less efficient than recalculating $Hv(S \cup \{y\})$ directly. For two-three objectives the typical size of SY was up to 50 in our experiments.

5. Numerical experiments

The methods based on constructing underestimates of the objective functions are most suitable for MO optimization problems, especially those based on Pareto front learning and hypervolume criterion. The sequences of underestimates converge to the scalar objective uniformly, thus providing accurate approximations to the whole Pareto set. The suggested use of the DIRECT method in [17,45–47] is based on that logic, and hence it is appropriate to compare it to other deterministic global optimization approaches such as ECAM.

The DIRECT method [41,42] also falls into the category of Lipschitz global opimisation, even though it is marketed as not requiring the knowledge of the Lipschitz constant. The DIRECT method partitions the feasible set X into hyper-rectangles, and calculates a lower bound on the value of the objective on each hyper-rectangle from the assumed Lipschitz condition. It then dynamically selects which hyper-rectangles to partition next.

While the lower approximations to the objective function on the hyper-rectangular partition of X are calculated based on the size of the rectangle and the Lipschitz constant (in the same spirit as in the CAM/ECAM), the latter is treated as a parameter, allowing the algorithm to select the subsequent evaluation points x^k based on two rather than one criteria: the size of the rectangle and its ratio to the Lipschitz constant (which is equal to the value of the local minimum of the underestimate on that rectangle). Hence the subsequent evaluation points are chosen from the Pareto sets of an auxiliary bi-objective problem. In fact one can convert any Lipschitz optimization algorithm into a Lipschitz-free method in the same way as in the DIRECT method, by treating the Lipschitz constant as a parameter. In this work we prefer to fix a reasonable overestimate of the Lipschitz constant for the ECAM, while comparing it to both Lipschitz-free DIRECT and its version with a fixed Lipschitz constant.

We used Ganso library [11] for an implementation of the ECAM algorithm. We used $n20^k$ overestimate of the Lipschitz constant for k-objective n-variate problems. The DIRECT method implementation was taken from [42], converted into C language https://github.com/rlnx/DiRect, it does not require any user-specified parameters.

We used the algorithm WFG to compute the hypervolumes from [62,63]. WFG is polynomial $O(|P|^{k-1})$ in terms of the cardinality of the Pareto set, and it shows very good average case performance, which is suitable for multiple evaluations of hypervolumes of related Pareto sets. The program code in C is available from https://github.com/MOEAFramework/Hypervolume/tree/master/WFG.

We selected the following set of MO test problems from a comprehensive review in [37], from where we adopted their abbreviations, the default domains and parameter setting. We excluded univariate problems as in this case ECAM becomes the standard Pijavski-Shubert algorithm, and we also excluded less challenging problems with convex objectives. Where possible we used different dimensionalities n. The computations were performed on a Linux-based workstation with Intel I7-6700K CPU with 32 Gb Ram, clocked at 4GHz. The hypervolumes were calculated with chosen reference points. For a reference we also included the hypervolume found by the baseline method – random search. This way the difference between the two methods is put in the context. All methods were given the budget of 100,000 function evaluations.

For benchmarking we also used several standard evolutionary MO optimization methods, namely (Non-dominated Sorting Genetic Algorithm) NSGA2, NSGA3, reference point based NSGA (RNSGA) and Adaptive Geometry Estimation Multiobjective Evolutionary Algorithm (AGEMOEA) implemented in the package pyMOO [14] with the same budget and different numbers of iteration and population size. The description of these methods is given in [3,21,22,40,50,66].

As the evaluation and efficiency criteria we used the hypervolume (relative to a suitably chosen point for each problem, namely $N = (\max f_1(x), \ldots, \max f_k(x))$ over X, the (Inverted Generational Distance) IGD+ indicator [38] and the CPU wall time. Even though the budget (in terms of function evaluations) was the same for each algorithm,

Table 1. The results of numerical experiments expressed as Hypervolume and IGD+ on a selection of challenging test problems of different dimensions *n* and number of objective *k*.

		ECAM			DIRECT			baseline
Problem	n/k	Hv	IGD+	CPU (sec)	Hv	IGD+	CPU (sec)	Hv
Kur1	2/2	99.53	0.003	38	99.22	0.01	21	99.4
Kur1	3/2	*272.3*	0.11	82	276.3	0.019	45	275.1
Kur1	4/2	498.9	0.32	118	519.1	0.05	48	452
Kur1	5/2	828.1	0.02	311	826.2	0.02	141	666
VU1	2/2	165.8	0.003	91	165.8	0.003	61	165.8
VU2	2/2	110.9	0.004	83	*91.9*	0.91	58	110.8
SK2	4/2	80.1	0.06	219	*67.5*	0.41	129	68.4
TKLY1	4/2	24.75	0	211	24.2	0.001	152	23.6
LTDZ1	3/3	28.25	0	291	*26.99*	0.1	175	26.99
ZDT1	10/2	23.1	0.11	356	*16.1*	1.8	103	20
ZDT1	20/2	24.2	0.017	751	*10.3*	3.1	141	20
ZDT2	10/2	24.0	0.012	361	*13.1*	3.0	101	20
ZDT2	20/2	*21.5*	0.14	713	*4.6*	4.1	155	20
ZDT3	10/2	27.2	0.022	386	*19.2*	1.1	114	20.2
ZDT3	20/2	29.4	0	812	*13.4*	2.1	112	20.2

Note: The hypervolumes (Hv) which are the same or worse than the baseline random search are marked with italics.

the CPU times varied, which is explained by the more sophisticated and expensive inner workings of each algorithm.

First we illustrate the behaviours of different algorithms in the objective functions space. Consider Figure 1. In part (a) we show the images of uniform randomly selected points in the domain D, which correspond to the base case of pure random search. Clearly this method is inefficient, as only a small proportion of the selected points falls into a proximity of the Pareto set. The other parts on this figure show the images of the sequences of points x^k chosen by different algorithms. The faster and closer these sequences converge to the Pareto set, the more efficient is the algorithm. Tables 1 and 2 present the numerical results for comparison. In both tables the hypervolume in italic indicates the cases where the respective algorithm was no better (and sometimes much worse) than the baseline random search (which is obviously the least sophisticated approach with no overheads). First we focus on Table 1 which compares the two hypervolume-based methods.

We see that both methods show comparable results on Kur1 and VU1 problems, and the DIRECT method is faster than ECAM, as we expected. However on the ZDT problems the DIRECT algorithm surprisingly was unable to determine Pareto fronts and was significantly worse than the random search. Comparatively smaller CPU times for these dimensionalities indicate premature convergence to a limited subset of Pareto points.

This may reflect the view in [67] about the relative efficiency of the passive MO optimization algorithms compared to those that use the computed values of the objectives. In contrast, ECAM has shown solid performance in most cases, although it was more expensive and did not find Pareto fronts in a few cases. For higher dimension ECAM is substantially better than the baseline and DIRECT methods, but at the expense of the CPU time. Since the budget of all methods was the same, it is the internal workings of ECAM building more accurate model of the objective which contributes to the computational cost. Consequently it appears that the performance of the DIRECT method applied to MO optimization is not much better than that of a faster random search. ECAM delivers better quality Pareto fronts but is substantially more expensive.

Table 2. The results of numerical experiments on a test problems of different dimensions *n* and number of objective *k* using the Evolutionary Computing algorithms, for benchmarking against Table 1.

		NSGA2			NSGA3			RNSGA			AGEMOEA		
Problem	n/k	Hv	IGD+	CPU	Hv	IGD+	CPU	Hv	IGD+	CPU	Hv	IGD+	CPU
Kur1	2/2	99.54	0.0026	10	99.4	0.004	11	99.3	0.01	21	99.5	0.006	26
Kur1	3/2	277.3	0.0048	9	276.3	0.017	11	*232.5*	1.18	12	276.8	0.016	25
Kur1	4/2	521.9	0.024	10	511.8	0.1	12	520.7	0.047	20	522.06	0.018	25
Kur1	5/2	834.3	0.01	10	*773.4*	0.59	12	*739.8*	1.60	18	833.8	0.01	26
VU1	2/2	165.8	0.003	9	165.4	0.008	12	*160.9*	0.05	23	165.77	0.0058	17
VU2	2/2	110.9	0.0056	10	110.8	0.019	12	*103.9*	0.25	19	110.8	0.008	38
SK2	4/2	82.0	0.018	10	80.7	0.06	12	*53.8*	0.61	19	81.9	0.023	24
TKLY1	4/2	*23.9*	0.023	10	24.66	0.00001	11	*20.48*	0.13	23	*23.88*	0.005	29
LTDZ1	3/3	*26.38*	0.41	10	*26.29*	0.42	13	*18.7*	0.79	27	*26.41*	0.41	46
ZDT1	10/2	24.7	0.0011	9	24.6	0.0013	11	24.59	0.015	14	24.7	0.0024	21
ZDT1	20/2	24.7	0	9	24.6	0.0001	11	24.6	0.0015	20	24.7	0.0001	31
ZDT2	10/2	24.3	0.0014	9	24.3	0.002	10	24.3	0.0015	11	24.3	0.002	27
ZDT2	20/2	24.3	0.0015	9	24.3	0.002	11	24.2	0.0015	24	24.3	0.002	28
ZDT3	10/2	28.1	0.0005	9	28.1	0.001	10	28.3	0.0015	12	28.1	0.001	31
ZDT3	20/2	28.1	0.11	9	28.1	0.11	10	28.1	0.12	12	28.1	0.11	31

Note: The hypervolumes (Hv) which are the same or worse than the baseline random search are marked with italics.

We also compared the hypervolume-based methods against some state-of-the-art MO algorithms, namely NSGA2, NSGA3, RNSGA and AGEMOEA [3,21,22,40,50,66]. The results here are as follows (see Table 2). For low dimensionality all methods show similar results in terms of CPU time and the calculated hypervolume and IGD+ criterion. The CPU time of these methods is smaller than those of ECAM and DIRECT and stayed the same for all dimensionalities, indicating that their cost is bounded by the allocated budget. The performance varied across the board, each of those methods failed to deliver better results than the baseline random search on various occasions, with NSGA2 being the fastest method. For two test problems LTDZ1 and ZDT3 all these methods were significantly worse than the ECAM hypervolume-based algorithm, and even were unable to match it when increasing their budget tenfold. This is consistent with [59] where the hypervolume criterion was used with the budget of up to 5.5×10^6 function evaluations. We refer to [17,20,29,59,69,70] for detailed benchmarking of the EC-based methods.

6. Conclusions

We translated the vector optimization problem into a single objective problem by using hypervolume indicator as the objective, taking advantage of the algorithms for relatively fast calculation of that objective. We have shown that this function is Lipschitz (as long as the objectives of MO problem are Lipschitz), and detailed how to evaluate the hypervolume efficiently. We then applied two global optimization strategies which allow one to track the whole Pareto front of the MO problem based on Lipschitz optimization. The DIRECT and the ECAM methods build a piecewise linear lower approximation to the single objective, which converges to the true objective uniformly. The DIRECT method avoids explicit use of the Lipschitz constant of the objective, whereas ECAM requires its overestimate. While the DIRECT method uses a much more simple model of the objective and hence is much faster in more than 2 variables, its results are inconsistent, and are sometimes worse than the baseline random selection of function evaluation points. ECAM delivers higher hypervolumes due to its more accurate lower approximation model, but at the expense of CPU time. We also compared the numerical results to some state-of-the-art methods based on evolutionary computing and found ECAM to be competitive but computationally more expensive. Our research establishes the usefulness and competitiveness of using the combination of Lipschitz optimization and the hypervolume criterion in MO optimization with limited computational budget. From the rationale provided here and expressed in [3,26,46,47,59,64], as well as computational experience, it appears that for a reasonably small number of objectives the use of deterministic global optimization together with the hypervolume criterion is a viable competitive alternative method for solving multiobjective problems, however the question of more efficient computation of the hypervolume in higher dimensions is crucial for the success of such methods.

Disclosure statement

No potential conflict of interest was reported by the author(s).

Funding

Partial support by the Australian Research Council, Linkage project LP170100458 is gratefully acknowledged.

Notes on contributors

Gleb Beliakov received the Ph.D. degree in physics and mathematics in Moscow, Russia, in 1992. He worked at the Universities of Melbourne and South Australia, and is currently at Deakin University in Melbourne, Burwood, VIC, Australia. He is currently a Professor in the School of Information Technology, Deakin University. His research interests include the areas of aggregation operators, multivariate approximation, global optimization, and numerical computing. He is the author of nearly 200 research papers and two monographs in the mentioned areas, and a number of software packages. He was an Associate Editor of the IEEE Transactions on Fuzzy Systems and Fuzzy Sets and Systems journals.

Longxiang Gao received the Ph.D. degree in computer science from Deakin University, VIC, Australia, in 2014. He was a Postdoctoral Research Fellow with IBM Research and Development, Australia. He is currently a Lecturer with the School of Information Technology, Deakin University. He has over 30 publications, including patents, monographs, book chapters, and journal and conference papers. Some of his publications have been published in the top venues, such as IEEE TMC, IEEE IoT, IEEE TDSC, and IEEE TVT. His research interests include data processing, mobile social networks, fog computing, and network security. Dr. Gao received the 2012 Chinese Government Award for Outstanding Students Abroad (Ranked No.1 in Victoria and Tasmania consular districts). He is active in the IEEE Communication Society. He has served as the TPC Co-Chair, a publicity Co-Chair, an organization chair, and a TPC member for many international conferences.

Yong Xiang received the Ph.D. degree in electrical and electronic engineering from The University of Melbourne, Parkville, VIC, Australia. He is currently a Professor and the Director of the Artificial Intelligence and Data Analytics Research Cluster, School of Information Technology, Deakin University, Burwood, VIC, Australia. His research interests include information security and privacy, multimedia speech/image/video) processing, wireless sensor networks and IoT, and biomedical signal processing. He has published 2 monographs, more than 90 refereed journal articles, and numerous conference papers in these areas. He is an Associate Editor of the IEEE Signal Processing Letters and the IEEE Access. He was the Program Chair, the TPC Chair, the Symposium Chair, and the Session Chair for a number of international conferences.

Wanlei Zhou received the B.E. and M.E. degrees in computer science and engineering from the Harbin Institute of Technology, Harbin, China, in 1982 and 1984, respectively, the Ph.D. degree in computer science and engineering from The Australian National University in 1991, and the D.Sc. degree (a higher doctorate degree) from Deakin University in 2002. He is currently the Vice Rector (Academic Affairs) and the Dean of the Institute of Data Science, City University of Macau, Macau, China. Before joining the City University of Macau, he held various positions including the Head of the School of Computer Science, University of Technology Sydney, Australia, and the Alfred Deakin Professor, the Chair of Information Technology, the Associate Dean, and the Head of the School of Information Technology, Deakin University, Australia. He was also a Lecturer with the University of Electronic Science and Technology of China; a System Programmer with HP, MA, USA; a Lecturer with Monash University, Melbourne, Australia; and with the National University of Singapore. He has authored or coauthored more than 400 papers in refereed international journals and refereed international conferences proceedings, including many articles in IEEE Transactions and journals. His main research interests include security, privacy, and distributed computing.

References

[1] M. Asikuzzaman and M.R. Pickering, *An overview of digital video watermarking*, IEEE Trans. Circuits Syst. Video Technol. 28(9) (2018), pp. 2131–2153.

[2] A. Auger, J. Bader, D. Brockhoff, and E. Zitzler, *Hypervolume-based multiobjective optimization: Theoretical foundations and practical implications*, Theor. Comput. Sci. 425 (2012), pp. 75–103. Theoretical foundations of evolutionary computation.

[3] J. Bader and E. Zitzler, *HypE: An algorithm for fast hypervolume-based many-objective optimization*, Evol. Comput. 19(1) (2011), pp. 45–76.

[4] A. Bagirov and A. Rubinov, *Global minimization of increasing positively homogeneous function over the unit simplex*, Ann. Oper. Res. 98 (2000), pp. 171–187.
[5] L.M. Batten and G. Beliakov, *Fast algorithm for the cutting angle method of global optimization*, J. Glob. Optim. 24 (2002), pp. 149–161.
[6] G. Beliakov, *Geometry and combinatorics of the cutting angle method*, Optimization 52(4–5) (2003), pp. 379–394.
[7] G. Beliakov, *The cutting angle method – a tool for constrained global optimization*, Optim. Methods Softw. 19 (2004), pp. 137–151.
[8] G. Beliakov, *Extended cutting angle method of global optimization*, Pacific J. Optim. 4 (2008), pp. 153–176.
[9] G. Beliakov, H. Bustince, and T. Calvo, *A Practical Guide to Averaging Functions*, Springer, Berlin, Heidelberg, 2016.
[10] G. Beliakov, A. Pradera, and T. Calvo, *Aggregation Functions: A Guide for Practitioners*, Springer, Berlin, Heidelberg, 2007.
[11] G. Beliakov and J. Ugon, *Implementation of novel methods of global and non-smooth optimization: GANSO programming library*, Optimization 56 (2007), pp. 543–546.
[12] N. Beume, C. Fonseca, M. López-Ibánez, L. Paquete, and J. Vahrenhold, *On the complexity of computing the hypervolume indicator*, IEEE Trans. Evol. Comput. 13 (2009), pp. 1075–1082.
[13] N. Beume, B. Naujoks, and M. Emmerich, *Sms-emoa: Multiobjective selection based on dominated hypervolume*, Eur. J. Oper. Res. 181(3) (2007), pp. 1653–1669.
[14] J. Blank and K. Deb, *Multi-objective optimization in python https://pymoo.org/*, IEEE Access 8 (2020), pp. 89497–89509.
[15] K. Bringmann and T. Friedrich, *Approximating the volume of unions and intersections of high-dimensional geometric objects*, in *International Symposium on Algorithms and Computation, LNCS 5369*, S. H. Hong, H. Nagamochi and T. Fukunaga, eds., Springer, Berlin, 2008, pp. 436–447.
[16] D. Brockhoff and E. Zitzler, *Objective reduction in evolutionary multiobjective optimization: Theory and applications*, Evol. Comput. 17(2) (2009), pp. 135–166.
[17] E.F. Campana, M. Diez, and G. Liuzzi, *A multi-objective direct algorithm for ship hull optimization*, Comput. Optim. Appl. 71 (2018), pp. 53–72.
[18] P. Campigotto, A. Passerini, and R. Battiti, *Active learning of pareto fronts*, IEEE Trans. Neural Netw. Learn. Syst. 25(3) (2014), pp. 506–519.
[19] A. Chinchuluun and P.M. Pardalos, *A survey of recent developments in multiobjective optimization*, Ann. Oper. Res. 154 (2007), pp. 29–50.
[20] T. Chugh, K. Sindhya, J. Hakanen, and K. Miettinen, *A survey on handling computationally expensive multiobjective optimization problems with evolutionary algorithms*, Soft Comput. 23 (2019), pp. 3137–3166.
[21] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan, *A fast and elitist multiobjective genetic algorithm: NSGA-II*, IEEE Trans. Evol. Comput. 6(2) (2002), pp. 182–197.
[22] K. Deb and J. Sundar, *Reference point based multi-objective optimization using evolutionary algorithms*, in *GECCO '06: Proceedings of the 8th Annual Conference on Genetic and Evolutionary Computation*, 2006, pp. 635–642.
[23] M. Ehrgott, *Multicriteria Optimization*, Springer, Berlin, 2005.
[24] M. Ehrgott, *A discussion of scalarization techniques for multiple objective integer programming*, Ann. Oper. Res. 147 (2006), pp. 343–360.
[25] M. Ehrgott and D. Tenfelde-Podehl, *Computation of ideal and nadir values and implications for their use in mcdm methods*, Eur. J. Oper. Res. 151(1) (2003), pp. 119–139.
[26] Y.G. Evtushenko and M.A. Posypkin, *A deterministic algorithm for global multi-objective optimization*, Optim. Methods Softw. 29 (2014), pp. 1005–1019.
[27] P.J. Fleming, R.C. Purshouse, and R.J. Lygoe, *Many-objective optimization: An engineering design perspective*, Proceedings Evolutionary Multi-Criterion Optimization, 2005, pp. 14–32.
[28] C.M. Fonseca, L. Paquete, and M. López-Ibánez, *An improved dimension-sweep algorithm for the hypervolume indicator*, in *22nd In Congress on Evolutionary Computation*, G. Syme, D.

Hatton MacDonald, B. Fulton, and J. Piantadosi, eds., IEEE Press, Vancouver, BC Canada, 2006, pp. 1157–1163.
[29] I. Giagkiozis and P. Fleming, *Methods for multi-objective optimization: An analysis*, Inf. Sci. 293 (2015), pp. 338–350.
[30] A. Gimbutas and A. Zilinskas, *Generalization of Lipschitzian global optimization algorithms to the multi-objective case*, in *International Workshop on Optimization and Learning: Challenges and Applications*, Alicante, Spain, 2018.
[31] A.P. Guerreiro, C.M. Fonseca, and L. Paquete, *The hypervolume indicator: Problems and algorithms*, preprint (2020). Available at arXiv:2005.00515.
[32] P. Hansen and B. Jaumard, *Lipschitz optimization*, in *Handbook of Global Optimization*, R. Horst and P. Pardalos, eds., Kluwer, Dordrecht, 1995, pp. 407–493.
[33] M. Hartikainen, K. Miettinen, and M.M. Wiecek, *PAINT: Pareto front interpolation for nonlinear multiobjective optimization*, Comput. Optim. Appl. 52 (2012), pp. 845–867.
[34] R. Horst and P.M. Pardalos, *Handbook of Global Optimization*, Kluwer, Dordrecht, Boston, 1995.
[35] R. Horst, P. Pardalos, and N.V. Thoai (eds.), *Introduction to Global Optimization*, 2nd ed., Kluwer, Dordrecht, 2000.
[36] G. Hua, L. Zhao, H. Zhang, G. Bi, and Y. Xiang, *Random matching pursuit for image watermarking*, IEEE Trans. Circuits Syst. Video Technol. 29(3) (2019), pp. 625–639.
[37] S. Huband, P. Hingston, L. Barone, and L. While, *A review of multiobjective test problems and a scalable test problem toolkit*, IEEE Trans. Evol. Comput. 10(5) (2006), pp. 477–506.
[38] H. Ishibuchi, H. Masuda, Y. Tanigaki, and Y. Nojima, *Modified distance calculation in generational distance and inverted generational distance*, in *Evolutionary Multi-Criterion Optimization*, A. Gaspar-Cunha, C. Henggeler Antunes, and C. Coello Coello, eds., Springer, Cham, 2015, pp. 110–125.
[39] J. Jahn, *Vector Optimization: Theory, Applications, and Extensions*, Springer-Verlag, Berlin, 2004.
[40] H. Jain and K. Deb, *An evolutionary many-objective optimization algorithm using reference-point based nondominated sorting approach*, IEEE Trans. Evol. Comput. 18 (2014), pp. 602–622.
[41] D.R. Jones and J.R.R.A. Martins, *The DIRECT algorithm: 25 years later*, J. Glob. Optim. 79 (2021), pp. 521–566.
[42] D.R. Jones, C.D. Perttunen, and B.E. Stuckman, *Lipschitzian optimization without the Lipschitz constant*, J. Optim. Theory Appl. 79 (1993), pp. 157–181.
[43] E. Karasakal and M. Koksalan, *Generating a representative subset of the nondominated frontier in multiple criteria decision making*, Oper. Res. 57(1) (2009), pp. 187–199.
[44] A. Lovison, *Singular continuation: Generating piecewise linear approximations to Pareto sets via global analysis*, SIAM J. Optim. 21(2) (2011), pp. 463–490.
[45] A. Lovison and M.E. Hartikainen, *On generalizing Lipschitz global methods for multiobjective optimization*, in *EMO 2015, Part II, LNCS 9019*, A. Gaspar-Cunha et al., ed., 2015, pp. 264–278.
[46] A. Lovison and K. Miettinen, *Exact extension of the DIRECT algorithm to multiple objectives*, in *AIP Conference Proceedings*, Vol. 2070, 2019.
[47] A. Lovison and K. Miettinen, *On the extension of the DIRECT algorithm to multiple objectives*, J. Glob. Optim. 79 (2021), pp. 387–412.
[48] V.K. Mehta and B. Dasgupta, *Parametric approximation of the Pareto set in multi-objective optimization problems*, J. Multi-Criteria Decis. Anal. 21 (2014), pp. 335–362.
[49] K. Miettinen, *Nonlinear Multiobjective Optimization*, Kluwer, Boston, 1999.
[50] A. Panichella, *An adaptive evolutionary algorithm based on non-euclidean geometry for many-objective optimization*, in *GECCO '19: Proceedings of the Genetic and Evolutionary Computation Conference*, 2019, pp. 595–603.
[51] P.M. Pardalos, A. Zilinskas, and J. Zilinskas, *Non-Convex Multi-Objective Optimization*, Springer, Cham, 2017.
[52] S.A. Pijavski, *An algorithm for finding the absolute extremum of a function*, USSR Comput. Math. Math. Phys. 2 (1972), pp. 57–67.

[53] J. Pinter, *Global Optimization in Action: Continuous and Lipschitz Optimization–Algorithms, Implementations, and Applications*, Kluwer, Dordrecht, Boston, 1996.
[54] B. Qu, Y. Zhu, Y. Jiao, M. Wu, P. Suganthan, and J. Liang, *A survey on multi-objective evolutionary algorithms for the solution of the environmental/economic dispatch problems*, Swarm Evol. Comput.38(1) (2018), pp. 1–11.
[55] A.M. Rubinov, *Abstract Convexity and Global Optimization*, Kluwer, Dordrecht, Boston, 2000.
[56] D.K. Saxena, J.A. Duro, A. Tiwari, K. Deb, and Q. Zhang, *Objective reduction in many-objective optimization: Linear and nonlinear algorithms*, IEEE Trans. Evol. Comput. 17(1) (2013), pp. 77–99.
[57] B. Shubert, *A sequential method seeking the global maximum of a function*, SIAM J. Numer. Anal.9 (1972), pp. 379–388.
[58] R.G. Strongin and Y.D. Sergeyev, *Global optimization with non-convex constraints: Sequential and parallel algorithms*, Nonconvex Optimization and Its Applications Vol. 45, Kluwer Academic Publishers, Dordrecht, London, 2000.
[59] Y. Sun, G.G. Yen, and Z. Yi, *IGD indicator-based evolutionary algorithm for many-objective optimization problems*, IEEE Trans. Evol. Comput. 23(2) (2019), pp. 173–187.
[60] T. Wagner, N. Beume, and B. Naujoks, *Pareto-, aggregation-, and indicator-based methods in many-objective optimization*, in *Evolutionary Multi-Criterion Optimization*, Springer, Berlin/Heidelberg, 2007, pp. 742–756.
[61] R. Wang, Z. Zhou, H. Ishibuchi, T. Liao, and T. Zhang, *Localized weighted sum method for many-objective optimization*, IEEE Trans. Evol. Comput. 22 (2018), pp. 3–18.
[62] L. While, L. Bradstreet, and L. Barone, *A fast way of calculating exact hypervolumes*, IEEE Trans. Evol. Comput. 16 (2012), pp. 86–95.
[63] L. While, P. Hingston, L. Barone, and S. Huband, *A faster algorithm for calculating hypervolume*, IEEE Trans. Evol. Comput. 10(1) (2006), pp. 29–38.
[64] C.S.Y. Wong, A. Al-Dujaili, and S. Suresh, *Hypervolume-based DIRECT for multi-objective optimisation*, in *Proceedings of the 2016 ACM on Genetic and Evolutionary Computation Conference Companion*, 2016, pp. 1201–1208.
[65] Y. Xiang, I. Natgunanathan, D. Peng, G. Hua, and B. Liu, *Spread spectrum audio watermarking using multiple orthogonal pn sequences and variable embedding strengths and polarities*, IEEE/ACM Trans. Audio Speech Lang. Process. 26(3) (2018), pp. 529–539.
[66] Q. Zhang and H. Li, *Moea/d: A multiobjective evolutionary algorithm based on decomposition*, IEEE Trans. Evol. Comput. 11 (2007), pp. 712–731.
[67] A. Zilinskas, *On the worst-case optimal multi-objective global optimization*, Optim. Lett. 7 (2013), pp. 1921–1928.
[68] A. Zilinskas and J. Zilinskas, *Adaptation of a one-step worst-case optimal univariate algorithm of bi-objective Lipschitz optimization to multidimensional problems*, Commun. Nonlinear Sci. Numer. Simul. 21 (2015), pp. 89–98.
[69] E. Zitzler, K. Deb, and L. Thiele, *Comparison of multiobjective evolutionary algorithms: Empirical results*, Evol. Comput. 8(2) (2000), pp. 173–195.
[70] E. Zitzler, L. Thiele, M. Laumanns, C.M. Fonseca, and V.G. da Fonseca, *Performance assessment of multiobjective optimizers: An analysis and review*, IEEE Trans. Evol. Comput. 7(2) (2003), pp. 117–132.