Efficient computation of expected hypervolume improvement using box decomposition algorithms

Kaifeng Yang[1] · Michael Emmerich[1] · André Deutz[1] · Thomas Bäck[1]

Abstract

In the field of multi-objective optimization algorithms, multi-objective Bayesian Global Optimization (MOBGO) is an important branch, in addition to evolutionary multi-objective optimization algorithms. MOBGO utilizes Gaussian Process models learned from previous objective function evaluations to decide the next evaluation site by maximizing or minimizing an infill criterion. A commonly used criterion in MOBGO is the Expected Hypervolume Improvement (EHVI), which shows a good performance on a wide range of problems, with respect to exploration and exploitation. However, so far, it has been a challenge to calculate exact EHVI values efficiently. This paper proposes an efficient algorithm for the exact calculation of the EHVI for in a generic case. This efficient algorithm is based on partitioning the integration volume into a set of axis-parallel slices. Theoretically, the upper bound time complexities can be improved from previously $O(n^2)$ and $O(n^3)$, for two- and three-objective problems respectively, to $\Theta(n \log n)$, which is asymptotically optimal. This article generalizes the scheme in higher dimensional cases by utilizing a new hyperbox decomposition technique, which is proposed by Dächert et al. (Eur J Oper Res 260(3):841–855, 2017). It also utilizes a generalization of the multilayered integration scheme that scales linearly in the number of hyperboxes of the decomposition. The speed comparison shows that the proposed algorithm in this paper significantly reduces computation time. Finally, this decomposition technique is applied in the calculation of the Probability of Improvement (PoI).

Keywords Expected hypervolume improvement · Probability of improvement · Time complexity · Multi-objective Bayesian global optimization · Hypervolume indicator · Kriging

1 Introduction

In multi-objective design optimization, the objective function evaluations are generally computationally costly, mainly due to the long convergence times of simulation models. A simple and common remedy to this problem is to use a statistical model learned from previous evaluations as the fitness function, instead of the 'true' objective function. This method is also known as Bayesian Global Optimization (BGO) [30]. In BGO, a Gaussian Process (GP) model is used as a statistical model. In each iteration, the algorithm evaluates a new solution and updates the Gaussian Process model. A new solution is chosen by the score of an infill criterion, given a statistical model. For multi-objective problems, the family of these algorithms is called *Multi-objective Bayesian global optimization* (MOBGO). Compared to evolutionary multi-objective optimization algorithms (EMOAs), MOBGO requires only a small budget of function evaluations to achieve a similar result with respect to hypervolume indicator, and it has already been used in real-world applications to solve expensive evaluation problems [40]. According to the authors' knowledge, BGO was used for the first time in the context of airfoil optimization in [27], and then applied in the field of biogas plant controllers [16], detection in water quality management [41], machine learning algorithm configuration [23], and structural design optimization [33].

In the context of Bayesian global optimization, an infill or pre-selection criterion is used to evaluate how promising a new point is. In single-objective optimization, the *Expected Improvement* (EI) is widely used as the infill criterion, and it was first introduced by Mockus et al. [30] in 1978. The EI exploits both the Kriging prediction and the variance in order to give a quantitative measure of a solution's improvement. Later, the EI became more popular due to the work of Jones et al [21]. In MOBGO, a commonly used criterion is *Expected Hypervolume Improvement* (EHVI), which is a straightforward generalization of the EI and was proposed by Emmerich [10] in 2005. Compared with other criteria, EHVI leads to an excellent convergence to—and coverage of—the true Pareto front [5,37]. Nevertheless, the calculation of EHVI itself so far has been time-consuming [23,32,36,41], even in the 2-D case[1]. Moreover, EHVI has to be computed many times by an optimizer to search for a promising solution in every iteration. For these reasons, a fast algorithm for computing EHVI is needed.

The first method suggested for EHVI calculation was Monte Carlo integration and was proposed by Emmerich [10,13]. This method is simple and straightforward. However, the accuracy of EHVI highly depends on the number of iterations. The first exact EHVI calculation algorithm in the 2-D case was derived in [12], with time complexity of $O(n^3 \log n)$. Here, n is the number of non-dominated points in the archive. The EHVI calculation algorithm in [12] partitions an objective space into boxes and then calculates the EHVI by summing all the EHVI values of each box. Couckuyt et al. [5] introduced an exact EHVI calculation algorithm (CDD13) for $d > 2$ by representing a non-dominated space with three types of boxes, where d represents the number of objective functions. The method in [5] was also practically much faster than those discussed in [12], though a detailed complexity analysis was missing. Hupkens et al. [19] reduced the time complexity to $O(n^2)$ and $O(n^3)$ in the 2-D and 3-D cases, respectively. The algorithms in [19] improve the algorithms in [12] by two ways: (1) only summing the EHVI values of each box in a non-dominated space; (2) reusing some intermediate integrations during the EHVI calculation. The algorithms in [19] further improve the practical efficiency of EHVI on test data in comparison to [5]. Recently, Emmerich et al. [14] proposed an asymptotically optimal algorithm with time complexity of

---

[1] In this paper, 2-D and 3-D represent two and three objective functions, respectively.

$\Theta(n \log n)$ in the bi-objective case. More recently, Yang et al. proposed an asymptotically optimal algorithm with time complexity $\Theta(n \log n)$ in the 3-D case [38]. The algorithm, KMAC[2] in [38], partitions a non-dominated space by slices linearly and re-derives the EHVI calculation formulas. However, a generalization of this technique to more than three dimensions/objectives and the empirical testing of MOBGO algorithms on benchmark optimization problems, are still missing so far.

This paper mainly contributes to extending the state-of-the-art EHVI calculation methods into higher dimensional cases. The paper is structured as follows: Sect. 2 introduces the nomenclature, Kriging, and the framework of MOBGO; Sect. 3 provides some fundamental definitions used in this paper; Sect. 4 describes how to partition an integration space into (hyper)boxes efficiently, and how to calculate EHVI based on this partitioning method; Sect. 5 shows experimental results of speed comparison and MOBGO based algorithms' performance on 10 well-known scientific benchmarks in 6- and 18-dimensional search spaces; Sect. 6 draws the main conclusions and discusses some potential topics for further research.

2 Multi-objective Bayesian global optimization

A multi-objective optimization (MOO) problem is an optimization problem that involves multiple objective functions. A MOO problem can be formulated as:

$$\text{``max''}\big(y_1(\mathbf{x}), y_2(\mathbf{x}), \ldots, y_d(\mathbf{x})\big) \qquad \mathbf{x} \in \mathbb{R}^m \tag{1}$$

where d is the number of objective functions, $y_i (i = 1, \ldots, d)$ are the objective functions, and a decision vector $\mathbf{x}$ is in an m-dimensional space (Table 1).

2.1 Notations

The following table summarizes the notations used in this paper.

2.2 Kriging

As a statistical interpolation method, Kriging is a Gaussian process based multivariate regression method. Compared with simulator-based evaluations in design optimization, one prediction/evaluation of the Kriging model is typically cheap [28]. Therefore, Kriging is widely used as a popular surrogate model to approximate noise-free data in computer experiments. Kriging models are fitted from previously evaluated points. Given a set of n decision vectors $\mathbf{X} = \{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(n)}\}, \mathbf{x}^{(i)} \in \mathbb{R}^m |_{i,=1,\ldots,n}$ in an m-dimensional search space, and associated function values $\mathbf{Y}(\mathbf{X}) = \big(y(\mathbf{x}^{(1)}), y(\mathbf{x}^{(2)}), \ldots, y(\mathbf{x}^{(n)})\big)^\top$, Kriging assumes $\mathbf{Y}(\mathbf{X})$ to be a realization of a random process Y of the following form [3,21]:

$$Y(\mathbf{x}) = \mu(\mathbf{x}) + \epsilon(\mathbf{x}) \tag{2}$$

where $\mu(\mathbf{x})$ is the estimated mean value over all given sampled points, and $\epsilon(\mathbf{x})$ is a realization of a Gaussian process with zero mean and variance σ^2. The regression part $\mu(\mathbf{x})$ approximates

---

[2] KMAC stands for the authors' given names of the EHVI exact calculation algorithm.

Table 1 Notations

Symbol	Type	Description
m	$\mathbb{N}^+$	Dimension of a search space
d	$\mathbb{N}^+$	Dimension of an objective space
$\boldsymbol{\mu}$	$\mathbb{R}^d$	Mean values of predictive distribution
$\boldsymbol{\sigma}$	$(\mathbb{R}_0^+)^d$	Standard deviations of predictive distribution
$\mathcal{P}$	$(\mathbb{R}^d)^n$	A Pareto-front approximation set
$n, |\mathcal{P}|$	$\mathbb{N}^+$	Number of the non-dominated points in $\mathcal{P}$
$\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}$	$\mathbb{R}^d$	The vectors in $\mathcal{P}$, where $\mathcal{P} = (\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)})$
$\mathbf{r}$	$\mathbb{R}^d$	Reference point
S_d	$(\mathbb{R}^d)^2$	Integration slices in a d-dimensional space
N_i	$\mathbb{N}^+$	Number of integration slice in i-dimensional case
$\mathbf{l}_d^{(1)}, \ldots, \mathbf{l}_d^{(N_d)}$	$\mathbb{R}^d$	Lower bounds of integration slices
$\mathbf{u}_d^{(1)}, \ldots, \mathbf{u}_d^{(N_d)}$	$\mathbb{R}^d$	Upper bounds of integration slices
$\mathbf{x}^t$	$\mathbb{R}^d$	A target solution in a search space
$\mathbf{x}^*$	$\mathbb{R}^d$	A promising solution in a search space
M_i	$i = 1, \ldots, d$	The Kriging model for the i-th objective function
g	$\mathbb{N}^+$	Counter/Number of function evaluations
T_c	$\mathbb{N}^+$	Termination criterion
η	$\mathbb{N}^+$	The number of initialized solutions
D	$(\mathbb{R}^{(d+m)})^g$	Training dataset for the Kriging models M
λ_i	$i = 1, \ldots, d$	The Lebesgue measure on $\mathbb{R}^i$
$\boldsymbol{\xi}$		The multivariate independent normal distribution
S_i	$i = 1, \ldots, d$	The integration slices in an i-dimensional case
y		Objective functions

the function Y globally and the Gaussian process $\epsilon(\mathbf{x})$ takes local variations into account. Opposed to other regression methods (such as support vector machine), Kriging/GP also provides an uncertainty qualification of a prediction. The correlation between the deviations at two decision vectors ($\mathbf{x}$ and $\mathbf{x}'$) is defined as:

$$Corr[\epsilon(\mathbf{x}), \epsilon(\mathbf{x}')] = R(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{m} R_i(x_i, x_i') \tag{3}$$

Here $R(.,.)$ is the correlation function, which decreases with the distance between two points. It is common practice to use a Gaussian correlation function (also known as a squared exponential kernel):

$$R(\mathbf{x}, \mathbf{x}') = \prod_{i=1}^{m} \exp(-\theta_i (x_i - x_i')^2) \qquad (\theta_i \geq 0) \tag{4}$$

where θ_i are parameters of the correlation model. They can be interpreted as a measurement of the variables' importance. The optimal $\boldsymbol{\theta} = (\theta_1^{opt}, \ldots, \theta_m^{opt})$ in the Kriging models are usually optimized by a continuous optimization algorithm. In this paper, the optimal $\boldsymbol{\theta}$ is

optimized by the simplex search method of Lagarias et al. ($fminsearch$) [26], with the parameter of max function evaluations equal to 1000. The covariance matrix can then be expressed through the correlation function:

$$Cov(\epsilon(\mathbf{x})) = \sigma^2 \mathbf{\Sigma}, \quad \text{where} \quad \mathbf{\Sigma}_{i,j} = R(\mathbf{x_i}, \mathbf{x_j}) \tag{5}$$

When $\mu(\mathbf{x})$ is assumed to be an unknown constant, the unbiased prediction is called ordinary Kriging (OK). In OK, the Kriging model determines the hyperparameters $\boldsymbol{\theta} = [\theta_1, \theta_2, \ldots, \theta_n]$ by maximizing the likelihood over the observed dataset. The expression of the likelihood function is:

$$L = -\frac{n}{2}\ln(\sigma^2) - \frac{1}{2}\ln(|\mathbf{\Sigma}|) \tag{6}$$

The maximum likelihood estimates of the mean $\hat{\mu}$ and the variance $\hat{\sigma}^2$ are given by:

$$\hat{\mu} = \frac{\mathbf{1}_n^\top \mathbf{\Sigma}^{-1} \mathbf{y}}{\mathbf{1}_n^\top \mathbf{\Sigma}^{-1} \mathbf{1}_n} \tag{7}$$

$$\hat{\sigma}^2 = \frac{1}{n}(\mathbf{y} - \mathbf{1}_n\hat{\mu})^T \mathbf{\Sigma}^{-1}(\mathbf{y} - \mathbf{1}_n\hat{\mu}) \tag{8}$$

Then the predictor of the mean and the variance at a target point $\mathbf{x}^t$ can be derived. They are shown in [21]:

$$\mu(\mathbf{x}^t) = \hat{\mu} + \mathbf{c}^\top \mathbf{\Sigma}^{-1}(\mathbf{y} - \hat{\mu}\mathbf{1}_n) \tag{9}$$

$$\sigma^2(\mathbf{x}^t) = \hat{\sigma}^2 \left[1 - \mathbf{c}^\top \mathbf{\Sigma}^{-1}\mathbf{c} + \frac{1 - \mathbf{c}^T \Sigma^\top \mathbf{c}}{\mathbf{1}_n^\top \mathbf{\Sigma}^{-1} \mathbf{1}_n} \right] \tag{10}$$

where $\mathbf{c} = (Corr[y(x^t), y(x_1)], \ldots, Corr[y(x^t), y(x_n)])^\top$.

2.3 Structure of MOBGO

In MOBGO, it is assumed that d objective functions are mutually independent in an objective space. Each objective function is approximated by a Kriging model individually, based on the η existing evaluated data $D = ((\mathbf{x}^{(1)}, \mathbf{y}^{(1)} = y(\mathbf{x}^{(1)})), \ldots, (\mathbf{x}^{(\eta)}, \mathbf{y}^{(\eta)} = y(\mathbf{x}^{(\eta)})))$. Each Kriging model is a one-dimensional normal distribution, with a mean μ and a standard deviation σ. Given a target solution $\mathbf{x}^t$, the Kriging models can predict the multivariate outputs by means of an independent joint normal distribution with means $\mu_1, \ldots, \mu_d$ and standard deviations $\sigma_1, \ldots, \sigma_d$. These predictive means and standard deviations are used to calculate the score of an infill criterion, which can quantitatively measure how promising the target point $\mathbf{x}^t$ is when compared with the current Pareto-front approximation set. A promising solution $\mathbf{x}^*$ can be found by maximizing/minimizing[3] the score of the infill criterion. Then, this promising solution $\mathbf{x}^*$ is evaluated by the 'true' objective functions, and both the dataset D and the Pareto-front approximation set $\mathcal{P}$ are updated.

The basic structure of the MOBGO algorithm is shown in Algorithm 1. It mainly contains three parts: initialization of a sampling dataset, searching for an optimal solution and updating the Kriging models, and returning the Pareto-front approximation set $\mathcal{P}$.

---

[3] It depends on which criterion is chosen.

Algorithm 1: MOBGO algorithm

Input: Objective functions y, initialization size η, termination criterion T_c
Output: Pareto-front approximation $\mathcal{P}$
 1: Initialize η solutions $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(\eta)}\}$ using LHS algorithm;
 2: Evaluate the initial set of η points: $(\mathbf{y}^{(1)} = y(\mathbf{x}^{(1)}), \ldots, \mathbf{y}^{(\eta)} = y(\mathbf{x}^{(\eta)}))$;
 3: Store $\{\mathbf{x}^{(1)}, \ldots, \mathbf{x}^{(\eta)}\}$ and $(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(\eta)})$ in D: $D = ((\mathbf{x}^{(1)}, \mathbf{y}^{(1)}), \ldots, (\mathbf{x}^{(\eta)}, \mathbf{y}^{(\eta)}))$;
 4: Compute the non-dominated subset of D and store it in $\mathcal{P}$;
 5: $g = \eta$;
 6: **while** $g <= T_c$ **do**
 7:     Train surrogate models M based on D;
 8:     Use an optimizer to find the promising point $\mathbf{x}^*$ based on surrogate models M, with the infill
         criterion C;
 9:     Update D: $D = D \cup ((\mathbf{x}^*, y(\mathbf{x}^*)))$;
10:      Update $\mathcal{P}$ as a non-dominated subset of D;
11:      $g = g + 1$;
12: **end while**
13: Return $\mathcal{P}$.

First, a dataset D is initialized and a Pareto-front approximation set $\mathcal{P}$ is computed, as shown in Algorithm 1 from Step 1 to Step 5. The initialization of D contains the generation of the decision vectors using Latin Hypercube Sampling method (LHS) [29] (Step 1), calculation of the corresponding objective values (Step 2) and storage of this information in dataset D (Step 3). This dataset D will be utilized to build the Kriging models in the second part.

The second part of MOBGO is the main loop, as shown in Algorithm 1 from Step 6 to Step 12. This main loop starts with training Kriging models M_i based on dataset D (Step 7). Note that M contains d independent models for each objective function, and these models will be used as temporary objective functions instead of 'true' objective functions in Step 8. Then, an optimizer finds a promising solution $\mathbf{x}^*$ by maximizing or minimizing an infill criterion C (Step 8). Here, an infill criterion is calculated by its corresponding calculation formula, whose inputs include Kriging models M, the current Pareto-front approximation set $\mathcal{P}$, a target decision vector $\mathbf{x}^t$, etc. Theoretically, any single-objective optimization algorithm can be utilized as an optimizer to search for a promising solution $\mathbf{x}^*$. In this paper, the BI-population CMA-ES is chosen for its favorable performance on BBOB function testbed [18]. Step 9 and Step 10 will update the dataset D by adding $(\mathbf{x}^*, y(\mathbf{x}^*))$ into D and update the Pareto-front approximation set $\mathcal{P}$. The main loop from Step 6 to Step 12 will continue until g meets the termination criterion T_c. The last part of MOBGO returns Pareto-front approximation set $\mathcal{P}$.

The choice of infill criterion C at Step 8 distinguishes different types of MOBGO based algorithms. In this paper, EHVI-MOBGO and PoI-MOBGO, which set EHVI and PoI [22, 24,35] as the infill criterion C respectively, are compared in Sect. 5.2.

3 Definitions

Pareto dominance, or briefly *dominance*, is a fundamental concept in MOO and provides an ordering relation on the set of potential solutions.[4] *Dominance* is defined as follows:

---

[4] For the convenience of visualization and consistency, this paper only considers maximization problems. Minimization problems can always be re-written as maximization problems by multiplying the corresponding objective functions by (-1).

Definition 1 (*Dominance* [4]) Given two decision vectors $\mathbf{x}^{(1)}, \mathbf{x}^{(2)} \in \mathbb{R}^m$ and their corresponding objective values $\mathbf{y}^{(1)} = y(\mathbf{x}^{(1)}), \mathbf{y}^{(2)} = y(\mathbf{x}^{(2)})$ in a maximization problem, it is said that $\mathbf{y}^{(1)}$ dominates $\mathbf{y}^{(2)}$, being represented by $\mathbf{y}^{(1)} \prec \mathbf{y}^{(2)}$, iff $\forall i \in \{1, 2, \ldots, d\} : y_i(\mathbf{x}^{(1)}) \geq y_i(\mathbf{x}^{(2)})$ and $\exists j \in \{1, 2, \ldots, d\} : y_j(\mathbf{x}^{(1)}) > y_j(\mathbf{x}^{(2)})$.

From the perspectives of searching and optimization, non-dominated points are of greater interest. The concept of *non-dominance* is defined as:

Definition 2 (*Non-dominance* [14]) Given a decision vector set $\mathbf{X} \in \mathbb{R}^m$, and the image of the vector set $\mathbf{Y} = \{y(\mathbf{x}) | \mathbf{x} \in \mathbf{X}\}$, the non-dominated subset of $\mathbf{Y}$ is defined as:

$$nd(\mathbf{Y}) := \{\mathbf{y} \in \mathbf{Y} | \nexists \mathbf{z} \in \mathbf{Y} : \mathbf{z} \prec \mathbf{y}\} \tag{11}$$

A vector $\mathbf{y} \in nd(\mathbf{Y})$ is called a non-dominated point of $\mathbf{Y}$.

Definition 3 (*Dominated subspace of a set*) Let $\mathcal{P}$ be a subset of $\mathbb{R}^d$. The dominated subspace of $\mathcal{P}$ in $\mathbb{R}^d$, notation dom ($\mathcal{P}$), is then defined as:

$$\mathrm{dom}(\mathcal{P}) := \{\mathbf{y} \in \mathbb{R}^d \mid \exists \mathbf{p} \in \mathcal{P} \text{ with } \mathbf{p} \prec \mathbf{y}\} \tag{12}$$

Definition 4 (*Non-dominated space of a set*) Let $\mathcal{P}$ be a subset of $\mathbb{R}^d$ and let $\mathbf{r} \in \mathbb{R}^d$ be such that $\forall \mathbf{p} \in \mathcal{P} : \mathbf{p} \prec \mathbf{r}$. The non-dominated space of $\mathcal{P}$ with respect to $\mathbf{r}$, denoted as $\mathrm{ndom}(\mathcal{P})$, is then defined as:

$$\mathrm{ndom}(\mathcal{P}) := \{\mathbf{y} \in \mathbb{R}^d \mid \mathbf{y} \prec \mathbf{r} \text{ and } \nexists \mathbf{p} \in \mathcal{P} \text{ such that } \mathbf{p} \prec \mathbf{y}\} \tag{13}$$

Note that the notion of dominated space as well as the notion of non-dominated space of a set can also be defined for (countably and non-countably) infinite sets $\mathcal{P}$.

The *Hypervolume Indicator* (HV), introduced in [42], is one of the essential unary indicators for evaluating the quality of a Pareto-front approximation set. Its theoretical properties are discussed in [43]. Notably, HV does not require the knowledge of the Pareto front in advance. The maximization of HV leads to a high-qualified and diverse Pareto-front approximation set. The *Hypervolume Indicator* is defined as follows:

Definition 5 (*Hypervolume indicator*) Given a finite approximation to a Pareto front, say $\mathcal{P} = \{\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}\} \subset \mathbb{R}^d$, the *Hypervolume Indicator* of $\mathcal{P}$ is defined as the d-dimensional Lebesgue measure of the subspace dominated by $\mathcal{P}$ and bounded below by a reference point $\mathbf{r}$:

$$\mathrm{HV}(\mathcal{P}) = \lambda_d(\cup_{\mathbf{y} \in \mathcal{P}}[\mathbf{r}, \mathbf{y}]) \tag{14}$$

with λ_d being the Lebesgue measure on $\mathbb{R}^d$.

The hypervolume indicator measures the size of the dominated subspace bounded below by a reference point $\mathbf{r}$. This reference point needs to be provided by users. Theoretically, in order to get the extreme non-dominated points, this reference point should be chosen in a way that it is dominated by all elements of a Pareto-front approximation set $\mathcal{P}$ during the optimization process. However, there is no requirement of setting the reference point in practice if the user is not interested in extreme non-dominated points.

Another important infill criterion is *Hypervolume Improvement*, which is also called the *Improvement of Hypervolume* in [11]. The definition of *Hypervolume Improvement* is:
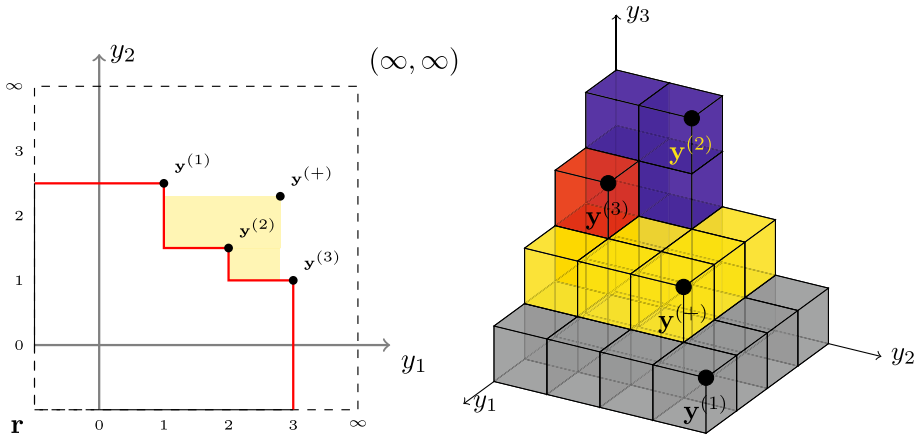
Fig. 1 The left and right figures illustrate *Hypervolume Improvement* in a 2-D and a 3-D example, respectively

Definition 6 (*Hypervolume improvement*) Given a finite collection of vectors $\mathcal{P} \subset \mathbb{R}^d$, the *Hypervolume Improvement* of a vector $\mathbf{y} \in \mathbb{R}^d$ is defined as:

$$\mathrm{HVI}(\mathbf{y}, \mathcal{P}) = \mathrm{HV}(\mathcal{P} \cup \{\mathbf{y}\}) - \mathrm{HV}(\mathcal{P}) \tag{15}$$

When we want to emphasize the reference point $\mathbf{r}$, the notation $\mathrm{HVI}(\mathbf{y}, \mathcal{P}, \mathbf{r})$ will be used to denote *Hypervolume Improvement*.

Example 1 Figure 1 illustrates the concept of *Hypervolume Improvement* using two examples. The first example, on the left, is a 2-D example: Suppose a Pareto-front approximation set is $\mathcal{P}$, which is composed by $\mathbf{y}^{(1)} = (1, 2.5)^\top$, $\mathbf{y}^{(2)} = (2, 1.5)^\top$ and $\mathbf{y}^{(3)} = (3, 1)^\top$. When a new point $\mathbf{y}^{(+)} = (2.8, 2.3)^\top$ is added, the *Hypervolume Improvement* $HVI(\mathbf{y}^{(+)}, \mathcal{P}, \mathbf{r})$ is the area of the yellow polygon. The second example (on the right in Fig. 1) illustrates the *Hypervolume Improvement* by means of a 3-D example. Assume a Pareto-front approximation set is $\mathcal{P} = (\mathbf{y}^{(1)} = (4, 4, 1)^\top, \mathbf{y}^{(2)} = (1, 2, 4)^\top, \mathbf{y}^{(3)} = (2, 1, 3)^\top)$. The *Hypervolume Improvement* of $\mathbf{y}^{(+)} = (3, 3, 2)^\top$ relative to $\mathcal{P}$ is given by the joint volume covered by the yellow slices.

Probability of Improvement (PoI) is an important criterion in MOBGO. It was first introduced by Stuckman in [34]. Later, Emmerich et al. [13] generalized it to multi-objective optimization. PoI is defined as:

Definition 7 (*Probability of improvement*) Given parameters of the multivariate predictive distribution $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and the Pareto-front approximation set $\mathcal{P}$, the *Probability of Improvement* is defined as:

$$\mathrm{PoI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}) := \int_{\mathbb{R}^d} \mathrm{I}(\mathbf{y} \text{ impr } \mathcal{P}) \xi_{\boldsymbol{\sigma}, \boldsymbol{\mu}}(\mathbf{y}) d\mathbf{y} \qquad \mathrm{I}(v) = \begin{cases} 1 & \text{if } v = \text{true} \\ 0 & \text{if } v = \text{false} \end{cases} \tag{16}$$

where $\xi_{\boldsymbol{\mu}, \boldsymbol{\sigma}}$ is a multivariate independent normal distribution with the mean values $\boldsymbol{\mu} \in \mathbb{R}^d$ and the standard deviations $\boldsymbol{\sigma} \in \mathbb{R}_+^d$. Here, $(\mathbf{y} \text{ impr } \mathcal{P})$ represents $\mathbf{y} \in \mathbb{R}^d$ as an improvement with respect to $\mathcal{P}$, if and only if the following holds: $\mathbf{y} \prec \mathbf{r}$ and $\forall \mathbf{p} \in \mathcal{P} : \neg(\mathbf{p} \prec \mathbf{y})$.
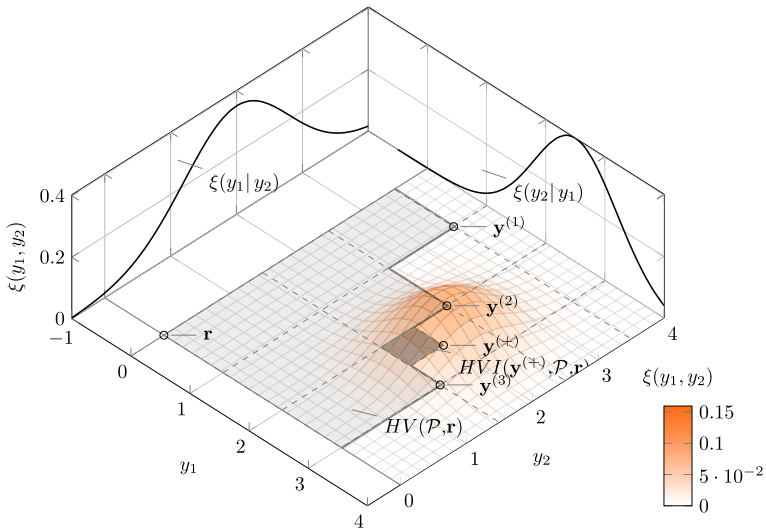
Fig. 2 Expected hypervolume improvement in 2-D (cf. Example 2)

In Eq. (16), I($\mathbf{y}$ impr $\mathcal{P}$) = 1 means that $\mathbf{y}$ is an element of the non-dominated space of $\mathcal{P}$. In other words, $\mathbf{y} \in [\mathbf{r}, \infty^d] \backslash \text{dom}(\mathcal{P})$ if I($\mathbf{y}$ impr $\mathcal{P}$) = 1. A reference point $\mathbf{r}$ is not indicated in Eq. (16) because $\mathbf{r}$ must be chosen as $[-\infty]^d$ in PoI. Therefore, PoI is a reference-free infill criterion.

Definition 8 (*Expected hypervolume improvement*) Given parameters of the multivariate predictive distribution $\boldsymbol{\mu}, \boldsymbol{\sigma}$ and the Pareto-front approximation set $\mathcal{P}$, the *expected hypervolume improvement* is defined as:

$$EHVI(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) := \int_{\mathbb{R}^d} \text{HVI}(\mathcal{P}, \mathbf{y}, \mathbf{r}) \cdot \boldsymbol{\xi}_{\boldsymbol{\sigma},\boldsymbol{\mu}}(\mathbf{y}) d\mathbf{y} \tag{17}$$

Example 2 An illustration of the 2-D EHVI is shown in Fig. 2. The light gray area is the dominated subspace of $\mathcal{P} = \{\mathbf{y}^{(1)} = (3, 1)^\top, \mathbf{y}^{(2)} = (2, 1.5)^\top, \mathbf{y}^{(3)} = (1, 2.5)^\top\}$ bounded by the reference point $\mathbf{r} = (0, 0)$. The bivariate Gaussian distribution has the parameters $\mu_1 = 2.5$, $\mu_2 = 2$, $\sigma_1 = 0.7$, $\sigma_2 = 0.8$. The probability density function ($\boldsymbol{\xi}$) of the bivariate Gaussian distribution is indicated as a 3-D plot. Here $\mathbf{y}^{(+)}$ is a sample from this distribution and the area of improvement relative to $\mathcal{P}$ is indicated by the dark shaded area. Variables y_1 and y_2 stand for the first and the second objective values, respectively.

For computing integrals of EHVI in Sect. 4, it is useful to define Δ and Ψ_∞ functions.

Definition 9 (Δ *function* (see also [14,39])) For a given vector of objective function values $\mathbf{y} \in \mathbb{R}^d$ and $\mathbf{y} \notin \mathcal{P}$, $\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r})$ is the subset of the vectors in $\mathbb{R}^d$ which are exclusively dominated by the vector $\mathbf{y}$ but not by elements in $\mathcal{P}$, and which dominate the reference point $\mathbf{r}$, that is:

$$\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r}) = \{\mathbf{z} \in \mathbb{R}^d \mid \mathbf{y} \prec \mathbf{z} \text{ and } \mathbf{z} \prec \mathbf{r} \text{ and } \nexists \mathbf{q} \in \mathcal{P} : \mathbf{q} \prec \mathbf{z}\} \tag{18}$$

Definition 10 (Ψ_∞ *function* (see also [19])) Let $\phi(s) = 1/\sqrt{2\pi} e^{-\frac{1}{2}s^2}$ ($s \in \mathbb{R}$) denote the PDF (ξ) of the standard normal distribution. Moreover, let $\Phi(s) = \frac{1}{2}\left(1 + \text{erf}\left(\frac{s}{\sqrt{2}}\right)\right)$ denote

its cumulative probability distribution function (CDF), and erf denote the Gaussian error function. The general normal distribution with mean μ and standard deviation σ has PDF $\xi_{\mu,\sigma}(s) = \phi_{\mu,\sigma}(s) = \frac{1}{\sigma}\phi(\frac{s-\mu}{\sigma})$ and its CDF is $\Phi_{\mu,\sigma}(s) = \Phi(\frac{s-\mu}{\sigma})$. Then the function $\Psi_{\infty}(a, b, \mu, \sigma)$ is defined as:

$$\Psi_{\infty}(a, b, \mu, \sigma) := \int_b^{\infty} (z - a)\frac{1}{\sigma}\phi\left(\frac{z - \mu}{\sigma}\right) dz \qquad (19)$$

One can easily show that $\psi_{\infty}(a, b, \mu, \sigma) = \sigma\phi\left(\frac{b - \mu}{\sigma}\right) + (\mu - a)\left[1 - \Phi\left(\frac{b - \mu}{\sigma}\right)\right]$.

4 Efficient EHVI calculation

This section mainly discusses an efficient partitioning method for a non-dominated space and how to employ this partitioning method to calculate EHVI and PoI.

4.1 Partitioning a non-dominated space

The efficiency of an infill criterion calculation is determined by a non-dominated search algorithm and the number of integration slices. The main idea of the partitioning method is to separate the integration volume (a non-dominated space) into as few integration slices as possible. Then, the integral of the criterion is calculated within each integration slice. The value of the criterion is the sum of its contribution in every integration slice.

4.1.1 The 2-D case

In the 2-D case, the partitioning method is simple and has already been published by Emmerich et al. [14]. Given a Pareto-front approximation set $\mathcal{P}$ with n elements, the algorithm in [14] adopts a new way to derive the EHVI calculation formulas and only partitions a non-dominated space into $n + 1$ integration slices, instead of $(n + 1)^2$ grids in [5,19]. For the sake of completeness, we will introduce this integration technique here briefly.

Suppose $\mathbf{Y} = \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}$ and $d = 2$, an integration space (a non-dominated space) of $\mathbf{Y}$ can be divided into $n + 1$ disjoint integration slices ($S_2^{(i)}, i = 1, \ldots, n + 1$) by drawing lines parallel to y_2-axis at each element in $\mathbf{y}$, as indicated in Fig. 3. Then, each integration slice can be expressed by its lower bound ($\mathbf{l}_2^{(i)}$) and upper bound ($\mathbf{u}_2^{(i)}$). In order to define the slices formally, we argue a Pareto-front approximation set $\mathcal{P}$ with two sentinels: $\mathbf{y}^{(0)} = (r_1, \infty)^{\top}$ and $\mathbf{y}^{(n+1)} = (\infty, r_2)^{\top}$. Then, the integration slices for the 2-D case are defined by:

$$S_2^{(i)} = (\mathbf{l}_2^{(i)}, \mathbf{u}_2^{(i)}) = \left(\begin{pmatrix} l_1^{(i)} \\ l_2^{(i)} \end{pmatrix}, \begin{pmatrix} u_1^{(i)} \\ u_2^{(i)} \end{pmatrix}\right) = \left(\begin{pmatrix} y_1^{(i-1)} \\ y_2^{(i)} \end{pmatrix}, \begin{pmatrix} y_1^{(i)} \\ \infty \end{pmatrix}\right), i = 1, \ldots, N_2 \quad (20)$$

In the 2-D case, the number of integration slices is straightforward, namely, $N_2 = n + 1$.

4.1.2 The 3-D case

Similar to the 2-D partitioning method, in the 3-D case, each integration slice can be defined by its lower bound ($\mathbf{l}_3$) and upper bound ($\mathbf{u}_3$). Since the upper bound of each integration slice is always ∞ in the y_3 axis, we can describe each integration slice as follows:
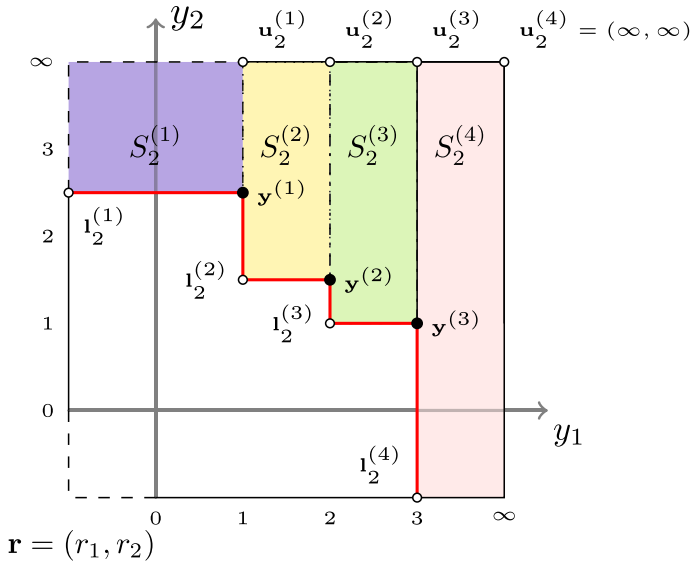
Fig. 3 Partitioning of the 2-D integration region into slices

$$S_3^{(i)} = (\mathbf{l}_3^{(i)}, \mathbf{u}_3^{(i)}) = \left(\begin{pmatrix} l_1^{(i)} \\ l_2^{(i)} \\ l_3^{(i)} \end{pmatrix}, \begin{pmatrix} u_1^{(i)} \\ u_2^{(i)} \\ \infty \end{pmatrix} \right), \qquad i = 1, \ldots, N_3 \qquad (21)$$

Example 3 An illustration of integration slices is shown in Fig. 4. A Pareto front set $\mathcal{P}$ is composed by 4 points ($\mathbf{y}^{(1)} = (1, 3, 4)^\top, \mathbf{y}^{(2)} = (4, 2, 3)^\top, \mathbf{y}^{(3)} = (2, 4, 2)^\top$ and $\mathbf{y}^{(4)} = (3, 5, 1)^\top$), and this Pareto front is shown in the upper left figure. The upper right figure represents the partitioned integration slices of $\mathcal{P}$. The lower center figure illustrates the projection of the upper right figure onto the $y_1 y_2$-plane with rectangle slices and $\mathbf{l}, \mathbf{u}$. The rectangular slices, which share a similar color but of different opacity, represent integration slices with the same value of y_3 in their lower bound. The lower bound of the 3-D integration slice B_4 is $\mathbf{l}_3^{(4)} = (1, 2, 2)^\top$, and the upper bound of the slice is $\mathbf{u}_3^{(4)} = (2, 4, \infty)^\top$.

Algorithm 2 describes how to obtain the slices $S_3^{(1)}, \ldots, S_3^{(i)}, \ldots, S_3^{(N_3)}$ with the corresponding lower and upper bounds ($\mathbf{l}_3^{(i)}$ and $\mathbf{u}_3^{(i)}$). The partitioning algorithm is similar to the sweep line algorithm described in [15]. The basic idea of this algorithm is to use an AVL tree to process points in descending order of the y_3 coordinate. For each such point, say $\mathbf{y}^{(i)}$, the algorithm finds all the points ($\mathbf{y}^{(d[1])}, \ldots, \mathbf{y}^{(d[s])}$) which are dominated by $\mathbf{y}^{(i)}$ in the $y_1 y_2$-plane and inserts $\mathbf{y}^{(i)}$ into the tree. Moreover, because of $\mathbf{y}^{(i)}$, the algorithm will also discard all the points ($\mathbf{y}^{(d[1])}, \ldots, \mathbf{y}^{(d[s])}$) from the AVL tree. See Fig. 5 for describing one such iteration. In each iteration, $s + 1$ slices are created by coordinates of the points $\mathbf{y}^{(t)}$, $\mathbf{y}^{(d[1])}, \ldots, \mathbf{y}^{(d[s])}, \mathbf{y}^{(r)}$, and $\mathbf{y}^{(i)}$ as illustrated in Fig. 5.

The number of the integration slices in the 3-D case N_3 is $2n + 1$ where all points are in general position (for each $i, i = 1, \ldots, d$: the i-th coordinate is different for each pair of points in $\mathbf{Y}$). Otherwise, $2n + 1$ provides an upper bound for the obtained number of slices.

Proof In the algorithm, each point $\mathbf{y}^{(i)}|_{i=1,\ldots,n}$ creates two slices. The first one, say slice $A^{(i)}$, is created when the point $\mathbf{y}^{(i)}$ is *added* to the AVL tree. Another slice, say slice
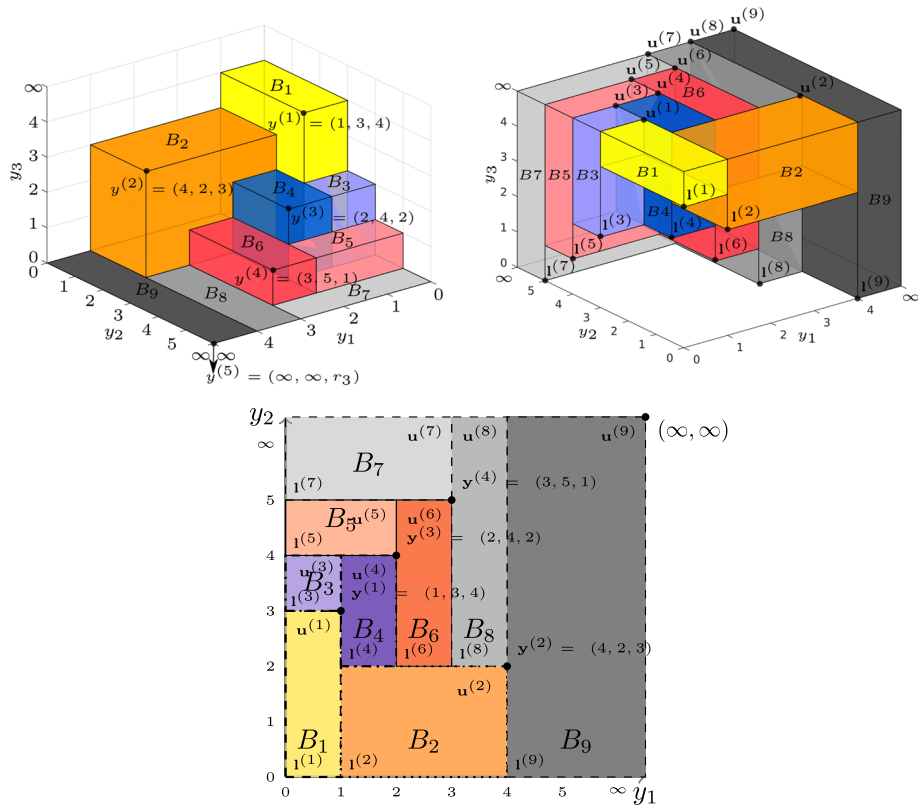
Fig. 4 Upper left: 3-D Pareto-front approximation. Upper right: Integration slices in 3-D. Lower center: The projection of 3-D integration slices into the $y_1 y_2$-plane, each slice can be described by lower bound and upper bound

$S_3^{(i)}$, is created when the point $\mathbf{y}^{(i)}$ is *discarded* from the AVL tree due to domination by another point, say $\mathbf{y}^{(s)}$, in the $y_1 y_2$-plane. These two slices are defined as follows $A^{(i)} = ((y^{(t)}, y_2^{(l2)}, y_3^{(i)})^\top, (y_1^{(u1)}, y_2^{(i)}, \infty)^\top)$ whereas $y_2^{(l2)}$ is either $y_2^{(r)}$ if no point is dominated by $\mathbf{y}^{(i)}$ in the $y_1 y_2$-plane, or $y_2^{(d[1])}$, otherwise. Moreover, $S_3^{(i)} = ((y_1^{(i)}, y_2^{(r)}, y_3^{(s)})^\top$, $(y_1^{(u)}, y_2^{(s)}, \infty)^\top)$ and $\mathbf{y}^{(u)}$ denote either the right neighbor among the newly dominated points in the $y_1 y_2$-plane, or $\mathbf{y}^{(s)}$ if $\mathbf{y}^{(i)}$ is the rightmost point among all newly dominated points. In this way, each slice can be attributed to exactly one point in $\mathcal{P}$, except the slice that is created in the final iteration. In the final iteration, one additional point $\mathbf{y}^{(n+1)} = (\infty, \infty, \infty)^\top$ is added to the AVL tree. This point will create a new slice when it is added, but because it is never discarded, it adds only a single slice. Therefore, $2n + 1$ slices are created in total. □

4.1.3 Higher dimensional cases

In higher dimensional cases, the non-dominated space can be partitioned into axis aligned hyperboxes, similar to the 3-D case. In the d-dimensional case ($d \geq 4$), the hyperboxes can be denoted by $S_d^{(1)}, \ldots, S_d^{(i)}, \ldots, S_d^{N_d}$ with their lower bounds $(\mathbf{l}^{(1)}, \ldots, \mathbf{l}^{(N_d)})$ and upper

---

Algorithm 2: Integration slice acquisition in 3-D case

Input: $(\mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)})$: mutually non-dominated $\mathbb{R}^3$-points sorted by third coordinate (y_3) in descending order

Output: $S_3^{(1)}, \ldots, S_3^{(i)}, \ldots, S_3^{(N_3)}$

1   $\mathbf{y}^{(n+1)} = (\infty, \infty, r_3)$;

2   Initialize AVL tree T for 3-D points

    Insert $\mathbf{y}^{(1)}$, $(\infty, r_2, \infty)^\top$ and $(r_1, \infty, \infty)^\top$ into T;

3   Initialize the number of integration slices $n_b = 1$;

4   Initialize $EHVI = 0$;

5   **for** $i = 2$ **to** $n+1$ **do**   `/* Main loop */`

6   Retrieve the following information from tree T:

7   r: index of the successor of $\mathbf{y}^{(\mathbf{i})}$ in y_1-coordinate (right neighbor);

8   t: index of the successor of $\mathbf{y}^{(\mathbf{i})}$ in y_2-coordinate (left neighbor);

9   d[1], ..., d[s]: indices of points dominated by $\mathbf{y}^{(\mathbf{i})}$ in $y_1 y_2$-plane, sorted ascendingly in the first coordinate(y_1);

10    $S_3^{(n_b)}.l_3 = y_3^{(i)}, \quad S_3^{(n_b)}.u_2 = y_2^{(i)}, \quad S_3^{(n_b)}.u_3 = \infty$;

11   **if** $s == 0$ **then**   `/* Case 1 */`

12     $S_3^{(n_b)}.l_1 = y_1^{(t)}, \quad S_3^{(n_b)}.l_2 = y_2^{(r)}, \quad S_3^{(n_b)}.u_1 = y_1^{(i)}$;

13     $n_b = n_b + 1$;

14   **else**   `/* Case 2 */`

15   **for** $j = 1$ **to** $s + 1$ **do**

16   **if** $j == 1$ **then**

17       $S_3^{(n_b)}.l_1 = y_1^{(t)}, \quad S_3^{(n_b)}.l_2 = y_2^{(d[1])}, \quad S_3^{(n_b)}.u_1 = y_1^{(d[1])}$;

18   **else if** $j == s + 1$ **then**

19       $S_3^{(n_b)}.l_1 = y_1^{(d[s])}, \quad S_3^{(n_b)}.l_2 = y_2^{(r)}, \quad S_3^{(n_b)}.u_1 = y_1^{(i)}$;

20   **else**

21       $S_3^{(n_b)}.l_1 = y_1^{(d[j-1])}, \quad S_3^{(n_b)}.l_2 = y_2^{(d[j])}, \quad S_3^{(n_b)}.u_1 = y_1^{(d[j])}$;

22      $n_b = n_b + 1$;

23   Discard $\mathbf{y}^{(d[1])}, \ldots, \mathbf{y}^{(d[s])}$ from tree T;

24   Insert $\mathbf{y}^{(i)}$ in tree T.

25 Return $S_3^{(1)}, \ldots, S_3^{(i)}, \ldots, S_3^{(N_3)}$

---

bounds $(\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(N_d)})$. Here, N_d is the number of hyperboxes and has the same definition as N_2 and N_3. The hyper-integral box $S_d^{(i)}$ is defined as:

$$S_d^{(i)} = (\mathbf{l}_d^{(i)}, \mathbf{u}_d^{(i)}) = \left((l_1^{(i)}, \ldots, l_d^{(i)})^\top, (u_1^{(i)}, \ldots, \infty)^\top \right) \quad i = 1, \ldots, N_d \qquad (22)$$

An efficient algorithm for partitioning a higher dimensional, non-dominated space is proposed in this section, which is based on two state-of-the-art algorithms DKLV17 [6] by Dächert et al. and LKF17 [25] by Lacour et al. Here, algorithm DKLV17 is an efficient algorithm to locate the local lower bound points[5] ($\mathbf{L}_d$) in a dominated space for maximization problems, based on a specific neighborhood structure among local lower bounds. Moreover, LKF17 is an efficient algorithm to calculate the HVI by partitioning the dominated space. In other words, LKF17 is also efficient in partitioning the dominated space and provides the boundary information for each hyperbox in the dominated space.

---

[5] For the definition of the local lower bound points $\mathbf{L}_d$, see [6].
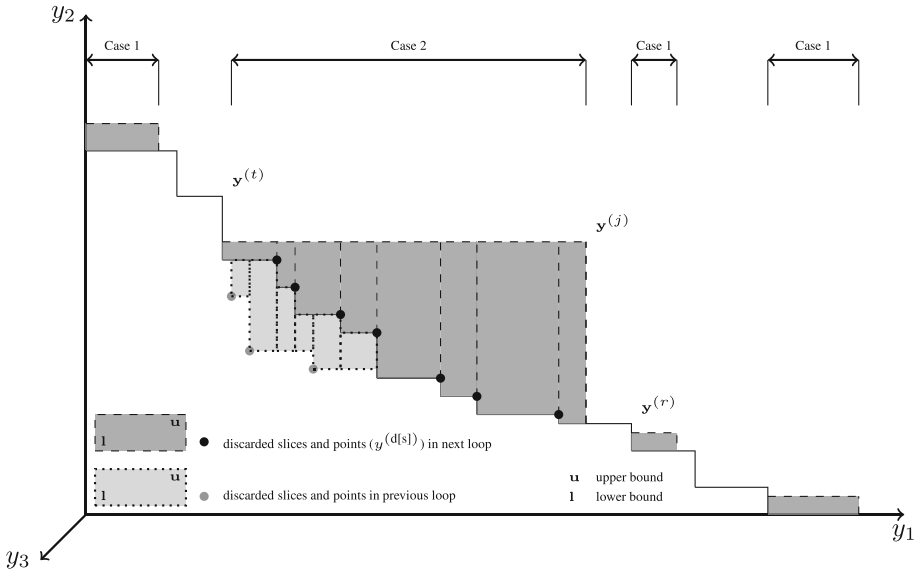
Fig. 5 Boundary search for slices in 3-D case

The idea behind the proposed algorithm is transforming the problem of partitioning a non-dominated space into the problem of partitioning the dominated space, by means of introducing an intermediate Pareto-front approximation set $\mathcal{P}'$. This transformation is done by the following steps. Suppose that we have a current Pareto-front approximation set $\mathcal{P}$ for a maximization problem and we want to partition the non-dominated space of $\mathcal{P}$. Firstly, DKLV17 is applied to locate the local lower bound points ($\mathbf{L}_d$) of $\mathcal{P}$ in the dominated space. Secondly, regard $\mathbf{L}_d$ as a new Pareto-front approximation $\mathcal{P}'$ for a minimization problem with a reference point $\{\infty\}^d$. The dominated space of $\mathcal{P}'$ is actually the non-dominated space of $\mathcal{P}$. Then, LKF17 can be applied to partition the dominated space of $\mathcal{P}'$ by locating the lower bound points $\mathbf{l}_d$ and the upper bound points $\mathbf{u}_d$. These bound points ($\mathbf{l}_d,\mathbf{u}_d$) of $\mathcal{P}'$ in the dominated space for a minimization problem are exact the lower/upper bound points of the partitioned, non-dominated hyperboxes of $\mathcal{P}$ for a maximization problem. The pseudo code of partitioning non-dominated space in higher dimensional cases is shown in Algorithm 3.

Algorithm 3: Partitioning a non-dominated space in higher dimensional cases

Input: Pareto-front approximation set $\mathcal{P}$ (maximization problem), a reference point $\mathbf{r}$
Output: Hyperboxes S_d

1 Locate local lower bound points $\mathbf{L}_d$: $\mathbf{L}_d = DKLV17(\mathcal{P}, \mathbf{r})$;
2 Set new Pareto front $\mathcal{P}'$ using $\mathbf{L}_d$: $\mathcal{P}' = \mathbf{L}_d$;
3 Set a new reference point $\mathbf{r}'$: $\mathbf{r}' = \{\infty\}^d$;
4 Get lower bound points $\mathbf{l}_d$ and upper bound points $\mathbf{u}_d$: $(\mathbf{l}_d, \mathbf{u}_d) = LKF17(\mathcal{P}', \mathbf{r}')$;
5 $S_d = (\mathbf{l}_d, \mathbf{u}_d)$;
6 Return S_d
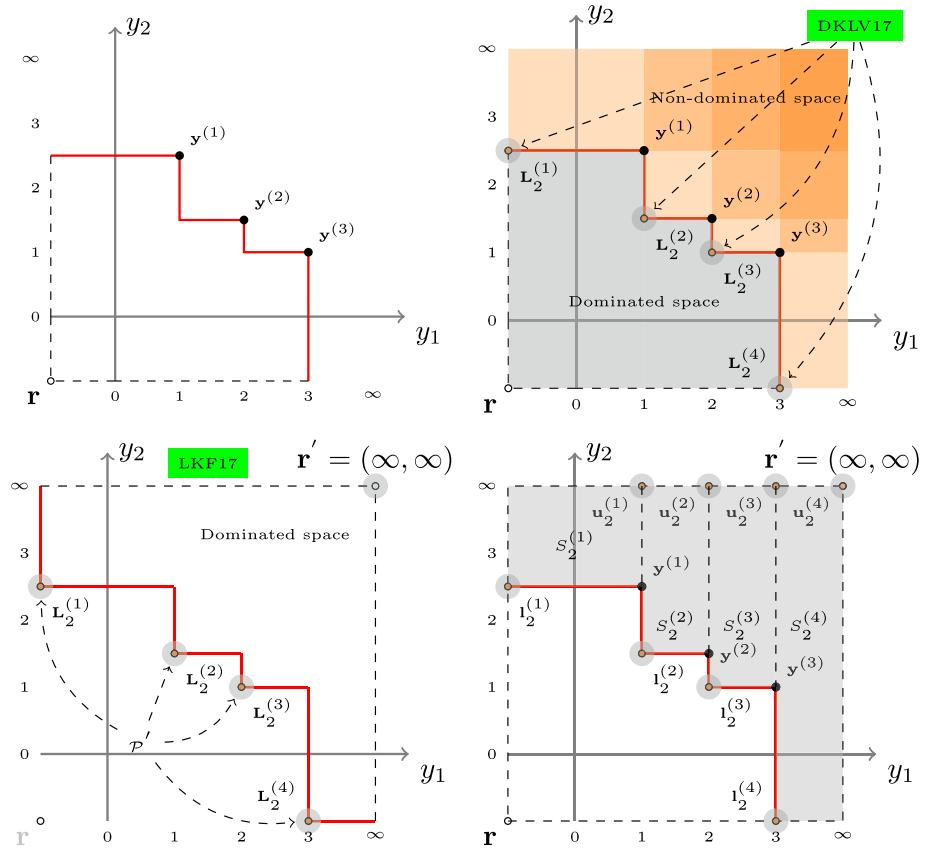
Fig. 6 The illustration of partitioning a non-dominated space in higher dimensional cases. Above left: Pareto-front approximation set $\mathcal{P}$. Above right: Locating $\mathbf{L}_2$ points using DKLV17. Below left: Partitioning the dominated space of $\mathcal{P}'$ using LKF17. Below right: The partitioned non-dominated space of $\mathcal{P}$

Example 4 Figure 6 illustrates Algorithm 3. In the 2-D case, suppose a Pareto-front approximation set is $\mathcal{P}$, which consists of $\mathbf{y}^{(1)} = (1, 2.5)^{\top}$, $\mathbf{y}^{(2)} = (2, 1.5)^{\top}$ and $\mathbf{y}^{(3)} = (3, 1)^{\top}$. The reference point is $\mathbf{r} = (0, 0)$, see Fig. 6 (above left). Use DKLV17 to locate the local lower bound points $\mathbf{L}_2$, which consist of $\mathbf{L}_2^{(1)} = (0, 2.5)^{\top}$, $\mathbf{L}_2^{(2)} = (1, 1.5)^{\top}$, $\mathbf{L}_2^{(3)} = (2, 1)^{\top}$ and $\mathbf{L}_2^{(4)} = (3, 0)^{\top}$, see Fig. 6 (above right). Regard all of the local lower bound points $\mathbf{L}_2$ as the elements of a new Pareto-front approximation set $\mathcal{P}' = (\mathbf{L}_2^{(1)}, \ldots, \mathbf{L}_2^{(4)})$. Set a new reference point $\mathbf{r}' = (\infty, \infty)$ and utilize LKF17 to partition the dominated space of $\mathcal{P}'$, by considering minimization, see Fig. 6 (below left). The partitioned non-dominated space of $\mathcal{P}$ is then the partitioned dominated space of $\mathcal{P}'$, see Fig. 6 (below right).

4.2 EHVI calculation

[6]This section discusses the problem of exact EHVI calculation. Moreover, a new and efficient algorithm is also derived. Sections 4.2.1 and 4.2.2 introduce the proposed method in the 2-

---

[6] Both C++ and MATLAB source code for computing the EHVI are available on http://liacs.leidenuniv.nl/~csmoda/index.php?page=code or on request from the authors.

D and 3-D cases, respectively. Section 4.2.3 illustrates the general calculation formulas in higher dimensional cases, based on the proposed method.

In order to simplify the notation, $\Delta(\mathbf{y})$ is used whenever $\mathcal{P}, \mathbf{r}$ are given by the context. Based on $\Delta(\mathbf{y})$, the expected hypervolume improvement function can be re-defined as:

$$
\begin{aligned}
\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) &= \int_{\mathbb{R}^d} \text{HVI}(\mathcal{P}, \mathbf{y}, \mathbf{r}) \cdot \boldsymbol{\xi}_{\boldsymbol{\sigma}, \boldsymbol{\mu}}(\mathbf{y}) d\mathbf{y} \\
&= \int_{y_1=-\infty}^{\infty} \cdots \int_{y_d=-\infty}^{\infty} \lambda_d[S_d \cap \Delta(\mathbf{y})] \boldsymbol{\xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}
\end{aligned}
\tag{23}
$$

For the convenience of expressing the EHVI formula in the remaining parts of this paper, two functions (ℓ and ϑ) are defined as follows:

Definition 11 (ℓ *function*) Given the parameters of an integration slice $S_d^{(i)}$ in a d-dimensional space, the *Hypervolume Improvement* of slice $S_d^{(i)}$ in dimension $k \leq d$ is defined as:

$$
\ell(u_k^{(i)}, y_k, l_k^{(i)}) := \lambda_1[S_d^{(i)} \cap \Delta(y_k)] = |[l_k^{(i)}, u_k^{(i)}] \cap [l_k^{(i)}, y_k]| = \min\{u_k^{(i)}, y_k\} - l_k^{(i)} \tag{24}
$$

Definition 12 (ϑ *function*) Given the parameters of an integration slice $S_d^{(i)}$ in a d-dimensional space and multivariate predictive distribution $\boldsymbol{\mu}, \boldsymbol{\sigma}$, the function $\vartheta(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k)$ is then defined as:

$$
\begin{aligned}
\vartheta(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) &:= \int_{y_k=u_k^{(i)}}^{\infty} \lambda_1[S_d^{(i)} \cap \Delta(y_k)] \cdot \boldsymbol{\xi}_{\mu_k, \sigma_k}(y_k) dy_k \\
&= \int_{y_k=u_k^{(i)}}^{\infty} (u_k^{(i)} - l_k^{(i)}) \cdot \boldsymbol{\xi}_{\mu_k, \sigma_k}(y_k) dy_k \\
&= (u_k^{(i)} - l_k^{(i)}) \cdot \left(1 - \Phi\left(\frac{u_k^{(i)} - \mu_k}{\sigma_k}\right) \right) \quad k = 1, \ldots, d-1
\end{aligned}
\tag{25}
$$

4.2.1 2-D EHVI calculation

According to the definition of the 2-D integration slice in Eq. 4.1.1, the *Hypervolume Improvement* $\mathbf{y} \in \mathbb{R}^2$ in the 2-D case is:

$$
\text{HVI}_2(\mathbf{y}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_2} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})]
\tag{26}
$$

HVI_2 gives rise to the compact integral for the original EHVI:

$$
\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{y_1=-\infty}^{\infty} \int_{y_2=-\infty}^{\infty} \sum_{i=1}^{N_2} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}
\tag{27}
$$

Here $\mathbf{y} = (y_1, y_2)$, the intersection of $S_2^{(i)}$ with $\Delta(y_1, y_2)$ is non-empty if and only if $(\mathbf{y})$ dominates the lower left corner of $S_2^{(i)}$. Therefore:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_2} \int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \qquad (28)$$

In Eq. (28), the summation is done after integration. This operation is allowed, because integration is a linear mapping. Moreover, the integration interval $\int_{y_1=l_1^{(i)}}^{\infty}$ can be divided into $(\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} + \int_{y_1=u_1^{(i)}}^{\infty})$, because the HVI in one dimension $\lambda_1[S_2^{(i)} \cap \Delta(y_1)]$ differs in these two integration intervals. Here $\lambda_1[B_i \cap \Delta(y_k)]$ is the HVI in dimension k, i.e., a 1-D HVI. Equation (28) can then be expressed as:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_2} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \qquad (29)$$

$$+ \sum_{i=1}^{N_2} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \qquad (30)$$

According to the definition of HVI, $\ell(u_1^{(i)}, y_1, l_1^{(i)})$ is constant and is $(u_1^{(i)} - l_1^{(i)})$ in $\int_{y_1=u_1^{(i)}}^{\infty}$. Therefore, the *Expected Improvement* in dimension y_1 is also a constant and it is: $\vartheta(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1)$. Recall the Ψ_{∞} function, by which the terms (29) and (30) can be expressed as follows:

$$\text{Term (29)} = \sum_{i=1}^{N_2} \left(\Psi_{\infty}(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_{\infty}(l_1^{(i)}, u_1^{(i)}, \mu_1, \sigma_1) \right) \cdot \Psi_{\infty}(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) \qquad (31)$$

$$\text{Term (30)} = \sum_{i=1}^{N_2} \vartheta(l_1^{(i)}, u_1^{(i)}, \mu_1, \sigma_1) \cdot \Psi_{\infty}(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) \qquad (32)$$

According to Eq. (28), the exact EHVI calculation needs to compute the terms (29) and (30) $n+1$ times, and each calculation requests $O(1)$ computation. To keep $\mathcal{P}$ sorted in the first coordinate requires an effort of amortized time complexity $O(\log n)$ per iteration. Hence, the time complexity of the expected hypervolume improvement in the 2-D case is in $O(n \log n)$. In the case when $\mathcal{P}$ is sorted, we can show that the time complexity is in $\Theta(n)$.

4.2.2 3-D EHVI calculation

Given a partitioning of the non-dominated space into integration slices $S_3^{(1)}, \ldots, S_3^{(i)}, \ldots,$ $S_3^{(2n+1)}$, the EHVI integrations over each slice can be computed separately. To see how this calculation can be done, the *Hypervolume Improvement* of a point $\mathbf{y} \in \mathbb{R}^3$ is rewritten as:

$$\text{HVI}_3(\mathcal{P}, \mathbf{y}, \mathbf{r}) = \sum_{i=1}^{N_3} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \qquad (33)$$

where $\Delta(\mathbf{y})$ is the part of the objective space that is dominated by $\mathbf{y}$. The HVI expression in the definition of EHVI in Eq. (17) can be replaced by HVI_3 in Eq. (33):

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \qquad (34)$$

Similar to the 2-D case, we can divide the integration interval $\int_{y_1=l_1^{(i)}}^{\infty}$ and $\int_{y_2=l_2^{(i)}}^{\infty}$ into $(\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} + \int_{y_1=u_1^{(i)}}^{\infty})$ and $(\int_{y_2=l_2^{(i)}}^{u_2^{(i)}} + \int_{y_2=u_2^{(i)}}^{\infty})$, respectively. Also, again we can swap integration and summation based on the fact that integration is a linear mapping. Based on this subdivision, Eq. (34) can be expressed as:

$$\text{Eq. (34)} = \sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \qquad (35)$$

$$+ \sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=u_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \qquad (36)$$

$$+ \sum_{i=1}^{N_3} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \qquad (37)$$

$$+ \sum_{i=1}^{N_3} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=u_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \qquad (38)$$

Recalling the definition of the ϑ function and calculation of $\lambda_1[B_i \cap \Delta(y_k)]$, the term (35) can be rewritten as follows:

$$\text{Term (35)} = \sum_{i=1}^{N_3} (\Psi_\infty(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_\infty(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1))$$
$$\cdot (\Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) - \Psi_\infty(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2)) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (39)$$

Similar to the derivation of the term (35), the terms (36), (37) and (38) can be written as follows:

$$\text{Term (36)} = \sum_{i=1}^{N_3} (\Psi_\infty(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_\infty(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1)) \cdot \vartheta(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2)$$
$$\cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \qquad (40)$$

$$\text{Term (37)} = \sum_{i=1}^{N_3} \vartheta(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1) \cdot (\Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) - \Psi_\infty(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2))$$
$$\cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \qquad (41)$$

$$\text{Term } (38) = \sum_{i=1}^{N_3} \vartheta(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1) \cdot \vartheta(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (42)$$

The final EHVI formula is the sum of the terms (39), (40), (41) and (42).

During the EHVI calculation, $y_1 y_2$-projections are mutually non-dominated. Moreover, the points are sorted by the y_2 coordinate in the AVL tree. Therefore, identifying a neighboring/discard point takes time $O(\log n)$. Then the EHVI for these integration slices is calculated by the summation of the term (39), (40), (41) and (42), with the parameters of $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and $S_3^{(N_3)}$. The EHVI time complexity for each slice is $O(1)$. Moreover, the dominated points ($\mathbf{y}^{(\mathrm{d}[s])}$) are removed from the AVL tree, and the new points ($\mathbf{y}^{(j)}$) are inserted into the AVL tree. Since the points dominated by the new point $\mathbf{y}^{(j)}$ are deleted at the end of the current loop, they will not occur again in later computations. Hence, the total number of open slices does not exceed N_3, as mentioned before, and the total computational cost is $O(n \log n)$.

4.2.3 Higher dimensional EHVI

The interval of integration in each coordinate (except the last) can be divided into 2 parts: $[l, u]$ and $[u, \infty]$. Therefore, the equation for EHVI for each hyperbox can be decomposed into 2^{d-1} parts. For the interval of $[u, \infty]$, the improvements ($\lambda_k[S_d^{(i)} \cap \Delta(y_k)]$) are constant values, and the Ψ_∞ function can be simplified by calculating function Φ and the improvement in these coordinates. For the last coordinate, there is no need to separate the interval, because the improvement in this coordinate ($\lambda_d[S_d^{(i)} \cap \Delta(y_d)]$) is a variable in $[l, \infty]$.

According to the definition of higher dimensional integral boxes in Sect. 4.1.3, the EHVI ($d \geqslant 4$) can be calculated by the following equation:

$$\text{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r})$$

$$= \sum_{i=1}^{N_d} \int_{y_1=l_1^{(i)}}^{\infty} \cdots \int_{y_d=l_d^{(i)}}^{\infty} \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}$$

$$= \sum_{i=1}^{N_d} \left(\left(\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} + \int_{y_1=u_1^{(i)}}^{\infty} \right) \cdots \left(\int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} + \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} \right) \cdot \int_{y_d=l_d^{(i)}}^{\infty} \right) \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}$$

$$= \left\{ \begin{array}{cccc}
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{u_{d-2}^{(i)}} \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} \int_{y_d=l_d^{(i)}}^{p_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{u_{d-2}^{(i)}} \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} \int_{y_d=p_d^{(i)}}^{\infty} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{u_{d-2}^{(i)}} \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} \int_{y_d=l_d^{(i)}}^{p_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{u_{d-2}^{(i)}} \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} \int_{y_d=p_d^{(i)}}^{\infty} \\
\cdots \\
\int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{\infty} \int_{y_{d-1}=l_{d-1}^{(i)}}^{\infty} \int_{y_d=l_d^{(i)}}^{p_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \cdots \int_{y_{d-2}=l_{d-2}^{(i)}}^{\infty} \int_{y_{d-1}=l_{d-1}^{(i)}}^{\infty} \int_{y_d=p_d^{(i)}}^{\infty}
\end{array} \right\} \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot \boldsymbol{\xi}_{\boldsymbol{\mu}, \boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y}$$

$$
\begin{aligned}
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, \sigma_{l}^p\right) &\quad \vartheta\!\left(l_{(i)}^{p-1}, \sigma_{p-2}, \sigma_{p-2}, l_{(i)}^{p-1}\right) &\quad \cdots &\quad \vartheta\!\left(l_{(i)}^2, \, u_{(i)}^2, \sigma_2\right) &\quad \vartheta\!\left(l_{(i)}^1, \, u_{(i)}^1, \sigma_1\right) \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, \sigma_{l}^p\right) &\quad \vartheta\!\left(l_{(i)}^{p-1}, \sigma_{p-2}, \sigma_{p-2}, l_{(i)}^{p-1}\right) &\quad \cdots &\quad \vartheta\!\left(l_{(i)}^2, \, u_{(i)}^2, \sigma_2\right) &\quad \vartheta\!\left(l_{(i)}^1, \, u_{(i)}^1, \sigma_1\right) \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, \sigma_{l}^p\right) &\quad \vartheta\!\left(\sigma_{p-2}, \sigma_{p-2}, u_{(i)}^{p-1}\right) &\quad \cdots &\quad \vartheta\!\left(l_{(i)}^2, \, u_{(i)}^2, \sigma_2\right) &\quad \vartheta\!\left(l_{(i)}^1, \, u_{(i)}^1, \sigma_1\right) \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, \sigma_{l}^p\right) &\quad \vartheta\!\left(\sigma_{p-2}, \sigma_{p-2}, u_{(i)}^{p-1}\right) &\quad \cdots &\quad \vartheta\!\left(l_{(i)}^2, \, u_{(i)}^2, \sigma_2\right) &\quad \vartheta\!\left(l_{(i)}^1, \, u_{(i)}^1, \sigma_1\right) \\
&\vdots & \vdots &\quad \cdots & \vdots & \vdots \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, u_{(i)}^p\right) &\quad \vartheta\!\left(u_{(i)}^{p-1}, \sigma_{p-2}, \sigma_{p-2}, u_{(i)}^{p-1}\right) &\quad \cdots &\quad \vartheta\!\left(u_{(i)}^2, \, u_{(i)}^2, \sigma_2\right) &\quad \vartheta\!\left(u_{(i)}^1, \, u_{(i)}^1, \sigma_1\right) \\
&\cdots & \cdots &\quad \cdots & \cdots & \cdots \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, u_{(i)}^p\right) &\quad \vartheta\!\left(l_{(i)}^{p-1}, l_{(i)}^{p-1}, u_{(i)}^{p-1}\right) &\quad \cdots \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, u_{(i)}^p\right) &\quad \vartheta\!\left(l_{(i)}^{p-1}, l_{(i)}^{p-1}, u_{(i)}^{p-1}\right) &\quad \cdots \\
&\varphi_\infty\!\left(l_{(i)}^p, l_{(i)}^p, u_{(i)}^p\right) &\quad \vartheta\!\left(l_{(i)}^{p-1}, l_{(i)}^{p-1}, u_{(i)}^{p-1}\right) &\quad \cdots
\end{aligned}
\qquad (43)
$$

$$
= \left(
\begin{aligned}
&\omega(i,1,0)\,\omega(i,2,0)\,\cdots\,\omega(i,d-2,0)\,\omega(i,d-1,0)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,0)\,\omega(i,2,0)\,\cdots\,\omega(i,d-2,0)\,\omega(i,d-1,1)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,0)\,\omega(i,2,0)\,\cdots\,\omega(i,d-2,1)\,\omega(i,d-1,0)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,0)\,\omega(i,2,0)\,\cdots\,\omega(i,d-2,1)\,\omega(i,d-1,1)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\cdots \qquad\cdots\qquad \ddots \qquad\cdots\qquad\cdots\qquad \cdots \\
&\omega(i,1,1)\,\omega(i,2,1)\,\cdots\,\omega(i,d-2,0)\,\omega(i,d-1,0)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,1)\,\omega(i,2,1)\,\cdots\,\omega(i,d-2,0)\,\omega(i,d-1,1)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,1)\,\omega(i,2,1)\,\cdots\,\omega(i,d-2,1)\,\omega(i,d-1,0)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d) + \\
&\omega(i,1,1)\,\omega(i,2,1)\,\cdots\,\omega(i,d-2,1)\,\omega(i,d-1,1)\,\Psi_\infty(l_d^{(i)},\,P_d^{(i)},\,l_d^{(i)},\,\mu_d,\,\sigma_d)
\end{aligned}
\right)
$$

$$
=
\begin{aligned}
&\omega(i,1,C_1^{(0)_2})\ \omega(i,2,C_2^{(0)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(0)_2})\ \omega(i,d-1,C_{d-1}^{(0)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(1)_2})\ \omega(i,2,C_2^{(1)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(1)_2})\ \omega(i,d-1,C_{d-1}^{(1)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(2)_2})\ \omega(i,2,C_2^{(2)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(2)_2})\ \omega(i,d-1,C_{d-1}^{(2)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(3)_2})\ \omega(i,2,C_2^{(3)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(3)_2})\ \omega(i,d-1,C_{d-1}^{(3)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\cdots\qquad\cdots\qquad\ddots\qquad\cdots\qquad\cdots\qquad\cdots \\
&\omega(i,1,C_1^{(2^{d-1}-4)_2})\ \omega(i,2,C_2^{(2^{d-1}-4)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(2^{d-1}-4)_2})\ \omega(i,d-1,C_{d-1}^{(2^{d-1}-4)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(2^{d-1}-3)_2})\ \omega(i,2,C_2^{(2^{d-1}-3)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(2^{d-1}-3)_2})\ \omega(i,d-1,C_{d-1}^{(2^{d-1}-3)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(2^{d-1}-2)_2})\ \omega(i,2,C_2^{(2^{d-1}-2)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(2^{d-1}-2)_2})\ \omega(i,d-1,C_{d-1}^{(2^{d-1}-2)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ + \\
&\omega(i,1,C_1^{(2^{d-1}-1)_2})\ \omega(i,2,C_2^{(2^{d-1}-1)_2})\ \cdots\ \omega(i,d-2,C_{d-2}^{(2^{d-1}-1)_2})\ \omega(i,d-1,C_{d-1}^{(2^{d-1}-1)_2})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)
\end{aligned}
$$

$$
= \sum_{i=1}^{N_d} \underbrace{\sum_{j=0}^{2^{d-1}-1}\left(\underbrace{\prod_{k=1}^{d-1}\omega(i,k,C_k^{(j)_2})\cdot\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)}_{\text{An integration for a divided interval.}}\right)}_{\text{The EHVI value for a hyperbox }S_d^{(i)},\text{ denoted as }EHVI_{S_d^{(i)}}.}
\tag{44}
$$

In the term (43), the integral of each dimension $\int_{y_k=l_k^{(i)}}^{u_k^{(i)}} \lambda_k[S_k^{(i)} \cap \Delta(y_1, \ldots, y_k)] \cdot \xi_{\mu,\sigma}(\mathbf{y})dy|_{1 \leq k \leq d-1}$ has two and only two different expressions (Ψ_∞ or ϑ), except that in the last dimension ($k = d$), the expression of $\int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot \xi_{\mu,\sigma}(\mathbf{y})dy|_{k=d}$ is always Ψ_∞. The final expression of the EHVI is the sum of the $EHVI_{S_d^{(i)}}$ in Eq. (44) for all the partitioned, non-dominated hyperboxes $S_d^{(i)}|_{i=1,\ldots,N_d}$. In Eq. (44), $EHVI_{S_d^{(i)}}$ has 2^{d-1} terms because the integration has two different expressions in dimension k ($1 \leq k \leq d-1$) and has only one expression in d-th dimension ($k = d$).

In Eq. (44), $(j)_2$ stands for the binary string representation of the integer j. The length of $(j)_2$ is $d - 1$. $C_k^{(j)_2}$ is a bit and represents the k-th bit of $(j)_2$ in the binary string. For example, if $d = 5$, $j = 8$, then $(j)_2 = (1\,0\,0\,0)$, $C_{k=4}^{(j)_2} = 1$ and $C_{k=1,2,3}^{(j)_2} = 0$. In Eq. (44), $\omega(i, k, C_k^{(j)_2})$ is defined as:

$$\omega(i, k, C_k^{(j)_2}) := \begin{cases} \Psi_\infty(l_k^{(i)}, l_k^{(i)}, \mu_k, \sigma_k) - \Psi_\infty(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) & \text{if } C_k^{(j)_2} = 0 \\ \vartheta(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) & \text{if } C_k^{(j)_2} = 1 \end{cases} \quad (45)$$

Equation (44) shows how to calculate EHVI in the case of d objectives. According to Eq. (44), the runtime complexity of the proposed algorithm can be calculated. The exact EHVI is given by $\sum_{j=0}^{2^{d-1}-1} \left(\prod_{k=1}^{d-1} \omega(i, k, C_k^{(j)_2}) \cdot \Psi_\infty(l_d^{(i)}, l_d^{(i)}, \mu_d, \sigma_d) \right)$, which requires $O(1)$ computation steps for each hyperbox calculation. Currently, the exact number of hyperboxes $N_d|_{d\geq4}$ for a non-dominated space is still unknown. It is hypothesized by the authors that N_d is the exact number of the local lower bound points, which can be calculated by the DKLV17 algorithm. The LKF17 algorithm partitions the non-dominated space into $\Theta(n^{\lfloor d/2 \rfloor})$ hyperboxes and the time complexity for computing these hyperboxes grows linearly with the number of boxes (see, e.g., Lacour et al. [25]). Given a fixed dimension d, the time complexity of our EHVI computation algorithm is therefore also in $\Theta(n^{\lfloor d/2 \rfloor})$.

Note that every hyperbox requires $O(2^{d-1})$ time complexity. Therefore, the complexity in terms of d and n is given by $O(2^{d-1} \cdot n^{\lfloor d/2 \rfloor})$. Due to the exponential dependence on d, the EHVI computation algorithm is only useful in moderate dimensional cases. Note, however, that a much faster computation cannot be expected, as the time complexity of the hypervolume indicator itself scales superpolynomially with the number of objectives d, under the assumption P $\neq$ NP. It is easy to show, that the EHVI computation has at least the same time complexity than the hypervolume indicator computation [14], and it is therefore also an NP hard problem in d—but polynomial in n for any fixed value of d.

4.3 Probability of improvement (PoI)

According to the partitioning method in Sect. 4.1, PoI can be calculated as follows:

$$\begin{aligned} \text{PoI}(\mu, \sigma, \mathcal{P}) &= \int_{y_1=-\infty}^{\infty} \cdots \int_{y_d=-\infty}^{\infty} \text{I}(\mathbf{y} \text{ impr } \mathcal{P}) \xi_{\mu,\sigma}(\mathbf{y}) dy_1 \ldots dy_d \\ &= \sum_{i=1}^{N_d} \prod_{j=1}^{d} \text{I}(\mathbf{y} \text{ impr } \mathcal{P}) \left(\Phi\left(\frac{u_j^{(i)} - \mu_j}{\sigma_j}\right) - \Phi\left(\frac{l_j^{(i)} - \mu_j}{\sigma_j}\right) \right) \end{aligned} \quad (46)$$

Here, N_d is the number of integration slices, and $N_2 = n + 1$, $N_3 = 2n + 1$ in the 2-D and 3-D case respectively. Since PoI is a reference-free indicator, a reference point $\mathbf{r} = \{-\infty\}^d$ is only used in order to obtain the correct boundary information ($\mathbf{l}_d$, $\mathbf{u}_d$).

5 Experiments

5.1 Speed comparison

The test benchmarks from Emmerich and Fonseca [15] were used to generate Pareto-front sets. The Pareto-front sets and evaluated points were randomly generated based on CONVEX-SPHERICAL and CONCAVESPHERICAL functions. Two EHVI calculation algorithms, CDD13 [5] and KMAC, were compared using the same benchmarks in this experiment.[7] Note that KMAC algorithm in this paper includes the KMAC_2D for 2-D EHVI calculation in [14], the KMAC_3D algorithm for 3-D EHVI calculation in [38], and the extended KMAC algorithm for higher dimensional cases ($d \geq 4$) in this paper.

The parameters: $\sigma_d = 2.5$, $\mu_d = 10$, $d = 2, \ldots, 5$ are used in the experiments. Pareto front sizes are $|\mathcal{P}| \in \{10, 20, \ldots, 200\}$ and Batch Size is 1, which represents the number of the evaluated points under the same Pareto-front approximation set. Ten $\mathcal{P}$ sets are randomly generated by the same parameters. Average runtimes over 100 repetitions (10 repetitions for 10 $\mathcal{P}$ sets) were computed. All the experiments were performed on the same computer, and the hardware is: Intel(R) Xeon(R) CPU I7 3770 3.40GHz, RAM 16GB. The operating system is Ubuntu 16.04 LTS (64 bit), the compiler of KMAC is g++ 4.9.2 with compiler flag -Ofast, and CDD13 is based on MATLAB 8.4.0.150421 (R2014b), 64 bit. The experiments are set to halt when the algorithms cannot finish the EHVI computation within 30 min.

The experimental results in Fig. 7 show that KMAC is much faster than CDD13, especially when $|\mathcal{P}|$ is increased. Moreover, CDD13 can not calculate the exact EHVI value within 30 min when $|\mathcal{P}|$ is bigger than 30 in the 5-D case.

5.2 Benchmark performance

Five state-of-the-art algorithms are compared in this section, namely: EHVI-MOBGO, PoI-MOBGO[8], NSGA-II [8], NSGA-III [7,17] and SMS-EMOA [1]. The benchmarks are DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ5, DTLZ7 [9], MaF1, MaF5, MaF12 and MaF13 [2]. The dimension m of all the benchmarks is 6 or 18. The parameter settings for all of these algorithms are shown in Table 2. The number of function evaluations (T_c in Algorithm 1) for MOBGO based algorithms is 300. The Reference points for each benchmark are shown in Table 3. For DTLZ1 and DTLZ2, the reference points are chosen from the article [5]. For the other test problems, the reference points are revised from the articles [2,5]. All experiments were repeated 10 times.

The final Pareto fronts are evaluated by means of the *Hypervolume* indicator. Tables 4 and 5 show the experimental results (statistical means and standard deviations) in 6- and 18-dimensional search spaces, respectively. Compared with EMOAs in 6- and 18-dimensional search spaces, either EHVI-MOBGO or PoI-MOBGO yields the best result on the 10 benchmark function with 300 function evaluations. On the test problems of DTLZ7 and MaF13,

---

[7] Another EHVI calculation algorithm, IRS_fast, is not compared in this paper, as it only works when $d = 2, 3$. For the detailed speed comparisons between IRS_fast and KMAC in the 3-D case, see [38].

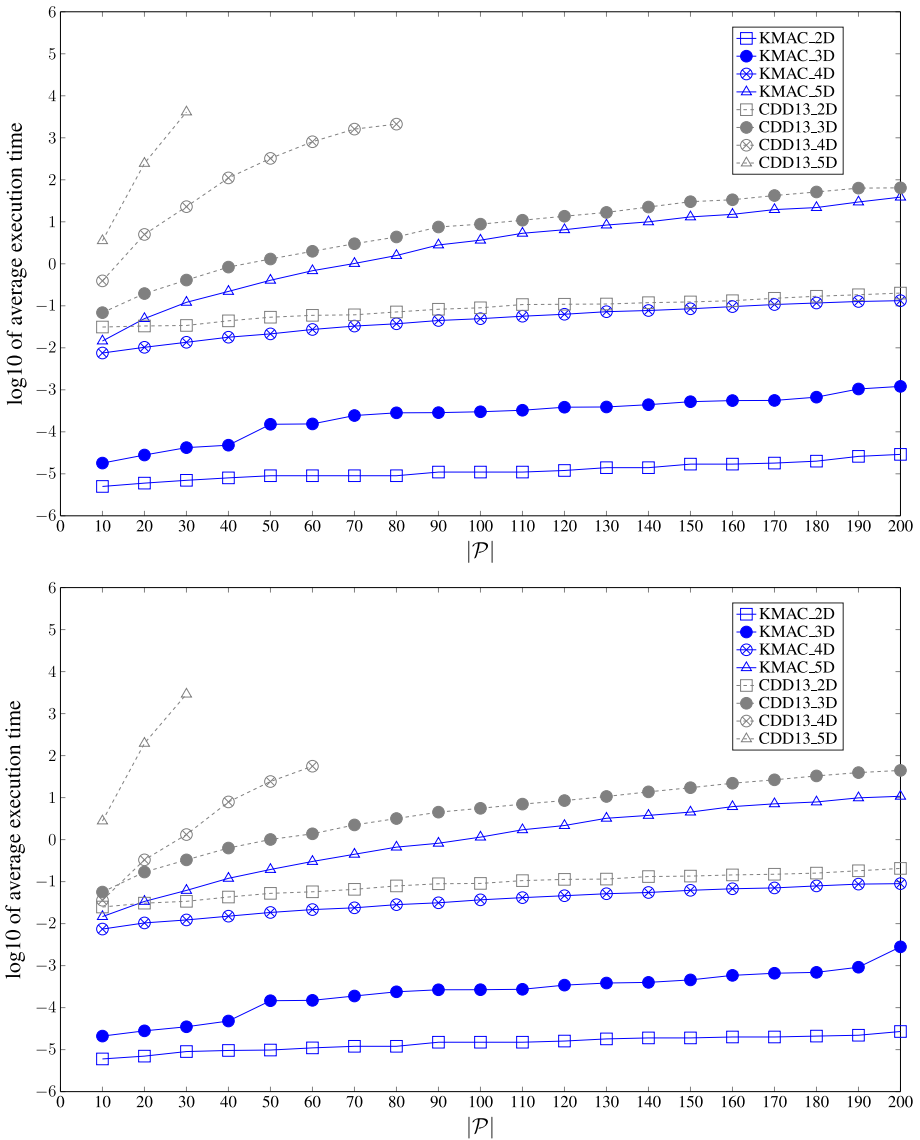[8] Using EHVI or PoI as the infill criterion in Algorithm 1.

Fig. 7 Speed comparison of EHVI calculation. Above: concave random Pareto-front set; Below: convex random Pareto-front set

even if the function evaluation budget of the EMOAs is increased to 2000, EHVI-MOBGO can still outperform the EMOAs in 6- and 18-dimensional search spaces. Among EHVI-MOBGO and PoI-MOBGO, EHVI-MOBGO outperforms PoI-MOBGO in most cases, but PoI-MOBGO yields better results on two and three (out of ten) test problems when $m = 6$ and $m = 18$, respectively. The reason is that the dominated space of the PoI is $\mathbf{y} \in$

Table 2 Algorithm parameter settings

	EHVI-MOBGO	PoI-MOBGO	NSGA-II	NSGA-III	SMS-EMOA
μ	30	30	30	/	30
λ	1	1	30		/
Evaluation	300	300	300/2000	300/2000	300/2000
divisions_outer	/	/	/	12	/
p_c	/	/	0.9		0.9
p_m	/	/	1/6		1/6
Platform	MATLAB	MATLAB	MATLAB	Python	MATLAB

Table 3 Reference Points

	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ7	MaF Problems
$\mathbf{r}$	(400,400,400) [5]	(2.5,2.5,2.5)[5]	(1500,1500,1500)	(2.5,2.5,2.5)	(11,11,11)	(1,1,10)	(5,5,5)

$(-\infty, \infty)^d\backslash\mathrm{dom}\,(\mathcal{P})^9$, which is bigger than that of the EHVI $\mathbf{y} \in (\mathbf{r}, \infty^d)\backslash\mathrm{dom}\,(\mathcal{P})$. Therefore, PoI performs better when searching for extreme non-dominated points. In other words, EHVI is a reference based infill criterion and it cannot indicate any improvement of an evaluated point in a discarded part of the non-dominated space, namely, $\mathbf{y} \in (-\infty^d, \mathbf{r})\backslash\mathrm{dom}\,(\mathcal{P})$.

6 Conclusions and outlook

This paper describes an efficient algorithm for EHVI calculation. It reviews and benchmarks recently proposed asymptotically optimal algorithms with $\Theta(n \log n)$ time complexity and generalizes them to higher dimensional cases with $d \geq 4$. By using the fast box decomposition techniques, which were recently developed by Dächert et al. [6] and Lacour et al. [25], a non-dominated space can be partitioned with only $\Theta(n^{\lfloor d/2 \rfloor})$ hyperboxes. The time complexity of our new EHVI computation algorithm scales linearly with the number of hyperboxes of arbitrary box decompositions. Unlike previous EHVI computation algorithms, the new algorithm does not require full grid partitionings with $\Theta(n^d)$ boxes. The new algorithm is, therefore, a significant improvement in terms of asymptotic time complexity. In addition, our benchmarks on random non-dominated sets show that this new algorithm is also many orders of magnitude faster for computations of typically sized problems where $n \leq 1000$.

This paper also compares the performance of MOBGO based algorithms with three other state-of-the-art EMOAs on 10 benchmark test problems. For budgets of function evaluations up to 300, MOBGO based algorithms can yield the Pareto-front approximation sets with higher HV values than that of EMOAs in both 6- and 18-dimensional search spaces. In most cases, EHVI-MOBGO yields better performance than PoI-MOBGO. However, PoI-MOBGO performs better than EHVI-MOBGO on up to 3 test problems in 6- and 18- dimensional search spaces. The reason is that the PoI can imply an improvement of an evaluated point in the whole non-dominated space, but the EHVI can only indicate the improvement in the non-

---

^{9} A reference point in Eq. (16) is defined as $\mathbf{r} = (-\infty, \infty, \infty)$ in this paper. For PoI-MOBGO, a reference point in Table 3 is only used to evaluate the experimental results.

Table 4 Empirical Comparisons w.r.t HV in the case of $m = 6$

Algorithm			MOBGO		EAs			EAs		
			EHVI	PoI	NSGA-II	NSGA-III	SMS-EMOA	NSGA-II	NSGA-III	SMS-EMOA
Eval.			300	300	300	300	300	2000	2000	2000
DTLZ1	HV	mean	**6.39587E+7**	6.33975E+7	6.34583E+7	6.35749E+7	6.15372E+7	6.39914E+7	**6.39984E+7**	6.39946E+7
		std.	**4.99505E+4**	2.30970E+5	2.22275E+5	1.75929E+5	9.64758E+5	7.78434E+3	**2.01767E+3**	5.68083E+3
DTLZ2	HV	mean	**1.50203E+1**	1.49975E+1	1.34232E+1	1.44291E+1	1.30529E+1	1.39295E+1	**1.49765E+1**	1.46605E+1
		std.	1.02673E−2	**5.15545E−3**	2.20199E-1	1.14701E-1	2.53151E-1	3.02290E-1	**5.18375E−3**	9.54696E−2
DTLZ3	HV	mean	**3.37451E+9**	3.36952E+9	3.35561E+9	3.36144E+9	3.28391E+9	3.37463E+9	**3.37499E+9**	3.37459E+9
		std.	**1.41276E+5**	4.75297E+6	8.29054E+6	3.54949E+6	2.51542E+7	2.75889E+5	**5.94791E+3**	2.20550E+5
DTLZ4	HV	mean	1.37964E+1	**1.44561E+1**	1.24064E+1	1.22748E+1	9.64531E+0	1.31946E+1	**1.42249E+1**	1.32491E+1
		std.	2.50980E-1	**1.60714E-1**	1.36386E+0	5.77561E-1	7.48353E-1	1.60385E+0	**7.42880E-1**	8.12735E-1
DTLZ5	HV	mean	1.31728E+3	**1.31883E+3**	1.31089E+3	1.31549E+3	1.30092E+3	1.31742E+3	**1.31881E+3**	1.31794E+3
		std.	3.69904E-1	**1.43867E−2**	5.50786E+0	1.46033E+0	9.43520E+0	3.57031E-1	**4.52836E−2**	3.43975E-1
DTLZ7	HV	mean	**5.08646E+0**	4.06894E+0	2.45578E+0	1.59954E-1	2.43627E+0	**4.96201E+0**	4.91579E+0	1.06682E-1
		std.	**7.59397E−2**	1.42676E-1	7.00249E-1	5.81027E-1	7.96762E-1	**1.14735E-1**	1.61886E-1	7.87267E-1
MaF1	HV	mean	**1.17445E+2**	1.17153E+2	1.14050E+2	1.05551E+2	1.11801E+2	**1.16459E+2**	1.16079E+2	1.13299E+2
		std.	**4.96840E−2**	6.87363E−2	1.05600E+0	1.45637E+0	1.54941E+0	**1.42786E-1**	4.47650E-1	1.19772E+0
MaF5	HV	mean	7.23424E+1	6.76919E+1	5.62684E+1	4.27219E+1	2.75989E+1	**8.61379E+1**	8.13403E+1	8.32843E+1
		std.	**1.32736E+1**	2.76597E+1	2.07352E+1	1.83150E+1	3.10755E+1	**1.98541E+1**	2.01754E+1	1.99339E+1
MaF12	HV	mean	8.35314E+1	8.20753E+1	7.05794E+1	5.99779E+1	7.07943E+1	9.17876E+1	8.33820E+1	**9.40144E+1**
		std.	**3.47560E+0**	8.10423E+0	4.83604E+0	3.53278E+0	4.09591E+0	8.85050E-1	3.21738E+0	**4.42730E-1**
MaF13	HV	mean	**1.24153E+2**	1.22594E+2	1.07438E+2	1.13489E+2	1.09264E+2	1.22595E+2	**1.23203E+2**	1.22414E+2
		std.	**3.02702E-1**	7.76629E-1	5.69950E+0	3.83089E+0	4.07713E+0	7.81655E-1	**5.29024E-1**	9.04852E-1

Table 5 Empirical comparisons w.r.t HV in the case of $m = 18$

Algorithm			MOBGO		EAs			EAs		
			EHVI	PoI	NSGA-II	NSGA-III	SMS-EMOA	NSGA-II	NSGA-III	SMS-EMOA
Eval.			300	300	300	300	300	2000	2000	2000
DTLZ1	HV	mean	**5.1288E+7**	2.7077E+7	2.2253E+7	2.3745E+7	1.8500E+7	5.9953E+7	**6.3317E+7**	5.9134E+7
		std	**3.0564E+6**	7.8473E+6	4.6439E+6	3.3495E+6	6.9285E+6	1.0099E+6	**2.5485E+5**	1.5575E+6
DTLZ2	HV	mean	1.2639E+1	**1.4239E+1**	8.5169E+0	1.2543E+1	9.2854E+0	1.0758E+1	**1.4859E+1**	1.2128E+1
		std	1.8477E+0	3.9014E−1	1.0017E+0	**3.8405E−1**	6.2299E−1	5.8787E−1	**3.6867E−2**	4.2366E−1
DTLZ3	HV	mean	**3.2911E+9**	2.6798E+9	2.1654E+9	2.6382E+9	2.2338E+9	3.2974E+9	**3.3655E+9**	3.2967E+9
		std	**1.1090E+7**	1.8750E+8	1.9180E+8	1.7309E+8	2.6704E+8	2.4809E+8	**3.8824E+6**	3.2490E+8
DTLZ4	HV	mean	8.3732E+0	**1.2054E+1**	6.5989E+0	7.5245E+0	7.1119E+0	7.9902E+0	**1.3305E+1**	9.2049E+0
		std	2.7638E+0	**1.1977E+0**	1.3956E+0	1.4967E+0	1.9694E+0	**1.7487E+0**	2.2255E+0	2.0953E+0
DTLZ5	HV	mean	**1.3128E+3**	1.3050E+3	1.2852E+3	1.3027E+3	1.2883E+3	1.3042E+3	**1.3176E+3**	1.3077E+3
		std	**1.5919E+0**	1.7630E+0	1.0882E+1	4.0443E+0	1.4893E+1	2.6870E+0	**8.5620E−1**	1.9797E+0
DTLZ7	HV	mean	**4.5911E+0**	1.0198E+0	0	0	0	0	**3.5493E+0**	3.8917E−3
		std	**6.1310E−1**	8.4290E−1	/	/	/	/	**2.4150E−1**	1.2307E+2
MaF1	HV	mean	1.0473E+2	1.0381E+2	6.7074E+1	9.9441E+1	8.8914E+1	9.6273E+1	**1.1563E+2**	1.0034E+2
		std	4.4597E+0	**2.4946E+0**	3.5408E+0	2.0910E+0	2.7782E+0	3.5475E+0	**3.6416E−1**	2.2847E+0
MaF5	HV	mean	**3.8302E+1**	2.9227E+1	6.7114E+0	2.4487E+1	1.0390E+1	2.0824E+1	**8.1290E+1**	1.6117E+1
		std	**1.4669E+1**	2.1667E+1	1.3320E+1	2.2095E+1	1.4802E+1	2.0555E+1	2.0022E+1	**1.9612E+1**
MaF12	HV	mean	**7.1659E+1**	6.3981E+1	4.2264E+1	5.6579E+1	5.1103E+1	5.4358E+1	**7.9808E+1**	6.5401E+1
		std	1.2235E+1	1.9204E+1	7.8510E+0	**3.4781E+0**	6.6979E+0	6.5554E+0	**3.2876E+0**	3.8870E+0
MaF13	HV	mean	9.5580E+1	**1.2260E+2**	3.1497E+1	6.9621E+1	3.9234E+1	5.4296E+1	**1.1731E+2**	6.8469E+1
		std	2.7625E+1	**1.1757E+0**	6.8408E+0	6.2208E+0	6.4776E+0	9.4731E+0	**1.9190E+0**	7.4302E+0

dominated space bounded by a reference point. A remedy to the EHVI's disadvantage can be achieved by setting a large reference point or using the dynamic reference point strategy. However, the selected reference point must not be too large. Otherwise, EHVI at any evaluated points would be similar, even identical, because of the insufficient numerical stability of the computations involved.

For future research, it is recommended to further investigate on reference-free computation of EHVI. Moreover, it is still an open question of how to obtain fast EHVI calculations for a larger number of objective functions. Although it is conjectured that in the worst case time complexity will increase superpolynomially with the number of objectives, a better average case time complexity could be obtained by adopting concepts from recently proposed divide-and-conquer algorithms for computing the hypervolume indicator [20,31].

References

1. Beume, N., Naujoks, B., Emmerich, M.: SMS–EMOA: multiobjective selection based on dominated hypervolume. Eur. J. Oper. Res. **181**(3), 1653–1669 (2007)
2. Cheng, R., Li, M., Tian, Y., Zhang, X., Yang, S., Jin, Y., Yao, X.: A benchmark test suite for evolutionary many-objective optimization. Complex Intell. Syst. **3**(1), 67–81 (2017). https://doi.org/10.1007/s40747-017-0039-7
3. Chugh, T.: Handling expensive multiobjective optimization problems with evolutionary algorithms. Ph.D. thesis, Faculty of Information Technology, University of Jyväskylä (2017)
4. Coello Coello, C.A.: Evolutionary multi-objective optimization: Basic concepts and some applications in pattern recognition. In: J.F. Martínez-Trinidad, J.A. Carrasco-Ochoa, C. Ben-Youssef Brants, E.R. Hancock (eds.) Proceedings of the Third Mexican Conference on Pattern Recognition, pp. 22–33. Springer, Berlin (2013). https://doi.org/10.1007/978-3-642-21587-2_3
5. Couckuyt, I., Deschrijver, D., Dhaene, T.: Fast calculation of multiobjective probability of improvement and expected improvement criteria for Pareto optimization. J. Glob. Optim. **60**(3), 575–594 (2014)
6. Dächert, K., Klamroth, K., Lacour, R., Vanderpooten, D.: Efficient computation of the search region in multi-objective optimization. Eur. J. Oper. Res. **260**(3), 841–855 (2017)
7. Deb, K., Jain, H.: An evolutionary many-objective optimization algorithm using reference-point-based nondominated sorting approach, part I: solving problems with box constraints. IEEE Trans. Evol. Comput. **18**(4), 577–601 (2014). https://doi.org/10.1109/TEVC.2013.2281535
8. Deb, K., Pratap, A., Agarwal, S., Meyarivan, T.: A fast and elitist multiobjective genetic algorithm: NSGA-II. IEEE Trans. Evol. Comput. **6**(2), 182–197 (2002). https://doi.org/10.1109/4235.996017
9. Deb, K., Thiele, L., Laumanns, M., Zitzler, E.: Scalable multi-objective optimization test problems. In: Proceedings of the 2002 Congress on Evolutionary Computation. CEC'02 (Cat. No.02TH8600), vol. 1, pp. 825–830 (2002). https://doi.org/10.1109/CEC.2002.1007032
10. Emmerich, M.: Single-and multi-objective evolutionary design optimization assisted by Gaussian random field metamodels. Ph.D. thesis, Fachbereich Informatik, Chair of Systems Analysis, University of Dortmund (2005)
11. Emmerich, M., Deutz, A., Klinkenberg, J.W.: The computation of the expected improvement in dominated hypervolume of Pareto front approximations. Technical Report, Leiden University, vol. 34 (2008)
12. Emmerich, M., Deutz, A.H., Klinkenberg, J.W.: Hypervolume-based expected improvement: monotonicity properties and exact computation. In: 2011 IEEE Congress on Evolutionary Computation (CEC), pp. 2147–2154. IEEE (2011)
13. Emmerich, M., Giannakoglou, K.C., Naujoks, B.: Single-and multiobjective evolutionary optimization assisted by Gaussian random field metamodels. IEEE Trans. Evol. Comput. **10**(4), 421–439 (2006)

14. Emmerich, M., Yang, K., Deutz, A., Wang, H., Fonseca, C.M.: A multicriteria generalization of Bayesian global optimization. In: Pardalos, P.M., Zhigljavsky, A., Žilinskas, J. (eds.) Advances in Stochastic and Deterministic Global Optimization, pp. 229–243. Springer, Berlin (2016)

15. Emmerich, M.T.M., Fonseca, C.M.: Computing hypervolume contributions in low dimensions: asymptotically optimal algorithm and complexity results. In: Takahashi, R.H.C., Deb, K., Wanner, E.F., Greco, S. (eds.) International Conference on Evolutionary Multi-criterion Optimization, pp. 121–135. Springer, Berlin (2011)

16. Gaida, D.: Dynamic real-time substrate feed optimization of anaerobic co-digestion plants. Ph.D. thesis, Leiden Institute of Advanced Computer Science (LIACS), Faculty of Science, Leiden University (2014)

17. Hadka, D.: Platypus-multiobjective optimization in Python. https://github.com/Project-Platypus/Platypus (2015). Accessed Aug 2017

18. Hansen, N.: Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In: Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers (GECOO), pp. 2389–2396. ACM, New York (2009). https://doi.org/10.1145/1570256.1570333

19. Hupkens, I., Deutz, A., Yang, K., Emmerich, M.: Faster exact algorithms for computing expected hypervolume improvement. In: A. Gaspar-Cunha, C. Henggeler Antunes, C.C. Coello (eds.) International Conference on Evolutionary Multi-Criterion Optimization, pp. 65–79. Springer, Cham (2015)

20. Jaszkiewicz, A.: Improved quick hypervolume algorithm. Comput. Oper. Res. **90**, 72–83 (2018). https://doi.org/10.1016/j.cor.2017.09.016

21. Jones, D.R., Schonlau, M., Welch, W.J.: Efficient global optimization of expensive black-box functions. J. Glob. Optim. **13**(4), 455–492 (1998)

22. Keane, A.J.: Statistical improvement criteria for use in multiobjective design optimization. Am. Inst. Aeronaut. Astronaut. (AIAA) J. **44**(4), 879–891 (2006)

23. Koch, P., Wagner, T., Emmerich, M.T., Bäck, T., Konen, W.: Efficient multi-criteria optimization on noisy machine learning problems. Appl. Soft Comput. **29**, 357–370 (2015). https://doi.org/10.1016/j.asoc.2015.01.005

24. Kushner, H.J.: A new method of locating the maximum point of an arbitrary multi-peak curve in the presence of noise. J. Basic Eng. **86**(1), 97–106 (1964)

25. Lacour, R., Klamroth, K., Fonseca, C.M.: A box decomposition algorithm to compute the hypervolume indicator. Comput. Oper. Res. **79**, 347–360 (2017)

26. Lagarias, J., Reeds, J., Wright, M., Wright, P.: Convergence properties of the Nelder–Mead simplex method in low dimensions. SIAM J. Optim. **9**(1), 112–147 (1998). https://doi.org/10.1137/S1052623496303470

27. Łaniewski-Wołłk, Ł., Obayashi, S., Jeong, S.: Development of expected improvement for multi-objective problems. In: Proceedings of 42nd Fluid Dynamics Conference/Aerospace Numerical, Simulation Symposium (CD ROM). Varna, Bulgaria (2010)

28. Li, R., Emmerich, M.T.M., Eggermont, J., Bovenkamp, E.G.P., Bäck, T., Dijkstra, J., Reiber, J.H.C.: Metamodel-assisted mixed integer evolution strategies and their application to intravascular ultrasound image analysis. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp. 2764–2771 (2008). https://doi.org/10.1109/CEC.2008.4631169

29. McKay, M.D., Beckman, R.J., Conover, W.J.: Comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics **21**(2), 239–245 (1979). https://doi.org/10.1080/00401706.1979.10489755

30. Mockus, J., Tiešis, V., Žilinskas, A.: The application of Bayesian methods for seeking the extremum. In: Dixon, L., Szegö, G. (eds.) Towards Global Optimization, vol. 2, pp. 117–131. North-Holland, Amsterdam (1978)

31. Russo, L.M.S., Francisco, A.P.: Quick hypervolume. IEEE Trans. Evol. Comput. **18**(4), 481–502 (2014). https://doi.org/10.1109/TEVC.2013.2281525

32. Shimoyama, K., Jeong, S., Obayashi, S.: Kriging-surrogate-based optimization considering expected hypervolume improvement in non-constrained many-objective test problems. In: 2013 IEEE Congress on Evolutionary Computation, pp. 658–665 (2013). https://doi.org/10.1109/CEC.2013.6557631

33. Shimoyama, K., Sato, K., Jeong, S., Obayashi, S.: Updating Kriging surrogate models based on the hypervolume indicator in multi-objective optimization. J. Mech. Des. **135**(9), 094503 (2013)

34. Stuckman, B.E.: A global search method for optimizing nonlinear systems. IEEE Trans. Syst. Man Cybern. **18**(6), 965–977 (1988)

35. Ulmer, H., Streichert, F., Zell, A.: Evolution strategies assisted by Gaussian processes with improved preselection criterion. In: The 2003 Congress on Evolutionary Computation, 2003. CEC '03, vol. 1, pp. 692–699 (2003). https://doi.org/10.1109/CEC.2003.1299643

36. Wagner, T., Emmerich, M., Deutz, A., Ponweiser, W.: On expected-improvement criteria for model-based multi-objective optimization. In: Schaefer, R., Cotta, C., Kołodziej, J., Rudolph, G. (eds.) International Conference on Parallel Problem Solving from Nature-PPSN XI, pp. 718–727. Springer, Berlin (2010)
37. Yang, K., Deutz, A., Yang, Z., Bäck, T., Emmerich, M.: Truncated expected hypervolume improvement: exact computation and application. In: 2016 IEEE Congress on Evolutionary Computation (CEC), pp. 4350–4357. IEEE (2016). https://doi.org/10.1109/CEC.2016.7744343
38. Yang, K., Emmerich, M., Deutz, A., Fonseca, C.M.: Computing 3-D expected hypervolume improvement and related integrals in asymptotically optimal time. In: Trautmann, H., Rudolph, G., Klamroth, K., Schütze, O., Wiecek Margaret Jin, Y., Grimme, C. (eds.) International Conference on Evolutionary Multi-criterion Optimization, pp. 685–700. Springer, Cham (2017)
39. Yang, K.: Multi-objective Bayesian global optimization for continuous problems and applications. Ph.D. thesis, Leiden Institute of Advanced Computer Science (LIACS), Faculty of Science, Leiden University (2017)
40. Yang, K., Gaida, D., Bäck, T., Emmerich, M.: Expected hypervolume improvement algorithm for PID controller tuning and the multiobjective dynamical control of a biogas plant. In: 2015 IEEE Congress on Evolutionary Computation (CEC), pp. 1934–1942. IEEE (2015). https://doi.org/10.1109/CEC.2015.7257122
41. Zaefferer, M., Bartz-Beielstein, T., Naujoks, B., Wagner, T., Emmerich, M.: A case study on multi-criteria optimization of an event detection software under limited budgets. In: Purshouse, R.C., Fleming, P.J., Fonseca, C.M., Greco, S., Shaw, J. (eds.) International Conference on Evolutionary Multi-criterion Optimization, pp. 756–770. Springer, Berlin (2013)
42. Zitzler, E., Thiele, L.: Multiobjective evolutionary algorithms: a comparative case study and the strength Pareto approach. IEEE Trans. Evol. Comput. **3**(4), 257–271 (1999)
43. Zitzler, E., Thiele, L., Laumanns, M., Fonseca, C.M., Da Fonseca, V.G.: Performance assessment of multiobjective optimizers: an analysis and review. IEEE Trans. Evol. Comput. **7**(2), 117–132 (2003)

Affiliations

Kaifeng Yang[1] · **Michael Emmerich[1]** · **André Deutz[1]** · **Thomas Bäck[1]**

Michael Emmerich
m.t.m.emmerich@liacs.leidenuniv.nl

André Deutz
a.h.deutz@liacs.leidenuniv.nl

Thomas Bäck
t.h.w.baeck@liacs.leidenuniv.nl

[1]   LIACS, Leiden University, Niels Bohrweg 1, #164, 2333 CA Leiden, The Netherlands