Multi-objective Bayesian global optimization for continuous problems and applications

Yang, K.

Chapter 3

Efficient EHVI Calculation

In chapter 2, the basic structure of MOBGO was introduced and some common infill criteria were mentioned. Among these infill criteria in MOO, EHVI outperforms other criteria for its inherent ability to balance exploitation and exploration [61]. However, EHVI is seldom applied in real application because the computational complexity of EHVI is very expensive[1]. In this chapter, an asymptotically optimal algorithm for the computation of the exact expected hypervolume improvement (EHVI) is proposed, based on partitioning the integration volume into a set of axis-parallel slices. Theoretically, the upper bound time complexities are improved from previously $O(n^3 \log n)$ and $O(n^4 \log n)$ in [60], for two and three objectives problems respectively, to now $O(n \log n)$ for both two and three objective problems, which is asymptotically optimal, as we have proved. This scheme is also generalized in the case of high dimension in this chapter.

This chapter mainly contributes to the thesis by introducing the state-of-the-art EHVI calculation methods. This chapter is structured as follows: Section 3.1 provides the definition of EHVI; Section 3.2 explains the reason why EHVI is an important criterion in MOO and introduces some current algorithms to calculate EHVI; Section 3.3 provides the partitioning methods for non-dominated space; Section 3.4 shows the final formula expression of EHVI, based on the partitioning method described in Section 3.3; Section 3.6 shows the EHVI calculation speed comparison and empirical experimental results on benchmarks, with respect to state-of-the-art multi-objective optimization algorithms.

[1]The computational complexity of an infill criterion is crucial in multi-objective Bayesian global optimization, because this criterion needs to be called frequently during the execution of such an algorithm.

3.1 EHVI Definition

Definition 3.1 (Δ function (see also [2])) *For a given vector of objective function values, $\mathbf{y} \in \mathbb{R}^d$, $\Delta(\mathbf{y}, \mathcal{P}, \mathrm{r})$ is the subset of the vectors in $\mathbb{R}^d$ which are exclusively dominated by a vector $\mathbf{y}$ and not by elements in $\mathcal{P}$ and that dominate the reference point, in symbols*

$$\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r}) = \lambda_d\{\mathbf{z} \in \mathbb{R} \mid \mathbf{y} \prec \mathbf{z} \text{ and } \mathbf{z} \prec \mathbf{r} \text{ and } \nexists \mathbf{q} \in \mathcal{P} : \mathbf{q} \prec \mathbf{z}\} \tag{1-1}$$

For the simplicity, the notation $\Delta(\mathbf{y})$ will be used to express $\Delta(\mathbf{y}, \mathcal{P}, \mathbf{r})$ in this paper.

EHVI is a generalization of EI for multi-objective cases, and it is based on the theory of the HV. Similar to EI, the calculation of EHVI is based on the predictions in the Gaussian random field. EHVI measures how much hypervolume improvement could be achieved by evaluating the new point, considering the uncertainty of the prediction. It is defined as:

Definition 3.2 (Expected Hypervolume Improvement) [1] *Given parameters of the multivariate predictive distribution $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and the Pareto-front approximation $\mathcal{P}$ the expected hypervolume improvement (EHVI) is defined as:*

$$EHVI(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) := \int_{\mathbb{R}^d} HVI(\mathcal{P}, \mathbf{y}) \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \tag{1-2}$$

where $PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}$ is the multivariate independent normal distribution for mean values $\boldsymbol{\mu} \in \mathbb{R}^d$, and standard deviations $\boldsymbol{\sigma} \in \mathbb{R}^d_+$.

Example 3.1 *An illustration of the 2-D EHVI is shown in Figure 4.1. The light gray area is the dominated subspace of $\mathcal{P} = \{\mathbf{y}^{(1)} = (3,1)^\top, \mathbf{y}^{(2)} = (2,1.5)^\top, \mathbf{y}^{(3)} = (1,2.5)^\top\}$ cut by the reference point $\mathbf{r} = (0,0)^\top$. The bivariate Gaussian distribution has the parameters $\mu_1 = 2$, $\mu_2 = 1.5$, $\sigma_1 = 0.7$, $\sigma_2 = 0.6$. The probability density function (PDF) 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. The variable y_1 stands for the f_1 value and y_2 for the f_2 value.*

[1] The prediction of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ depends on a Kriging model and a target point $\mathbf{x}$ in the search space. Explicitly, EHVI is dependent on the target point $\mathbf{x}$.

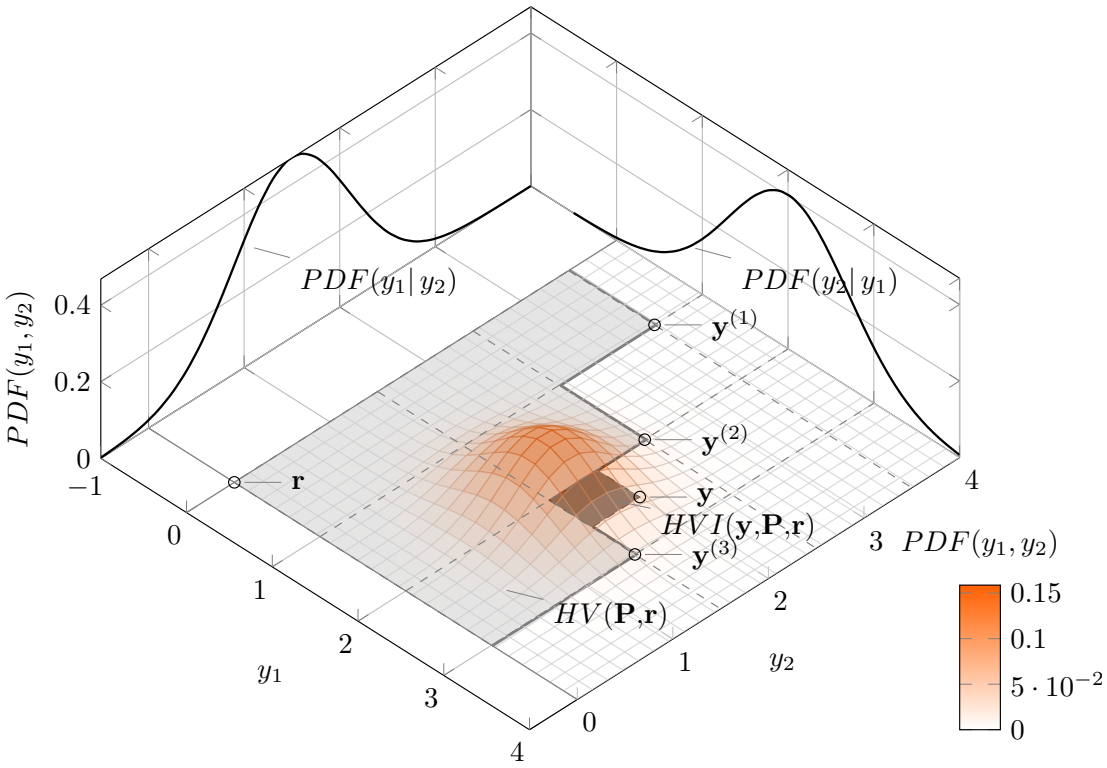

Figure 3.1: EHVI in 2-D (cf. Example 3.1).

For the convenience of expressing the formula of EHVI and EHVIG in later sections, it is useful to define a function we call Ψ_∞.

Definition 3.3 (Ψ_∞ function (see also [1])) *Let $\phi(s) = 1/\sqrt{2\pi}e^{-\frac{1}{2}s^2} (s \in \mathbb{R})$ denote the probability density function (PDF) of the standard normal distribution. Moreover, let $\Phi(s) = \frac{1}{2}\left(1 + erf\left(\frac{s}{\sqrt{2}}\right)\right)$ denote its cumulative probability distribution function (CDF), and erf is Gaussian error function. The general normal distribution with mean μ and standard deviation σ has as PDF, $\phi_{\mu,\sigma}(s) = \phi_{\mu,\sigma}(s) = \frac{1}{\sigma}\phi(\frac{s-\mu}{\sigma})$ 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:*

$$\begin{aligned}\Psi_\infty(a, b, \mu, \sigma) &= \int_b^\infty (z - a)\frac{1}{\sigma}\phi\left(\frac{z-\mu}{\sigma}\right) dz \\ &= \sigma\phi\left(\frac{b-\mu}{\sigma}\right) + (\mu - a)\left[1 - \Phi\left(\frac{b-\mu}{\sigma}\right)\right] \end{aligned} \tag{1-3}$$

3.2 State-of-the-art

In the context of MOBGO, an infill criterion is used to evaluate the improvement for a new point, as introduced in Chapter 2. A common criterion for a single-objective optimization problem is *Expected Improvement* (EI), which was firstly introduced by Mockus et al. [37] in 1978 and it exploits both the Kriging prediction and the variance to give a quantitative measure of the improvements for the points in the search space. Later, EI became more popular due to the work of Jones et al. [41], in which it servers as an infill criterion in the so-called Efficient Global Optimization (EGO) algorithm[1]. In each iteration, EGO evaluates the design point with maximal EI. Its convergence properties are discussed in [62], where a proof of global convergence under mild assumptions on the global covariance and the smoothness of the function is given. Roughly speaking, global convergence occurs due to the fact that EI rewards high variance and also high mean values.

Various generalizations of EI in the field of multi-objective optimization have been discussed in the literature, e.g., [45, 55, 61, 63, 64]. See also [47] for an overview. In the case of multiple objectives, it is possible to consider a Gaussian process model for each objective function separately and independently, resulting in a multivariate distribution with d mean values $\mu_i(x)$ and standard deviation $\sigma_i(x)$. A key question when generalizing the expected improvement is how to define improvement of a given Pareto-front approximation. In indicator-based multi-objective optimization, the performance of a Pareto-front approximation is assessed by a unary indicator, typically the *Hypervolume Indicator*, which allows a simple generalization of the *Expected Improvement* – the EHVI. EHVI is a straightforward generalization of the *single-objective expected improvement* and was proposed by Emmerich [56] in 2005. Since then, EHVI has been used in Evolutionary Algorithms for airfoil optimization [53] and quantum control [65]. It is also applied in multi-objective generalizations of Bayesian Global Optimization for applications, such as fluid dynamics [42], event controllers in wastewater treatment [44], efficient algorithm tuning [66], electrical component design [58], and bio-fuel power-generation [10]. In all of these applications, the bi-objective EHVI was used. Due to its high computation time for problems which contains three and more objectives, it is not recommended to use EHVI as an infill criterion in such cases. Fast, but imprecise, alternatives were sought [67].

The expected hypervolume improvement (EHVI) is the expected value of the increment of the hypervolume indicator given a Pareto-front approximation and

[1] Efficient Global Optimization is another name of Bayesian Global Optimization.

a predictive multivariate Gaussian distribution predicting the outcome at a new point. When compared to other criteria, EHVI leads to better convergence towards the true Pareto front, and to a higher diversity of the Pareto-front approximation set [6, 58, 67, 68]. However, the calculation of EHVI itself has so far been time-consuming [44, 47, 69], even in the case of two dimensions. It still remains unknown whether the integration algorithms used in the literature achieved optimal performance. Hence, it is important to study whether, and to what extent, the computational efficiency of the exact computation of the EHVI can be further improved. In addition, EHVI is called multiple times in every iteration. For the above reasons, a fast algorithm for computing the EHVI is crucial.

The first method suggested for EHVI calculation was Monte Carlo integration and it was first proposed by Emmerich in [56] and [53]. This method is simple and straightforward. However, the accuracy of EHVI highly depends on the number of the iterations. The first exact EHVI calculation algorithm was derived by Emmerich et al. [60], with the computational complexity $O(n^3 \log n)$ and $O(n^4 \log n)$ in the cases of 2-D and 3-D, respectively. Couckuyt et al. introduced a faster exact EHVI calculation algorithm for $d > 2$ in [58], but did not provide a detailed complexity analysis. Recently, Hupkens et al. reduced the time complexity to $O(n^2)$ and $O(n^3)$ [1] for two- and three-dimensional cases, respectively. These algorithms further improved the practical efficiency of EHVI on test data in comparison to [58]. More recently, Emmerich et al. proposed an asymptotically optimal algorithm for the bi-objective case with time complexity $O(n \log n)$ [2], where n is the number of non-dominated points in the archive. So far the best known bounds for the time complexity of exact computations have been $O(n \log n)$ for $d = 2$, and $O(n^3)$ for $d = 3$. It is notable that the number of transcendental function evaluations, such as erf and exp, scales only linearly in n in the algorithm presented in [1]. A lower bound of $\Omega(n \log n)$ is provided for a given approximation set of size n. However, it makes sense to assume that non-dominated points are sorted in the first coordinate. In that case, as will be shown, a lower bound of $\Omega(n)$ still holds.

3.3 Non-dominated Space Partitioning Algorithm

3.3.1 Low Dimensional case

2-D case: Suppose $\mathbf{y} = \mathbf{y}^{(1)}, \ldots, \mathbf{y}^{(n)}$ and $d = 2$, then the integration area (non-dominated area) can be divided into $n+1$ disjoint integration slices ($S_2^{(i)}, i =$

$1, \ldots, n+1$) by drawing parallel to y_2-axis lines at each element in $\mathbf{y}$, as indicated in Figure 3.2 (left). 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 stripes formally, augment $\mathcal{P}$ with two sentinels: $\mathbf{y}^{(0)} = (r_1, \infty)$ and $\mathbf{y}^{(n+1)} = (\infty, r_2)$. Then, the integration slices for 2-D case are now defined by:

$$\begin{aligned} S_2^{(i)} = (\mathbf{l}_2^{(i)}, \mathbf{u}_2^{(i)}) &= \left((l_1^{(i)}, l_2^{(i)})^T, (u_1^{(i)}, u_2^{(i)})^T\right) \\ &= \left((y_1^{(i-1)}, y_2^{(i)}, (y_1^{(i)}, \infty)^T\right) \qquad i = 1, \ldots, N_2 \end{aligned} \tag{3-4}$$

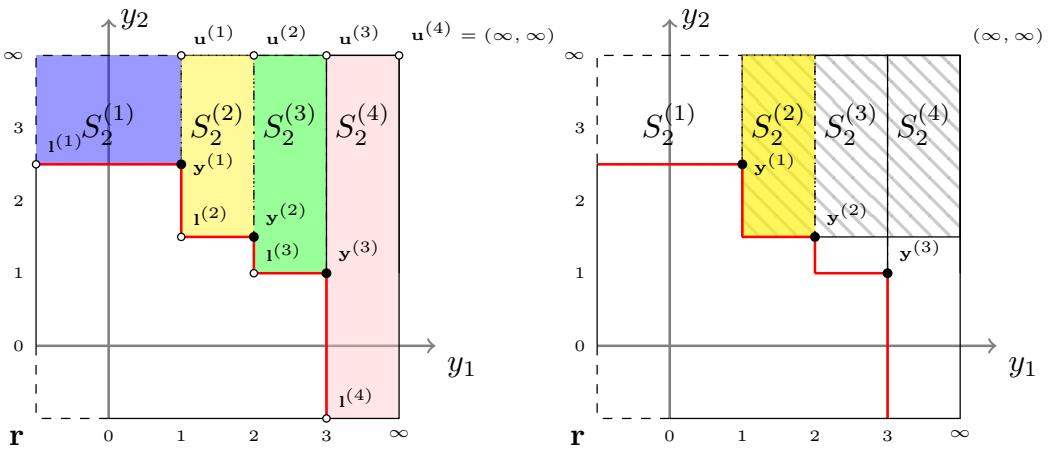

Figure 3.2: Left: Partitioning of the integration region into stripes. Right: New partitioning of the reduced integration region after the first iteration of the algorithm.

For the 2-D case, it is straightforward that the number of integration slices N_2 is $n + 1$.

3-D case: Similar to the 2-D partitioning method, in the 3-D case, each integration slice can also 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:

$$S_3^{(i)} = (\mathbf{l}_3^{(i)}, \mathbf{u}_3^{(i)}) = \left((l_1^{(i)}, l_2^{(i)}, l_3^{(i)})^T, (u_1^{(i)}, u_2^{(i)}, \infty)^T\right) \qquad i = 1, \ldots, N_3 \tag{3-5}$$

Example 3.2 *An illustration of integration slices is shown in Figure 3.3. A Pareto front set is composed by* $n = 4$ *points* ($\mathbf{y}^{(1)} = (1, 3, 4), \mathbf{y}^{(2)} = (4, 2, 3), \mathbf{y}^{(3)} = (2, 4, 2)$ *and* $\mathbf{y}^{(4)} = (3, 5, 1)$*), and this Pareto front is shown in Figure 3.3 (a).*

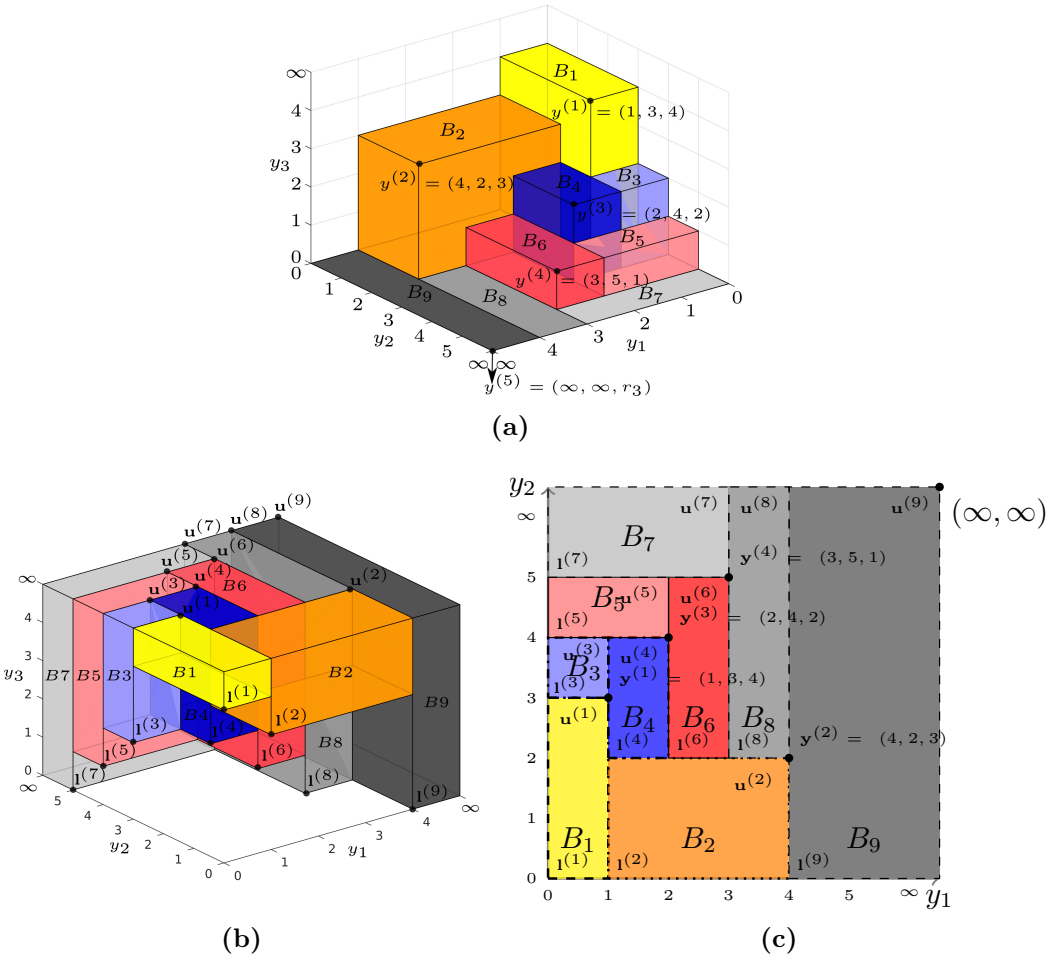

Figure 3.3: Figure (a): 3-D Pareto-front Approximation. Figure (b): Integration slices in 3-D. Figure (c): The projection of 3-D integration slices into the y_1y_2-plane, each slice can be described by lower bound and upper bound.

The added point $\mathbf{y}^{(n+1)}$ *is* $\mathbf{y}^{(5)} = (\infty, \infty, r_3)$. *The integration slices in the non-dominated space are represented by boxes in Figure 3.3 (b). Figure 3.3 (c) illustrates the projection onto the* y_1y_2*-plane with rectangle slices and* $\mathbf{l}, \mathbf{u}$. *The rectangular slices, which share the similar color but differ in opacity, represent integration slices with the same value of* y_3 *in their lower bound. For example, the lower bound of the 3-D integration slice* B_4 *is* $\mathbf{l}_3^{(4)} = (1, 2, 2)$, *and the upper bound of the slice is* $\mathbf{u}_3^{(4)} = (2, 4, \infty)$.

The basic idea of the efficient partitioning algorithm in 3-D non-dominated space

is that the transforming the 3-D Pareto front into 2-D Pareto front. This transforming consists of the following steps. Firstly, sort all the n elements $(\mathbf{y}^{(1)}, \cdots, \mathbf{y}^{(n)})$ in Pareto-front approximation set $\mathcal{P}$ in descending order by coordinate y_3. Secondly, set a new point $\mathbf{y}^{(n+1)} = (\infty, \infty, r_3)$. Thirdly, insert the element $\mathbf{y}^{(i)}$ $(i = 1, \cdots, n+1)$ into the 2-D Pareto-front set $\mathcal{P}'$ one by one using coordinate y_1 and y_2 value [1], and discard the dominated points $\mathbf{y}^{[d]}$ which are dominated by the new inserted point $\mathbf{y}^{(i)}$. During the third step, when a new point $\mathbf{y}^{(i)}$ is inserted, one and only one integration box is created on its below left side. When there exists a discarded point $\mathbf{y}^{[d]}$, one and only one integration box is created on its above right side. In other words, an integration box is only created when a new point $\mathbf{y}^{(i)}$ is inserted to the $\mathcal{P}'$ or a point is $\mathbf{y}^{[d]}$ is discarded from the $\mathcal{P}'$. The third step will not stop until the last point $\mathbf{y}^{(n+1)}$ is inserted. Since all the elements in $\mathcal{P}$ are dominated by $\mathbf{y}^{(n+1)}$ and no point can dominate it in the non-dominated space cut by a reference point $\mathbf{r}$, all the elements in $\mathcal{P}$ will be discarded and $\mathcal{P}'$ only consist of $\mathbf{y}^{(n+1)}$ in the last iteration of the third step. Then, the number of integration boxes for 3-D case is the sum of the number of the points that are inserted and the number of the points that are discarded, i.e., $N_3 = (n+1) + n = 2n + 1$.

Algorithm 4 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 $\mathbf{u}_3^{(i)}$) and how to compute the integrals for them. The partitioning algorithm is similar to the sweep line algorithm described in [20]. 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 $\mathbf{y}^{(i)}$, add this point to the AVL tree and find all the points ($\mathbf{y}^{(\mathrm{d}[1])}, \ldots,$ $\mathbf{y}^{(\mathrm{d}[s])}$) which are dominated by $\mathbf{y}^{(i)}$ in the y_1y_2-plane and discard them from the AVL tree. See Figure 3.4 for describing one such iteration. In each iteration, $s+1$ slices are created using coordinates of the points $\mathbf{y}^{(\mathrm{t})}$, $\mathbf{y}^{(\mathrm{d}[1])}, \ldots, \mathbf{y}^{(\mathrm{d}[s])}$, $\mathbf{y}^{(r)}$, and $\mathbf{y}^{(i)}$ as illustrated in Figure 3.4.

Here, the number of the integration slices for 3-D case N_3 is $2n + 1$, when all points are in general position (the coordinate of each point is different). Otherwise $2n + 1$ provides an upper bound for the obtained number of slices. The reason is as follows: In the algorithm each point $\mathbf{y}^{(i)}, i = 1, \ldots, n$ creates a slice, say slice $A^{(i)}$, when it is created and a slice, say slice $S_3^{(i)}$, when it is discarded from the AVL tree due to domination by another point, say $\mathbf{y}^{(s)}$, in the y_1y_2-plane.

The two slices are defined as follows $A^{(i)} = ((y^{(t)}, y_2^{(l2)}, y_3^{(i)}), (y_1^{(u1)}, y_2^{(i)}, \infty))$ whereas

[1] The coordinate value of y_3 is hidden for action of inserting and discarding, but y_3 value still exist.

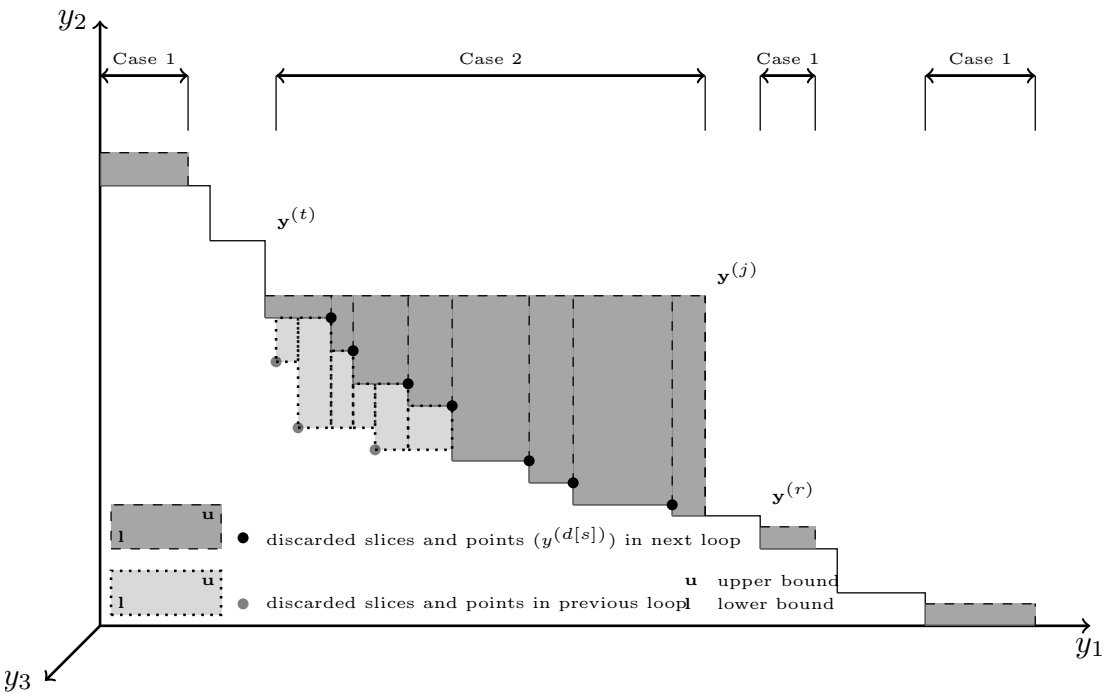

Figure 3.4: Boundary search for slices in 3-D case.

$y_2^{(l2)}$ is either $y_2^{(r)}$ if no points are dominated by $\mathbf{y}^{(i)}$ in the y_1y_2-plane or $y_2^{(d[1])}$, otherwise. Moreover, $S_3^{(i)} = ((y_1^{(i)}, y_2^{(r)}, y_3^{(s)}), (y_1^{(u)}, y_2^{(s)}, \infty))$, and $\mathbf{y}^{(u)}$ denotes either the right neighbour among the newly dominated points in the y_1y_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 for the slice that is created in the final iteration. In the final iteration one additional point $\mathbf{y}^{(n+1)} = (\infty, \infty, \infty)$ is added in the y_1y_2-plane. This point leads to the creation of a slice when it is added, but it adds only a single slice, because it is never discarded. Therefore, $2n + 1$ slices are created in total.

As opposed to previous techniques, which required grid decomposition of the non-dominated subspace into $O(n^3)$ integration slices, the new integration technique can make use of efficient partitioning of the dominated space into only $2n + 1$ axis-aligned integration slices. In practice, the new computation scheme will be of great advantage to making the EHVI and related integrals applicable in multi-objective optimization with three objectives, especially in Bayesian Optimization and surrogate-assisted multi-objective evolutionary algorithms.

Algorithm 4: Integration slices aquiring in 3-D Case

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

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

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

2: Initialize AVL tree T for 3-D points
Insert $\mathbf{y}^{(1)}$, $(\infty, r_2, \infty)^T$ and $(r_1, \infty, \infty)^T$ 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 $\mathbf{y^{(i)}}$ in x-coordinate (right neighbour);

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

9: d[1], $\cdots$, d[s]: indices of points dominated by $\mathbf{y^{(i)}}$ in y_1y_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: $S_3^{(n_b)}.l_1 = y_1^{(t)}, \quad S_3^{(n_b)}.l_2 = y_2^{(\mathrm{d[1]})}, \quad S_3^{(n_b)}.u_1 = y_1^{(d[1])}$;

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

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

20: **else**

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

22: $n_b = n_b + 1$;

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

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

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

3.3.2 High Dimensional case

In higher dimensional cases, the non-dominated space can be partitioned into axis aligned hyperboxes, similar to 3-D case. In d dimensional case, the hyperboxes can be denoted by $S_d^{(1)}, \ldots, S_d^{(i)}, \ldots, S_d^{N_d}$ with their lower bound $(\mathbf{l}^{(1)}, \ldots, \mathbf{l}^{(N_d)})$ and upper bound $(\mathbf{u}^{(1)}, \ldots, \mathbf{u}^{(N_d)})$. Here, N_d is the number of hyperboxes. The hyper-integral box $S_d^{(i)}$ is defined as:

$$S_d^{(i)} = (\mathbf{l}_d^{(i)}, \mathbf{u}_d^{(i)}) = \left((l_1^{(i)}, \cdots, l_d^{(i)})^T, (u_1^{(i)}, \cdots, \infty)^T\right) \qquad i = 1, \ldots, N_d \tag{3-6}$$

An efficient algorithm for partitioning high-dimensional non-dominated space is proposed in this chapter. This new proposed algorithm is based on two state-of-the-art algorithms DKLV17 [70] by Dächert et al. and LKF17 [71] by Lacour et al. Here, algorithm DKLV17 is an efficient algorithm to locate the local lower bound points in a dominated space for maximization problem, based on a specific neighborhood structure among local lower bounds. Meanwhile, LKF17 is an efficient algorithm to calculated hypervolume improvement by partitioning the dominated space. In other words, LKF17 is also efficient to partition the dominated space and provides the boundary information for each hyerbox in the dominated space.

Algorithm 5: Partitioning non-dominated space for high dimensional cases

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

Output: Hyperboxes S_d

1: Locate local lower bound points $\mathbf{L}$: $\mathbf{L} = DKLV17(\mathcal{P}, \mathbf{r})$;
2: Set new Pareto front $\mathcal{P}^{'}$ using $\mathbf{L}$: $\mathcal{P}^{'} = \mathbf{L}$;
3: Set a reference point $\mathbf{r}^{'}$: $\mathbf{r}^{'} = \{\infty\}^d$;
4: Get local lower bound points $\mathbf{L}^{'}$ and local upper bound points $\mathbf{U}^{'}$: $(\mathbf{L}^{'}, \mathbf{U}^{'}) = LKF17(\mathcal{P}^{'}, \mathbf{r}^{'})$;
5: $S_d = (\mathbf{L}^{'}, \mathbf{u}^{'})$;
6: Return S_d

The basic idea of the proposed algorithm to partition high-dimensional non-dominated space is transforming the problem of partitioning non-dominated space into the problem of partitioning the dominated space by introducing an intermediate Pareto-front approximation $\mathcal{P}^{'}$. This transforming is done by the following

steps. Suppose that we have a current Pareto-front approximation $\mathcal{P}$ and we want to partition the non-dominated space of $\mathcal{P}$. Firstly, DKLV17 is applied to locate the local lower bound points ($\mathbf{L}$) of $\mathcal{P}$ in dominated space. If we regard the local lower bound points $\mathbf{L}$ as a new Pareto-front approximation $\mathcal{P}'$, the dominated space of $\mathcal{P}'$ is exact the non-dominated space of $\mathcal{P}$. The pseudo code of partitioning non-dominated space for high dimensional cases is shown in Algorithm 5.

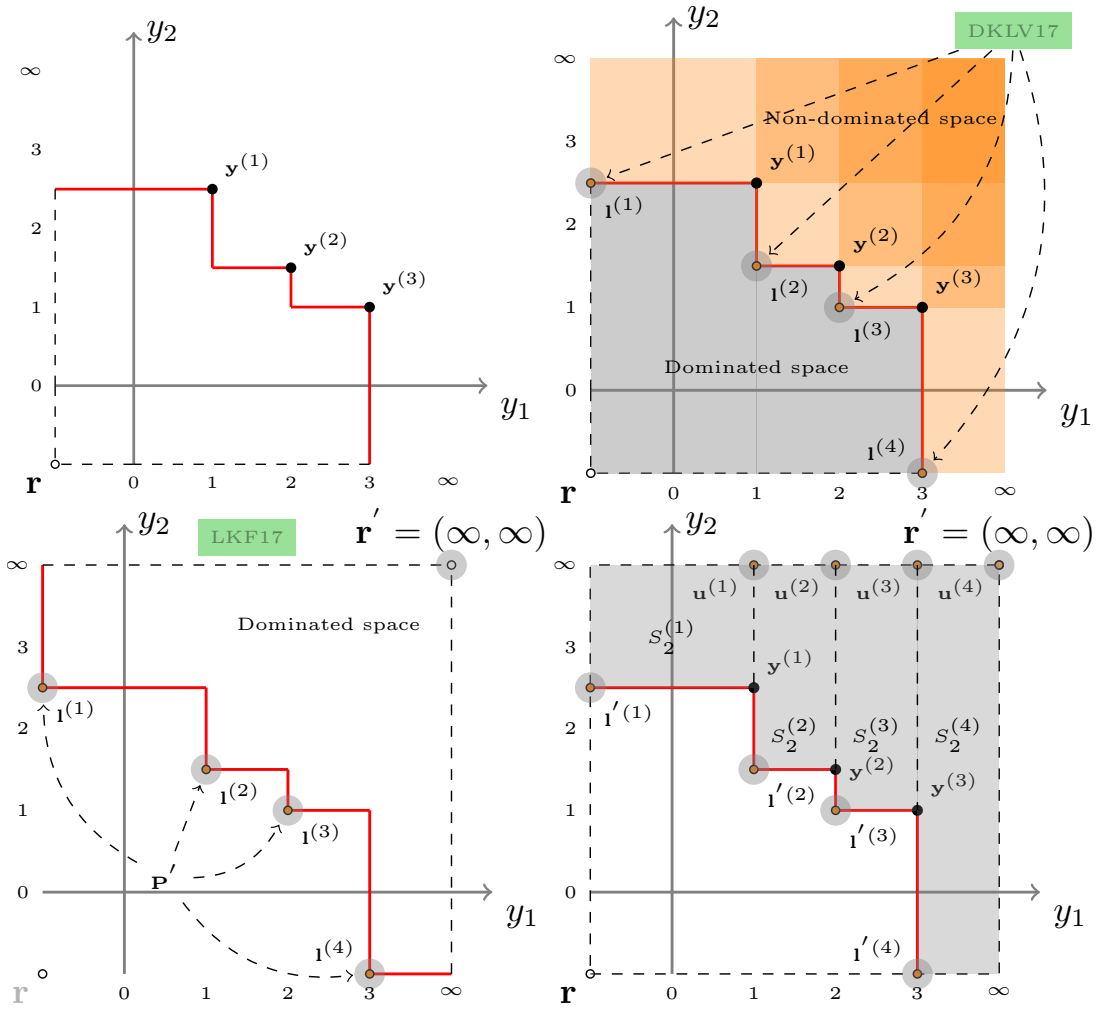

Figure 3.5: The illustration of partitioning non-dominated space for high dimensional case. Above left: Pareto-front approximation $\mathcal{P}$. Above right: Locating $\mathbf{L}$ points using DKLV17. Below left: Partition the dominated space of $\mathcal{P}'$ using LKF17. Below right: The partitioned non-dominated space of $\mathcal{P}$.

Example 3.3 *Figure 3.5 illustrates Algorithm 5. For the 2-D maximization case,*

suppose the Pareto-front approximation is $\mathcal{P}$*, which is composed by* $\mathbf{y}^{(1)} = (1, 2.5)$*,* $\mathbf{y}^{(2)} = (2, 1.5)$ *and* $\mathbf{y}^{(3)} = (3, 1)$*. The reference point is* $\mathbf{r} = (0, 0)$*, see Figure 3.5 (above left). Use DKLV17 to locate the local lower bound points* $\mathbf{l}$*, which consist of* $\mathbf{l}^{(1)} = (0, 2.5)$*,* $\mathbf{l}^{(2)} = (1, 1.5)$*,* $\mathbf{l}^{(3)} = (2, 1)$ *and* $\mathbf{l}^{(4)} = (3, 0)$*, see Figure 3.5 (above right). Regard all the local lower bound points* $\mathbf{l}$ *as the elements of a new Pareto-front approximation set* $\mathcal{P}^{'} = (\mathbf{l}^{(1)}, \cdots, \mathbf{l}^{(4)})$*. Set a new reference point* $\mathbf{r}^{'} = (\infty, \infty)$ *and utilize LKF17 to partition the dominated space of* $\mathcal{P}^{'}$*, considering minimization case, see Figure 3.5 (below left). Then the partitioned non-dominated space of* $\mathcal{P}$ *is actually the partitioned dominated space of* $\mathcal{P}^{'}$*, see Figure 3.5 (below right).*

3.4 Computing the integrals

Before introducing the EHVI formula deviation, it is useful to define an important function ϑ:

Definition 3.4 (ϑ function) *Let* $\phi(s) = 1/\sqrt{2\pi}e^{-\frac{1}{2}s^2} (s \in \mathbb{R})$ *denote the probability density function (PDF) of the standard normal distribution. Moreover, let* $\Phi(s) = \frac{1}{2}\left(1 + erf\left(\frac{s}{\sqrt{2}}\right)\right)$ *denote its cumulative probability distribution function (CDF), and erf is Gaussian error function. The general normal distribution with mean* μ *and standard deviation* σ *has as PDF,* $\phi_{\mu,\sigma}(s) = \phi_{\mu,\sigma}(s) = \frac{1}{\sigma}\phi(\frac{s-\mu}{\sigma})$ *and its CDF is* $\Phi_{\mu,\sigma}(s) = \Phi(\frac{s-\mu}{\sigma})$*, the integration box (or hyper-box)* B_i *consist of a lower bound point* $\mathbf{l}^{(i)}$ *and a upper bound point* $\mathbf{u}^{(i)}$*. Then the function* $\vartheta(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k)$ *is defined as:*

$$\begin{aligned}
\vartheta(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) :&= \int_{y_k=u_k^{(i)}}^{\infty} \lambda_1[B_i \cap \Delta(y_k)] \cdot PDF_{\mu_k,\sigma_k}(y_k)dy_k \\
&= \int_{y_k=u_k^{(i)}}^{\infty} (u_k^{(i)} - l_k^{(i)}) \cdot PDF_{\mu_k,\sigma_k}(y_k)dy_k \\
&= (u_k^{(i)} - l_k^{(i)}) \cdot (1 - \Phi(\frac{u_k^{(i)} - \mu_k}{\sigma_k})) \quad \textit{where } k = 1, \cdots, d-1
\end{aligned} \tag{4-7}$$

In the definition of ϑ function, $\lambda_1[B_i \cap \Delta(y_k)]$ is the *Hypervolume Improvement*

of the i^{th} integration box in dimension k, i.e., a 1-D *Hypervolume Improvement*. Considering the partitioning methods in Chapter 3.3, $\lambda_1[B_i \cap \Delta(y_k)] = |[l_k^{(i)}, u_k^{(i)}] \cap [l_k^{(i)}, y_k]| = \min\{u_k^{(i)}, y_k\} - l_k^{(i)}$, where $k = 1, \cdots, d$. The idea of introducing the ϑ function is that the improvement for $\int_{y_k=u_k^{(i)}}^{\infty} \lambda_1[B_i \cap \Delta(y_k)] \cdot PDF_{\mu_k,\sigma_k}(y_k)dy_k$, where $k = 1, \cdots, d-1$, is a constant, that is ϑ itself. This is very useful during the calculation process of EHVI, because ϑ function for each integration box B_i can be calculated once and reused to save calculation time.

3.4.1 2-D EHVI

According to the definition of the 2-D integration slice in Equation 3.3.1, the *Hypervolume Improvement* $\mathbf{y} \in \mathbb{R}^2$ for the 2-D case is:

$$\mathrm{HVI}_2(\mathcal{P}, \mathbf{y}, \mathbf{r}) = \sum_{i=1}^{N_2} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y}, \mathcal{P}, \mathbf{r})] \tag{4-8}$$

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

$$\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \int_{y_1=-\infty}^{\infty} \int_{y_2=-\infty}^{\infty} \sum_{i=1}^{N_2} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot \boldsymbol{PDF}_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \tag{4-9}$$

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 $S_2^{(i)}$. In other words, if and only if $\mathbf{y}$ is located in the rectangle with lower left corner $(l_1^{(i)}, l_2^{(i)})$ and upper right corner (∞, ∞). See Figure 3.2 (right) for an illustration. Therefore:

$$\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_2} \int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y} \tag{4-10}$$

In Equation (4-10), the summation is done after integration. This 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 *Hypervolume Improvement* $\lambda_1[S_2^{(i)} \cap \Delta(y_1)]$ differs in these two integration intervals. Then Equation (4-10)

is expressed by:

$$\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_2} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} + \quad (4\text{-}11)$$

$$\sum_{i=1}^{N_2} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \lambda_2[S_2^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \quad (4\text{-}12)$$

Here $\lambda_1[B_i \cap \Delta(y_k)]$ is the *Hypervolume Improvement* in dimension k, i.e., a 1-D *Hypervolume Improvement*. According to the definition of *Hypervolume Improvement*, $\lambda_1[B_i \cap \Delta(y_k)]$ is constant and it is $(u_1^{(i)} - l_1^{(i)})$. 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 Ψ_∞ function, then the Equation (4-11) and (4-12) are:

$$\mathrm{Cp.}(4-11) = \sum_{i=1}^{N_2} \Big(\Psi_\infty(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_\infty(l_1^{(i)}, u_1^{(i)}, \mu_1, \sigma_1) \Big) \cdot \Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) \quad (4\text{-}13)$$

$$\mathrm{Cp.}(4-12) = \sum_{i=1}^{N_2} \vartheta(l_1^{(i)}, u_1^{(i)}, \mu_1, \sigma_1) \cdot \Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) \quad (4\text{-}14)$$

3.4.2 3-D EHVI

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

$$\mathrm{HVI}_3(\mathcal{P}, \mathbf{y}, \mathbf{r}) = \sum_{i=1}^{N_3} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \quad (4\text{-}15)$$

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 Equation (1-2) can be replaced by HVI_3 in Equation (4-15):

$$\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) = \sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \quad (4\text{-}16)$$

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. Based on this division, Equation (4-16) can be expressed by:

$$\text{Cp.4} - 16 = \sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} + \quad (4\text{-}17)$$

$$\sum_{i=1}^{N_3} \int_{y_1=l_1^{(i)}}^{u_1^{(i)}} \int_{y_2=u_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} + \quad (4\text{-}18)$$

$$\sum_{i=1}^{N_3} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=l_2^{(i)}}^{u_2^{(i)}} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} + \quad (4\text{-}19)$$

$$\sum_{i=1}^{N_3} \int_{y_1=u_1^{(i)}}^{\infty} \int_{y_2=u_2^{(i)}}^{\infty} \int_{y_3=l_3^{(i)}}^{\infty} \lambda_3[S_3^{(i)} \cap \Delta(\mathbf{y})] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) d\mathbf{y} \quad (4\text{-}20)$$

Recalling the definition of ϑ function and calculation of $\lambda_1[B_i \cap \Delta(y_k)]$, component (4-17) can be written as follows:

$$\text{Cp.4} - 17 = \sum_{i=1}^{N_3} (\Psi_\infty(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_\infty(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1)) \cdot (\Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) - \Psi_\infty(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2)) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (4\text{-}21)$$

Similar to the derivation of Component (4-17), components (4-18), (4-19) and (4-20) can be written as follows:

$$\text{Cp.4} - 18 = \sum_{i=1}^{N_3} (\Psi_\infty(l_1^{(i)}, l_1^{(i)}, \mu_1, \sigma_1) - \Psi_\infty(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1)) \cdot \vartheta(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (4\text{-}22)$$

$$\text{Cp.4} - 19 = \sum_{i=1}^{N_3} \vartheta(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1) \cdot (\Psi_\infty(l_2^{(i)}, l_2^{(i)}, \mu_2, \sigma_2) - \Psi_\infty(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2)) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (4\text{-}23)$$

$$\text{Cp.4} - 20 = \sum_{i=1}^{N_3} \vartheta(l_1^{(i)}, u_1^{(i)}, \sigma_1, \mu_1) \cdot \vartheta(l_2^{(i)}, u_2^{(i)}, \sigma_2, \mu_2) \cdot \Psi_\infty(l_3^{(i)}, l_3^{(i)}, \mu_3, \sigma_3) \quad (4\text{-}24)$$

The final EHVI formula is the sum of Components (4-21), (4-22), (4-23) and (4-24).

3.4.3 High Dimensional Case

The interval of integration in each coordinate, except the last coordinate, can be divided into two parts: $[l, u]$ and $[u, \infty]$. Therefore, the equation for EHVI for each hyperboxes 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 numbers, and the Ψ function can be simplified by calculating function Φ and the improvement in these coordinate. For the last coordinate, there is no need to separate the interval, because the improvement in this coordinate $(\lambda_m[S_d^{(i)} \cap \Delta(y_m)])$ is a variable in $[l, \infty]$.

According to the definition of high dimensional integral boxes in Section 3.3.2, the formula of EHVI for a high dimensional case($d \geqslant 4$) can be calculated by the following equation:

$$
\begin{aligned}
\mathrm{EHVI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}, \mathbf{r}) &= \int_{y_{d}^{(i)}=l_{d}^{(i)}}^{\infty} \cdots \int_{y_{d}^{(i)}=l_{d}^{(i)}}^{\infty} \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{y}) d\boldsymbol{y} \\
&= \sum_{i=1}^{N_d} \left(\int_{y_1=l_1^{(i)}}^{y_1=u_1^{(i)}} \int_{y_1=u_1^{(i)}}^{\infty} \cdots \left(\int_{y_{d-1}=l_{d-1}^{(i)}}^{y_{d-1}=u_{d-1}^{(i)}} \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} \left(\int_{y_d=l_d^{(i)}}^{\infty} \cdot \right) \right) \right) \\
&\quad \cdot \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{y}) d\boldsymbol{y} \\
&= \sum_{i=1}^{N_d} \begin{pmatrix}
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} & \int_{y_d=l_d^{(i)}}^{\infty} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} & \int_{y_d=u_d^{(i)}}^{\infty} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=u_{d-1}^{(i)}}^{\infty} & \int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=u_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=l_{d-1}^{(i)}}^{\infty} & \int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=u_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_{d-1}=l_{d-1}^{(i)}}^{u_{d-1}^{(i)}} & \int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \\
\int_{y_1=l_1^{(i)}}^{u_1^{(i)}} & \int_{y_1=l_1^{(i)}}^{\infty} & \cdots & \int_{y_{d-1}=l_{d-1}^{(i)}}^{\infty} & \int_{y_{d-1}=l_{d-1}^{(i)}}^{\infty} & \int_{y_d=l_d^{(i)}}^{u_d^{(i)}}
\end{pmatrix} \\
&\quad \cdot \lambda_d[S_d^{(i)} \cap \Delta(y_1, \ldots, y_d)] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\boldsymbol{y}) d\boldsymbol{y}
\end{aligned}
$$

$$
\begin{aligned}
&\left(\begin{matrix}\Psi_\infty(l_1^{(i)},l_1^{(i)},\mu_1,\sigma_1)-\\ \Psi_\infty(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_2^{(i)},l_2^{(i)},\mu_2,\sigma_2)-\\ \Psi_\infty(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\end{matrix}\right)\cdots\left(\begin{matrix}\Psi_\infty(l_{d-2}^{(i)},l_{d-2}^{(i)},\mu_{d-2},\sigma_{d-2})-\\ \Psi_\infty(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_{d-1}^{(i)},l_{d-1}^{(i)},\mu_{d-1},\sigma_{d-1})-\\ \Psi_\infty(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\end{matrix}\right)\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\left(\begin{matrix}\Psi_\infty(l_1^{(i)},l_1^{(i)},\mu_1,\sigma_1)-\\ \Psi_\infty(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_2^{(i)},l_2^{(i)},\mu_2,\sigma_2)-\\ \Psi_\infty(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\end{matrix}\right)\cdots\left(\begin{matrix}\Psi_\infty(l_{d-2}^{(i)},l_{d-2}^{(i)},\mu_{d-2},\sigma_{d-2})-\\ \Psi_\infty(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\end{matrix}\right)\vartheta(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\left(\begin{matrix}\Psi_\infty(l_1^{(i)},l_1^{(i)},\mu_1,\sigma_1)-\\ \Psi_\infty(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_2^{(i)},l_2^{(i)},\mu_2,\sigma_2)-\\ \Psi_\infty(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\end{matrix}\right)\cdots\vartheta(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\left(\begin{matrix}\Psi_\infty(l_{d-1}^{(i)},l_{d-1}^{(i)},\mu_{d-1},\sigma_{d-1})-\\ \Psi_\infty(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\end{matrix}\right)\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\left(\begin{matrix}\Psi_\infty(l_1^{(i)},l_1^{(i)},\mu_1,\sigma_1)-\\ \Psi_\infty(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_2^{(i)},l_2^{(i)},\mu_2,\sigma_2)-\\ \Psi_\infty(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\end{matrix}\right)\cdots\vartheta(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\ \vartheta(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\vdots\qquad\qquad\vdots\qquad\qquad\ddots\qquad\qquad\vdots\qquad\qquad\vdots\\
&\left(\begin{matrix}\Psi_\infty(l_1^{(i)},l_1^{(i)},\mu_1,\sigma_1)-\\ \Psi_\infty(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\end{matrix}\right)\vartheta(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\cdots\left(\begin{matrix}\Psi_\infty(l_{d-2}^{(i)},l_{d-2}^{(i)},\mu_{d-2},\sigma_{d-2})-\\ \Psi_\infty(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_{d-1}^{(i)},l_{d-1}^{(i)},\mu_{d-1},\sigma_{d-1})-\\ \Psi_\infty(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\end{matrix}\right)\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\vartheta(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\left(\begin{matrix}\Psi_\infty(l_2^{(i)},l_2^{(i)},\mu_2,\sigma_2)-\\ \Psi_\infty(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\end{matrix}\right)\cdots\left(\begin{matrix}\Psi_\infty(l_{d-2}^{(i)},l_{d-2}^{(i)},\mu_{d-2},\sigma_{d-2})-\\ \Psi_\infty(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\end{matrix}\right)\left(\begin{matrix}\Psi_\infty(l_{d-1}^{(i)},l_{d-1}^{(i)},\mu_{d-1},\sigma_{d-1})-\\ \Psi_\infty(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\end{matrix}\right)\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\vartheta(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\ \vartheta(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\cdots\left(\begin{matrix}\Psi_\infty(l_{d-2}^{(i)},l_{d-2}^{(i)},\mu_{d-2},\sigma_{d-2})-\\ \Psi_\infty(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\end{matrix}\right)\vartheta(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\vartheta(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\ \vartheta(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\cdots\vartheta(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\left(\begin{matrix}\Psi_\infty(l_{d-1}^{(i)},l_{d-1}^{(i)},\mu_{d-1},\sigma_{d-1})-\\ \Psi_\infty(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\end{matrix}\right)\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\vartheta(l_1^{(i)},u_1^{(i)},\sigma_1,\mu_1)\ \vartheta(l_2^{(i)},u_2^{(i)},\sigma_2,\mu_2)\cdots\vartheta(l_{d-2}^{(i)},u_{d-2}^{(i)},\sigma_{d-2},\mu_{d-2})\ \vartheta(l_{d-1}^{(i)},u_{d-1}^{(i)},\sigma_{d-1},\mu_{d-1})\ \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)
\end{aligned}
$$

$$
=
$$

$$
\begin{aligned}
&\omega(i,1,0)\quad\omega(i,2,0)\quad\cdots\quad\omega(i,d-2,0)\quad\omega(i,d-1,0)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,0)\quad\omega(i,2,0)\quad\cdots\quad\omega(i,d-2,0)\quad\omega(i,d-1,1)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,0)\quad\omega(i,2,0)\quad\cdots\quad\omega(i,d-2,1)\quad\omega(i,d-1,0)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,0)\quad\omega(i,2,0)\quad\cdots\quad\omega(i,d-2,1)\quad\omega(i,d-1,1)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\qquad\vdots\qquad\qquad\vdots\qquad\qquad\ddots\qquad\qquad\vdots\qquad\qquad\vdots\qquad\qquad\vdots\ +\\
&\omega(i,1,1)\quad\omega(i,2,1)\quad\cdots\quad\omega(i,d-2,0)\quad\omega(i,d-1,0)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,1)\quad\omega(i,2,1)\quad\cdots\quad\omega(i,d-2,0)\quad\omega(i,d-1,1)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,1)\quad\omega(i,2,1)\quad\cdots\quad\omega(i,d-2,1)\quad\omega(i,d-1,0)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\ +\\
&\omega(i,1,1)\quad\omega(i,2,1)\quad\cdots\quad\omega(i,d-2,1)\quad\omega(i,d-1,1)\quad\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)
\end{aligned}
\tag{4-25}
$$

$$
\begin{aligned}
&\left(\begin{array}{cccc}
\omega(i,1,C_1^{(0)^2}) & \omega(i,2,C_2^{(0)^2}) & \cdots & \omega(i,d-2,C_{(d-2)}^{(0)^2}) \\
\omega(i,1,C_1^{(1)^2}) & \omega(i,2,C_2^{(1)^2}) & \cdots & \omega(i,d-2,C_{(d-2)}^{(1)^2}) \\
\omega(i,1,C_1^{(2)^2}) & \omega(i,2,C_2^{(2)^2}) & \cdots & \omega(i,d-2,C_{(d-2)}^{(2)^2}) \\
\omega(i,1,C_1^{(3)^2}) & \omega(i,2,C_2^{(3)^2}) & \cdots & \omega(i,d-2,C_{(d-2)}^{(3)^2}) \\
\vdots & \vdots & \ddots & \vdots \\
\omega(i,1,C_1^{(2^{d-1}-4)^2}) & \omega(i,2,C_2^{(2^{d-1}-4)^2}) & \cdots & \omega(i,d-2,C_{d-2}^{(2^{d-1}-4)^2}) \\
\omega(i,1,C_1^{(2^{d-1}-3)^2}) & \omega(i,2,C_2^{(2^{d-1}-3)^2}) & \cdots & \omega(i,d-2,C_{d-2}^{(2^{d-1}-3)^2}) \\
\omega(i,1,C_1^{(2^{d-1}-2)^2}) & \omega(i,2,C_2^{(2^{d-1}-2)^2}) & \cdots & \omega(i,d-2,C_{d-2}^{(2^{d-1}-2)^2}) \\
\omega(i,1,C_1^{(2^{d-1}-1)^2}) & \omega(i,2,C_2^{(2^{d-1}-1)^2}) & \cdots & \omega(i,d-2,C_{d-2}^{(2^{d-1}-1)^2})
\end{array}\right. \\
&\quad \left.\begin{array}{cc}
\omega(i,d-1,C_{(d-1)}^{(0)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{(d-1)}^{(1)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{(d-1)}^{(2)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{(d-1)}^{(3)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\vdots & \vdots \\
\omega(i,d-1,C_{d-1}^{(2^{d-1}-4)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{d-1}^{(2^{d-1}-3)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{d-1}^{(2^{d-1}-2)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d) \\
\omega(i,d-1,C_{d-1}^{(2^{d-1}-1)^2}) & \Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)
\end{array}\right) \\
&= \sum_{i=1}^{N_d}\left(\sum_{j=0}^{2^{d-1}-1}\left(\prod_{k=1}^{d-1}\omega(i,k,C_k^{(j)^2})\cdot\Psi_\infty(l_d^{(i)},l_d^{(i)},\mu_d,\sigma_d)\right)\right)
\end{aligned}
\tag{4-26}
$$

In the component of (4-25), the integral of each dimension $\int_{y_k=l_k^{(i)}}^{u_k^{(i)}} \lambda_k[S_k^{(i)} \cap \Delta(y_1, \cdots, y_k)] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y}, 1 \leq k \leq d-1$ has two and only two different expressions (Ψ_∞ or ϑ), except for the last dimension, the expression of $\int_{y_d=l_d^{(i)}}^{u_d^{(i)}} \lambda_d[S_d^{(i)} \cap \Delta(y_1, \cdots, y_d)] \cdot PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y})d\mathbf{y}$ is always Ψ_∞. The final expression of EHVI is the sum of the combination of the product of each different expressions. Since the integral of dimension $1 \leq k \leq d-1$ has two different expressions and dimension $k = d$ has one expression, the final EHVI expression is the sum of 2^{d-1} terms.

In Equation (4-26), j_2 stands for the binary string of j in the integer system. The length of j_2 is $d-1$. $C_k^{(j)_2}$ is a binary bit and represents the k-th bit of j in binary string. For example, if $d = 5$, $j = 8$ and $k = 4$, then $j_2 = (1\ 0\ 0\ 0)$ and $C_k^{(j)_2} = 1$. Still in Equation (4-26), $\omega(i, k, C_k^{(j)_2})$ is defined as:

$$\omega(i,k,C_k^{(j)_2}) := \begin{cases} \Psi_\infty(l_k^{(i)}, l_k^{(i)}, \mu_k, \sigma_k) - \Psi_\infty(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) & \text{if } C_k^{(j)_2} = 0 \\ \vartheta_d(l_k^{(i)}, u_k^{(i)}, \sigma_k, \mu_k) & \text{if } C_k^{(j)_2} = 1 \end{cases} \tag{4-27}$$

Equation (4-26) shows how to calculate EHVI in the case of d objectives, and based on it, the runtime complexity of the proposed algorithm can be calculated. The exact EHVI is calculated by the sum of $\prod_{k=1}^{d-1} \omega(i, k, C_k^{(j)_2}) \cdot \Psi_\infty(l_d^{(i)}, l_d^{(i)}, \mu_d, \sigma_d)$ for 2^{d-1} times, which performs $O(1)$ for each hyperboxes calculation. Currently, the minimum number of hyperboxes $N_d, d \geq 4$ for a non-dominated space is still unknown. It is hypothesized by the author that N_d is equal to the number of the local lower bound points, which can be calculated by DKLV17 algorithm. The upper bound of runtime complexity is $O(n\tau)$, where $O(\tau)$ is the computation complexity of the search algorithm. For the case of $d = 1, 2, 3$, $O(\tau) \in O(\log n)$.

3.5 Other Related Criterion

Probability of Improvement is another important criterion in MOBGO, and it was first introduced by Stuckman [72], and then generalized by Emmerich et al. [53] to multi-objective optimization. It was also considered in MOBGO in Couckuyt et al. [58] and in Keane et al. [55]. It is defined as:

Definition 3.5 (Probability of Improvement) *Given parameters of the multivariate predictive distribution $\boldsymbol{\mu}$, $\boldsymbol{\sigma}$ and the Pareto-front approximation $\mathcal{P}$, the*

Probability of Improvement *(PoI) is defined as:*

$$PoI(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}) := \int_{\mathbb{R}^d} PDF_{\sigma,\mu}(\mathbf{y}) d\mathbf{y} \tag{5-28}$$

where $PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}$ *is the multivariate independent normal distribution for mean values* $\boldsymbol{\mu} \in \mathbb{R}^d$*, and standard deviations* $\boldsymbol{\sigma} \in \mathbb{R}^d_+$*.*

According to the partitioning method in Section 3.4.3, the calculation of PoI can be achieved by the following expression:

$$\begin{aligned} \mathrm{PoI}(\boldsymbol{\mu}, \boldsymbol{\sigma}, \mathcal{P}) &= \int_{y_1=-\infty}^{\infty} \cdots \int_{y_d=-\infty}^{\infty} PDF_{\boldsymbol{\mu},\boldsymbol{\sigma}}(\mathbf{y}) \mathrm{d}y_1 \ldots \mathrm{d}y_d \\ &= \sum_{i=1}^{N_d} \prod_{j=1}^{d} \Phi(\frac{u_j^{(i)} - \mu_j}{\sigma_j}) - \Phi(\frac{l_j^{(i)} - \mu_j}{\sigma_j}) \end{aligned} \tag{5-29}$$

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

3.6 Empirical Experiments

3.6.1 Speed Comparison

Three EHVI calculation algorithms, CDD13 [58], IRS_fast [1] and KMAC[1] [4, 5], are compared using the same benchmarks in this experiment. The test benchmarks from Emmerich and Fonseca [20] are used to generate Pareto-front sets. The Pareto-front sets and evaluated points were randomly generated based on convexSpherical and concaveSpherical functions.

The parameters: $\sigma_d = 2.5$, $\mu_d = 10, d = 2, \cdots, 5$ were used in the experiments. Pareto front sizes $|P| \in \{10, 20, \cdots, 200\}$ and the number of predictions (candidate points) Batch Size[1] $\in \{1\}$ are used together with σ_d and μ_d. Ten trials were

[1]This means that the integration space for PoI is unbounded and covers the entire non-dominated space.

[1]KMAC stands for the authors' given names.

[1]Batch Size means the number of the evaluated points under the same Pareto-front approximation set.

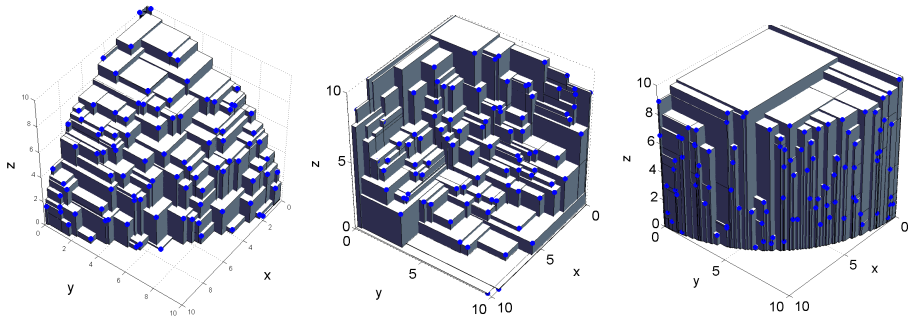

Figure 3.6: Randomly generated fronts of type convexSpherical, concave-Spherical, and cliff3D from [20] with $|P| = 100$ (left, middle and right).

randomly generated by the same parameters, and average runtimes (10 runs) for whole trails with the same parameters were computed. All the experiments were performed on the same computer and the hardware were: Intel(R) Xeon(R) CPU I7 3770 3.40GHz, RAM 16GB. The operating system was Ubuntu 16.04 LTS (64 bit), and software were gcc 4.9.2 with compiler flag -Ofast, except for SUMO code, MATLAB 8.4.0.150421 (R2014b), 64 bit. The experiments were set to halt if the algorithms could not finish the EHVI computation within 30 minutes. The results are shown in Figure 3.7.

The experimental results in Figure 3.7 show that KMAC is much faster than CDD13, especially when $|P|$ is increased. Sometimes, we need to calculate EHVI for multiple points under the same Pareto-front set and test whether the algorithms handle this problem efficiently. Since execution time would be increased dramatically when Batch Size is increased for high dimension ($d \geq 4$), here we only consider the 3-D case. Table 3.1 shows the experimental results with different Batch Size.

The parameters: $\boldsymbol{\sigma} = (2.5, 2.5, 2.5)$, $\boldsymbol{\mu} = (10, 10, 10)$ were used in the experiments. Pareto front sizes $|P| \in \{10, 100, 1000\}$ and the number of predictions (candidate points) or Batch Size $\in \{1, 10, 100, 1000\}$ are used together with $\boldsymbol{\sigma}$ and $\boldsymbol{\mu}$. Ten trials were randomly generated by the same parameters, and average runtimes (10 runs) for the whole 10 trails with the same parameters were computed. The data for 3-D case with $|P| = 100$ are visualized in Figure 3.6, and these figures are originally from [20]. All the experiments were run on the same hardware: Intel(R) Xeon(R) CPU E5-2667 v2 3.30GHz, RAM 48GB. The operating system was Ubuntu 12.04 LTS (64 bit), and the compiler was gcc 4.9.2 with compiler flag -Ofast, except for SUMO code, MATLAB 8.4.0.150421 (R2014b), 64 bit. The experiments were set to halt if the algorithms could not finish the EHVI

computation within 3 hours. The results are shown in Table 3.1.

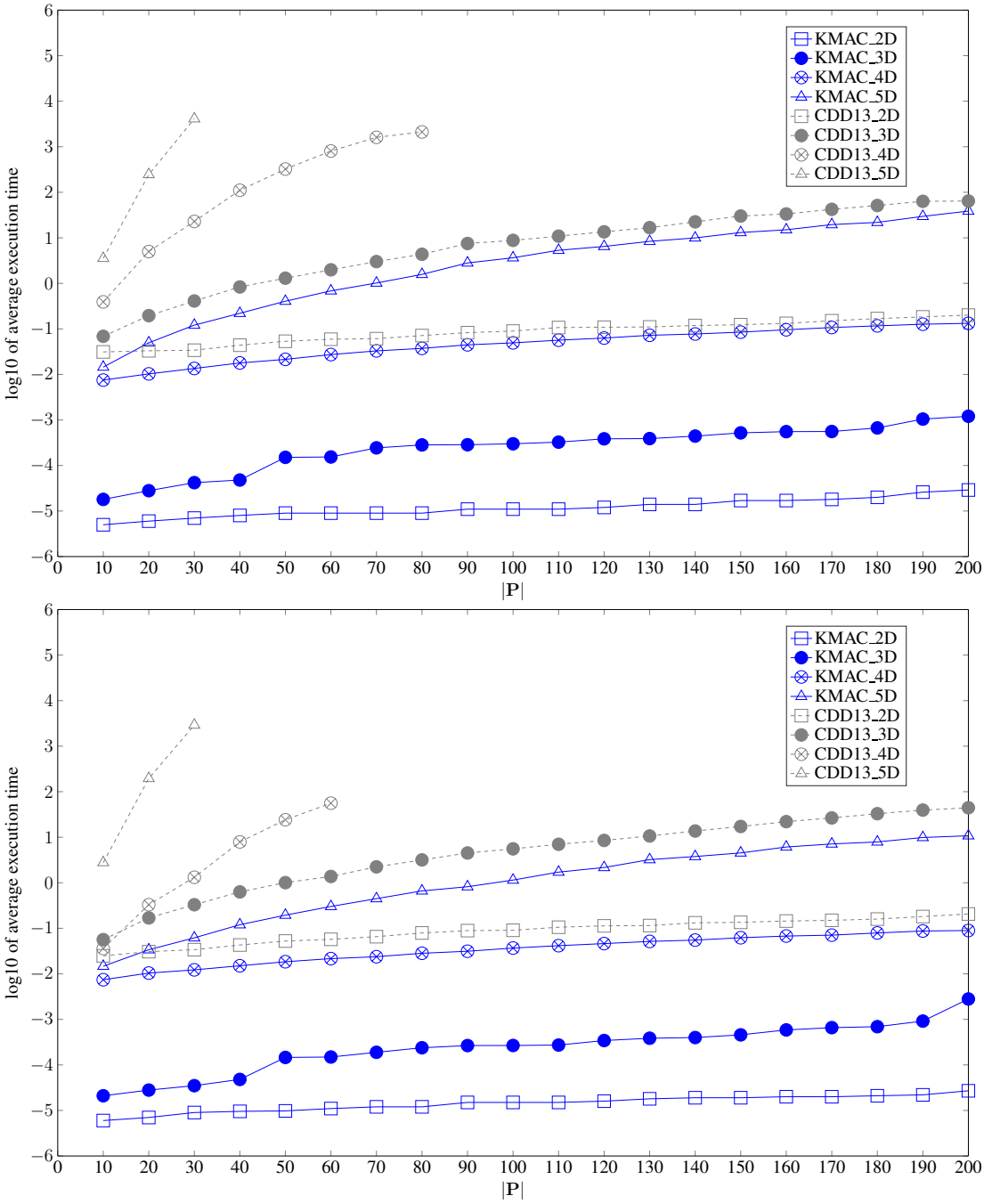

Figure 3.7: Speed comparison of EHVI calculation. Above: concave random Pareto front set; Below: convex random Pareto front set.

Table 3.1: Empirical comparisons of strategies for 3-D EHVI calculation.

Type	$\|P\|$	Batch Size	Time Average (s)		
			CDD13 [58]	IRS_fast [1]	KMAC
CONVEX	10	1	0.13785	0.00037	**0.00005**
CONVEX	10	10	0.14090	0.00056	**0.00021**
CONVEX	10	100	0.16500	0.00304	**0.00095**
CONVEX	10	1000	0.69104	0.02778	**0.00754**
CONVEX	100	1	13.97556	0.05337	**0.00038**
CONVEX	100	10	17.05551	0.13730	**0.00099**
CONVEX	100	100	45.90095	0.93196	**0.00831**
CONVEX	100	1000	422.31263	8.38585	**0.06462**
CONVEX	1000	1	>3 hours	94.72402	**0.00390**
CONVEX	1000	10	>3 hours	155.77306	**0.01067**
CONVEX	1000	100	>3 hours	795.11319	**0.06517**
CONVEX	1000	1000	>3 hours	2838.31854	**0.53801**
CONCAVE	10	1	0.11209	0.00026	**0.00007**
CONCAVE	10	10	0.12790	0.00054	**0.00014**
CONCAVE	10	100	0.14002	0.00294	**0.00077**
CONCAVE	10	1000	0.36697	0.02597	**0.00840**
CONCAVE	100	1	10.62329	0.04895	**0.00031**
CONCAVE	100	10	12.63582	0.12927	**0.00146**
CONCAVE	100	100	27.51827	0.85124	**0.00768**
CONCAVE	100	1000	314.32314	7.67280	**0.06285**
CONCAVE	1000	1	>3 hours	91.51055	**0.00332**
CONCAVE	1000	10	>3 hours	149.58491	**0.01079**
CONCAVE	1000	100	>3 hours	744.46691	**0.06696**
CONCAVE	1000	1000	>3 hours	2499.29737	**0.50981**
CLIFF3D	10	1	0.12514	0.00026	**0.00007**
CLIFF3D	10	10	0.13222	0.00055	**0.00013**
CLIFF3D	10	100	0.14432	0.00278	**0.00075**
CLIFF3D	10	1000	0.44964	0.02725	**0.00761**
CLIFF3D	100	1	10.90605	0.04730	**0.00029**
CLIFF3D	100	10	12.85031	0.12709	**0.00112**
CLIFF3D	100	100	44.79395	0.80735	**0.00689**
CLIFF3D	100	1000	679.51368	7.46205	**0.06099**
CLIFF3D	1000	1	>3 hours	136.37944	**0.00344**
CLIFF3D	1000	10	>3 hours	165.34537	**0.01007**
CLIFF3D	1000	100	>3 hours	731.03794	**0.06480**
CLIFF3D	1000	1000	>3 hours	2543.16864	**0.51032**

The results show that the proposed algorithm, KMAC, is the fastest one for all the test problems. Empirical comparisons on randomly generated Pareto fronts of different shape show that the new algorithm is by a factor of 7 to 3.9×10^4 faster than previously published implementations in the 3-D case.

3.6.2 Benchmark Performance

Five state-of-the-art algorithms are compared in this section, they are: EHVI-MOBGO, PoI-MOBGO, NSGA-II [25], NSGA-III [73][74] and SMS-EMOA [19]. The benchmarks are DTLZ1, DTLZ2, DTLZ3, DTLZ4, DTLZ5 and DTLZ7. The parameter settings for all these test algorithms are shown in Table 3.2. Here, EHVI-EGO and PoI-EGO were only tested with evaluation budget as 300, because these two algorithms are time-consuming[1]. The reference points for each benchmark are shown in Table 3.3. Each setting was repeated ten times.

Table 3.2: 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 3.3: Reference Points.

	DTLZ1	DTLZ2	DTLZ3	DTLZ4	DTLZ5	DTLZ7
REF	(400,400,400)	(2.5,2.5,2.5)	(1500,1500,1500)	(2.5,2.5,2.5)	(11,11,11)	(1,1,10)

The final Pareto fronts were evaluated by *Hypervolume*. The empirical experimental results, with respect to statistical mean and standard deviation, are shown in Table 3.4 and 3.5. MOBGO based algorithms perform better than EAs (NSGA-II, NSGA-III and SMS-EMOA), despite the fact that the evaluation budget was increased to 2000. Among EHVI-MOBGO and PoI-MOBGO, EHVI-MOBGO

[1]Updating Kriging model and finding the optimal solution using CMA-ES are expensive.

Table 3.4: Empirical Comparisons.

	MOBGO		EAs			EAs		
Algorithm	EHVI	PoI	NSGA2	NSGA3	SMS-EMOA	NSGA2	NSGA3	SMS-EMOA
Eval.	300	300	300	300	300	2000	2000	2000
DTLZ1	6.38174E+7	6.31810E+7	6.34392E+7	6.36393E+7	6.04776E+7	6.39818E+7	6.39986E+7	6.39985E+7
	6.39662E+7	6.33303E+7	6.32707E+7	6.38025E+7	6.18341E+7	6.39958E+7	6.39990E+7	6.39932E+7
	6.39672E+7	6.35670E+7	6.31858E+7	6.35901E+7	6.14465E+7	6.39958E+7	6.39991E+7	6.39995E+7
	6.39729E+7	6.35214E+7	6.37121E+7	6.36915E+7	6.29733E+7	6.39993E+7	6.39984E+7	6.39995E+7
	6.39722E+7	6.36307E+7	6.37121E+7	6.36159E+7	5.99827E+7	6.39840E+7	6.39992E+7	6.39801E+7
	6.39833E+7	6.33890E+7	6.30707E+7	6.36498E+7	6.23334E+7	6.39969E+7	6.39999E+7	6.39977E+7
	6.39776E+7	6.34682E+7	6.36282E+7	6.32780E+7	6.22081E+7	6.39969E+7	6.39990E+7	6.39951E+7
	6.39790E+7	6.28641E+7	6.34714E+7	6.37243E+7	6.18742E+7	6.39919E+7	6.39996E+7	6.39951E+7
	6.39723E+7	6.35972E+7	6.34714E+7	6.34629E+7	6.18742E+7	6.39764E+7	6.39928E+7	6.39926E+7
	6.39791E+7	6.34263E+7	6.36216E+7	6.32948E+7	6.03677E+7	6.39947E+7	6.39985E+7	6.39946E+7
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	1.50123E+1	1.49961E+1	1.35131E+1	1.45495E+1	1.28434E+1	1.39116E+1	1.49691E+1	1.47435E+1
	1.50293E+1	1.49869E+1	1.35313E+1	1.43761E+1	1.36917E+1	1.35030E+1	1.49737E+1	1.46302E+1
	1.50133E+1	1.50001E+1	1.36346E+1	1.44741E+1	1.29611E+1	1.40951E+1	1.49713E+1	1.42623E+1
	1.50302E+1	1.50036E+1	1.33895E+1	1.41762E+1	1.31983E+1	1.40951E+1	1.49893E+1	1.47413E+1
	1.50178E+1	1.49990E+1	1.33895E+1	1.42859E+1	1.24597E+1	1.38371E+1	1.49763E+1	1.47010E+1
	1.50288E+1	1.49972E+1	1.34931E+1	1.43059E+1	1.28168E+1	1.44331E+1	1.49705E+1	1.48084E+1
	1.50325E+1	1.50030E+1	1.34931E+1	1.45558E+1	1.29619E+1	1.41515E+1	1.49737E+1	1.46117E+1
	1.50263E+1	1.50075E+1	1.28980E+1	1.44279E+1	1.30089E+1	1.34422E+1	1.49844E+1	1.46629E+1
	1.50263E+1	1.49913E+1	1.29148E+1	1.44522E+1	1.31828E+1	1.34422E+1	1.49747E+1	1.47318E+1
	1.49865E+1	1.49901E+1	1.39749E+1	1.46875E+1	1.34046E+1	1.43841E+1	1.49816E+1	1.47118E+1
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

outperforms PoI-MOBGO in most cases, except for DTLZ4 and DTLZ5. The reason is that PoI is a reference-free indicator, and it considers all the possibilities of an evaluated point in the subspace which dominates $\mathcal{P}$. Compared to PoI, however, EHVI only considers the subspace, which is dominated by $\mathcal{P}$ and is cut by a reference point $\mathbf{r}$. In other words, EHVI cannot indicate any improvement of an evaluated point in the rest of non-dominated space, which is cut by a reference point.

3.7 Summary

This chapter described the *Expected Hypervolume Improvement* as the criterion used in MOBGO. The exact calculation of EHVI in the 2-D and the 3-D cases was introduced with the computational complexity of $O(n \log n)$. Compared to [1], the computational complexity is improved by the factor $n^2/\log n$ for 2-D and 3-D cases. This meets the lower bound for the time complexity of the EHVI com-

Table 3.5: Empirical Comparisons.

	MOBGO		EAs			EAs		
Algorithm	EHVI	PoI	NSGA2	NSGA3	SMS-EMOA	NSGA2	NSGA3	SMS-EMOA
Eval.	300	300	300	300	300	2000	2000	2000
DTLZ3	3.37465E+9	3.37431E+9	3.35455E+9	3.36011E+9	3.27636E+9	3.37471E+9	3.37498E+9	3.37459E+9
	3.37462E+9	3.37188E+9	3.34970E+9	3.35426E+9	3.27336E+9	3.37477E+9	3.37498E+9	3.37494E+9
	3.37405E+9	3.36607E+9	3.33783E+9	3.36719E+9	3.26915E+9	3.37446E+9	3.37499E+9	3.37413E+9
	3.37433E+9	3.34920E+9	3.35935E+9	3.36343E+9	3.28850E+9	3.37443E+9	3.37498E+9	3.37440E+9
	3.37460E+9	3.36956E+9	3.33889E+9	3.36678E+9	3.26551E+9	3.37478E+9	3.37500E+9	3.37469E+9
	3.37455E+9	3.37459E+9	3.36864E+9	3.35810E+9	3.34960E+9	3.37471E+9	3.37498E+9	3.37467E+9
	3.37466E+9	3.37371E+9	3.35894E+9	3.36364E+9	3.20942E+9	3.37498E+9	3.37499E+9	3.37469E+9
	3.37452E+9	3.37278E+9	3.36435E+9	3.36349E+9	3.30056E+9	3.37499E+9	3.37500E+9	3.37496E+9
	3.37465E+9	3.36970E+9	3.36268E+9	3.36183E+9	3.30523E+9	3.37361E+9	3.37499E+9	3.37413E+9
	3.37443E+9	3.37337E+9	3.36111E+9	3.35554E+9	3.30145E+9	3.37484E+9	3.37498E+9	3.37467E+9
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	1.38157E+1	1.45550E+1	1.17541E+1	1.08884E+1	9.55215E+0	1.47844E+1	1.49959E+1	1.48530E+1
	1.36793E+1	1.44860E+1	9.30831E+0	1.16457E+1	1.08876E+1	1.32053E+1	1.36099E+1	9.33000E+0
	1.30832E+1	1.45194E+1	9.33753E+0	1.18060E+1	8.85919E+0	1.47539E+1	1.35738E+1	1.31180E+1
	1.38658E+1	1.48104E+1	1.29697E+1	1.28858E+1	9.20911E+0	9.32150E+0	1.49421E+1	1.32434E+1
	1.36991E+1	1.45245E+1	1.31589E+1	1.21828E+1	1.06302E+1	1.35101E+1	1.49398E+1	1.48101E+1
	1.44512E+1	1.42640E+1	1.42940E+1	1.32904E+1	8.86695E+0	1.45135E+1	1.49798E+1	1.32412E+1
	1.42156E+1	1.45332E+1	1.44560E+1	1.19631E+1	8.91678E+0	1.47941E+1	1.35860E+1	1.34922E+1
	1.35453E+1	1.38725E+1	1.30766E+1	1.24391E+1	9.20227E+0	9.30187E+0	1.30364E+1	1.33983E+1
	1.38884E+1	1.44282E+1	1.30079E+1	1.26858E+1	9.16894E+0	1.48205E+1	1.36039E+1	1.33954E+1
	1.37201E+1	1.45676E+1	1.27012E+1	1.29606E+1	1.11599E+1	1.29413E+1	1.49813E+1	1.36090E+1
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	1.31781E+3	1.31885E+3	1.29108E+3	1.31652E+3	1.28384E+3	1.31670E+3	1.31872E+3	1.31681E+3
	1.31609E+3	1.31883E+3	1.31533E+3	1.31687E+3	1.31455E+3	1.31755E+3	1.31888E+3	1.31803E+3
	1.31703E+3	1.31885E+3	1.31556E+3	1.31650E+3	1.30785E+3	1.31759E+3	1.31878E+3	1.31815E+3
	1.31754E+3	1.31885E+3	1.30788E+3	1.31583E+3	1.28715E+3	1.31804E+3	1.31889E+3	1.31744E+3
	1.31741E+3	1.31882E+3	1.31565E+3	1.31663E+3	1.29529E+3	1.31714E+3	1.31887E+3	1.31827E+3
	1.31706E+3	1.31883E+3	1.30716E+3	1.31333E+3	1.30770E+3	1.31731E+3	1.31881E+3	1.31844E+3
	1.31711E+3	1.31878E+3	1.31571E+3	1.31034E+3	1.30873E+3	1.31675E+3	1.31875E+3	1.31785E+3
	1.31802E+3	1.31884E+3	1.31501E+3	1.31654E+3	1.31039E+3	1.31769E+3	1.31876E+3	1.31813E+3
	1.31735E+3	1.31882E+3	1.31564E+3	1.31595E+3	1.30349E+3	1.31785E+3	1.31881E+3	1.31832E+3
	1.31742E+3	1.31883E+3	1.30992E+3	1.31640E+3	1.29023E+3	1.31760E+3	1.31883E+3	1.31795E+3
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	5.09516E+0	4.40942E+0	-1.64477E+0	1.94194E+0	-3.70070E+0	-1.36049E+0	4.90652E+0	1.97405E+0
	5.17729E+0	4.03866E+0	-2.52335E+0	2.08332E+0	-5.84918E+0	-4.14008E+0	5.08696E+0	4.34746E-2
	5.15481E+0	4.28059E+0	-4.49931E+0	2.70067E+0	-4.80969E+0	1.48952E+0	4.66387E+0	2.56120E-1
	4.89191E+0	4.02969E+0	-1.46708E+0	1.18938E+0	-4.38641E+0	2.25216E+0	4.71677E+0	1.98754E-1
	5.06238E+0	4.42348E+0	-1.54354E+0	2.12846E+0	-4.43218E+0	-4.30157E+0	4.99910E+0	1.60383E+0
	4.93139E+0	4.22680E+0	-2.94045E+0	1.27511E+0	-6.29530E-1	-1.04859E+0	5.08663E+0	2.61729E+0
	5.16743E+0	4.52667E+0	-3.76357E+0	2.76559E+0	-3.21233E+0	-2.91679E+0	4.95629E+0	1.78240E+0
	5.02720E+0	4.16430E+0	-2.93363E+0	2.23855E+0	-4.61563E+0	-9.60784E-1	4.66007E+0	1.01542E+0
	5.06158E+0	4.31499E+0	-4.74578E+0	1.94897E+0	-4.83294E+0	-3.53033E+0	4.67642E+0	5.15294E-1
	5.08646E+0	4.06894E+0	-5.35736E+0	1.59954E-1	-4.29017E+0	-6.53071E-1	4.91579E+0	1.06682E-1
mean	**5.08646E+0**	4.06894E+0	-5.35736E+0	1.59954E-1	-4.29017E+0	-6.53071E-1	**4.91579E+0**	1.06682E-1
std.	**7.59397E-2**	1.42676E-1	9.78696E-1	5.81027E-1	1.02525E+0	1.76415E+0	**1.61886E-1**	7.87267E-1

putation for $d = 2, 3$, shown by reduction of the *Hypervolume Indicator* problem, see [1]. Thus, the algorithm is asymptotically optimal and the time complexity of 2-D and 3-D EHVI computation is in $\Theta(n \log n)$. For the arbitrary dimensional case when $d \geq 2$, the formula for exact EHVI calculation is generalized in this chapter. In the speed-comparison experiments, the average execution time of KMAC is compared with that of CDD13. The experimental results show that KMAC is much faster than CDD13, especially for high dimensional cases.

This chapter also compared EHVI-MOBGO with other state-of-the-art multi-objective optimization algorithms. Multi-objective Bayesian global optimization algorithms yield better results, compared to evolutionary multi-objective optimization algorithms. Among multi-objective Bayesian global optimization algorithms, the Pareto-front approximation sets generated by EHVI-MOBGO are usually closer to the true Pareto front. However, PoI-MOBGO is better than EHVI-MOBGO when dealing with DTLZ4 and DTLZ5 problems. The reason is that PoI is a reference-free criterion and EHVI is a reference-based criterion, and EHVI only implies the improvement in the non-dominated space which is cut above by the reference point. A remedy to this problem can be achieved by setting a large reference point or using dynamic reference point. The reference point cannot be too large, otherwise, EHVI at any evaluated points would be similar, even the same, which is due to the numerical stability.