Stochastic and Deterministic Algorithms for Continuous Black-Box Optimization

Proefschrift

ter verkrijging van
de graad van Doctor aan de Universiteit Leiden,
op gezag van Rector Magnificus prof.mr. C.J.J.M. Stolker,
volgens besluit van het College voor Promoties
te verdedigen op donderdag 1 november 2018
klokke 10.00 uur

door

Hao Wang

geboren te Baoji, China
in 1989

Promotiecommissie

Promotor:	Prof. Dr. T.H.W. Bäck	
Co-promotor:	Dr. M.T.M. Emmerich	
Overige leden:	Prof. Dr. A. Plaat	(voorzitter)
	Prof. Dr. H. Trautmann	(WWU Münster and LIACS)
	Dr. C. Dörr	(CNRS and Sorbonne University, FR)
	Prof. Dr. S. Manegold	(secretaris, CWI and LIACS, NL)
	Dr. W.A. Kosters	
	Prof. X. Liu	(Brunel University, UK)

This research is financially supported by the Dutch funding agency NWO, under project number 650.002.001 (the PROMIMOOC project), in collaboration with Tata Steel IJmuiden, BMW Group Regensburg, Centrum voor Wiskunde en Informatica (CWI) and MonetDB.

Figures and diagrams are generated using GGPLOT2, PGF/TIKZ and MATPLOTLIB.

Abstract

Continuous optimization is never easy: the exact solution is always a luxury demand and the theory of it is not always analytical and elegant. Continuous optimization, in practice, is essentially about the efficiency: how to obtain the solution with same quality using as minimal resources (e.g., CPU time or memory usage) as possible? In this thesis, the number of function evaluations is considered as the most important resource to save. To achieve this goal, various efforts have been implemented and applied successfully. One research stream focuses on the so-called stochastic variation (mutation) operator, which conducts an (local) exploration of the search space. The efficiency of those operator has been investigated closely, which shows a good stochastic variation should be able to generate a good coverage of the local neighbourhood around the current search solution. The first part (Chapter 2) of this thesis contributes on this issue by formulating a novel stochastic variation that yields good space coverage.

Alternative research stream approaches the efficiency issue differently: we should keep record of the evaluated solutions and re-use them as they carry partially information about the objective function. This leads to studies on the so-called surrogate modeling. Here, the second part (Chapter 3) of this thesis dives into the one specific surrogate modeling technique, called Kriging/Gaussian Process Regression (GPR). In addition, we try to keep a precise and theoretical treatment on Kriging/GPR and several improvements over it. Closely related to the surrogate modeling, it is crucial to exploit the surrogate model properly. A common approach is to take the so-called infill criteria/acquisition function, which measures the potential gain we could obtain by evaluate one candidate solution on the model. Lastly, the efficiency issues can also be tackled by generalizing a well-performing algorithm from a specific domain to a broader class of problems. One prominent example is to extend the gradient-based optimization algorithms (that are devised

for single objective problems) to the multi-objective scenario. The last part of this thesis (Chapter 5) is on this topic, where both the first- and second-order methods are generalized for multi-objective optimization problems.

Contents

List of Symbols

$\mathbf{1}$	Vector of ones, whose dimension is implied in context
$\mathbf{0}$	Vector of zeros, whose dimension is implied in context
$\mathbf{I}$	Identity matrix, whose shape is implied in context
$\mathbb{N}_{>0}$	Positive natural numbers
$\mathbb{R}^d$	d-dimensional Euclidean space
S	Search space/decision space/domain of the objective function
$\mathcal{X}$	Pareto efficient set
$P_{\mathcal{X}}$	Pareto front
$\mathcal{H}$	Hilbert space of functions $f\colon \mathrm{S} \to \mathbb{R}$
$L^2(\mathrm{S})$	Space of square-integrable functions on S
$\|\cdot\|$	Euclidean norm
$\|\cdot\|_{\mathbf{\Sigma}}$	Mahalanobis norm with respect to a covariance matrix $\mathbf{\Sigma}$
$\|\cdot\|_{\mathcal{H}}$	Norm in Hilbert space $\mathcal{H}$
$\|\cdot\|_{\infty}$	supremum norm
$\langle\cdot,\cdot\rangle$	Dot product in Euclidean spaces.
$\langle\cdot,\cdot\rangle_{\mathcal{H}}$	Inner product in Hilbert space $\mathcal{H}$
$\mathcal{U}(0,1)$	Uniform distribution over $[0,1]$
$\mathcal{N}(m,\sigma^2)$	Gaussian random variable with mean m and variance σ^2
$\boldsymbol{\mathcal{N}}(\mathbf{m},\mathbf{K})$	Gaussian random vector with mean $\mathbf{m}$ and covariance matrix $\mathbf{K}$

$\mathcal{GP}(m(\cdot), k(\cdot,\cdot))$	Gaussian process with mean function $m(\cdot)$ and kernel $k(\cdot,\cdot)$
$\Pr(\cdot)$	Probability of an event
$p(\cdot)$	Probability density function (p.d.f.)
$P(\cdot)$	Cumulative distribution function (c.d.f.)
$\mathbb{P}$	Probability measure
ϕ	Probability density function of $\mathcal{N}(0,1)$
ϕ_{m,σ^2}	Probability density function of $\mathcal{N}(m,\sigma^2)$
Φ	Cumulative distribution function of $\mathcal{N}(0,1)$
Φ_{m,σ^2}	Cumulative distribution function of $\mathcal{N}(m,\sigma^2)$
$\Phi^n_{\mathbf{m},\mathbf{K}}$	Cumulative distribution function of a multivariate Gaussian $\boldsymbol{\mathcal{N}}(\mathbf{m},\mathbf{K})$
$\mathbb{E}$	Expectation
Var	Variance
Cov	Covariance
$\perp\!\!\!\perp$	Statistical independence
σ_n^2	Variance of the white noise process
$\mathscr{A}$	Infill criteria/acquisition function
det	Determinant of square matrices
κ	Condition number of matrices

CHAPTER 1

Introduction

Optimization problems are of fundamental importance in mathematics, statistics, Machine Learning and real-world applications (e.g., optimization of a production process). In most cases, we aim at searching for an element (called candidate solution) in some pre-determined domain (called search space) of objective functions, such that that this element "outperforms" the remaining elements according to an (partial) order structure defined in the image of objective functions. Some examples of optimization problems are: searching for the minimum of a given function, the optimal linear predictor/estimator (statistics), the optimal linear separation boundary for binary classification problems (Machine Learning) and the optimal control parameters of an industrial production line. Prior to the detailed discussions, we shall give a brief explanation on some important aspects of optimization problems.

Domain It is also referred as the *search space* in the unconstrained optimization problems. The most intuitive domain is the subset of Euclidean spaces $\mathbb{R}^d$ (d is used as the dimensionality of the domain in this thesis). Some other important ones are: Hilbert (or Banach) spaces of functions, and mixed spaces, e.g., $\mathbb{R}^{d_1} \times \{0,1\}^{d_2} \times \{\text{Mon}, \text{Tue}, \ldots\}^{d_3}$. In this thesis, the discussion is restricted to the subset of $\mathbb{R}^d$. In addition, the Euclidean metric (or the related Mahalanobis metric) is always assumed on $\mathbb{R}^d$. Under such assumptions, we shall adopt the convention "**continuous optimization**" here[1].

[1]The specification of the metric is mandatory here because $\mathbb{R}^d$ can become a discrete space if any metric that yields isolated points is equipped to $\mathbb{R}^d$.

Objective function Although there are various types of objective functions in practice, this thesis is limited to the *real-valued* functions. In addition, the well-known **black-box** assumption is set on the objective function, meaning that no additional analytical property (e.g., continuity, smoothness and differentiability) is assumed on the objective function and the only available information is the evaluation of points in its domain.

Algorithm There are many numerical algorithm for solving the optimization problem. From the perspective of randomness, those algorithms can be categorized into **deterministic** and **stochastic** optimization algorithms. The former usually refers the classical mathematical optimization techniques (e.g., the Newton's method). The latter is mainly developed for the black-box optimization problems, which relies heavily on the statistical properties of random variables. In this thesis, both categories of algorithms are studies and improved.

1.1 Stochastic Optimization

In this thesis, the discussion is restricted to the real-valued objective function of the form:

$$f : \mathrm{S} \subseteq \mathbb{R}^d \to \mathbb{R}^m, \tag{1.1}$$

where its domain S is assumed to be a subset of the d-dimensional Euclidean space and its image is $\mathbb{R}^m$. The problem of minimizing (or maximizing) f is referred as a *single objective* problem if $m = 1$. For $m > 1$, it is called a *multi-objective* optimization problem and it is typically denoted by the boldface symbol $\mathbf{f}$. Note that, in the multi-objective scenario, it is usually not possible to define a *total order* on $\mathbb{R}^m$. The result of the multi-objective optimization is typically the "best" *anti-chain* w.r.t. some partial order defined on $\mathbb{R}^m$ (see the next section). In practice, domain S could represent the so-called "feasible region" in $\mathbb{R}^d$, that is restricted by a set of constraint functions. The subject of this thesis, the stochastic optimization paradigm, targets the so-called black-box optimization problem.

Definition 1.1 (Black-Box Optimization)**.** *An objective function f as defined in Eq.* (1.1) *is called* ***black-box*** *iff no prior knowledge is available on f and the only accessible posterior information is the objective value $f(\mathbf{x})$ for every point $\mathbf{x}$ on its domain.*

Remark. With no prior knowledge on f, many mathematical/numerical optimization methods, e.g., gradient descent and Newton's method, render inapplicable because the common assumptions, e.g., analytical expressions, differentiability as well as continuity no longer hold on f. In optimization, the domain S of f is more commonly referred as **search space** or **decision space**. We shall use those two terms interchangeably in this thesis. In the context of evolutionary computation (Bäck, 1996), the so-called **fitness value** depends on $f(\mathbf{x})$.

Throughout this thesis, the objective function f is assumed to be *minimized*, without loss of generality. In the single objective case, the goal of global minimization is to solve

$$\mathbf{x}^* = \arg\min_{\mathbf{x}\in \mathrm{S}} f(\mathbf{x}),$$

where the notion of global minimum is defined as follows.

Definition 1.2 (Global minimum)**.** *In the single objective case, point* $\mathbf{x}^* \in \mathrm{S}$ *is a global minimum (or minimum for short) of* f *iff* $\forall \mathbf{x} \in \mathrm{S}, f(\mathbf{x}^*) \leq f(\mathbf{x})$.

Approaching a global minimum is generally a difficult task due to the so-called *multimodality* of the objective function. Practically, it is only possible to guarantee the convergence to the so-called local optima.

Definition 1.3 (Local minimum)**.** *In the single objective case, a point* $\mathbf{x} \in \mathrm{S}$ *is a local minimum of* f *if there exists a neighborhood* $N_{\mathbf{x}}$ *of* $\mathbf{x}$ *such that* $\forall \mathbf{x}' \in N_{\mathbf{x}}, f(\mathbf{x}) \leq f(\mathbf{x}')$.

Remark. As the search space S is a subset of the *metric space* $\mathbb{R}^d$, it is straightforward to use any metric on $\mathbb{R}^d$ to define the neighborhood. For example, when taking the Euclidean norm $\|\cdot\|$, the neighborhood can be defined as a subset of S that contains an open Euclidean ball around $\mathbf{x}$: $B_\varepsilon(\mathbf{x}) = \{\mathbf{x}' \in \mathrm{S} : \|\mathbf{x} - \mathbf{x}'\| < \varepsilon\}$ for some $\varepsilon > 0$.

For solving black-box optimization problems, a very common mechanism is to progressively refine a point $\mathbf{x}$ by evaluating other candidate points in the neighborhood of $\mathbf{x}$ and moving to the point that improves $f(\mathbf{x})$. This is called *local search*. This mechanism requires two design choices: the determination of the neighborhood and a selection method to pick points in the neighborhood. In numerical optimization, this is usually achieved by directly using gradient increments (steps) or Newton increments. However, none of those techniques is applicable under the black-box assumption. Alternatively, in stochastic optimization, random perturbations are used for the local search, where commonly a parametric distribution family centered at $\mathbf{x}$ (e.g., Gaussian) is taken to generate candidate points in the neighborhood

(typically S). More precisely, when discussing multivariate random variables in $\mathbb{R}^d$, it is common to assume the probability space $(\Omega, \mathscr{F}, \mathbb{P})$ and a measurable space $(\mathbb{R}^d, \mathscr{B}^d)$, where $\mathscr{B}^d$ is the Borel algebra on $\mathbb{R}^d$. A $\mathbb{R}^d$-valued random variable (or random vector) $\mathbf{x} \in \mathbb{R}^d$ is a $\mathscr{F}$-measurable function, $\mathbf{x} : \Omega \to \mathbb{R}^d$. Then, the formal definition of stochastic optimization is given as follows.

Definition 1.4 (Stochastic Optimization)**.** *Taking the aforementioned probability settings, Stochastic Optimization is the procedure of applying one or many optimization algorithms on a black-box function f, yielding a process of $\mathbb{R}^d$-valued random variables: $\{\mathbf{x}_t : t \in \mathbb{N}_{>0}\}$, such that*

$$\forall \varepsilon > 0, \quad \lim_{t \to \infty} \Pr\left(D(\mathbf{x}_t - \tilde{\mathbf{x}}) > \varepsilon\right) = 0, \tag{1.2}$$

where D is a metric on $\mathbb{R}^d$ and $\tilde{\mathbf{x}}$ is a (local) minimum and the conditional density

$$p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \ldots, \mathbf{x}_1\right)$$

can be specified using the probability measure $\mathbb{P}$.

Remark. 1) The stochastic process $\{\mathbf{x}_t : t \in \mathbb{N}_{>0}\}$ could stand for the current best point or the best point found since the first iteration. 2) It is expressed explicitly that "applying one or many optimization algorithms" because in practice two or more stochastic optimizers can be combined for the task, e.g., in case of memetic algorithms (Moscato et al., 1989). 3) In the single objective case, the convergence criterion can be formulated equivalently:

$$\forall \varepsilon > 0, \quad \lim_{t \to \infty} \Pr\left(|f(\mathbf{x}_t) - f(\tilde{\mathbf{x}})| > \varepsilon\right) = 0.$$

4) If the (local) minimum $\tilde{\mathbf{x}}$ is forced to be the global minimum $\mathbf{x}^*$, then the stochastic optimization procedure is said to **converge globally**. Note that the convergence in probability (Eq. 1.2) is taken for the convergence criterion because stronger types of convergence (e.g., almost sure convergence) do not hold in some cases and thus it is generally safe to use a weaker convergence notion. 5) In the case where it is hard to verify the convergence criterion for some practically well-performing algorithms, the criterion is relaxed to the following:

$$\forall t \in \mathbb{N}_{>0}\ \exists n \in \mathbb{N}_{>0} \quad \text{s.t.} \quad \mathbb{E}\left\{f(\mathbf{x}_{t+n}) \mid f(\mathbf{x}_{t+n-1}), f(\mathbf{x}_{t+n-2}), \ldots, f(\mathbf{x}_1)\right\} \leq f(\mathbf{x}_t).$$

Or equivalently there exists a subprocess of $\{f(\mathbf{x}_t) : t \in \mathbb{N}_{>0}\}$, being a supermartingale[1]. 6) Markov property holds for some stochastic optimization algorithms, e.g., $(1+1)$-ES (Bäck, 1996), meaning that $p\left(\mathbf{x}_t \mid \mathbf{x}_{t-1}, \mathbf{x}_{t-2}, \ldots, \mathbf{x}_1\right) = p(\mathbf{x}_t \mid \mathbf{x}_{t-1})$.

[1]Loosely speaking, the discrete-time supermartingale indicates the situation where the conditional expectation on the whole history at each step is not bigger than the random variable at the last time step.

Many stochastic optimization algorithms has been proposed for single- (Kirkpatrick et al., 1983; Schwefel, 1993; Bäck, 1996) and multi-objective (Deb et al., 2000; Emmerich, 2005) black-box optimization problems. In the literature, some stochastic optimization algorithms are often referred to *metaheuristics* (Luke, 2009; Talbi, 2009), e.g., Particle Swarm Optimization (Kennedy and Eberhart, 1995). Those algorithms can be categorized according to different criteria:

- Local search/Global search: a well-known example of local search is the stochastic hill-climbing algorithm. Some example of global search are evolutionary algorithms (Bäck, 1996), Swarm Intelligence (Bonabeau et al., 1999) and Efficient Global Optimization (Jones et al., 1998).
- Single-point based/Population-based algorithms: if a stochastic optimizer employs only one point iteratively, it is called a single-point strategy, e.g., Simulated Annealing (Kirkpatrick et al., 1983). Otherwise, it is called a population-based algorithm, e.g., (multi-membered) evolutionary algorithms. It is worth mentioning that the so-called $(1+1)$-EAs that employ one parent and one offspring fall into the single-point category.
- Nature-inspired algorithms: examples are evolution strategies (Bäck and Schwefel, 1993; Bäck et al., 2013), genetic algorithms (Goldberg, 1989) and Swarm Intelligence (Bonabeau et al., 1999).

In this thesis, instead of focusing on some specific optimization algorithms, we illuminate and investigate several important aspects of this field, which underpin many optimization algorithms:

- *Stochastic variation* is the algorithmic component where the (local) random perturbation is generated to modify the current point. In evolutionary computation, this is typically called the *mutation operator* (Bäck and Schwefel, 1993). In $\mathbb{R}^d$, the most common method is to apply the simple random sampling method on the Gaussian distribution. Other stochastic variations include: differential vector in the Differential Evolution (Storn and Price, 1997) and polynomial mutation (Agrawal et al., 1995). In Chapter 2, we shall illustrate a drawback of the simple random sampling from a Gaussian distribution and propose an improved sampling method, whose effectiveness is validated when plugged into evolution strategies.
- *Surrogate modeling*: When the function evaluation is very *expensive* on f, e.g., due to the high time complexity, it is common to build models that are

less computational expensive on the evaluated points, in order to partially replace the actual function evaluation. The precision of the surrogate model is of vital importance when assisting the stochastic optimizer. However, this is usually a demanding requirement due to the lack of a sufficient number of data points, or irreducible modeling error when the objective function is noisy. In this scenario, it is helpful to quantify the uncertainty of the model prediction, e.g., by computing a confidence interval. We study a widely used surrogate model, Kriging/Gaussian process regression that it is naturally equipped with an uncertainty quantification.

- *Model utilization*: taking the model imprecision and uncertainty quantification into account, it is possible to determine the most trustworthy point, or alternatively which point possesses the highest potential to help the optimization procedure if the actual function evaluation were performed on it. Such decisions are made through an utility function of the surrogate model, called *infill criterion*. The infill criterion plays a vital role in many optimization paradigms, including the Surrogate-assisted Evolutionary Algorithms (Emmerich, 2005), Efficient Global Optimization (Močkus, 1975, 2012; Jones et al., 1998), Multi-armed Bandits (Auer et al., 2002) and Monte-Carlo Tree Search (Silver et al., 2016). In this thesis, we aim at summarizing the existing infill criterion and proposing a novel infill criteria that is theoretically better than the existing ones. Furthermore, the parallelization issue (Ginsbourger et al., 2010) of infill criteria is also considered in detail and several new parallelization methods are proposed and tested.

1.2 Multi-objective Optimization

In this section we introduce some definitions in the context of multi-objective problems. Due to the possibility of incomparable solutions, the notation of modality/local optimality is also modified and extended to the multi-objective scenario. The search space under consideration is $\mathrm{S} \subseteq \mathbb{R}^d$ and the objective space is $\mathbb{R}^m$. Most of our definitions can also be generalized to other spaces, however, due to space limitations, this will not be part of this section.

Now, let $\mathbf{f} : \mathrm{S} \to \mathbb{R}^m$ be a multi-objective function (which we want to "minimize") with component functions $f_i : \mathrm{S} \to \mathbb{R}$, $i = 1, \ldots, m$ and $\mathrm{S} \subseteq \mathbb{R}^d$. Given a totally

ordered set $(T, \leq)$, with total order $\leq$, the *Pareto order*[1] $\prec$ on T^k for any $k \in \mathbb{N}$ is defined as follows: Let $\mathbf{t}^{(1)} = (t_1^{(1)}, \ldots, t_k^{(1)}), \mathbf{t}^{(2)} = (t_1^{(2)}, \ldots, t_k^{(2)}) \in T^k$. We say $\mathbf{t}^{(1)} \prec \mathbf{t}^{(2)}$ if and only if (iff) $t_i^{(1)} \leq t_i^{(2)}$, $i = 1, \ldots, k$ and $\mathbf{t}^{(1)} \neq \mathbf{t}^{(2)}$. Instantiating $\leq$ to the natural total order on the real numbers, we obtain the Pareto order on $\mathbb{R}^m$. A point $\mathbf{x} \in \mathrm{S}$ is called *Pareto efficient* or *global efficient* or for short *efficient* iff there does not exist $\tilde{\mathbf{x}} \in \mathrm{S}$ such that $\mathbf{f}(\tilde{\mathbf{x}}) \prec \mathbf{f}(\mathbf{x})$. The set of all the (global) efficient points in S is denoted by $\mathcal{X}$ and is called the (Pareto) *efficient set* of $\mathbf{f}$. The image of $\mathcal{X}$ under $\mathbf{f}$ is called the *Pareto front* of $\mathbf{f}$, symbolically $P_{\mathcal{X}} = \mathbf{f}[\mathcal{X}] = \mathbf{f}(\mathbf{x}) : \mathbf{x} \in \mathcal{X}$.

Defining a locally efficient point in S (or of $\mathbf{f}$) is as straightforward as defining local minimizers (maximizers) for single-objective functions. This is in contrast to defining local efficient *sets*, which are needed for the multi-criteria setting.

Definition 1.5 (Locally Efficient Point)**.** *A point* $\mathbf{x} \in \mathrm{S}$ *is called* locally efficient point *of* $\mathbf{f}$ *if there is an open set* $U \subseteq \mathbb{R}^d$ *such that there is no point* $\mathbf{x}' \in U \cap \mathcal{X}$ *such that* $\mathbf{f}(\mathbf{x}') \prec \mathbf{f}(\mathbf{x})$*. The set of all the local efficient points in* S *is denoted by* $\mathcal{X}_L$*.*

Definition 1.6 (Globally Efficient Point)**.** *A point* $\mathbf{x} \in \mathrm{S}$ *is called* globally efficient point $\mathbf{f}$ *if there is no point* $\mathbf{x}' \in \mathbb{R}^d \cap \mathrm{S}$ *such that* $\mathbf{f}(\mathbf{x}') \prec \mathbf{f}(\mathbf{x})$*. The set of all the global efficient points in* S *is termed* efficient set *of* $\mathbf{f}$ *and denoted by* $\mathcal{X}$*.*

In order to extend the definition of the local optimality to multi-objective problems, it is necessary to first give a notation on the locality of (efficient) sets. It is defined using the so-called *connectedness*.

Definition 1.7 (Connectedness and Connected Component)**.** *Let* $A \subseteq \mathbb{R}^d$*. The subset* A *is called* connected *if and only if there do not exist two open subsets* U_1 *and* U_2 *of* $\mathbb{R}^d$ *such that* $A \subseteq U_1 \cup U_2$*,* $U_1 \cap A \neq \emptyset$*,* $U_2 \cap A \neq \emptyset$*, and* $U_1 \cap U_2 \cap A = \emptyset$*; or equivalently there do not exist two non-empty subsets* A_1 *and* A_2 *of* A *which are open in the relative topology of* A *such that* $A_1 \cup A_2 = A$ *and* $A_1 \cap A_2 = \emptyset$*. Let* B *be a non-empty subset of* $\mathbb{R}^d$*. A subset* C *of* B *is a* connected component *of* B *iff* C *is non-empty, connected, and there exists no strict superset of* C *that is connected.*

Definition 1.8 (Locally Efficient Set)**.** *A subset* $A \subseteq \mathrm{S}$ *is a* locally efficient set *of* $\mathbf{f}$ *if* A *is a connected component of* $\mathcal{X}_L$ *(= set of the locally efficient points in* S*).*

Definition 1.9 (Local Pareto Front)**.** *A subset* P *of the image of* $\mathbf{f}$ *is a* local Pareto front *of* $\mathbf{f}$*, if there exists a local efficient set* E *such that* $P = \mathbf{f}[E]$*.*

[1] It gives rise to a partial order vector space $(T^k, \prec)$.

Note that the (global) Pareto front of $\mathbf{f}$ is obtained by taking the image of the union of connected components of $\mathcal{X}$, under $\mathbf{f}$. If $\mathcal{X}$ is connected and $\mathbf{f}$ is continuous on $\mathcal{X}$, the Pareto front is also connected. In this thesis we use the notion of connectedness to define the locally efficient sets. There still remains the task of extending the notion of efficient set by looking at connectedness in the objective space. For instance it could happen that two different local efficient sets are mapped onto the same set in the objective space. This rises many questions, which need to be addressed in future work.

With a view towards algorithms that numerically approximates (locally) efficient sets and/or (local) Pareto fronts, it is necessary to generalize definition 1.8 to determine whether a *finite set* belongs to a connected component (i.e., a finite subset of $\mathcal{X}_L$ is a set of some locally efficient set). Here the issue is: a finite subset of Euclidean space is never a connected component (of some other subset) unless it consists of one point. To reconcile with definition 1.8, the notion of connectedness can be relaxed, using the ε neighborhood.

Definition 1.10 (ε-connectedness)**.** *Let $\varepsilon \in \mathbb{R}_{>0}$ and $\mathrm{S} \subseteq \mathbb{R}^d$. Set A is* ε-connected *if and only if for any distinct points $x, x' \in A$ there is a finite set of points $\{a_1, \ldots, a_k\} \subseteq A$ such that $D(x, a_1) \leq \varepsilon, D(a_1, a_2) \leq \varepsilon, \ldots, D(a_{k-1}, a_k) \leq \varepsilon, D(a_k, x') \leq \varepsilon$, where D is a metric in $\mathbb{R}^d$.*

A *finite set* $A \subseteq \mathrm{S}$ is locally efficient, if it consists of local efficient points in S and A is ε-connected: there exists $\varepsilon > 0$ on which definition 1.10 holds.

Definition 1.11 (finite ε-Local Efficient Set)**.** *Let A be a* finite *subset of $\mathcal{X}_L$. Then A is an* ε-local efficient set*, if $A \neq \emptyset$, and A is ε-connected.*

1.3 Matrix Calculus

In this thesis, the compact notation, called *Matrix Calculus* (Kollo and von Rosen, 2005) is extensively used for the derivations (e.g., Section 3.1.4, 5.2 and 5.3). For the readability of the technical part, we shall specify this notation and give some examples. Intuitively, the matrix differentiation is a collection of many partial derivatives, e.g., the gradient vector of a real-valued function $f : \mathbb{R}^d \to \mathbb{R}$:

$$\nabla f = \frac{\partial f}{\partial \mathbf{x}} = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_d} \right)^\top .$$

The notation $\partial f/\partial \mathbf{x}$ is called a *scalar-by-vector* derivative. Throughout this thesis, the gradient is assumed to be a *column vector* and thus $\partial f/\partial \mathbf{x}$ has a column-wise layout. To avoid confusions, the *layout* of matrix derivatives like $\partial \mathbf{f}/\partial \mathbf{x}$ is determined according to that of $\mathbf{f}^\top$ or $\mathbf{x}$. This is called the *denominator layout convention*. Some common layouts are given as follows:

- *scalar-by-vector*

$$\frac{\partial f}{\partial \mathbf{x}} = \left(\frac{\partial f}{\partial x_1}, \frac{\partial f}{\partial x_2}, \ldots, \frac{\partial f}{\partial x_d}\right)^\top, \quad f : \mathbb{R}^d \to \mathbb{R},\ \mathbf{x} \in \mathbb{R}^d.$$

- *vector-by-scalar*

$$\frac{\partial \mathbf{f}}{\partial x} = \left(\frac{\partial f_1}{\partial x}, \frac{\partial f_2}{\partial x}, \ldots, \frac{\partial f_m}{\partial x}\right), \quad \mathbf{f} : \mathbb{R} \to \mathbb{R}^m,\ x \in \mathbb{R}.$$

- *vector-by-vector*

$$\frac{\partial \mathbf{f}}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \frac{\partial f_2}{\partial x_1} & \cdots & \frac{\partial f_m}{\partial x_1} \\ \frac{\partial f_1}{\partial x_2} & \frac{\partial f_2}{\partial x_2} & \cdots & \frac{\partial f_m}{\partial x_2} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial f_1}{\partial x_d} & \frac{\partial f_2}{\partial x_d} & \cdots & \frac{\partial f_m}{\partial x_d} \end{bmatrix}, \quad \mathbf{f} : \mathbb{R}^d \to \mathbb{R}^m,\ \mathbf{x} \in \mathbb{R}^d.$$

The major benefit of using such notations is that the common rules for derivatives, e.g., chain rule, product rule and quotient rule still hold for the matrix notation. For example, consider the following functions: $\mathbf{f} : \mathbb{R}^d \to \mathbb{R}^m, g : \mathbb{R}^m \to \mathbb{R}$. The composition $g \circ \mathbf{f} : \mathbb{R}^d \to \mathbb{R}$ can be differentiated using the chain rule:

$$\frac{\partial (g \circ \mathbf{f})}{\partial \mathbf{x}} = \frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} \frac{\partial g(\mathbf{f})}{\partial \mathbf{f}}$$

For example, when differentiating a quadratic form w.r.t. a vector, we have:

$$\frac{\partial \mathbf{u}^\top \mathbf{K} \mathbf{v}}{\partial \mathbf{x}} = \frac{\partial \mathbf{u}}{\partial \mathbf{x}} \mathbf{K} \mathbf{v} + \frac{\partial \mathbf{v}}{\partial \mathbf{x}} \mathbf{K}^\top \mathbf{u},$$

where $\mathbf{u} \in \mathbb{R}^m, \mathbf{v} \in \mathbb{R}^n, \mathbf{K} \in \mathbb{R}^{m \times n}, \mathbf{x} \in \mathbb{R}^d$ and $\mathbf{K}$ is not a function of $\mathbf{x}$. One can easily verify that the shape of the LHS (left-hand-side) admits that of the RHS (right-hand-side), assuming the denominator layout.

1.4 Outline of the Dissertation

The outline of this thesis is as follows. The motivation, content and research questions of each chapter are briefly introduced, which is followed by a publication list on each chapter.

Chapter 2 discusses several sampling methods designed to reduce the sampling error from a multivariate Gaussian distribution. The proposed *mirrored orthogonal sampling* method is applied to Evolution Strategies. The convergence property of the resulting optimization algorithm is investigated both theoretically and empirically. In addition, the stochastic variation behind the Efficient Global Optimization algorithm is extracted and formulated as a stand-alone stochastic variation method.

- Wang, H., M. Emmerich, and T. Bäck (2014). Mirrored orthogonal sampling with pairwise selection in evolution strategies. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, New York, NY, USA, pp. 154–156. ACM.

- van Rijn, S., H. Wang, B. van Stein, and T. Bäck (2017). Algorithm configuration data mining for CMA evolution strategies. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, New York, NY, USA, pp. 737–744. ACM.

- Wang, H., M. Emmerich, and T. Bäck (2018). Mirrored Orthogonal Sampling for Covariance Matrix Adaptation Evolution Strategies. *Evolutionary computation* (27), to appear.

- Emmerich, M., O. M. Shir, and H. Wang (2018). *Evolution Strategies*, pp. 1–31. Cham: Springer International Publishing.

Chapter 3 aims at giving a precise and unified treatment of the commonly used surrogate modeling method, Kriging/Gaussian process regression. This estimation method is summarized and compared from many perspectives, including the theory on the best linear predictor, reproducing kernel Hilbert Space and Bayesian inference. In the second half of the chapter, a novel algorithmic framework called *Cluster Kriging* is proposed to relax the high time/space complexity of the original Kriging method, when applied to large data sets. Moreover, it is shown that Cluster Kriging can effectively support the efficient global optimization algorithm.

van Stein, B., H. Wang, W. Kowalczyk, T. Bäck, and M. Emmerich (2015). Optimally weighted cluster kriging for big data regression. In E. Fromont, T. De Bie, and M. van Leeuwen (Eds.), *Advances in Intelligent Data Analysis XIV: 14th International Symposium, IDA 2015, Saint Etienne. France, October 22 -24, 2015. Proceedings*, Cham, pp. 310–321. Springer International Publishing.

van Stein, B., H. Wang, W. Kowalczyk, M. Emmerich, and T. Bäck (2016). Fuzzy clustering for optimally weighted cluster kriging. In *Proceedings of the Conference on Evolutionary Computation*, CEC '16, pp. 154–163.

Wang, H., B. van Stein, M. Emmerich, and T. Bäck (2017b). Time complexity reduction in efficient global optimization using cluster kriging. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, New York, NY, USA, pp. 889–896. ACM.

van Stein, B., H. Wang, W. Kowalczyk, and T. Bäck.
A Novel Uncertainty Quantification Method for Efficient Global Optimization. In *Proceedings of 17th International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems*, IPMU 2018.

Chapter 4 focuses on the issue on how to use the surrogate model properly. The utility of each location on a surrogate model is quantified by a well-defined function, called *Infill Criterion*. Various infill criteria are compared in this chapter, where the trade-offs between criterion are discovered. In addition, a novel infill criterion, *Moment-Generating Function of Improvement* (MGFI) is proposed as the extension of all improvement-based criteria. Lastly, we investigate the multi-point generalization to the existing infill criteria, allowing for the parallel evaluation of candidate solutions.

Wang, H., M. Emmerich, and T. Bäck (2016). Balancing risk and expected gain in kriging-based global optimization. In *Proceedings of the Conference on Evolutionary Computation*, CEC '16, pp. 154–163.

Emmerich, M., K. Yang, A. Deutz, H. Wang, and C. M. Fonseca (2016). *A Multicriteria Generalization of Bayesian Global Optimization*, pp. 229–242. Cham: Springer International Publishing.

Wang, H., B. van Stein, M. Emmerich, and T. Bäck (2017a, Oct). A New Acquisition Function for Bayesian Optimization based on the Moment-Generating Function. In *2017 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, pp. 507–512.

Wang, H., T. Bäck, and M. T. M. Emmerich (2018). Multi-point efficient global optimization using niching evolution strategy. In A.-A. Tantar, E. Tantar, M. Emmerich, P. Legrand, L. Alboaie, and H. Luchian (Eds.), *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation VI*, Cham, pp. 146–162. Springer International Publishing.

Wang, H., M. Emmerich, and T. Bäck (2018). Cooling Strategies for the Moment-Generating Function in Bayesian Global Optimization. In *Proceedings of the Conference on Evolutionary Computation*, CEC '18, to appear.

Chapter 5 discusses numerical multi-objective optimization (MOO). The demand on this topic originates from many numerical multi-objective tasks that arise in the study of stochastic optimization, e.g., the multi-objective treatment of infill criteria in Chapter 4. The contribution in this chapter is three-fold: firstly, we mathematically analyze the so-called *Mixed-Peak* bi-objective test problem. Secondly, the gradient field and Hessian matrix of the hypervolume indicator are studied in depth. Thirdly, two novel numerical MOO algorithms, namely the hypervolume-based first- (gradient) and second-order (Hessian) methods are proposed and tested.

Kerschke, P., H. Wang, M. Preuss, C. Grimme, T. Heike, and E. Michael (2016). Towards analyzing multimodality of multiobjective landscapes. In *International Conference on Parallel Problem Solving from Nature*, pp. 206–215. Springer.

Wang, H., A. Deutz, T. Bäck, and M. Emmerich (2017). Hypervolume indicator gradient ascent multi-objective optimization. In *9th International Conference on Evolutionary Multi-Criterion Optimization - Volume 10173*, EMO 2017, New York, NY, USA, pp. 654–669. Springer-Verlag New York, Inc.

Wang, H., Y. Ren, A. Deutz, and M. Emmerich (2017). *On Steering Dominated Points in Hypervolume Indicator Gradient Ascent for Bi-Objective Optimization*, pp. 175–203. Cham: Springer International Publishing.

Kerschke, P., H. Wang, M. Preuss, C. Grimme, T. Heike, and E. Michael (2018). Search Dynamics on Multimodal Multi-Objective Problems. *Evolutionary computation* (30), to appear.

van der Blom, K., S. Boonstra, H. Wang, H. Hofmeyer, and M. Emmerich *Evaluating Memetic Building Spatial Design Optimisation Using Hypervolume Indicator Gradient Ascent*. Cham: Springer International Publishing, to appear.

In addition to this description of the chapters, some closely linked sections are shown in the dependence graph below.

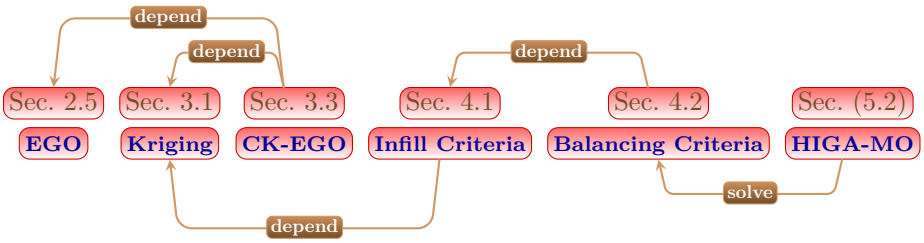


Figure 1.1: Dependences between several sections.

CHAPTER 2

Stochastic Variation

In the continuous search space $S \subseteq \mathbb{R}^d$, the most common stochastic variation operator is the **multivariate Gaussian distribution**. It is denoted as $\boldsymbol{\mathcal{N}}(\mathbf{m}, \mathbf{C})$ where $\mathbf{m}$ is the mean vector and $\mathbf{C}$ is the covariance matrix. The definition and some properties of the Gaussian distribution can be found in Appendix A. Generating d-dimensional random vectors from a multivariate Gaussian distribution is the key source of stochastic variations in many stochastic optimization algorithms, e.g., evolution strategies (Bäck et al., 2013). The standard method to achieve this, *simple random sampling* (or random sampling for short), samples pseudo-random numbers directly from a certain distribution. However, it also results in a high *sampling error* or sampling variation, which would lead to "bad" samples (explained in the following). The sampling error occurs when we estimate the statistical properties of a distribution from its realizations. By sampling error, we mean the estimation errors of statistical properties (e.g., mean, covariance) of a distribution, which are caused by unrepresentative or biased samples.

An example of biased samples is illustrated in Fig. 2.1, in which four i.i.d. mutation vectors are sampled from a multivariate Gaussian distribution $\boldsymbol{\mathcal{N}}(\mathbf{m}, \mathbf{C})$. The black solid ellipsoid represents the expectation of the mutations and reflects the covariance matrix $\mathbf{C}$. The diversity of the four samples is not satisfactory because the minimal distance between samples is relatively small compared to the axis length of the black solid ellipsoid. A strong sampling error incurs in this case because if the mean and covariance of the distribution are estimated from these four vectors, the results would deviate largely from $\mathbf{m}$ and $\mathbf{C}$.

As a result of the biased samples, a large portion of space is not reached (at least half the space in this case). Moreover, if the objective function is twice differentiable, the contour lines should be locally convex near the optimum (the dashed ellipsoids). The

probability that a new search point represents an improvement can be very small, shown by the area with vertical lines intersecting the solid ellipsoid. Therefore, if the population size is small, an undesired sampling case can take place such that none of the mutations represents an improvement, which renders the current generation inefficient. The sampling error has an even bigger side effect in modern evolution strategies (e.g., CMA-ES (Hansen, 2006)) because those algorithms tend to exploit small populations to speed up their convergence. To overcome this problem, it is proposed here to develop special sampling approaches for the reduction of sampling error in a small population, such that the statistical properties estimated from mutation samples are more similar to their underlying true distribution.

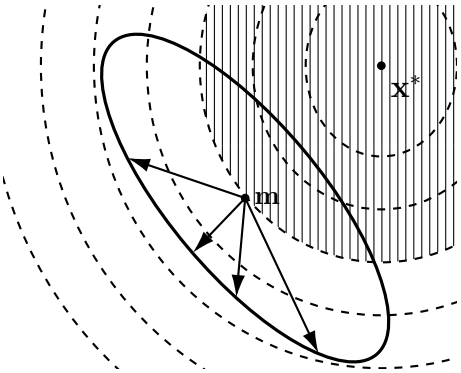


Figure 2.1: Illustration of a set of unsuccessful mutation samples. Four offspring are generated here while none of them is an improvement. This phenomenon reduces the convergence velocity of the algorithm.

The sampling method proposed in this chapter is plugged into evolution strategies (ES) for testing. To make this chapter self-contained, the algorithmic structure of $(\mu \overset{+}{,} \lambda)$-ES is given in Alg. 1. For the details on evolution strategies, please see Emmerich, Shir, and Wang (2018).

2.1 Quasi-Random Sampling

There are some techniques proposed to reduce the sampling error as much as possible and to enhance the diversity. The first method is called quasi-random sampling or low-discrepancy sequences (Dick and Pillichshammer, 2010). Low-discrepancy sequences are commonly used as a replacement of uniformly distributed

Algorithm 1 $(\mu \dot{+} \lambda)$ Evolution Strategy

procedure $(\mu \dot{+} \lambda)$-ES$(\mu, \lambda, f, \sigma_0)$
  $\mathbf{C} \leftarrow \mathbf{I}, \quad \sigma \leftarrow \sigma_0$ ▷ initialization
  while not terminated **do**
    $\mathbf{m} \leftarrow \frac{1}{\mu} \sum_{i=1}^{\mu} \mathbf{x}_i$ ▷ recombination
    for $i = 1 \rightarrow \lambda$ **do**
      $\mathbf{x}'_i \leftarrow \mathbf{m} + \sigma \mathbf{C}^{-1/2} \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})$ ▷ mutation/stochastic variation
      $f'_i \leftarrow f(\mathbf{x}'_i)$ ▷ function evaluation
    end for
    if comma selection is enabled **then**
      select the best μ solutions from $\{\mathbf{x}'_i\}_{i=1}^{\lambda}$.
    else
      select the best μ solutions from $\{\mathbf{x}'_i\}_{i=1}^{\lambda} \cup \{\mathbf{x}_i\}_{i=1}^{\lambda}$.
    end if
    Set the new population $\{\mathbf{x}_i\}_{i=1}^{\mu}$ to the selected points.
    Control step-size σ and covariance matrix $\mathbf{C}$.
  end while
  return the best solution found since the beginning.
end procedure

numbers. Intuitively, such sequences span the search space more "evenly" than the pseudo-random numbers. It is widely used in numerical problems like the quasi-Monte-Carlo method (Niederreiter, 1992) to achieve a faster rate of convergence. The discrepancy of a random sequence can be viewed as a quantitative measure for the deviation from the uniform distribution. Thus, the low-discrepancy sequence is able to solve the same problem as the one treated here, namely to create more evenly distributed samples.

Due to the advantages of quasi-random sampling, it is also applied in genetic algorithms (Kimura and Matsumura, 2005) and evolution strategies (Teytaud and Gelly, 2007). Specifically, it has already been applied to the well-known Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001; Hansen et al., 2003). Teytaud and Gelly (2007) propose to replace the independent random Gaussian samples by a low-discrepancy sequence in the mutation operator. The method for generating quasi-random samples according to the Gaussian distribution is also proposed because the quasi-random samples are usually related to a uniform distribution. It is also argued that the efficiency of CMA-ES is

improved due to the bigger diversity of quasi-random samples. However, when applying the quasi-random sample and recombination operator, a systematic bias on the step-size adaptation is induced: the quasi-random samples are no longer independent from each other and thus for each highly anti-correlated samples, their recombination is much smaller on average compared to the Gaussian mutation. As the step-size adaptation mechanism typically depends on the expected size of the recombinations, quasi-random samples causes a downward bias in step-sizes.

2.2 Mirroring and Orthogonalization

The mirrored sampling technique (Brockhoff et al., 2010) is another method for obtaining "good" samples and it is successfully accelerating the convergence of ESs (Auger et al., 2010). It is a quite simple and elegant idea in which a single random mutation vector is used to create two search points. More specifically, instead of generating λ i.i.d. search points, only half of the mutation vectors are sampled during each ES generation, namely $\{\mathbf{z}_{2i-1}\}_{1\leq i\leq \lambda/2}$, $\mathbf{z}_i \sim \mathcal{N}(\mathbf{0}, \sigma^2\mathbf{C})$, where σ is the current global step size and $\mathbf{C}$ is the current covariance matrix. Each mutation vector $\mathbf{z}_{2i-1}$ is used to generate two offspring, the usual one $\mathbf{x}_{2i-1} = \mathbf{m} + \mathbf{z}_{2i-1}$ and the mirrored offspring $\mathbf{x}_{2i} = \mathbf{m} - \mathbf{z}_{2i-1}$. Those two offspring are *symmetric* or *mirrored* to the parental point $\mathbf{m}$.

In order to make the argument here clearer, the mutations sampled from the distribution are denoted as *realized* mutations. The mirrored sampling method is described in Algorithm 2, acting as an alternative to the random mutation operator in evolution strategies. For odd λ, it begins by generating $\lceil \lambda/2 \rceil$ offspring in the first generation, which results in $\lceil \lambda/2 \rceil$ mirrored offspring. Then, all of the realized offspring and $\lceil \lambda/2 \rceil - 1$ mirrored ones are used immediately while the extra one mirrored mutation is kept to the next iteration (Lines $18-21$). In the next iteration, the extra mirrored offspring is used (Lines $3-9$) and only $\lfloor \lambda/2 \rfloor$ realized mutations need to be drawn. The following generations repeat this procedure. The static variable $\mathbf{z}_{\text{last}}$ in Algorithm 2 stores the extra realized mutation vector. Here, the notation proposed in Brockhoff et al. (2010) is used such that any ES algorithm with the mirrored sampling is denoted by $(1 \overset{+}{,} \lambda_{\text{m}})$-ES.

By using mirrored sampling, the mirrored mutations are entirely dependent on the realized mutation samples and explore the reverse (or mirrored) directions such

that the mirrored counterpart of an unsuccessful mutation has a certain chance to realize an improvement.

Algorithm 2 Mirrored Sampling

procedure MIRRORED($\mathbf{m}, \sigma, \mathbf{C}, \lambda$)
  $\mathbf{B}, \mathbf{D} \leftarrow$ EIGEN-DECOMPOSITION($\mathbf{C}$)
  if $\lambda \mod 2 \neq 0$ **and** $\mathbf{z}_{\text{last}}$ is set **then**
    $\mathbf{x}_\lambda \leftarrow \mathbf{m} - \sigma\mathbf{B}\mathbf{D}\mathbf{z}_{\text{last}}$ ▷ mutation $\mathbf{z}_{\text{last}}$ from the last iteration
    $\lambda' \leftarrow \lambda - 1$
    Unset the static variable $\mathbf{z}_{\text{last}}$ ▷ Unset $\mathbf{z}_{\text{last}}$
  else
    $\lambda' \leftarrow \lambda$
  end if
  for $i = 1 \rightarrow \lambda'$ **do**
    if $i \mod 2 = 0$ **then**
      $\mathbf{x}_i \leftarrow \mathbf{m} - \sigma\mathbf{B}\mathbf{D}\mathbf{z}_{i-1}$ ▷ Mirroring
    else
      $\mathbf{z}_i \leftarrow \mathcal{N}(\mathbf{0}, \mathbf{I})$
      $\mathbf{x}_i \leftarrow \mathbf{m} + \sigma\mathbf{B}\mathbf{D}\mathbf{z}_i$
    end if
  end for
  if $\lambda' \mod 2 \neq 0$ **then** ▷ Odd number of mutations are created
    Set the static variable $\mathbf{z}_{\text{last}} \leftarrow \mathbf{z}_\lambda$ ▷ Save $\mathbf{z}_\lambda$ for the next iteration
  end if
  return $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\lambda\}$
end procedure

2.2.1 Deterministic Orthogonal Sampling

Orthogonal sampling, which denotes a the sampling approach utilizing orthogonal search directions, is another solution to enhance the mutation diversity. This sampling scheme can be found in Coordinate Descent (Schwefel, 1993), Adaptive Coordinate Descent (ACiD) (Loshchilov et al., 2011) and Rosenbrock's Local Search (Rosenbrock, 1960). Intuitively, by sampling on the mutually orthogonal directions, the samples spread quite diversely such that the search space would be

explored more evenly. This sampling method is not well suited for solving the problem discussed here, but gives a lot of inspiration for the proposed method.

Normally, in this approach, a set of orthogonal basis vectors $\Xi = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_n\}$ are maintained in each optimization iteration, determining the exploration directions. In each iteration, only a line search is conducted along a basis vector, which is achieved by sampling two trial points: one point is created by adding the basis to the current search point $\mathbf{m}$ while the other one is mirrored. In the next iteration, another basis vector in Ξ is picked for the exploration. The general framework of the optimization algorithm using this method is summarized below:

1. Initialize the search point $\mathbf{m}$, an orthonormal basis $\Xi = \{\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \ldots, \boldsymbol{\xi}_n\}$ and the step sizes $\{\sigma_1, \sigma_2, \ldots, \sigma_n\}$ for each vector in the basis.

2. If the termination condition is not satisfied, perform the following steps until (e) for each iteration. Let g be the iteration counter:

   (a) Choose vector $\boldsymbol{\xi}_i$ as the exploration direction where $i = g \mod n$ and generate one trial point: $\mathbf{x}_1 = \mathbf{m} + \sigma_i \boldsymbol{\xi}_i$.

   (b) For Rosenbrock's local search, goto (c). For the other methods, use vector $\boldsymbol{\xi}_i$ to generate the other trial point: $\mathbf{x}_2 = \mathbf{m} - \sigma_i \boldsymbol{\xi}_i$.

   (c) Evaluate the trial points $\mathbf{x}_1, \mathbf{x}_2$ (if $\mathbf{x}_2$ exists). Set the search point $\mathbf{m}$ to the one with the best fitness value.

   (d) Update the step size σ_i according to a deterministic or stochastic rule and increase the iteration counter g by one.

   (e) If $g \mod n = 0$, then update the basis Ξ according to the search points of the most recent n iterations.

When all vectors in Ξ are tried, the orthogonal basis Ξ is either unchanged or updated based on the successful vectors in the history. Note that the rules of the update may be different in every optimization algorithm. In Coordinate Descent, the basis is fixed to the standard basis in $\mathbb{R}^d$ during the process. In ACiD, the basis is updated by Adaptive Encoding (Hansen, 2008), which is the generalization of the covariance matrix update in CMA-ES. We deliberately term this sampling method as *deterministic* orthogonal sampling due to the fact that the update of the orthonormal basis is completely deterministic and it is easier to distinguish this sampling method from the *random* orthogonal sampling proposed here.

2.2.2 Mirrored Orthogonal Sampling

In this section, we propose a new sampling method based on the mirrored sampling technique. The motivation, the algorithm and the implementation are provided. This new method is motivated by the following observation: In mirrored sampling, half of the mutation vectors (the mirrored or dependent ones) completely depend on the other half (the realized or independent ones). Between these two sets of mutations, mirrored sampling is able to guarantee a significant difference between a realized mutation and its mirrored counterpart. In addition, the mirrored mutation is anti-parallel to the realized one and thus a mirrored pair would span two half-spaces almost surely, no matter how the search space is partitioned (such a pair can stay on the partition boundary with zero probability). However, within the realized half of mutations, everything is still purely random and not arranged evenly in high-dimensional space. Thus, the mirrored sampling technique still suffers from undesirable clustering of samples.

In order to improve the realized half of the mutations, we resort to the deterministic orthogonal sampling method (Section 2.2.1), where the mutations (new search points) are generated along a precomputed orthogonal basis and thus the minimal distance between samples is greatly enlarged. The disadvantage is that it only works in the single-parental evolutionary algorithm and only one of the orthogonal samples can be used in one evolution cycle, which limits its usability for the general (μ, λ)-ES. Instead of generating mutations along some orthogonal basis, it is proposed here to create half the mutations as "uniform random orthogonal vectors", in the sense that each vector is stochastic instead of the deterministic orthonormal basis and "uniform random" requires each vector to sample each direction evenly (Wang et al., 2014).

Definition 2.1. *The **uniform random orthogonal vectors** are defined as a set of random vectors $\{\boldsymbol{\mathcal{O}}_1, \boldsymbol{\mathcal{O}}_2, \ldots, \boldsymbol{\mathcal{O}}_k\} \subset \mathbb{R}^d$ $(k \leq d)$, satisfying the following three properties:*

1. *Orthogonality:* $\forall i \neq j \in \{1, 2, \ldots, k\}, \ \langle \boldsymbol{\mathcal{O}}_i, \boldsymbol{\mathcal{O}}_j \rangle = 0.$
2. $\chi(d)$*-distributed norm:* $\forall i \in \{1, 2, \ldots, k\}, \ \|\boldsymbol{\mathcal{O}}_i\| = \sqrt{\langle \boldsymbol{\mathcal{O}}_i, \boldsymbol{\mathcal{O}}_i \rangle} \sim \chi(d).$
3. *Uniformality: for each vector $\boldsymbol{\mathcal{O}}_i$, its normalization $\boldsymbol{\mathcal{O}}_i / \|\boldsymbol{\mathcal{O}}_i\|$ distributes* uniformly *on the unit sphere.*

Remark. 1) The norm of those vectors is restricted to $\chi(d)$-distribution for mimicking the behavior of the vector samples from the standard multivariate

Gaussian distribution. 2) The uniform distribution on the unit sphere is equivalent to the *rotation-invariant* property with respect to an arbitrary rotation matrix[1] $\mathbf{R} \in \mathbb{R}^{d\times d}$: the random vector $\mathbf{x}$ and the rotated one $\mathbf{x}' = \mathbf{R}\mathbf{x}$ are identically distributed. 3) Throughout this thesis, the dot product is taken for the inner product, namely $\langle \mathbf{x}, \mathbf{y} \rangle = \mathbf{x}^\top \mathbf{y}$.

The new mutation method is named *random orthogonal sampling*. For clarity, the mutation operator (that takes i.i.d. normal samples) in the canonical CMA-ES is called *standard random sampling*. In addition, the random orthogonal samples are rescaled and rotated according to the covariance matrix $\mathbf{C}$ before they are added to the parental point $\mathbf{m}$, which follows the same as the procedures as for the Gaussian mutations :

$$\mathbf{x}_{2i-1} \leftarrow \mathbf{m} + \sigma \mathbf{C}^{\frac{1}{2}} \boldsymbol{\mathcal{O}}_i, \ 1 \leq i \leq \lambda/2. \tag{2.1}$$

The $\mathbf{x}$'s are the new search points and σ denotes the step size. The implementation of the random orthogonal sampling algorithm and the validity of the implementation are discussed in the following section. Consider two i.i.d. vectors $\mathbf{x}$ and $\mathbf{y}$ drawn from a standard normal distribution. The expected value of the inner product of these two vectors is given as:

$$\mathbb{E}\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{n} \mathbb{E} x_i y_i = 0.$$

This indicates two independent standard normal vectors are orthogonal to each other in expectation. Intuitively, by generating random orthogonal samples, the mutations are derandomized such that the variance of the angle formed by a pair of mutations vanishes. Therefore, the search directions are guaranteed to be uncorrelated so that the mutation samples are spread over the space more evenly. In the next step, we combine mirrored sampling with random orthogonal sampling such that the remaining half of the search points are created by mirroring, which reads as follows:

$$\mathbf{x}_{2i} \leftarrow \mathbf{m} - \sigma \mathbf{C}^{\frac{1}{2}} \boldsymbol{\mathcal{O}}_i, \ 1 \leq i \leq \lambda/2. \tag{2.2}$$

Note that only using random orthogonal sampling is not sufficient for exploration due to the fact that random orthogonal vectors are only capable of spanning one orthant of the space, no matter how they are realized (just consider the

[1] A d-dimensional rotation matrix $\mathbf{R}$ satisfies conditions $\mathbf{R}^{-1} = \mathbf{R}^\top$ and $\det(\mathbf{R}) = 1$. All such matrices form so-called special orthogonal group $\mathrm{SO}(d)$.

Algorithm 3 Mirrored Orthogonal Sampling

1: **procedure** MIRRORED-ORTHOGONAL($\mathbf{m}, \sigma, \mathbf{C}, \lambda$)
2: $\mathbf{B}, \mathbf{D} \leftarrow$ EIGEN-DECOMPOSITION($\mathbf{C}$)
3: **if** $\lambda \mod 2 \neq 0$ **and** $\mathbf{z}_{\text{last}}$ is not set **then**
4: $\mathbf{x}_\lambda \leftarrow \mathbf{m} - \sigma\mathbf{B}\mathbf{D}\mathbf{z}_{\text{last}}$ ▷ $\mathbf{z}_{\text{last}}$: the unused mutation from the last iteration
5: $\lambda' \leftarrow \lambda - 1$ ▷ One offspring is already created
6: Unset the static variable $\mathbf{z}_{\text{last}}$. ▷ Unset $\mathbf{z}_{\text{last}}$ once it is used
7: **else**
8: $\lambda' \leftarrow \lambda$
9: **end if**
10: $p \leftarrow \lceil \lambda'/2 \rceil$
11: $\{\mathbf{z}_i\}_{i=1}^{p} \leftarrow$ ORTHOGONAL(p) ▷ sub-procedure, see Alg. 5
12: **for** $i = 1 \rightarrow p$ **do**
13: $\mathbf{x}_{2i-1} \leftarrow \mathbf{m} + \sigma\mathbf{B}\mathbf{D}\mathbf{z}_i$
14: $\mathbf{x}_{2i} \leftarrow \mathbf{m} - \sigma\mathbf{B}\mathbf{D}\mathbf{z}_i$ ▷ Mirroring
15: **end for**
16: **if** $\lambda' \mod 2 \neq 0$ **then** ▷ Save the unused mutation to the next iteration
17: Set the static variable $\mathbf{z}_{\text{last}} \leftarrow \mathbf{z}_p$
18: **end if**
19: **return** $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_\lambda\}$
20: **end procedure**

canonical basis in 3-D). Combining Eq. (2.1) and (2.2), the new sampling approach is completed and is called *mirrored orthogonal sampling*. In addition, any ES algorithm exploiting it is denoted as $(\mu \overset{+}{,} \lambda_{\text{m}}^{\text{o}})$-ES here. The detailed algorithm of the mirrored orthogonal sampling method is given as Algorithm 3. Note that an algorithm for generating random orthogonal Gaussian vectors (which is explained in the following) is invoked in line 10 and replaces the direct sampling of the Gaussian distribution. The remainder of this algorithm is basically the same as mirrored sampling (Alg. 2).

The mirrored orthogonal sampling method is a variant of mirrored sampling. In addition to mirroring, which ensures the difference within any mirrored pair, the orthogonalization method is exploited to guarantee the significant differences among realized mutations. Therefore, it is quite straightforward to compare the performance of mirrored orthogonal sampling to that of mirrored sampling and to

that of standard sampling. Such a comparison is presented in the experimental results (Section 2.4).

2.2.3 Implementation of Random Orthogonal Sampling

In order to implement random orthogonal sampling as introduced previously, the well-known Gram-Schmidt process (Björck, 1994) is exploited to generate the orthogonal samples. The Gram-Schmidt process is a method for orthogonormalizing a set of vectors in an inner product space, most commonly the Euclidean space $\mathbb{R}^d$. It takes a finite, linearly independent set $\mathcal{S} = \{\mathbf{v}_1, \ldots, \mathbf{v}_k\}$ for $k \leq d$ and generates an orthogonal set $\mathcal{S}' = \{\mathbf{u}_1, \ldots, \mathbf{u}_k\}$ that spans the same k-dimensional subspace of $\mathbb{R}^n$ as $\mathcal{S}$. The Gram-Schmidt process is shown in Alg. 4.

Algorithm 4 Gram-Schmidt orthonormalization

procedure GRAM-SCHMIDT($\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$)
  for $i = 2 \rightarrow k$ **do**
    for $j = 1 \rightarrow i - 1$ **do**
      $\mathbf{v}_i \leftarrow \mathbf{v}_i - \left(\mathbf{v}_i^\top \mathbf{v}_j / ||\mathbf{v}_j||^2\right) \mathbf{v}_j$ ▷ Othogonalizing $\mathbf{v}_i$ to $\mathbf{v}_j$
    end for
  end for
  for $i = 1 \rightarrow k$ **do**
    $\mathbf{v}_i \leftarrow \mathbf{v}_i / ||\mathbf{v}_i||$ ▷ Normalization
  end for
  return $\{\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k\}$
end procedure

Let p equal $\lambda/2$ again. In the first step, we sample p i.i.d. vectors from the standard normal distribution and record their norms (lengths), i.e.:

$$\mathcal{S} = \{\mathbf{s}_1, \ldots, \mathbf{s}_p\}, \quad \mathbf{s}_i \sim \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I}), \quad L_i = \|\mathbf{s}_i\|, \quad i = 1, \ldots, p. \tag{2.3}$$

Note that the Gram-Schmidt process is an orthonormalization method, normalizing the lengths of the i.i.d. samples. Therefore, the lengths have to be manually recorded so that we can restore mutation lengths for the samples. Then, processing $\mathcal{S}$ by the Gram-Schmidt process would give us a collection $\mathcal{S}'$ of random orthonormal vectors,

$$\mathcal{S}' = \left\{\mathbf{s}'_1, \ldots, \mathbf{s}'_p\right\} = \text{GRAM-SCHMIDT}(\mathcal{S}). \tag{2.4}$$

Note that each vector of $\mathbf{s}'_1, \ldots, \mathbf{s}'_p$ is of unit length and those vectors are orthogonal to each other. It is not very hard to see from Algorithm 4 that among all the resulting vectors, the direction of $\mathbf{s}'_1$ remains unchanged and the direction of $\mathbf{s}'_i$ depends on the set $\{\mathbf{s}_k\}_{k=1}^{i-1}$. Therefore, intuitively, the output vectors of the Gram-Schmidt process, $\{\mathbf{s}'_i\}_{i=1}^{p}$ are uniformly distributed on the unit sphere because the input vectors $\{\mathbf{s}_k\}_{k=1}^{p}$ are independent and identically distributed. Finally, we rescale all the $\mathbf{s}'_i$ by their corresponding original length:

$$\mathbf{z}_i = L_i \mathbf{s}'_i, \quad i = 1, \ldots, p. \tag{2.5}$$

The resulting random vectors are orthogonal Gaussian samples, which completes this process. A special situation takes place if p is greater than the dimensionality d: it is simply not possible to generate more than d distinct orthogonal vectors in $\mathbb{R}^d$. In this case, only d mutation samples are created using Equations (2.3), (2.4) and (2.5), and the remaining $p - d$ samples are created using the standard random sampling. The detailed procedure of orthogonal sampling is described in Algorithm 5. Lines $3 - 6$ correspond to Eq. (2.3). Through lines $7 - 17$, the Gram-Schmidt process is invoked and the number of samples p is handled properly. The advantage of this implementation is that there is no additional parameter to be considered. As for the time complexity, extra costs are spent in calling the Gram-Schmidt process, which is $O(k^2 d), k = \min\{p, d\}$.

To justify this implementation, it is possible to check the generated samples according to Definition 2.1: the orthogonality and restriction on the vectors length are immediately satisfied. The rotation-invariance of the vectors can be shown as follows. Firstly, the standard normal vectors are rotation-invariant, meaning that for every $\mathbf{s}_i \sim \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})$, it has the same distribution as $\mathbf{R}\mathbf{s}_i$, where $\mathbf{R}$ is the rotation matrix taken from $\mathrm{SO}(d)$. Second, the orthogonalization formula of the Gram-Schmidt process, which is encoded in Algorithm 4, reads as follows:

$$\mathbf{s}'_i = \mathbf{s}_i - \sum_{j=1}^{i-1} \frac{\langle \mathbf{s}_i, \mathbf{s}_j \rangle}{||\mathbf{s}_j||^2} \mathbf{s}_j, \quad i = 1, \ldots, p,$$

Now if an arbitrary rotation operator $\mathbf{R} \in \mathrm{SO}(d)$ is applied on $\mathbf{s}'_i$, the resulting vector is,

$$\mathbf{s}''_i = \mathbf{R}\mathbf{s}'_i = \mathbf{R}\mathbf{s}_i - \sum_{j=1}^{i-1} \frac{\langle \mathbf{R}\mathbf{s}_i, \mathbf{R}\mathbf{s}_j \rangle}{||\mathbf{R}\mathbf{s}_j||^2} \mathbf{R}\mathbf{s}_j, \quad i = 1, \ldots, p, \tag{2.6}$$

Note that it is valid to put $\mathbf{R}$ in the norm and the inner product (e.g., $||\mathbf{R}\mathbf{s}_j||$) because such matrices preserve the inner product. Finally, $\mathbf{R}\mathbf{s}_i$ is identically

Algorithm 5 Orthogonal sampling

procedure ORTHOGONAL(p)
  for $i = 1 \to p$ **do**
    $\mathbf{s}_i \leftarrow \boldsymbol{\mathcal{N}}(\mathbf{0}, \mathbf{I})$ ▷ generate standard normal vectors
    $L_i \leftarrow \|\mathbf{s}_i\|$ ▷ store the length
  end for
  $k \leftarrow \min\{p, n\}$ ▷ number of inputs for Gram-Schmidt
  $\{\mathbf{s}'_1, \ldots, \mathbf{s}'_k\} \leftarrow$ GRAM-SCHMIDT$(\{\mathbf{s}_1, \ldots, \mathbf{s}_k\})$ ▷ sub-procedure, see Alg. 4
  for $i = 1 \to k$ **do**
    $\mathbf{z}_i \leftarrow L_i \mathbf{s}'_i$ ▷ rescale the length
  end for
  if $k < p$ **then** ▷ more than n samples are needed
    for $i = 1 \to p - k$ **do**
      $\mathbf{z}_{k+i} \leftarrow \mathbf{s}_{k+i}$ ▷ copy the standard normal vectors
    end for
  end if
  return $\{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_p\}$
end procedure

distributed as $\mathbf{s}_i$ and it also holds for the remaining terms in the right-hand-side of Eq. (2.6). Therefore, $\mathbf{s}''_i$ is identically distributed as $\mathbf{s}'_i$ and therefore it is rotation-invariant. A more rigorous proof can be found in Eaton (1983).

2.3 Convergence Analysis of Mirroring and Orthogonalization

The theoretical analysis is twofold. First, the progress rate analysis for $(1, \lambda)$-ES, introduced in Beyer (1993), is applied to analyze mirrored sampling. In addition, such analysis gives a straightforward explanation why mirrored orthogonal sampling improves performance. There are no analytical results for mirrored orthogonal sampling yet while its empirical results are compared to random and mirrored sampling. Second, the progress rate analysis is applied again to provide an analytical result about the worst case performance of mirrored orthogonal sampling. This will (partially) explain the advantages of the new sampling method. For the analysis in the following, we will only consider the $(1, \lambda)$-ES with isotropic mutations on

the so-called sphere function[1], which is defined as:

$$f(\mathbf{x}) = (\mathbf{x} - \mathbf{x}^*)^\top (\mathbf{x} - \mathbf{x}^*),\ \mathbf{x} \in \mathbb{R}^d,$$

which has the global minimum $\mathbf{x}^*$. In addition, for the simplicity of our deviation, it is also assumed that the population size λ is **even** in the following analysis. In practice, when λ is odd, the corresponding progress rate can be bounded from below by using $\lambda - 1$ in the analysis and also be bounded from above by using $\lambda + 1$. Note that although some results (e.g., Fig. 2.3b) can be equivalently obtained, using the theoretical framework of *convergence rate analysis* (Brockhoff et al., 2010), we did not adopt such an analysis approach because the progress rate analysis gives more insight into why the proposed sampling method outperforms its counterparts. The link between progress rate and convergence rate is elaborated in Auger and Hansen (2011). For the convergence rate analysis on the mirrored sampling method, please see Auger et al. (2011a,b).

2.3.1 Mirrored Sampling

We will begin with the analysis of the $(1, \lambda_\mathrm{m})$-ES in order to show the reason why it outperforms random sampling and this analysis serves as a baseline for the comparison to mirrored orthogonal sampling, which is investigated here by the Monte Carlo simulation. The basics of the analysis are shown in Fig. 2.2a, following the same treatment as in Bäck (1995). Let $\mathbf{P}$ be the current parent which is at a distance R from the optimum $\mathbf{O}$. Due to the spherical symmetry, only the distance R is crucial, not the actual position of $\mathbf{P}$. The hypersphere centered at $\mathbf{P}$ has a radius of $\sigma\sqrt{d}$ and represents the mean length of isotropic Gaussian mutations: $\mathbf{z} = \boldsymbol{\mathcal{N}}(\mathbf{0}, \sigma^2\mathbf{I})$. The mirrored mutation is indicated as $-\mathbf{z}$. The progress of each mutation can be measured by the projection of $\mathbf{z}$ onto line $\mathbf{PO}$, which is the random variable z. Due to the invariance properties of isotropic Gaussian vectors, z is found to be normally distributed as $\mathcal{N}(0, \sigma^2)$, regardless of the actual direction of $\mathbf{PO}$. The progress made by mutation $\mathbf{z}$ is $R - r$. Furthermore, for a set of mutations $\{\mathbf{z}_i\}_{1 \le i \le \lambda}$, the actual progress made by all the mutations is $R - r_{1:\lambda}$, where $r_{1:\lambda}$ is the smallest order statistic among $\{r_i\}_{1 \le i \le \lambda}$. The progress rate is

[1] The sphere function is a standard function for theoretical analysis, reflecting local convergence properties of ESs.

defined in Beyer (2013):

$$\varphi_{1,\lambda} = \mathbb{E}\left\{R - r_{1:\lambda}\right\} \simeq \mathbb{E}\left\{R - \sqrt{(R - z_{\lambda:\lambda})^2 + \sigma^2 d}\right\}. \tag{2.7}$$

The approximation in Eq. (2.7) takes place when we replace $||\mathbf{z}||$ by $\sigma\sqrt{d}$. Note that $z_{\lambda:\lambda}$, the largest order statistic among the projections of all the mutations onto **PO**, determines the expectation above.

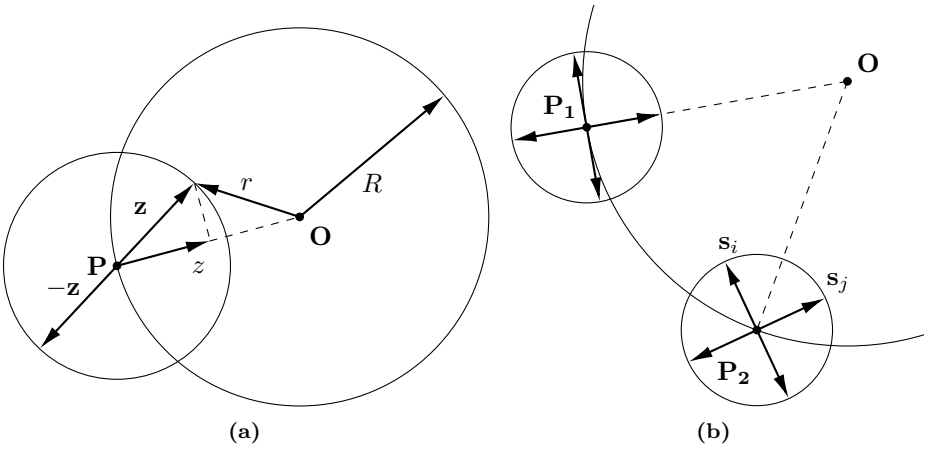


Figure 2.2: (a). Schematic diagram for the progress rate analysis on the sphere function. The mutations are centered at **P**, which is at distance R from the optimum **O**. (b) In 2-D, the diagram shows the best case ($\mathbf{P}_1$) of progress and the worst case ($\mathbf{P_2}$) for mirrored orthogonal sampling on the sphere function.

For the mirrored sampling, if z_i is the projection of mutation $\mathbf{z}_i$ onto **PO**, then the projection of its mirrored mutation $-\mathbf{z}_i$ is $-z_i$ by symmetry. Thus, the set of the projections of all the mutations of mirrored sampling can be written as $\{z_i, -z_i\}_{1\leq i\leq \lambda/2}$. Let $P^{\mathrm{m}}_{\lambda:\lambda}(Z \leq z)$ denote the cumulative probability distribution (c.d.f.) of the largest order statistic among $\{z_i, -z_i\}_{1\leq i\leq \lambda/2}$. Suppose for every $z \geq 0$, in order to facilitate the condition in $P^{\mathrm{m}}_{\lambda:\lambda}(Z \leq z)$, namely the largest order statistic is less than or equal to z, we must have $z_i \leq z, -z_i \leq z$ for all the z_i, which implies $-z \leq z_i \leq z$ for all the z_i. The intuition is that all random mutation points are required to be sampled less than or equal to z. In addition, because mirrored mutations are generated by reversing the signs of random mutations, every random mutation also needs to be bigger than $-z$, otherwise the mirrored

counterpart of an outlier would be larger than z and fails the condition. The argument reads as follows:

$$\begin{aligned} P^{\mathrm{m}}_{\lambda:\lambda}(Z \leq z) &= \left[\Pr\left(-z < Z \leq z\right)\right]^{\lambda/2} \\ &= \left[\Phi\left(\frac{z}{\sigma}\right) - \Phi\left(-\frac{z}{\sigma}\right)\right]^{\lambda/2} \\ &= \left[2\Phi\left(\frac{z}{\sigma}\right) - 1\right]^{\lambda/2}, \quad \forall z \geq 0. \end{aligned}$$

Note that $\Phi(\cdot)$ stands for the c.d.f. of a standard normal random variable. Then, in case of $z < 0$, the cumulative probability should be always 0. The reason is that if a realized mutation is sampled negative, then its mirrored counterpart would be positive. Therefore the largest order statistics could not be negative ever. In total, the c.d.f. of the largest order statistic is summarized as:

$$P^{\mathrm{m}}_{\lambda:\lambda}(Z \leq z) = \begin{cases} \left[2\Phi\left(\frac{z}{\sigma}\right) - 1\right]^{\lambda/2} & \forall z \geq 0, \\ 0 & \text{otherwise.} \end{cases}$$

And its probability density function is:

$$p^{\mathrm{m}}_{\lambda:\lambda}(z) = \begin{cases} \lambda p(\frac{z}{\sigma}) \left[2\Phi\left(\frac{z}{\sigma}\right) - 1\right]^{\lambda/2-1} & \forall z \geq 0, \\ 0 & \text{otherwise.} \end{cases} \tag{2.8}$$

where $p(\cdot)$ denotes the probability density function (p.d.f.) of a standard normal distribution. This density can be compared to the largest order statistic among the same projections of random samples (Beyer, 1993):

$$p_{\lambda:\lambda}(z) = \lambda p\left(\frac{z}{\sigma}\right) \Phi\left(\frac{z}{\sigma}\right)^{\lambda-1}.$$

In 5-D with $\lambda = 10$, we plot the c.d.f. and p.d.f. of mirrored sampling and random sampling in Fig. 2.3a. It is clear from the figure that the distribution of the largest projection for mirrored sampling is shifted to the right, compared to that for the Gaussian sampling and therefore the corresponding distribution of projections is shifted towards larger values. This advantage would affect the progress rate (as shown in the following) and is the main reason why mirrored sampling has a better performance than random sampling. By using the normalized quantities,

$$\varphi^* = \varphi\frac{d}{R}, \quad \sigma^* = \sigma\frac{d}{R},$$

and applying the same derivation as in Beyer (1993), the progress rate of $(1, \lambda_{\mathrm{m}})$-ES can be obtained by expanding the expectation in Eq. (2.7) (the details of the

simplification are not shown here):

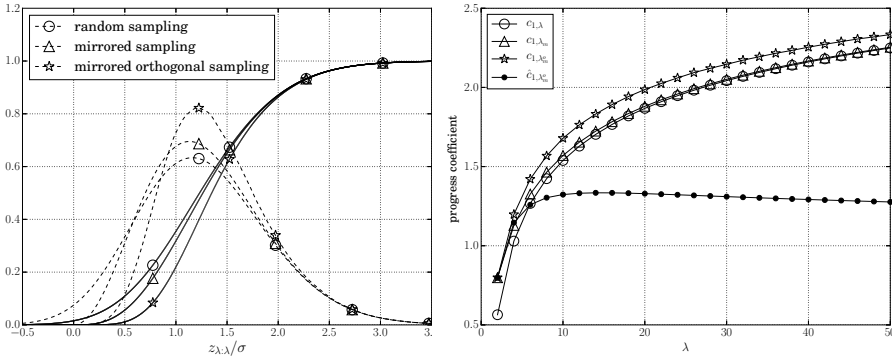
$$\begin{aligned}
\varphi^*_{1,\lambda_\mathrm{m}} &= \int_0^\infty z p^{\mathrm{m}}_{\lambda:\lambda}(z)\,\mathrm{d}z - \frac{(\sigma^*)^2}{2} \\
&= \lambda \int_0^\infty z p\left(\frac{z}{\sigma}\right)\left[2\Phi\left(\frac{z}{\sigma}\right)-1\right]^{\lambda/2-1}\,\mathrm{d}z - \frac{(\sigma^*)^2}{2} \\
&= \sigma^*\left(\lambda \int_0^\infty z' p(z')\left[2\Phi\left(z'\right)-1\right]^{\lambda/2-1}\,\mathrm{d}z'\right) - \frac{(\sigma^*)^2}{2} \\
&= c_{1,\lambda_\mathrm{m}}\sigma^* - \frac{(\sigma^*)^2}{2}. \qquad (2.9)
\end{aligned}$$

In the equation above, the integral about the normalized largest projection $z' = z/\sigma$ computes its expectation and it is known as the *progress coefficient* from (Beyer, 1993). We denote it by c_{1,λ_m} here. It can be compared to the progress coefficient of random sampling, which reads:

$$c_{1,\lambda} = \lambda \int_{-\infty}^{\infty} z p(z) \Phi(z)^{\lambda-1}\,\mathrm{d}z.$$

Note that the progress rate of random sampling can be easily obtained by replacing c_{1,λ_m} in Eq. (2.9) with $c_{1,\lambda}$. Numerically, we plot the progress coefficients of random sampling and mirrored sampling against population size in Fig. 2.3b. The mirrored sampling (the curve marked by triangles) shows a small yet obvious advantage compared to the random sampling for small population sizes. In larger populations, these two converging curves imply that mirrored sampling provides no speed-up compared to the standard ES algorithm. Thus, the application of mirrored sampling should be limited to the small population setting.

For mirrored orthogonal sampling, we would like to use the same approach as for the mirrored sampling analysis above. However, it is hard to analytically obtain the c.d.f. and the density function of the largest projection onto **PO** of the mirrored orthogonal sampling. Therefore, we compute its c.d.f. and density function empirically by Monte-Carlo simulation. For the simulation, the population size λ is set to $2d$. The mirrored orthogonal samples are projected onto **PO** and the largest projections are stored, from which the c.d.f. is estimated. The results are also summarized in Fig 2.3. In Fig. 2.3a, the c.d.f. of mirrored orthogonal sampling (the solid curve marked by stars) is more likely to distribute samples towards bigger values compared to the c.d.f. of mirrored sampling. As a consequence, in Fig. 2.3b, the progress coefficients of mirrored orthogonal sampling are significantly bigger than those of mirrored sampling, even in a large population.



(a) **(b)**

Figure 2.3: (a): The c.d.f. (solid) and p.d.f. (dashed) of the largest projection (normalized) onto **PO** for random, mirrored and mirrored orthogonal sampling. The dimension d is set to 5 and $\lambda = 10$ for all curves. There are 10^6 trials used in the estimation for mirrored orthogonal sampling. For the other sampling methods, the curves show the corresponding analytical results. (b): Progress coefficients against population size λ for random sampling, mirrored sampling and mirrored orthogonal sampling. The dimensionality d is set to $\lambda/2$ for all curves. The black dotted curve is the lower bound on the progress coefficient of mirrored orthogonal sampling.

2.3.2 Mirrored Orthogonal Sampling

The worst case analysis of mirrored orthogonal sampling is conducted when the population size is set to $2d$. We will call such population setting as "full mutations". Under this condition, the progress rate is maximized (as will be explained later) and it is possible to provide analytical results. The progress under the condition $\lambda < 2d$ will be also discussed later. In 2-D with $\lambda = 4$, the worst case (together with best case) of progress for $(1, \lambda^{\circ}_{\mathrm{m}})$ is shown in Fig. 2.2b. Suppose the step size $\sigma = 1$ here for simplification. In the mutations centered at $\mathbf{P_1}$, there is one mutation pointing to the optimum $\mathbf{O}$ and therefore this mutation performs optimally. We call this mutation scenario the best case of progresses. The progress coefficient in this case is the expectation of the standard norm mutation length. It serves as the upper bound of the progress coefficient and is the same for random, mirrored and mirrored orthogonal sampling.

The worst case of progress is indicated by the mutations centered at $\mathbf{P_2}$ in which

the angle formed by the line segment $\mathbf{P_2O}$ and mutation $\mathbf{s}_i$ is the same as the one ($\pi/4$ as shown in the figure) formed by $\mathbf{P_2O}$ and $\mathbf{s}_j$. In this scenario, the expected projections of $\mathbf{s}_i$ and $\mathbf{s}_j$ are the same. It is not possible to make the expected projection of one mutation smaller without rendering the expected projection of the other one larger. For example, if we rotate $\mathbf{s}_j$ a little bit clockwise, then its projection becomes smaller. However, in the meanwhile $\mathbf{s}_i$ is also rotated and its projection gets larger. Consequently, the largest projection of all the mutations becomes larger. Therefore, among all the possible mutation scenarios, $\mathbf{P}_2$ gives the lower bound of the largest projection of mutations onto $\mathbf{P_2O}$. Recall from Eq. (2.7) that the progress made by $(1, \lambda)$-ES is determined by the largest projection. Thus, the scenario $\mathbf{P}_2$ is the worst case of progress. Under the "full" mutation condition, we generalize the worst case for arbitrary dimensions. Let the mirrored orthogonal samples be denoted as $\{\boldsymbol{\mathcal{O}}_i, -\boldsymbol{\mathcal{O}}_i\}_{1 \le i \le \lambda/2}$. The unit vectors along the orthogonal mutations are defined as:

$$\mathbf{u}_i = \frac{\boldsymbol{\mathcal{O}}_\mathbf{i}}{\|\boldsymbol{\mathcal{O}}_i\|}. \tag{2.10}$$

Combining the unit vectors for mirrored mutations, all the unit vectors are $\{\mathbf{u}_i, -\mathbf{u}_i\}_{1 \le i \le \lambda/2}$. The worst case of progress is defined by the following conditions: for all the unit vectors, the linear combination with equal weights (denoted as $\mathbf{d}$ in the following) of $\lambda/2 = n$ unit vectors points to the optimum $\mathbf{O}$ and also to the reverse direction of the gradient of the sphere function, which reads:

$$\mathbf{d} = \sum_{k=1}^{\lambda/2} a_k \mathbf{u}_k = -\alpha \nabla f(\mathbf{x}), \quad \alpha > 0, a_k = \pm 1,$$

where a_k is a sign operator to select among $\mathbf{u}_k, -\mathbf{u}_k$. Then the scalar projection of mutation $\boldsymbol{\mathcal{O}}_i$ onto $\mathbf{d}$ is expressed as:

$$\mathrm{proj}_\mathbf{d}(\boldsymbol{\mathcal{O}}_i) = \frac{\langle \boldsymbol{\mathcal{O}}_i, \mathbf{d} \rangle}{\|\mathbf{d}\|} = \frac{\sum_{k=1}^{\lambda/2} a_k \langle \boldsymbol{\mathcal{O}}_i, \mathbf{u}_k \rangle}{\left\| \sum_{k=1}^{\lambda/2} a_k \mathbf{u}_k \right\|} = \frac{\sum_{k=1}^{\lambda/2} a_k \langle \boldsymbol{\mathcal{O}}_i, \boldsymbol{\mathcal{O}}_k \rangle / \|\boldsymbol{\mathcal{O}}_k\|}{\left\| \sum_{k=1}^{\lambda/2} a_k \mathbf{u}_k \right\|} = \frac{a_k \|\boldsymbol{\mathcal{O}}_i\|}{\sqrt{d}}.$$

Note that we substitute the expression of $\mathbf{u}_i$ (Eq. (2.10)) in the derivation above. The projections of all the mutations onto $\mathbf{d}$ can be written:

$$\mathrm{proj}_\mathbf{d} = \left\{ \frac{\|\boldsymbol{\mathcal{O}}_i\|}{\sqrt{d}}, -\frac{\|\boldsymbol{\mathcal{O}}_i\|}{\sqrt{d}} \right\}_{i=1}^{\lambda/2}.$$

The largest order statistic of all the projections is the maximum of $\mathrm{proj}_\mathbf{d}$:

$$\max\{\mathrm{proj}_\mathbf{d}\} = \max_{1 \le i \le \lambda/2} \left\{ \frac{\|\boldsymbol{\mathcal{O}}_i\|}{\sqrt{d}}, -\frac{\|\boldsymbol{\mathcal{O}}_i\|}{\sqrt{d}} \right\} = \frac{1}{\sqrt{d}} \max_{1 \le i \le \lambda/2} \{\|\boldsymbol{\mathcal{O}}_i\|\} = \frac{z}{\sqrt{d}}.$$

Here we denote the maximal mutation length by z. Note that the $\|\boldsymbol{\mathcal{O}}_i\|$ are independently distributed according to $\chi(n)$ (see Algorithm 5). Therefore, the density function of the maximal mutation length among $\lambda/2$ mutations reads:

$$p_{\frac{\lambda}{2}:\frac{\lambda}{2}}(z) = \frac{\lambda}{2} p_\chi(z) \left(F_\chi(z)\right)^{\frac{\lambda}{2}-1},$$

where $p_\chi(\cdot), F_\chi(\cdot)$ denote the density and c.d.f. of the $\chi(n)$ distribution, respectively. The worst case progress coefficient of mirrored orthogonal sampling, which is the expectation of $z/\sqrt{n}$, is denoted as $\hat{c}_{1,\lambda^{\text{o}}_{\text{m}}}$ and derived as follows:

$$\begin{aligned}
\hat{c}_{1,\lambda^{\text{o}}_{\text{m}}} &= \int_0^\infty \frac{z}{\sqrt{d}} p_{\frac{\lambda}{2}:\frac{\lambda}{2}}(z)\,\mathrm{d}z \\
&= \frac{\lambda}{2\sqrt{d}} \int_0^\infty z p_\chi(z) \left(F_\chi(z)\right)^{\frac{\lambda}{2}-1}\,\mathrm{d}z \\
&= \sqrt{d} \int_0^\infty z p_\chi(z) \left(F_\chi(z)\right)^{n-1}\,\mathrm{d}z. \qquad (2.11)
\end{aligned}$$

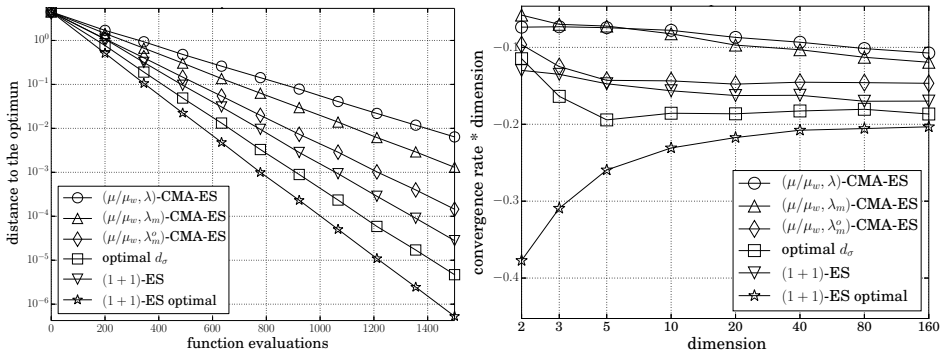
The last equation results from the fact that we picked the special population size $\lambda = 2d$ from the previous analysis setting. Eq. (2.11) is numerically evaluated and plotted in Fig. 2.3b. The curve for the worst case is above 1 and roughly stays constant when λ increases. It provides a non-zero lower bound of the progress coefficient of mirrored orthogonal sampling with "full mutations", which indicates no matter in what scenario, the mirrored orthogonal sampling with "full mutations" is going to guarantee positive progress on the sphere function. To compare, for random sampling, the lower bound of the progress coefficient is zero because it is possible to have all the mutations generated as in Fig. 2.1, where no mutation makes progress. For mirrored sampling, the lower bound of the progress coefficient is also zero because it is possible that all the mutations are generated in a tangent space of the local gradient, in which all the vectors are orthogonal to the gradient. Thus, the non-zero lower bound of mirrored orthogonal sampling with "full mutations" is its main advantage over the random and mirrored sampling.

In the case that mirrored orthogonal sampling does not use "full mutations", namely $\lambda < 2d$, the progress rate would be reduced in contrast to the "full mutations" case. This is because it can now happen that some subspace could not be covered when $\lambda < 2d$. Therefore, it is possible that the subspace in which the progress can be made is simply unexplored.

2.4 Empirical Results on Mirroring and Orthogonalization

For the multi-parental variants of ES, we only consider their empirical convergence rates here. Similar to the convergence rate estimation in Loshchilov et al. (2011), the effect of the mirrored orthogonal sampling technique on the sphere function is investigated empirically by incorporating it into the well-known CMA-ES algorithm.

On the 20-D sphere function, the convergence rates of the $(\mu, \lambda_{\mathrm{m}}^{\circ})$-CMA-ES and other comparable ES variants are illustrated in Fig. 2.4a. The empirical convergence rate is estimated as the average slope of the convergence curve over 200 runs. For



(a) Sphere 20-D **(b)** Convergence rate

Figure 2.4: The comparison of empirical convergence rates on the sphere function. All the results are estimated over 200 runs. The suggested λ setting $4 + \lfloor 3 \ln d \rfloor$ (Hansen, 2006) is used for all the CMA-ES variants (a): Plot of the history of distance to the global optimum against the number of function evaluations for four ES algorithms: $(\mu, \lambda_{\mathrm{m}}^{\circ})$-CMA-ES with standard d_σ and optimal d_σ, $(\mu, \lambda_{\mathrm{m}})$-CMA-ES, standard (μ, λ)-CMA-ES and $(1 + 1)$-ES in dimension 20. (b): Plot of convergence rate $\times$ dimensionality against the dimensionality for different algorithms on the sphere function, using 1500 function evaluations.

all the CMA-ES variants tested here, the default settings of population size are applied (Hansen, 2006): $\lambda = 4 + \lfloor 3 \ln d \rfloor, \mu = \lfloor \lambda/2 \rfloor$. The legend "$(1 + 1)$-ES" represents the $(1+1)$-ES with $1/5$ success rule step size control while the "$(1+1)$-ES optimal" is for the $(1 + 1)$-ES with scale-invariant step size setting $\sigma = \frac{1.2}{d} \|\mathbf{x}^{(k)}\|$,

which proves to be the optimal step size setting on the sphere function (Loshchilov et al., 2011).

The pairwise selection is always used if the mirroring operation is present in the sampling procedure. The mirrored sampling CMA-ES is denoted as "$(\mu, \lambda_{\mathrm{m}})$-CMA-ES ". The curve labeled by "$(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES " stands for the mirrored orthogonal CMA-ES. In addition, "optimal d_σ" represents the mirrored orthogonal CMA-ES using the optimal d_σ[1] tuning on the sphere function. Due to the empirical results, the convergence of $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES (marked by diamonds) is slower but close to that of the $(1+1)$-ES (marked by upside-down triangles) while the $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES using the optimal parameter settings gradually catches the convergence rates of the optimal $(1+1)$-ES in high dimensions.

The relation between the empirical convergence rate and the dimensionality is shown in Fig. 2.4b. The algorithms tested here are the same as Fig. 2.4a. It is obvious that there is a leap of convergence rates between the CMA-ES and its mirrored orthogonal competitor. The advantages of the mirrored orthogonal CMA-ES over the mirrored CMA-ES are significant and preserved even for large dimensions. The upper limit of the $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES on the sphere function is shown by the convergence rates achieved under the optimal d_σ tuning, which is even better than $(1+1)$-ES for almost all the dimensions. However, the optimal d_σ setting on the sphere function turned out to be not robust when considering other fitness functions and therefore is not used.

2.4.1 Experiments on BBOB

The mirrored orthogonal version of CMA-ES with pairwise selection has been tested on the noiseless Black-Box Optimization Benchmark (BBOB) (Hansen et al., 2010). By using the automatic comparison procedures provided in this benchmark, the BBOB results of $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES are compared to those of $(\mu, \lambda_{\mathrm{m}})$-CMA-ES and (μ, λ)-CMA-ES.

Experimental Settings The three algorithms, $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES, $(\mu, \lambda_{\mathrm{m}})$-CMA-ES and (μ, λ)-CMA-ES are benchmarked on BBOB-2012 and their results are compared and processed by the post-processing procedure of BBOB. The BBOB

[1]For the definition of the parameter d_σ, please see Hansen et al. (2003).

parameter settings of the experiment are the same for all the tested ES variants. The initial global step size σ is set to 1. The maximum number of function evaluations is set to $10^4 \times d$. The initial solution (initial parent) is uniformly sampled in the hyper-box $[-4, 4]^n$. The dimensions tested in the experiment are $d \in \{2, 3, 5, 10, 20, 40\}$. The experiment employs a relatively large population size, namely $2d$, the result of which is denoted as **large population**. In this experiment, the strategy parameters used are exactly the same for the three ES variants. The modified d_σ is not used because it is tuned under the default population setting instead of the large population setting.

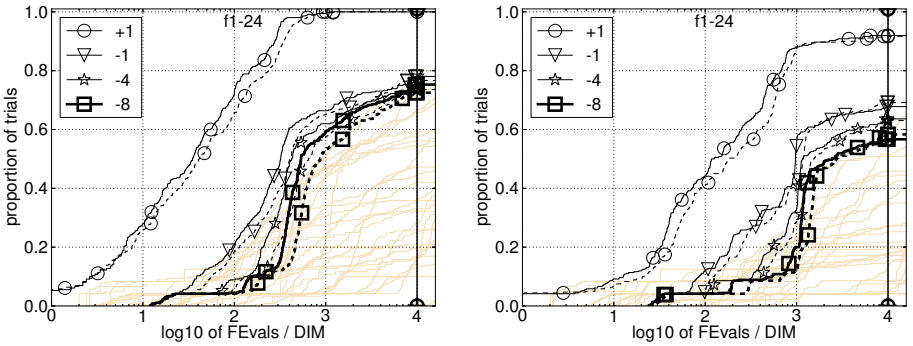


Figure 2.5: Left: $d = 5$. Right: $d = 20$. For the large population, the empirical cumulative distributions (ECDF) of run lengths (the number of function evaluations divided by dimension) for $(\mu, \lambda_{\mathrm{m}}^{\mathrm{o}})$-CMA-ES (solid lines) and $(\mu, \lambda_{\mathrm{m}})$-CMA-ES (dashed lines) needed to reach a target value.

Results and Discussion The BBOB noiseless testbed (Hansen et al., 2009) contains 24 test functions which are classified into several groups as separable, ill-conditioned or multi-modal functions. The performance of tested algorithms are compared using the aggregated empirical cumulative distribution functions (ECDFs) of run length over all the test functions are presented here. The ECDF of run length estimates the cumulative distribution of the function evaluations consumed in ESs, with respect to a given precision target. The comparisons between the mirrored orthogonal sampling and its mirrored sampling competitor are illustrated in Fig. 2.5. From the comparisons between the ECDFs of 5-D (left half) to that of 20-D (right half), it is obvious that the amount of the improvement is still significant when the dimensionality increases. The experimental results

for the large population suggest that the newly proposed mirrored orthogonal sampling technique would be most suitable in the case where the population size is about two times the dimensionality.

2.5 Efficient Global Optimization

Apart from the aforementioned mutation methods, that are directly defined by the realization of some probability distributions, in this section we shall extract and discuss the special method of creating new solutions from the *Efficient Global Optimization* (EGO) (Jones et al., 1998; Močkus, 1975, 2012) algorithm. Briefly, in this mutation method, the new candidate solution is obtained via the optimization on a well-specified utility function, which quantifies the potential "gain" in fitness value by evaluating this new solution. Therefore, we shall call this method **Mutation by Optimization**. In general, the utility function depends on a stochastic model (or statistical estimator) $\hat{f}$ of the fitness function f and statistical properties of this estimator $\hat{f}$, e.g., the *mean squared error* of the estimate: $s^2(\mathbf{x}) = \mathbb{E}\{\hat{f}(\mathbf{x}) - f(\mathbf{x})\}^2, \forall \mathbf{x} \in \mathrm{S}$. Note that $\hat{f}$ is usually called predictor in machine learning and we shall use these terminologies interchangeably here. Typically, the estimator $\hat{f}$ is a function of (random) sample $\mathrm{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathrm{S}$ and its corresponding fitness values $\mathbf{y} = (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))^\top$. The utility function is then denoted as $\mathscr{A}\colon \mathrm{S} \to \mathbb{R}$. The new candidate solution $\mathbf{x}'$ is proposed by solving the following problem[1]:

$$\mathbf{x}' = \arg\max_{\mathbf{x} \in \mathrm{S}} \mathscr{A}(\mathbf{x}; \Theta), \tag{2.12}$$

where Θ is a set of parameters that $\mathscr{A}$ might rely on (see Eq. (4.2) for example). In the literature, $\mathscr{A}$ is termed as *infill criterion* (Jones, 2001) or *acquisition function* (Martinez-Cantin, 2014) and we shall adopt the former throughout this thesis. Some commonly used infill criteria include: Expected Improvement, Probability of Improvement and Lower Confidence Bound. Typically, infill criteria are designed to make a balance between the model prediction $\hat{f}$ and the MSE of prediction (uncertainty) s^2. The detailed discussion on infill criteria can be found in Chapter 4. Built on the mutation by optimization mechanism, Efficient Global Optimization (also referred as Bayesian optimization (Močkus, 2012)) is able to perform a direct

[1] Some of the infill criteria are subject to minimization by the original definition. However, it can be equivalently transformed into the maximization task.

optimization efficiently on expensive objective functions. EGO is a *sequential* design strategy and it is presented in Alg. 6.

Algorithm 6 Efficient Global Optimization

procedure EGO($f, \mathscr{A}, \mathrm{S}$) ▷ f: objective function, $\mathscr{A}$: infill criterion, S: search space
  Sample the initial design $\mathrm{X} \subset \mathrm{S}$
  Evaluate $\mathbf{y} \leftarrow (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))^\top$
  Construct the fitness estimator $\hat{f}$ on $\mathrm{X}, \mathbf{y}$.
  while the stop criteria are not fulfilled **do**
    $\mathbf{x}' \leftarrow \arg\max_{\mathbf{x} \in \mathrm{S}} \mathscr{A}(\mathbf{x}; \Theta)$
    Evaluate $y' \leftarrow f(\mathbf{x}')$
    $\mathrm{X} \leftarrow \mathrm{X} \cup \{\mathbf{x}'\}, \mathbf{y} \leftarrow (\mathbf{y}^\top, y')^\top$
    Re-construct the estimator $\hat{f}$ on $\mathrm{X}, \mathbf{y}$
  end while
end procedure

Randomness It seems that the new location provided by solving Eq. (2.12) is *deterministic* and thus mutation by optimization is, by definition, not a stochastic variation method. However, although most of the infill criteria exhibit smooth landscapes, they are also highly *multi-modal* (see Section 4.2). This causes a practical difficulty in solving Eq. (2.12) globally. Although the exact solver, e.g., branch-and-bound (Jones et al., 1998) has been adapted for this task, such a solver only works a few types of infill criteria/covariance functions (see the discussion on the stochastic model below) and thus becomes inapplicable in practice. Instead, a stochastic optimization algorithm, e.g., evolutionary algorithms, is frequently applied for the infill criteria maximization, yielding random solutions. Thus, mutation by optimization is indeed **practically stochastic** and its randomness is determined by the underlying stochastic optimizer of the infill criterion $\mathscr{A}$.

Stochastic model To approximate the unknown objective function, the **Gaussian process regression** (**GPR**)/**Kriging** (Rasmussen and Williams, 2006; Krige, 1951) is used in EGO. It is a stochastic interpolation approach, which stems from earth science (Krige, 1951) and originally targets mining problems. It has been widely used as a surrogate model in the design and analysis of computer

experiments (Sacks et al., 1989; Santner et al., 2003), where the time-consuming simulations (computer models) are replaced by predictions from a Kriging model. In this technique, the objective function f is modeled as a realization of a Gaussian process Y. Conditioning on the data set $(\mathrm{X}, \mathbf{y})$, the so-called posterior process is obtained via Bayesian inference. The Gaussian Process Y is completely defined by a prescribed mean (trend) function $t(\mathbf{x})$ and a covariance function $k(\mathbf{x}, \mathbf{x}')$ (Rasmussen and Williams, 2006, Chapter 2.2):

$$\begin{aligned} t(\mathbf{x}) &= \mathbb{E}Y(\mathbf{x}), \\ k(\mathbf{x}, \mathbf{x}') &= \mathbb{E}\{(Y(\mathbf{x}) - t(\mathbf{x}))(Y(\mathbf{x}') - t(\mathbf{x}'))\}. \end{aligned}$$

When the mean function is assumed to be constant and unknown, the method is called *Ordinary Kriging* (OK) and is typically used in EGO. Now we wish to predict $f(\mathbf{x})$ at an unknown location $\mathbf{x} \in \mathrm{S}$. Without giving the derivation, the conditional distribution of Y on the observations $\mathbf{y}$ is a Gaussian distribution (Rasmussen and Williams, 2006):

$$Y(\mathbf{x}) \mid \mathbf{y} \sim \mathcal{N}\left(\hat{f}(\mathbf{x}), s^2(\mathbf{x})\right). \tag{2.13}$$

The conditional mean function $\hat{f}(\cdot)$ is used as the predictor for f while $s^2(\cdot)$ gives the MSE of the predictor $\hat{f}$. For the detailed discussion on Kriging/GPR, please see Chapter 3.

Step-wise risk Conceptually, EGO is a greedy step-wise search strategy. For instance, the model prediction can be set as the infill criterion, namely $\mathscr{A} \coloneqq \hat{f}$, giving the complete "trust" on the stochastic model. However, this is a highly *risky* action as the model is typically not accurate in the early stage of the optimization. To quantify the risk of the step-wise maximization of infill criteria, it is straightforward to calculate the *rate of failure*:

$$r = \Pr(f(\mathbf{x}) > f_{\min}) = 1 - \Pr(f(\mathbf{x}) < f_{\min}),$$

where $f_{\min} \coloneqq \min\{\mathbf{y}\}$ is the current minimal function value. Note that it is not feasible to calculate this rate due to the fact that there is a lack of the distribution information (e.g., which parametric family should be taken) about f, when assuming f is stochastic[1]. Thus, the typical approach is to approximate the

[1]Theoretically, this can be done by considering all the probabilistic models $\mathscr{M}$ (e.g., Gaussian/Student's t-process) for f and assuming a distribution over the models (e.g., a Dirichlet process). Then $r = \mathbb{E}\{\Pr(f(\mathbf{x}) > f_{\min} \mid \mathscr{M})\}$.

rate of failure under a specific distribution on f. When choosing the Kriging/GPR for f (Eq. (2.13)), the risk approximate is:

$$\hat{r} = 1 - \Pr(Y(\mathbf{x}) < f_{\min} \mid \mathbf{y}).$$

Note that $\Pr(Y(\mathbf{x}) < f_{\min} \mid \mathbf{y})$ is also a commonly used infill criterion, called *probability of improvement* (Eq. (4.6)). From the perspective of step-wise risks, it is interesting to compare EGO with the well-known *Simulated Annealing* (SA) algorithm (Agrawal et al., 1995):

- In SA, each candidate location $\mathbf{x}'$ that is worse than its parent $\mathbf{x}$ is accepted with the probability:

  $$r_{\text{SA}} = \exp\left(-\frac{f(\mathbf{x}') - f(\mathbf{x})}{t}\right),$$

  where $t \in \mathbb{R}_{>0}$ is the current temperature of SA. In other words, the step-wise risk of SA is r_{SA}.

- In EGO, each mutation $\mathbf{x}'$ obtained from Eq. (2.12) is always accepted and the step-wise risk of this action is $\hat{r}$.

From this conceptual comparison, it is obvious that EGO has no control over the step-wise risk if the probability of improvement is not chosen as the infill criterion. However, when using the probability of improvement, the resulting algorithm behaves very exploitative (see Section 4.2). To make a trade-off between exploitation and exploration, it is possible to enforce maximal risk (minimal probability of improvement) on the infill criterion maximization:

$$\begin{aligned} \arg\max_{\mathbf{x} \in \mathrm{S}} \quad & \mathscr{A}(\mathbf{x}; \Theta) \\ \text{subject to} \quad & \hat{r} < \upsilon, \end{aligned} \tag{2.14}$$

where υ is the threshold of the step-wise risk. It can be either determined by the user or controlled online as with r_{SA} in the Simulated Annealing. Intuitively, Eq. (2.14) pre-screens out highly risky regions in the search space. As will be described in Section 4.2, an alternative approach is to consider the step-wise risk and the other infill criterion as a *bi-objective* optimization task.

2.6 Summary

In this chapter, we discuss the stochastic variation operator, which is one of the most important component of stochastic optimization algorithms. Specifically, the so-called Gaussian sampling is re-visited: the sampling error of Gaussian random sampling could be very large when the sample size is quite small. The large sampling error could potentially reduce the efficiency of the stochastic variation. As a remedy, the mirrored orthogonal sampling is proposed to reduce the sampling error and therefore accelerate the convergence velocity for the small sample size. Apart from improving the existing stochastic variation operator, we manage to extract a stochastic variation operator from the well-known Efficient Global Optimization algorithm. The resulting operator is called mutation by optimization. In this manner, the EGO algorithm becomes conceptually similar to the canonical stochastic optimization algorithm, e.g., the Simulated Annealing.

CHAPTER 3

Kriging/Gaussian Process Regression

As nonparametric regression/interpolation methods, Kriging and Gaussian Process regression (GPR) (Stein, 1999; Rasmussen and Williams, 2006) are widely used as a (meta-)modeling tool in *Design and Analysis of Computer Experiments* (Sacks et al., 1989; Santner et al., 2003), Surrogate-assisted Evolutionary Algorithms (Emmerich, 2005; Jin, 2011), Global Optimization (Jones et al., 1998; Močkus, 2012) and Algorithm Configuration (Hutter et al., 2011; Bartz-Beielstein et al., 2005). Commonly, Kriging and GPR are used interchangeably in the literature due to the fact that they represent exactly the same estimator. However, they are motivated and derived differently and thus possess different assumptions and properties: Kriging is originated in geostatistics (Krige, 1951) while GPR is usually discussed in nonparametric Bayesian inference (van der Vaart and van Zanten, 2008). In this chapter, we shall compare these two methods conceptually and discuss to which extent they can be used interchangeably.

Moreover, it is well-known that Kriging/GPR suffers from the cubic time complexity and quadratic space complexity as the number of the data points increases. Several existing solutions to this issue are summarized and compared in this chapter. In addition, a novel solution framework, **Cluster Kriging** (CK), is proposed, in which the data set is divided into several folds and Kriging estimators constructed on each fold are combined in multiple ways. Similar to our argument on Kriging/GPR above, two parallel derivations of Cluster Kriging are presented: one approach taking the properties of Gaussian process (section 3.2.3) and the other one built on the theory of the *best linear unbiased prediction* (BLUP). In addition, the ability of reducing the time complexity is validated through experimental studies. To illustrate the usefulness of Cluster Kriging, it is then applied as the surrogate model in the efficient global optimization (EGO), aiming at reducing the running

time of EGO without slowing down its convergence rate. The resulting CK-EGO algorithm is tested on some benchmark functions in Section 3.3.

3.1 General Discussion

The discussion begins with assumptions that are common in both Kriging and GPR. Consider a (noiseless) real-valued function of interest $f : \mathrm{S} \subset \mathbb{R}^d \to \mathbb{R}$. It could serve as an objective function in optimization or a response variable in meta-modeling. Without loss of generality, we assume space $L^2(\mathrm{S})$ for such functions. A real-valued stochastic process $Y = \{Y(\mathbf{x}) : \mathbf{x} \in \mathrm{S}\}$ is a collection of random variables indexed by a set S, where random variables

$$\forall \mathbf{x} \in \mathrm{S}, \quad Y(\mathbf{x}) : \Omega \to \mathbb{R},$$

are defined between the probability space $(\Omega, \mathscr{F}, \mathbb{P})$ and the measurable space $(\mathbb{R}, \mathscr{B})$ ($\mathscr{F}$ is the σ-algebra on Ω and $\mathscr{B}$ is the Borel algebra on the real line). In order to make clear arguments, it is convenient to define the stochastic process Y as a measurable function of two variables (Øksendal, 2003),

$$Y : \mathrm{S} \times \Omega \to \mathbb{R}, \quad (\mathbf{x}, \omega) \mapsto Y(\mathbf{x}, \omega).$$

Using this notation, for every point $\mathbf{x} \in \mathrm{S}$, $Y(\mathbf{x}, \cdot)$ denotes the random variable indexed by $\mathbf{x}$ and for every outcome $\omega \in \Omega$, $Y(\cdot, \omega) : \mathrm{S} \to \mathbb{R}$ is a real-valued function and is called **sample path/sample function** of process Y. In the following discussion, when the outcome ω is not explicitly given, we shall abbreviate $Y(\mathbf{x}, \cdot)$ as $Y(\mathbf{x})$.

The general assumption of Kriging/GPR is: f **is a sample function of** Y. Commonly, the stochastic process Y is specified by two components: a *deterministic* trend function t and a *centered* stochastic process Z:

$$Y(\mathbf{x}) = t(\mathbf{x}) + Z(\mathbf{x}). \tag{3.1}$$

In general, instead of specifying the distribution for Z, only the mean and the covariance structure are given: $\forall \mathbf{x}, \mathbf{x}' \in \mathrm{S}, \mathbb{E}Z(\mathbf{x}) = 0, \mathrm{Cov}\{Z(\mathbf{x}), Z(\mathbf{x}')\} = k(\mathbf{x}, \mathbf{x}')$. Note that $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a *positive-definite kernel*, called *covariance function*. Suppose the target function f is evaluated at n points[1]: $\mathbf{y} =$

[1]Those points are typically obtained via a design of experiment, e.g., Latin hypercube sampling.

$(f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))^\top$. According to the assumption of Kriging, $\mathbf{y}$ also represents the realization of the random vector $\boldsymbol{\psi} = (Y(\mathbf{x}_1), Y(\mathbf{x}_2), \ldots, Y(\mathbf{x}_n))^\top$:

$$\exists \omega \in \Omega, \quad \mathbf{y} = \boldsymbol{\psi}(\omega) = (Y(\mathbf{x}_1, \omega), Y(\mathbf{x}_2, \omega), \ldots, Y(\mathbf{x}_n, \omega))^\top.$$

Then the task is to *approximate* the value $f(\mathbf{x})$ at an unobserved location $\mathbf{x}$ using vector $\mathbf{y}$. In the following, each component of process Y is specified. Normally, the trend function takes a parametric form. For example, it could be either a constant

$$t(\mathbf{x}) = \beta,$$

or the linear combination of a few basis functions,

$$t(\mathbf{x}) = \sum_{i=0}^{p} \beta_i b_i(\mathbf{x}) = \mathbf{b}(\mathbf{x})^\top \boldsymbol{\beta}, \quad b_0 = 1, \tag{3.2}$$

where $p+1$ fixed basis functions b_i, abbreviated as $\mathbf{b} = (b_0, b_2, \ldots, b_p)^\top$, are typically specified by the user. Typically, the first or second order polynomial basis functions (Lophaven et al., 2002) are used. Depending on the form of the trend function and whether the coefficients β are known, Kriging methods are further categorized into *Simple Kriging*, *Ordinary Kriging* and *Universal Kriging* (Zimmerman et al., 1999; Stein, 1999). For detailed discussions on the history of Kriging variants, please see Cressie (2015, 1990). Those terms are clarified in Tab. 3.1. Note that, it is unnecessary to distinguish the constant and

Table 3.1: Taxonomy of Kriging methods.

	known β	β to estimate
Constant trend	Simple	Ordinary
Basis functions	None	Universal

basis function because the former is special case of the latter when $p = 0$. Thus, we shall always refer to Eq. (3.2) for the trend function. For brevity, the trend component of all observations is denoted as:

$$\mathbf{t} = (t(\mathbf{x}_1), t(\mathbf{x}_2), \ldots, t(\mathbf{x}_n))^\top = \mathbf{B}\boldsymbol{\beta}, \quad \mathbf{B} = [\mathbf{b}(\mathbf{x}_1), \mathbf{b}(\mathbf{x}_2), \ldots, \mathbf{b}(\mathbf{x}_n)]^\top.$$

The covariance function is required to be a positive-definite kernel. A symmetric function $k : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is positive definite (p.d.) if the following condition

$$\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j k(\mathbf{x}_i, \mathbf{x}_j) \geq 0 \tag{3.3}$$

holds for $\forall n \in \mathbb{N}, \forall \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathrm{S}$ and $\forall c_1, c_2, \ldots, c_n \in \mathbb{R}$. Some commonly used kernels include: Gaussian kernel, also known as radial basis functions (RBF) (Buhmann, 2003):

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \exp\left(-\sum_{i=1}^{d} \frac{(x_i - x_i')^2}{2\theta_i^2}\right), \tag{3.4}$$

and the Matérn 3/2 kernel (Rasmussen and Williams, 2006):

$$k(\mathbf{x}, \mathbf{x}') = \sigma^2 \prod_{i=1}^{d} \left(1 + \sqrt{3}\frac{h_i}{\theta_i}\right) \exp\left(-\sqrt{3}\frac{h_i}{\theta_i}\right), \quad h_i = |x_i - x_i'|. \tag{3.5}$$

Note that $\forall \mathbf{x} \in \mathrm{S}, k(\mathbf{x}, \mathbf{x}) = \sigma^2$. The parameters σ^2 and $\boldsymbol{\theta} = (\theta_1, \theta_2, \ldots, \theta_d)^\top$ are the so-called *hyper-parameters* and are usually estimated from the data (please see the discussion on the likelihood function below). Throughout this thesis, the Matérn 3/2 kernel is applied to Kriging modeling as the Matérn family of kernels allows for accurate approximations of the local variation in the data (Stein, 1999). Please see Rasmussen and Williams (2006) for more kernel functions. As the kernel function governs the covariance structure, it is necessary to discuss the statistical properties of Z, when choosing the Matérn 3/2 kernel:

- Stationary: a stochastic process Z is called *weakly stationary* if for all $\mathbf{x}, \mathbf{x}'$ in its index set, the mean function is constant and the covariance only depends on $\mathbf{x} - \mathbf{x}'$, namely $\mathrm{Cov}\{Z(\mathbf{x}), Z(\mathbf{x}')\} = k(\mathbf{x} - \mathbf{x}', 0)$. This is a common assumption made on stochastic processes and it is assured by the Matérn 3/2 kernel.

- Isotropy: a stochastic process Z is called *weakly isotropic* if for all the locations of its index set, its mean function is constant and its covariance of $Z(\mathbf{x}), Z(\mathbf{x}')$ only depends on the Euclidean distance between the location, namely $\mathrm{Cov}\{Z(\mathbf{x}), Z(\mathbf{x}')\} = k(\|\mathbf{x} - \mathbf{x}'\|, 0)$. Intuitively, isotropy indicates that the process is rotation-invariant because $\|\mathbf{x} - \mathbf{x}'\| = \|\mathbf{R}(\mathbf{x} - \mathbf{x}')\|$ holds for any orthogonal matrix $\mathbf{R}$. It is straightforward to check that Matérn 3/2 kernel does not imply this property. In practice, the isotropy is too strong to assume on the data and thus non-isotropic kernels are suggested.

Lastly, some notations are introduced: the covariance matrix of $\mathbf{y}$ is written

as:

$$\mathbf{K}(\sigma^2, \boldsymbol{\theta}) = \mathbb{E}\{(\mathbf{y}-\mathbf{t})(\mathbf{y}-\mathbf{t})^\top\} = \begin{bmatrix} k(\mathbf{x}_1,\mathbf{x}_1) & k(\mathbf{x}_1,\mathbf{x}_2) & \dots & k(\mathbf{x}_1,\mathbf{x}_n) \\ k(\mathbf{x}_2,\mathbf{x}_1) & k(\mathbf{x}_2,\mathbf{x}_2) & \dots & k(\mathbf{x}_2,\mathbf{x}_n) \\ \vdots & \vdots & \ddots & \vdots \\ k(\mathbf{x}_n,\mathbf{x}_1) & k(\mathbf{x}_n,\mathbf{x}_2) & \dots & k(\mathbf{x}_n,\mathbf{x}_n) \end{bmatrix},$$

and its covariances with $Y(\mathbf{x})$ is denoted as $\mathbf{k}(\mathbf{x}) = (k(\mathbf{x},\mathbf{x}_1), k(\mathbf{x},\mathbf{x}_2), \dots, k(\mathbf{x},\mathbf{x}_n))^\top$. The covariance matrix will be denoted as $\mathbf{K}$ for short in the following discussions. In addition, from the definition of positive definite kernel (Eq. (3.3)), it is straightforward to verify that $\mathbf{K}$ is a *positive semi-definite matrix*. Moreover, the singular case is ignored throughout this thesis and thus $\mathbf{K}$ is assumed to be a *positive-definite matrix*.

3.1.1 Best Linear Unbiased Predictor

Consider a finite collection of random variables: $\boldsymbol{\psi} = (Y(\mathbf{x}_1), Y(\mathbf{x}_2), \dots, Y(\mathbf{x}_n))^\top$. The basic idea is to construct a *nonparametric linear predictor* $\widehat{Y} = \boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0$ to predict $Y(\mathbf{x})$. The *best predictor*[1] is chosen such that the following *risk function* (expected quadratic loss/mean squared error) is minimized:

$$\begin{aligned} R(\widehat{Y}, Y) &= \mathbb{E}\{\boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0 - Y(\mathbf{x})\}^2 \\ &= \big(\mathbb{E}\{\boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0 - Y(\mathbf{x})\}\big)^2 + \mathrm{Var}\{\boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0 - Y(\mathbf{x})\} \end{aligned} \tag{3.6}$$

Note that, in this risk function the expectation is taken w.r.t. the joint distribution of $\boldsymbol{\psi}$ and $Y(\mathbf{x})$. In addition, a *linear unbiased predictor* (LUP) (Stein, 1999) is intended, which can be obtained by enforcing the following constraint:

$$\begin{aligned} \big(\mathbb{E}\{\boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0 - Y(\mathbf{x})\}\big)^2 = 0 &\iff \forall \boldsymbol{\beta} \in \mathbb{R}^{p+1} \left(\boldsymbol{\alpha}^\top \mathbf{B}\boldsymbol{\beta} + \alpha_0 - \mathbf{b}^\top \boldsymbol{\beta} = 0\right) \\ &\iff \alpha_0 = 0 \wedge \mathbf{B}^\top \boldsymbol{\alpha} = \mathbf{b}. \end{aligned}$$

Therefore the existence of the LUP depends on the solution of the linear system $\mathbf{B}^\top \boldsymbol{\alpha} = \mathbf{b}$. We suppose the solution to this system exists for now ($\mathbf{b}$ is in the column space of $\mathbf{B}^\top$). As the bias in Eq. (3.6) is restricted to zero, only the variance

[1] Note that, after obtaining the best predictor $\widehat{Y}$, the best estimator $\hat{f}$ for f can be given by taking a sample function from $\widehat{Y}$, namely $\hat{f}(\cdot) = \widehat{Y}(\cdot, \omega)$ for some $\omega \in \Omega$. Please see Section 3.1.2 for more details.

term remains. Then, the task of finding the **best linear unbiased predictor** (BLUP) (Stein, 1999) becomes the minimization of the variance:

$$R(\widehat{Y}, Y) = \mathrm{Var}\{\boldsymbol{\alpha}^\top \boldsymbol{\psi} + \alpha_0 - Y(\mathbf{x})\} = \sigma^2 + \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} - 2\mathbf{k}^\top \boldsymbol{\alpha}.$$

This is a convex optimization task ($\mathbf{K}$ is positive definite) with equality constraints:

$$\begin{aligned} \underset{\boldsymbol{\alpha}\in\mathbb{R}^n}{\text{minimize}} \quad & \sigma^2 + \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} - 2\mathbf{k}^\top \boldsymbol{\alpha} \\ \text{subject to} \quad & \mathbf{B}^\top \boldsymbol{\alpha} = \mathbf{b}. \end{aligned} \tag{3.7}$$

This optimization problem can be solved using *Lagrange Multipliers*. The first order condition of optimality is (Boyd and Vandenberghe, 2004):

$$\begin{bmatrix} \mathbf{K} & \mathbf{B} \\ \mathbf{B}^\top & \mathbf{O} \end{bmatrix} \begin{bmatrix} \boldsymbol{\alpha} \\ \boldsymbol{\lambda} \end{bmatrix} = \begin{bmatrix} -\mathbf{k} \\ \mathbf{b} \end{bmatrix},$$

where $\boldsymbol{\lambda} \in \mathbb{R}^{p+1}$ is the *dual* variable and $\mathbf{O}$ represents the matrix of zeros. Solving this linear system, we have

$$\begin{aligned} \boldsymbol{\alpha}^* &= \mathbf{K}^{-1}(\mathbf{k} - \mathbf{B}\boldsymbol{\lambda}^*) \\ \boldsymbol{\lambda}^* &= \left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{B}\right)^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k} - \mathbf{b}\right). \end{aligned}$$

Due to the convexity of this problem, $\boldsymbol{\alpha}^*$ is also sufficient to be the minimizer of Problem (3.7) (Nocedal and Wright, 2000). Plugging $\boldsymbol{\alpha}^*$ back, we have the **Kriging predictor**:

$$\widehat{Y} = \left[\mathbf{k} - \mathbf{B}\left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{B}\right)^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k} - \mathbf{b}\right)\right]^\top \mathbf{K}^{-1}\boldsymbol{\psi}. \tag{3.8}$$

To approximate the target function f, it is straightforward to take a sample function from $\widehat{Y}$:

$$\hat{f} = \mathbf{b}^\top \left[\left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{y}\right] + \mathbf{k}^\top \mathbf{K}^{-1} \left\{\mathbf{y} - \mathbf{B}\left[\left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{y}\right]\right\},$$

which is achieved by substituting the realization $\mathbf{y} = \boldsymbol{\psi}(\omega)$ into Eq. (3.8) and rearranging the terms. It is important to observe that $\hat{\boldsymbol{\beta}} \coloneqq \left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{y}$ is exactly the **Generalized Least Squares** (GLS) (Rao, Toutenburg, Shalabh, and Heumann, Rao et al.) estimate of $\boldsymbol{\beta}$ in the following sense. The trend function $t = \mathbf{b}^\top \boldsymbol{\beta}$ is treated as the regression function and Z is the stationary error process, whose second-order information (auto-covariance) is known. Then, the *best linear unbiased estimator* (BLUE) of $\boldsymbol{\beta}$ is $\hat{\boldsymbol{\beta}}$. Note that, 1) compared to Kriging, GLS considers $\mathbf{b}^\top \boldsymbol{\beta}$ as the predictor while in Kriging the counterpart is $\mathbf{b}^\top \boldsymbol{\beta} + Z$ and 2)

the expression of $\hat{f}$ can also be derived in a much simpler way by first estimating $\boldsymbol{\beta}$ using the GLS formula and then predicting process Z on the residuals $\mathbf{y} - \mathbf{B}\hat{\boldsymbol{\beta}}$ (cf. Eq. (3.13)). However, this approach requires the complete specification of the auto-covariance/kernel and thus is erroneous when hyper-parameters σ^2 and $\boldsymbol{\theta}$ of the kernel function are subject to estimation. Taking the compact notation $\hat{\boldsymbol{\beta}}$, the function approximation is re-written as:

$$\hat{f} = \mathbf{b}^\top \hat{\boldsymbol{\beta}} + \mathbf{k}^\top \mathbf{K}^{-1} \left(\mathbf{y} - \mathbf{B}\hat{\boldsymbol{\beta}} \right). \tag{3.9}$$

In addition, it is also possible to give the covariance of the predictor:

$$\begin{aligned} &\mathrm{Cov}\left\{ \widehat{Y}(\mathbf{x}), \widehat{Y}(\mathbf{x}') \right\} \\ &= \left[\mathbf{k} - \mathbf{B} \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B} \right)^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k} - \mathbf{b} \right) \right]^\top \mathbf{K}^{-1} \left(\mathbf{K} + \mathbf{B}\mathbf{M}\mathbf{B}^\top \right) \mathbf{K}^{-1} \\ &\left[\mathbf{k}' - \mathbf{B} \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B} \right)^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k}' - \mathbf{b}' \right) \right] - \mathbf{b}^\top \mathbf{M} \mathbf{b}', \end{aligned} \tag{3.10}$$

where $\mathbf{M} = \boldsymbol{\beta}\boldsymbol{\beta}^\top, \mathbf{b}' = \mathbf{b}(\mathbf{x}')$. Note that this covariance depends on the unknown parameter $\boldsymbol{\beta}$. When the kernel is completely specified, $\boldsymbol{\beta}$ can be substituted by its GLS estimate $\hat{\boldsymbol{\beta}}$. The minimal MSE of $\widehat{Y}$ can be obtained by putting $\boldsymbol{\alpha}^*$ back to Eq. (3.7) and it is called the **Kriging MSE**:

$$s^2 = \sigma^2 - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k} + \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k} \right)^\top \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B} \right)^{-1} \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k} \right). \tag{3.11}$$

Note that s^2 is the not the variance of the predictor $\widehat{Y}$. In addition, $s = \sqrt{s^2}$ shall be called *Kriging Root Mean Squared Error* (Kriging RMSE).

Remark. 1) In some literatures (den Hertog et al., 2006), s^2 is also called Kriging variance. When using this terminology, s^2 should not be confused with the stationary variance σ^2 of the process Z or the variance of the Kriging predictor. 2) It is important to point out that the MSE s^2 quantifies the uncertainty about predicting the stochastic process Y. It, however, does not directly measure the accuracy of the function approximation, namely to which degree $\hat{f}$ is close to f. To see how the approximation accuracy is related to s^2, please check Section 3.1.2.

The prediction residuals at different locations are correlated. It is possible to calculate the *covariance* among the residuals (please do not confuse with the covariance of the predictor defined in Eq. (3.10)):

$$\begin{aligned} &\mathrm{Cov}\left\{ \left(\widehat{Y}(\mathbf{x}) - Y(\mathbf{x}) \right) \left(\widehat{Y}(\mathbf{x}') - Y(\mathbf{x}') \right) \right\} \\ &= k(\mathbf{x}, \mathbf{x}') - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}' + (\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k})^\top (\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B})^{-1} (\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k}'), \end{aligned} \tag{3.12}$$

where $\mathbf{k} = \mathbf{k}(\mathbf{x}), \mathbf{k}' = \mathbf{k}(\mathbf{x}')$. It is straightforward to verify this covariance function is positive-definite.

Known trend function The discussion so far can be greatly simplified if the trend function is completely provided prior to the modeling. In principle, the trend effect can be subtracted from the random vector $\boldsymbol{\psi}$:

$$\boldsymbol{\psi}' = \boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta} = (Z(\mathbf{x}_1), Z(\mathbf{x}_2), \ldots, Z(\mathbf{x}_n))^\top$$

And the latent[1] realization of $\boldsymbol{\psi}'$ is: $\mathbf{z} = \mathbf{y} - \mathbf{B}\boldsymbol{\beta}$. It is then sufficient to search for the optimal linear predictor of Z. In addition, due to the stationarity assumption on Z, any linear predictor $\boldsymbol{\alpha}^\top \boldsymbol{\psi}'$ is unbiased. Therefore, the **best linear predictor** (BLP) (Stein, 1999) suffices for our aim. It is the minimizer of the unconstrained risk function (cf. Eq. (3.7)):

$$R(\boldsymbol{\alpha}^\top \boldsymbol{\psi}', Z) = \sigma^2 + \boldsymbol{\alpha}^\top \mathbf{K}\boldsymbol{\alpha} - 2\mathbf{k}^\top \boldsymbol{\alpha}.$$

The optimal coefficients are $\boldsymbol{\alpha}^* = \mathbf{K}^{-1}\mathbf{k}$ and the BLP of process Z is $\boldsymbol{\alpha}^{*\top} \boldsymbol{\psi}'$. The best linear predictor of Y is obtained by adding the trend function back to the BLP of Z:

$$\widehat{Y} = \mathbf{b}^\top \boldsymbol{\beta} + \mathbf{k}^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta}) \tag{3.13}$$

$$s^2 = \sigma^2 - \mathbf{k}^\top \mathbf{K}^{-1}\mathbf{k} \tag{3.14}$$

$$\mathrm{Var}\left\{\widehat{Y}(\mathbf{x})\right\} = \mathbf{k}^\top \mathbf{K}^{-1}\mathbf{k} \tag{3.15}$$

$$\mathrm{Cov}\left\{\widehat{Y}(\mathbf{x}), \widehat{Y}(\mathbf{x}')\right\} = \mathbf{k}^\top \mathbf{K}^{-1}\mathbf{k}' \tag{3.16}$$

The extreme of this treatment is to set $\boldsymbol{\beta}$ to zero and it is called *Simple Kriging*. In this case, $\widehat{Y} = \mathbf{k}^\top \mathbf{K}^{-1}\boldsymbol{\psi}$ and its variance and MSE are the same as Eq. (3.15) and (3.14).

[1] We use the term "latent" here as Z is not directly observable.

Discussion 1. It seems a daunting task to select an appropriate trend function. For local interpolations, theoretically it is known that BLPs exhibit the same performance asymptotically with BLUPs, even if the trend is zero ($\boldsymbol{\beta} = \mathbf{0}$) (Stein, 1999). For such modeling tasks, it is sufficient to set the trend function to zero. For the extrapolation, the Kriging estimator regresses back to the trend when the location is weakly correlated to most of the data points. Thus, choosing a proper trend function is necessary for the extrapolation purpose (Journel and Rossi, 1989). Generally, Universal Kriging is recommended for this scenario (Journel and Rossi, 1989) if no prior knowledge is available. However, thorough empirical/theoretical analyses are necessary before putting it as a conclusion. In addition, as will be shown later, the predictor of Simple Kriging is an element of the Hilbert space $\mathcal{H}$ induced by the kernel function. It would be interesting to investigate if polynomial trend functions can be fully expressed in $\mathcal{H}$. The incorporation of the trend function would be unnecessary if it is also an element of $\mathcal{H}$.

Noisy observations and Kriging nugget In practice, it is very likely that the observed response variable contains random measurement noises. Therefore it is, in general, helpful to consider the following *data generation process*:

$$\widetilde{Y} = Y + \varepsilon, \tag{3.17}$$

where $\{\varepsilon(\mathbf{x}) : \mathbf{x} \in \mathrm{S}\}$ is a white noise process (e.g., Gaussian white noise) that is independent from Y and has stationary variance $\sigma_n^2 < \infty$. Formally, ε is specified as:

$$\forall \mathbf{x}, \mathbf{x}' \in \mathrm{S}, \quad \mathbb{E}\varepsilon(\mathbf{x}) = 0, \quad \mathrm{Cov}\{\varepsilon(\mathbf{x}), \varepsilon(\mathbf{x}')\} = \sigma_n^2 \mathbb{1}_{\{\mathbf{x}\}}(\mathbf{x}'), \quad \varepsilon(\mathbf{x}) \perp\!\!\!\perp Y(\mathbf{x}').$$

Here $\mathbb{1}_{\{\mathbf{x}\}}$ is the characteristic function (or indicator function) and $\perp\!\!\!\perp$ denotes the statistical independence. It is important to point out that the goal is still to predict process Y. Under this setting, the task of predicting Y becomes a *nonparametric regression* task, in which the regression function $\hat{f}$ admits a nonparametric form. Again, consider the random vector $\widetilde{\boldsymbol{\psi}} = (\widetilde{Y}(\mathbf{x}_1), \widetilde{Y}(\mathbf{x}_2), \ldots, \widetilde{Y}(\mathbf{x}_n))^\top$ and its realizations $\widetilde{\mathbf{y}} = \widetilde{\boldsymbol{\psi}}(\omega), \omega \in \Omega$. For the sake of brevity, only simple Kriging ($\boldsymbol{\beta}$ is zero) is considered here. By minimizing the risk function (cf. Eq. (3.7)),

$$R(\widehat{Y}, Y) = \mathrm{Var}\{\boldsymbol{\alpha}^\top \widetilde{\boldsymbol{\psi}} + \alpha_0 - Y(\mathbf{x})\} = \sigma^2 + \boldsymbol{\alpha}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right) \boldsymbol{\alpha} - 2\mathbf{k}^\top \boldsymbol{\alpha},$$

the Kriging estimator under noisy observations is:

$$\widehat{Y} = \mathbf{k}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\boldsymbol{\psi}}. \tag{3.18}$$

$$s^2 = \sigma^2 - \mathbf{k}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{k}. \tag{3.19}$$

The noise variance σ_n^2 is also known as the **Kriging nugget** or **nugget effect** (Cressie, 2015). Historically, the Kriging nugget is introduced with the so-called *Semivariogram* in the geostatistics literature. The semivariogram is defined as half the variance of the differences between observations at two locations: $\gamma(\mathbf{x}, \mathbf{x}') = \frac{1}{2}\mathbb{E}\{Y(\mathbf{x}) - Y(\mathbf{x}')\}^2$. As with the kernel function, the semivariogram is an alternative quantification of the auto-correlation (spatial dependency) on process Y. The nugget effect is defined to be the amount of the jump of the

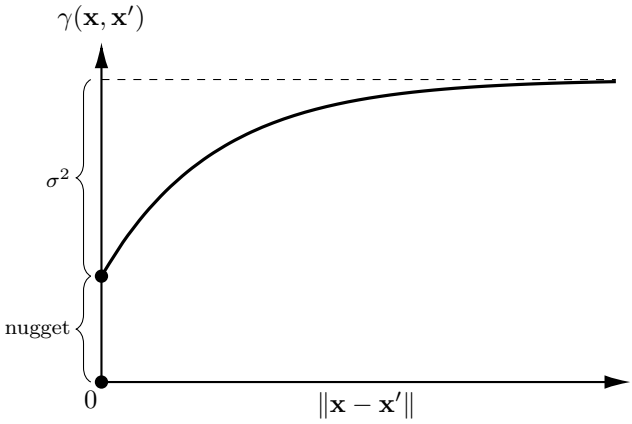


Figure 3.1: Illustration on the semivariogram and Kriging nugget.

semivariogram at the origin (Fig. 3.1) and can be attributed to measurement errors due to the inherent imprecision in measurement devices. Consider the noisy process $\widetilde{Y}$, its semivariogram is,

$$\begin{aligned}\widetilde{\gamma}(\mathbf{x}, \mathbf{x}') &= \frac{1}{2}\mathbb{E}\left\{\widetilde{Y}(\mathbf{x}) - \widetilde{Y}(\mathbf{x}')\right\}^2 = \frac{1}{2}\mathbb{E}\left\{Y(\mathbf{x}) - Y(\mathbf{x}')\right\}^2 + \sigma_n^2 \\ &= \gamma(\mathbf{x}, \mathbf{x}') + \sigma_n^2.\end{aligned}$$

The semivariogram $\widetilde{\gamma}$ in the noisy case is translated upwards from the noiseless case γ by an amount of σ_n^2. The correspondence between the noise variance and Kriging nugget is clearly seen. Moreover, the nugget effect is sometimes referred as *nugget variance* (Webster and Oliver, 2007).

Note that, the nugget effect is useful even when no measurement error is present, e.g., in computer experiments. It can help relaxing the conditional number of the covariance matrix $\mathbf{K}$ when it gets ill-conditioned (Andrianakis and Challenor, 2012). Let $\widetilde{\mathbf{K}} = \mathbf{K} + \sigma_n^2\mathbf{I}$ be the covariance matrix under the noisy assumption. It is then obvious that the eigenvalue $\tilde{\lambda}$ of $\widetilde{\mathbf{K}}$ admits the relation: $\tilde{\lambda} = \lambda + \sigma_n^2$, where λ is the eigenvalue of $\mathbf{K}$. Consequently, the condition number κ of $\widetilde{\mathbf{K}}$ is smaller than that of $\mathbf{K}$:

$$\kappa(\widetilde{\mathbf{K}}) = \frac{|\lambda_{\max} + \sigma_n^2|}{|\lambda_{\min} + \sigma_n^2|} < \frac{|\lambda_{\max}|}{|\lambda_{\min}|} = \kappa(\mathbf{K}).$$

Numerically, as the condition number increases, the covariance matrix becomes practically not invertible and therefore introducing the Kriging nugget can avoid numerical issues that are frequently encountered in the hyper-parameter estimation (Ababou et al., 1994).

3.1.2 Reproducing Kernel Hilbert Space

Here we shall take a different point of view on BLUP, namely from the Hilbert space associated with stochastic process Y. To simplify our discussions here, Y is assumed to have a zero mean and correspondingly the Kriging predictor is $\widehat{Y} = \mathbf{k}^\top\mathbf{K}^{-1}\boldsymbol{\psi}$ (obtained by setting $\boldsymbol{\beta}$ to zero in Eq. (3.13)). Moreover, the index set S is assumed to be a *separable space*. In addition, the covariance vector $\mathbf{k}$ is treated as a function from S to $\mathbb{R}^n$ and is denoted as $\mathbf{k}(\mathbf{x})$. More precisely, the approach posed in Section 3.1.1 is to predict a random variable $Y(\mathbf{x}, \cdot)$ using a linear combination of some other random variables on the process:

$$\widehat{Y}(\mathbf{x}, \cdot) = \sum_{i=1}^{n} \alpha_i(\mathbf{x}) Y(\mathbf{x}_i, \cdot), \quad n \in \mathbb{N}, \quad \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathrm{S}.$$

The optimal coefficient $\boldsymbol{\alpha}(\mathbf{x}) = \mathbf{K}^{-1}\mathbf{k}(\mathbf{x})$ is a function of $\mathbf{x}$ and is obtained by minimizing the risk function (Eq. (3.6)) as before. Note that the same prediction approach is applied for every $\mathbf{x} \in \mathrm{S}$, meaning that a predictor of the process Y is obtained:

$$\widehat{Y} = \left\{ \sum_{i=1}^{n} \alpha_i(\mathbf{x}) Y(\mathbf{x}_i, \cdot) : n \in \mathbb{N},\ \mathbf{x}, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathrm{S} \right\}. \tag{3.20}$$

We shall call $\widehat{Y}$ the **Kriging predictor**. The rationale is: $\widehat{Y}$ optimally predicts an unknown process Y only using partial information of its own. If we observe a

sample function of Y partially[1], namely $\mathbf{y} = (Y(\mathbf{x}_1, \omega), Y(\mathbf{x}_2, \omega), \ldots, Y(\mathbf{x}_n, \omega))^\top$ for some $\omega \in \Omega$, it is possible to interpolate this sample function by plugging $\mathbf{y}$ into $\widehat{Y}$ (Eq. (3.20)):

$$\begin{aligned}\hat{f}(\cdot) = \widehat{Y}(\cdot, \omega) &= \sum_{i=1}^{n} \alpha_i(\cdot) Y(\mathbf{x}_i, \omega) = \mathbf{y}^\top \mathbf{K}^{-1} \mathbf{k}(\cdot) \\ &= \sum_{i=1}^{n} \xi_i k(\cdot, \mathbf{x}_i), \quad \boldsymbol{\xi} = \mathbf{K}^{-1}\mathbf{y}. \end{aligned} \tag{3.21}$$

Being a sample function from the predictor $\widehat{Y}$, $\hat{f}$ approximates f (cf. Eq. (3.9)). In short, we shall show that the function form of Eq. (3.21) is in the **Reproducing Kernel Hilbert Space** (RKHS) attached to process Y. Given a positive definite kernel $k(\cdot, \cdot)$ on S, there is a unique Hilbert space of functions: $\mathrm{S} \to \mathbb{R}$ for which k is a reproducing kernel (Moore-Aronszajn theorem (Aronszajn, 1950)). This space $\mathcal{H}$ is the completion of the following linear space $\mathcal{H}_0$:

$$\mathcal{H}_0 = \left\{ \sum_{i=1}^{n} c_i k(\cdot, \mathbf{x}_i) : n \in \mathbb{N},\ c_1, c_2, \ldots, c_n \in \mathbb{R},\ \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathrm{S} \right\}.$$

The completion is conducted with respect to the RKHS norm $\|\cdot\|_{\mathcal{H}}$ that is induced by the inner product,

$$\left\langle \sum_{i=1}^{m} c_i k(\cdot, \mathbf{x}_i), \sum_{j=1}^{n} c'_j k(\cdot, \mathbf{x}_j) \right\rangle_{\mathcal{H}} = \sum_{i=1}^{m} \sum_{j=1}^{n} c_i c'_j k(\mathbf{x}_i, \mathbf{x}_j). \tag{3.22}$$

The function in $\mathcal{H}$ has the form: $f(\cdot) = \sum_{i=1}^{\infty} a_i k(\cdot, \mathbf{x}_i)$, where $\sum_{i=1}^{\infty} a_i^2 k(\mathbf{x}_i, \mathbf{x}_i) < \infty$. It is then obvious to see that the Kriging estimator $\hat{f}(\cdot)$ is an element of $\mathcal{H}$. Equivalently, the space of estimator $\hat{f}$ is the set of all the sample functions of $\widehat{Y}$, namely $\left\{ \widehat{Y}(\cdot, \omega) : \omega \in \Omega \right\} \subset \mathcal{H}$. The natural question is: how does the target function f related to $\hat{f}$ and $\mathcal{H}$ in general? Recall the assumption of Kriging: f is a realization of the process Y, or formally $f \in \mathrm{F}$, $\mathrm{F} \coloneqq \{Y(\cdot, \omega) : \omega \in \Omega\}$ (all sample functions of Y). Firstly, we will show that $\mathcal{H}$ is generally "smaller" than F. It is possible to construct a *surjection* from F to $\mathcal{H}$:

$$Y(\cdot, \omega) \mapsto \sum_{i=1}^{n} \tau_i Y(\mathbf{x}_i, \omega) k(\cdot, \mathbf{x}_i), \quad \tau_i \in \mathbb{R}.$$

[1] It is important to note that even countably many observations are partial information about the sample function because its domain S is separable.

It is obvious that for every function $\sum_{i=1}^{n} c_i k(\cdot, \mathbf{x}_i) \in \mathcal{H}$, there always exist $\tau_i \in \mathbb{R}$ and $\omega \in \Omega$ such that $\tau_i Y(\mathbf{x}_i, \omega) = c_i$. Thus, this mapping is surjective. However, it does not admit an inverse: $\{Y(\mathbf{x}_i, \omega)\}_i$ can be mapped back to infinitely many sample functions in F. Secondly, it is possible to quantify the difference between f and $\hat{f}$ using the supremum norm, $\left\| f - \hat{f} \right\|_\infty = \sup_{\mathbf{x} \in \mathrm{S}} \left\{ \left| f(\mathbf{x}) - \hat{f}(\mathbf{x}) \right| \right\}$. It is not hard to verify the following condition,

$$\left\| f - \hat{f} \right\|_\infty \leq \sup_{\omega \in \Omega} \left\{ \left| Y(\cdot, \omega) - \widehat{Y}(\cdot, \omega) \right| \right\}.$$

However, it is not straightforward to build a linkage between $\left\| f - \hat{f} \right\|_\infty$ and the Kriging MSE s^2 (cf. Eq. (3.11)) based on this condition. Alternatively, such a relation can be established point-wisely on f.

Theorem 3.1 (Approximation Error Bound)**.** *Let $\widehat{Y}$ be the BLUP of stochastic process Y. The MSE of $\widehat{Y}$ is $s^2 = \mathbb{E}\{Y(\mathbf{x}) - \widehat{Y}(\mathbf{x})\}^2$. Assume the target function $f : \mathrm{S} \to \mathbb{R}$ is a sample function of Y and it is approximated by a sample function of the BLUP: $\hat{f}(\cdot) = \widehat{Y}(\cdot, \omega), \omega \in \Omega$. Then for every point $\mathbf{x} \in \mathrm{S}$, the approximation error is bounded from above:*

$$\left| \hat{f}(\mathbf{x}) - f(\mathbf{x}) \right| \leq \sqrt{\frac{s^2}{C}},$$

where

$$C = \int_{\mathbb{R}} \Pr\left(|Y(\mathbf{x})| > |f(\mathbf{x})| \;\middle|\; \widehat{Y}(\mathbf{x}) = u \right) p_{\widehat{Y}(\mathbf{x})}(u)\, \mathrm{d}u.$$

Proof. Define a random variable $R = |Y(\mathbf{x}) - \widehat{Y}(\mathbf{x})|$. $r = |f(\mathbf{x}) - \hat{f}(\mathbf{x})|$ is a realization of R. According to Markov's inequality, we have,

$$\Pr(R \geq r) \leq \frac{\mathbb{E}R^2}{r^2}. \tag{3.23}$$

Note that $s^2 = \mathbb{E}R^2$ and $r^2 = |f(\mathbf{x}) - \hat{f}(\mathbf{x})|^2$. Now, we expand the probability on the left-hand-side of the inequality:

$$\begin{aligned} \Pr(R \geq r) &= \int_{\mathbb{R}} \Pr\left(|Y(\mathbf{x}) - \hat{f}(\mathbf{x})| > |f(\mathbf{x}) - \hat{f}(\mathbf{x})| \;\middle|\; \widehat{Y}(\mathbf{x}) = u \right) p_{\widehat{Y}(\mathbf{x})}(u)\, \mathrm{d}u \\ &\geq \int_{\mathbb{R}} \Pr\left(|Y(\mathbf{x})| > |f(\mathbf{x})| \;\middle|\; \widehat{Y}(\mathbf{x}) = u \right) p_{\widehat{Y}(\mathbf{x})}(u)\, \mathrm{d}u. \end{aligned}$$

Combining this inequality with Eq. (3.23), we have $r^2 \leq s^2/C$. □

Remark. The linkage between the approximation error on f and the MSE of BLUPs is clearly seen from this theorem: reducing the MSE s^2 leads to a more

precise function approximation, which is typically achieved by adding more data points/observations. The other factor C can be interpreted as the conditional probability $\Pr\left(|Y(\mathbf{x})| > |f(\mathbf{x})| \,\middle|\, \widehat{Y}(\mathbf{x}) = u\right)$, averaged over all possible predictions. Note that the smaller this conditional probability is (and thus the error bound is higher), it is less likely that $f(\mathbf{x})$ is a sample from $Y(\mathbf{x})$. This means the stochastic process Y tends to be **mis-specified** for f.

RKHS provides us another point of view on Kriging/GPR. In the following, it is shown that the same result in Eq. (3.13) can be obtained using the well-known *representer theorem* (Schölkopf et al., 2001), which gives the representer form of the solution to the regularized optimization problem in $\mathcal{H}$. We shall illustrate this theorem first and then build an alternative derivation of $\hat{f}$ based on it.

Theorem 3.2 (Representer Theorem)**.** *Let* S *be a nonempty set and* $k : \mathrm{S} \times \mathrm{S} \to \mathbb{R}$ *be a positive-definite kernel with corresponding reproducing kernel Hilbert space* $\mathcal{H}$*. The RKHS norm* $\|\cdot\|_{\mathcal{H}}$ *is induced from the inner product in Eq.* (3.22)*. Given a training sample* $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n) \in \mathrm{S} \times \mathbb{R}$*, a strictly monotonically increasing function* $g\colon [0, \infty) \to \mathbb{R}$*, and an arbitrary empirical risk function* R *of* $\{h(\mathbf{x}_i), y_i\}_{i=1}^n$ *and let* $h^* : \mathrm{S} \to \mathbb{R}$ *be the optimum of the regularized minimization problem:*

$$h^* = \underset{h \in \mathcal{H}}{\arg\min}\, R\left(\{h(\mathbf{x}_i), y_i\}_{i=1}^n\right) + g(\|h\|_{\mathcal{H}}),$$

then h^* *is represented as:*

$$h^*(\cdot) = \sum_{i=1}^n c_i k(\cdot, \mathbf{x}_i), \quad c_1, c_2, \ldots, c_n \in \mathbb{R}.$$

Proof. See Schölkopf et al. (2001). □

The representer form of h^* is exactly as Eq. (3.13), suggesting that the Kriging estimator can also be considered as the optimal function form that minimizes *empirical* risk. Furthermore, we will illustrate that under certain specifications, the Kriging coefficients in the noisy setting (Eq. (3.18)) can be obtained using this theorem.

Corollary 3.1. *Assume all the settings in Theorem 3.2, noisy observations* $\widetilde{\mathbf{y}}$ *generated from Eq.* (3.17) *and the following specifications: the empirical risk function* $R\left(\{\hat{f}(\mathbf{x}_i), \widetilde{y}_i\}_{i=1}^n\right) = \sum_{i=1}^n \left(\hat{f}(\mathbf{x}_i) - \widetilde{y}_i\right)^2$ *and* $g(\|\cdot\|_{\mathcal{H}}) = \sigma_n^2 \|\cdot\|_{\mathcal{H}}^2$*, then the coefficients* c_i *in Theorem 3.2 are given by* $\mathbf{c} = \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}}$ *and*

$$\hat{f}(\cdot) = \sum_{i=1}^n c_i k(\cdot, \mathbf{x}_i) = \widetilde{\mathbf{y}}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{k}(\cdot).$$

Proof. According to the representer theorem, $\hat{f}$ takes the form $\hat{f}(\cdot) = \sum_{i=1}^{n} c_i k(\cdot, \mathbf{x}_i)$. Then all predictions from $\hat{f}$ can be denoted as $(\hat{f}(\mathbf{x}_1), \ldots, \hat{f}(\mathbf{x}_n))^\top = \mathbf{Kc}$. Then the regularized minimization problem becomes:

$$\underset{\mathbf{c}\in\mathbb{R}^n}{\text{minimize}} \left\|\mathbf{Kc} - \widetilde{\mathbf{y}}\right\|^2 + \sigma_n^2 \mathbf{c}^\top \mathbf{Kc}. \tag{3.24}$$

The optimality condition of this problem is:

$$\frac{\partial}{\partial \mathbf{c}} \left(\left\|\mathbf{Kc} - \widetilde{\mathbf{y}}\right\|^2 + \sigma_n^2 \mathbf{c}^\top \mathbf{Kc} \right) = \mathbf{0}. \tag{3.25}$$

The solution of $\mathbf{c}$ results from this condition. □

This result can be interpreted as follows. Firstly, note that Problem (3.24) is equivalent to the following constrained convex optimization problem:

$$\begin{aligned} \underset{\mathbf{c}\in\mathbb{R}^n}{\text{minimize}} \quad & \left\|\mathbf{Kc} - \widetilde{\mathbf{y}}\right\|^2 \\ \text{subject to} \quad & \mathbf{c}^\top \mathbf{Kc} \leq t, \end{aligned} \tag{3.26}$$

where $t = \widetilde{\mathbf{y}}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{K} \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}}$. To see the equivalence, the *Karush-Kuhn-Tucker* conditions (KKT) (Boyd and Vandenberghe, 2004) of Problem (3.26) are,

$$\begin{aligned} \frac{\partial}{\partial \mathbf{c}} \left(\left\|\mathbf{Kc} - \widetilde{\mathbf{y}}\right\|^2 \right) + \eta \frac{\partial}{\partial \mathbf{c}} \left(\mathbf{c}^\top \mathbf{Kc} - t \right) &= \mathbf{0} \\ \eta \left(\mathbf{c}^\top \mathbf{Kc} - t \right) &= 0 \\ \mathbf{c}^\top \mathbf{Kc} - t &\leq 0 \\ \eta &\geq 0 \end{aligned} \tag{3.27}$$

The conditions above are also necessary because Slater's condition (Slater, 2014) obviously holds on Problem (3.26). It is not hard to verify that any solution of condition (3.25) is also a solution of conditions (3.27) and vice versa. Consider the convex constraint $\mathbf{c}^\top \mathbf{Kc} \leq t$. The LHS of it is the RKHS norm $\left\|\hat{f}\right\|_{\mathcal{H}}$ of the estimator $\hat{f}(\cdot) = \sum_{i=1}^{n} c_i k(\cdot, \mathbf{x}_i)$. It means that "complexity" of the estimator $\hat{f}$ should be smaller than a threshold t when minimizing the empirical risk. To understand the choice of threshold t, please consider the prediction of the data generation process process $\widetilde{Y} = Y + \varepsilon$ using observations $\widetilde{\mathbf{y}}$ (instead of predicting Y). The Kriging estimator $\widetilde{f}_{\text{est}} \in \mathcal{H}$ is obtained by applying Eq. (3.13) to the overall process $\widetilde{Y}$, whose covariance function is $\widetilde{k}(\cdot,\cdot) = k(\cdot,\cdot) + \sigma_n^2 \mathbb{1}_{\{\cdot\}}(\cdot)$:

$$\widetilde{f}_{\text{est}}(\cdot) = \sum_{i=1}^{n} \widetilde{\alpha}_i \widetilde{k}(\cdot, \mathbf{x}_i), \quad \widetilde{\boldsymbol{\alpha}} = \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}}.$$

Note that $\widetilde{f}_{\text{est}}$ is an element of the RKHS $\widetilde{\mathcal{H}}$ induced by kernel $\widetilde{k}$ and its norm $\left\|\widetilde{f}_{\text{est}}\right\|_{\widetilde{\mathcal{H}}} = \widetilde{\mathbf{y}}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}}$. It is clear that

$$\begin{aligned}\left\|\hat{f}\right\|_{\mathcal{H}} &\leq t = \widetilde{\mathbf{y}}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \mathbf{K} \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}} \\ &\leq \widetilde{\mathbf{y}}^\top \left(\mathbf{K} + \sigma_n^2 \mathbf{I}\right)^{-1} \widetilde{\mathbf{y}} \\ &= \left\|\widetilde{f}_{\text{est}}\right\|_{\widetilde{\mathcal{H}}},\end{aligned}$$

which means the estimator of the component f from the noisy function $\widetilde{f}$ should not be more complex than the estimator of $\widetilde{f}$. In summary, the Kriging estimator (Eq. (3.18)) under noisy observations is the solution to the following problem:

$$\begin{aligned}\underset{\hat{f} \in \mathcal{H}}{\text{minimize}} \quad & \sum_{i=1}^{n} \left(\hat{f}(\mathbf{x}_i) - \widetilde{y}_i\right)^2 \\ \text{subject to} \quad & \left\|\hat{f}\right\|_{\mathcal{H}} \leq \left\|\widetilde{f}_{\text{est}}\right\|_{\widetilde{\mathcal{H}}}.\end{aligned}$$

3.1.3 Bayesian Inference

Known trend function It is possible to give an alternative derivation of the Kriging estimator, using *Bayesian statistics*. Consider again the random vector $\boldsymbol{\psi} = (Y(\mathbf{x}_1), Y(\mathbf{x}_2), \ldots, Y(\mathbf{x}_n))^\top, \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n \in \mathrm{S}$ and its realization $\mathbf{y} = \boldsymbol{\psi}(\omega), \omega \in \Omega$. Bayesian inference requires the specification of the prior distribution on Y and the likelihood $p(\boldsymbol{\psi} \mid Y(\mathbf{x}))$. When the trend function is assumed to be known, the *posterior* of $Y(\mathbf{x})$ is

$$p(Y(\mathbf{x}) \mid \boldsymbol{\psi}) = \frac{p(\boldsymbol{\psi} \mid Y(\mathbf{x}))p(Y(\mathbf{x}))}{p(\boldsymbol{\psi})}. \tag{3.28}$$

Note that, this posterior probability is the conditional probability $p(Y(\mathbf{x}) \mid \boldsymbol{\psi})$ due to the fact that $Y(\mathbf{x})$ and $\boldsymbol{\psi}$ are taken from the same stochastic process Y in our setting. In some other processes, this conditional probability is even given in their definitions (e.g., Markov process). However, Eq. (3.28) gives a plausible rationale on using the conditional probability for the prediction and attaches a Bayesian interpretation to the Kriging predictor. The most common choice (and perhaps the most natural) on the prior distribution is Gaussian: the stochastic process Y is assumed to be a *Gaussian Process*. In addition to the first- (mean) and second-order (covariance) specifications (Section 3.1), the Gaussian process prior

on Y prescribes that random vector $\boldsymbol{\psi}$ is a multivariate Gaussian (See Appendix A for its definition). The following notation is used for a Gaussian process prior with kernel function k:

$$Y \sim t + \mathcal{GP}(0, k(\cdot, \cdot)),$$

where t is the trend function defined in Eq. (3.2) and it is called the *prior mean* function in this section. Note that, trend t is deliberately separated from the centered Gaussian Process $\mathcal{GP}(0, k(\cdot, \cdot))$ because t could admit a stochastic form and the addition of those two terms might not be Gaussian. Recall the basis expansion trend $t = \mathbf{b}^\top \boldsymbol{\beta}$ and $\boldsymbol{\beta}$ is known. It is then straightforward that $Y(\mathbf{x}) \sim \mathcal{N}(\mathbf{b}^\top \boldsymbol{\beta}, \sigma^2)$ and $\boldsymbol{\psi} \sim \boldsymbol{\mathcal{N}}(\mathbf{B}\boldsymbol{\beta}, \mathbf{K})$. Moreover, $Y(\mathbf{x})$ and $\boldsymbol{\psi}$ are jointly Gaussian:

$$\begin{bmatrix} Y(\mathbf{x}) \\ \boldsymbol{\psi} \end{bmatrix} \sim \boldsymbol{\mathcal{N}} \left(\begin{bmatrix} \mathbf{b}^\top \boldsymbol{\beta} \\ \mathbf{B}\boldsymbol{\beta} \end{bmatrix}, \begin{bmatrix} \sigma^2 & \mathbf{k}^\top \\ \mathbf{k} & \mathbf{K} \end{bmatrix} \right).$$

Recall the definition of the covariance vector $\mathbf{k}$ in Section 3.1. Directly applying the conditioning formula (Eq. (A.4)), the conditional distribution $p(Y(\mathbf{x}) \mid \boldsymbol{\psi})$ can be specified

$$Y(\mathbf{x}) \mid \boldsymbol{\psi} \sim \mathbf{b}^\top \boldsymbol{\beta} + \mathbf{k}^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta}) + \boldsymbol{\mathcal{N}} \left(0, \sigma^2 - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}\right). \tag{3.29}$$

Given this conditional distribution, it is obvious that the best unbiased predictor of Y is the conditional mean, i.e., $\widehat{Y} = \mathbf{b}^\top \boldsymbol{\beta} + \mathbf{k}^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta})$. The MSE of $\widehat{Y}$ is $s^2 = \mathbb{E}\{\widehat{Y} - Y\}^2 = \sigma^2 - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}$, which is also the conditional variance. Now, as the target function f is assumed to be a sample function of Y and we have observed some values on the target function, the approximation $\hat{f}$ is obtained by replacing $\boldsymbol{\psi}$ by its realization in $\widehat{Y}$ (cf. Eq. (3.21)):

$$\hat{f}(\cdot) = \widehat{Y}(\cdot, \omega) = \mathbf{b}^\top \boldsymbol{\beta} + \mathbf{k}^\top \mathbf{K}^{-1} (\mathbf{y} - \mathbf{B}\boldsymbol{\beta}), \quad \omega \in \Omega.$$

Note that those terms are exactly the same as the Kriging BLP estimator (cf. Eq. (3.13)). Note that, in terms of Bayesian statistics, the posterior mean in Eq. (3.29) can also be considered as a *Maximum a Posterior Probability* (MAP) estimate because the mode coincides with the mean in Gaussian distributions. This result is commonly referred to as **Gaussian Process Regression** (GPR) in the machine learning field (Rasmussen and Williams, 2006). Consequently, the Kriging MSE s^2 is also called **GPR variance** in this thesis.

Remark. In the standard treatment of GPR, there is no need to use the stochastic process Y because the prior Gaussian process is directly imposed on the target function f. In this section, process Y is taken to keep the consistence with the discussion on BLUP/BLP (Section 3.1.1).

Moreover, a posterior Gaussian process is implied by Eq. (3.29), whose mean function is $\widehat{Y}$. To see the covariance structure of the posterior process, consider two locations $\mathbf{x}_1, \mathbf{x}_2 \in \mathrm{S}$ in the query:

$$\begin{bmatrix} Y(\mathbf{x}_1) \\ Y(\mathbf{x}_2) \\ \boldsymbol{\psi} \end{bmatrix} \sim \boldsymbol{\mathcal{N}} \left(\begin{bmatrix} \mathbf{b}_1^\top \boldsymbol{\beta} \\ \mathbf{b}_2^\top \boldsymbol{\beta} \\ \mathbf{B}\boldsymbol{\beta} \end{bmatrix}, \begin{bmatrix} \sigma^2 & k(\mathbf{x}_1, \mathbf{x}_2) & \mathbf{k}_1^\top \\ k(\mathbf{x}_2, \mathbf{x}_1) & \sigma^2 & \mathbf{k}_2^\top \\ \mathbf{k}_1 & \mathbf{k}_2 & \mathbf{K} \end{bmatrix} \right),$$

in which $\mathbf{b}_1 = \mathbf{b}(\mathbf{x}_1), \mathbf{b}_2 = \mathbf{b}(\mathbf{x}_2)$ and $\mathbf{k}_1 = \mathbf{k}(\mathbf{x}_1), \mathbf{k}_2 = \mathbf{k}(\mathbf{x}_2)$. Conditioning on $\boldsymbol{\psi}$ again, we obtain:

$$\begin{aligned} \begin{bmatrix} Y(\mathbf{x}_1) \\ Y(\mathbf{x}_2) \end{bmatrix} \Bigg| \, \boldsymbol{\psi} \sim & \begin{bmatrix} \mathbf{b}_1^\top \boldsymbol{\beta} + \mathbf{k}_1^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta}) \\ \mathbf{b}_2^\top \boldsymbol{\beta} + \mathbf{k}_2^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta}) \end{bmatrix} \\ & + \boldsymbol{\mathcal{N}} \left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \sigma^2 - \mathbf{k}_1^\top \mathbf{K}^{-1} \mathbf{k}_1 & k(\mathbf{x}_1, \mathbf{x}_2) - \mathbf{k}_1^\top \mathbf{K}^{-1} \mathbf{k}_2 \\ k(\mathbf{x}_2, \mathbf{x}_1) - \mathbf{k}_2^\top \mathbf{K}^{-1} \mathbf{k}_1 & \sigma^2 - \mathbf{k}_2^\top \mathbf{K}^{-1} \mathbf{k}_2 \end{bmatrix} \right). \end{aligned}$$

In this posterior formulation, it is clear to see that the covariance at two arbitrary locations is expressed in the cross-term of the posterior covariance matrix. Consequently, we give the posterior mean (trend) $\widehat{Y}$ and posterior kernel k':

$$\widehat{Y}(\mathbf{x}) \coloneqq \mathbb{E}\{Y(\mathbf{x}) \mid \boldsymbol{\psi}\} = \mathbf{b}^\top \boldsymbol{\beta} + \mathbf{k}^\top \mathbf{K}^{-1} (\boldsymbol{\psi} - \mathbf{B}\boldsymbol{\beta}) \tag{3.30}$$

$$k'(\mathbf{x}, \mathbf{x}') \coloneqq \mathrm{Cov}\{Y(\mathbf{x}), Y(\mathbf{x}') \mid \boldsymbol{\psi}\} = k(\mathbf{x}, \mathbf{x}') - \mathbf{k}^\top \mathbf{K}^{-1} \mathbf{k}' \tag{3.31}$$

It is straightforward to show that k' is a stationary positive-definite kernel.

Unknown trend function When $\boldsymbol{\beta} \in \mathbb{R}^{p+1}$ is subject to estimation, the most common approach is to use hierarchical Bayesian inference by providing a prior on $\boldsymbol{\beta}$. For example, the Gaussian prior is assumed again $\boldsymbol{\beta} \sim \boldsymbol{\mathcal{N}}(\boldsymbol{\zeta}, \boldsymbol{\Sigma})$, with $\boldsymbol{\beta} \perp\!\!\!\perp Y$. It is important to note that when randomness on $\boldsymbol{\beta}$ is introduced the process Y is not necessarily Gaussian any longer. However, the conditional distribution/process on $\boldsymbol{\beta}$, e.g., $p(Y(\mathbf{x}) \mid \boldsymbol{\beta}, \boldsymbol{\psi})$ is still Gaussian. The posterior distribution of $\boldsymbol{\beta}$ is (Stein, 1999),

$$\begin{aligned} p(\boldsymbol{\beta} \mid \boldsymbol{\psi}) &= \frac{p(\boldsymbol{\psi} \mid \boldsymbol{\beta}) p(\boldsymbol{\beta})}{\int_{\mathbb{R}^{p+1}} p(\boldsymbol{\psi} \mid \boldsymbol{\beta}) p(\boldsymbol{\beta}) \, \mathrm{d}\boldsymbol{\beta}} \\ &= (2\pi)^{-\frac{p+1}{2}} \det(\boldsymbol{\Sigma}')^{\frac{1}{2}} \exp\left(-\frac{1}{2} (\boldsymbol{\beta} - \boldsymbol{\zeta}')^\top \boldsymbol{\Sigma}'^{-1} (\boldsymbol{\beta} - \boldsymbol{\zeta}') \right), \end{aligned}$$

where the posterior mean $\boldsymbol{\zeta}'$ and covariance $\boldsymbol{\Sigma}'$ are give below:

$$\boldsymbol{\zeta}' = \boldsymbol{\Sigma}' \left(\mathbf{B}^\top \mathbf{K}^{-1} \boldsymbol{\psi} + \boldsymbol{\Sigma}^{-1} \right), \quad \boldsymbol{\Sigma}' = \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B} + \boldsymbol{\Sigma}^{-1} \right)^{-1}.$$

Note that the conditional distribution $p(Y(\mathbf{x}) \mid \boldsymbol{\psi})$ is obtained by marginalizing $\boldsymbol{\beta}$ out,

$$p(Y(\mathbf{x}) \mid \boldsymbol{\psi}) = \int_{\mathbb{R}^{p+1}} p(Y(\mathbf{x}) \mid \boldsymbol{\beta}, \boldsymbol{\psi}) p(\boldsymbol{\beta} \mid \boldsymbol{\psi}) \, \mathrm{d}\boldsymbol{\beta}.$$

This marginalization can be interpreted as averaging $p(Y(\mathbf{x}) \mid \boldsymbol{\beta}, \boldsymbol{\psi})$ over the posterior of $\boldsymbol{\beta}$. Without giving the details on the derivation, the posterior mean and kernel are expressed as follows (Omre, 1987; O'Hagan and Kingman, 1978):

$$\widehat{Y}(\mathbf{x}) = \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k}\right)^\top \boldsymbol{\zeta}' + \mathbf{k}^\top \mathbf{K}^{-1} \boldsymbol{\psi} \tag{3.32}$$

$$k'(\mathbf{x}, \mathbf{x}') = k(\mathbf{x}, \mathbf{x}') - \mathbf{k}^\top \mathbf{K}^{-1}\mathbf{k}' + \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k}\right)^\top \boldsymbol{\Sigma}' \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k}'\right) \tag{3.33}$$

The formula above depends on the choice of prior parameter $\boldsymbol{\zeta}, \boldsymbol{\Sigma}$. Consider the limit $\boldsymbol{\Sigma} \to \mathbf{O}$ (matrix of zeros), meaning $\boldsymbol{\beta}$ becomes more and more *non-informative* because the prior is increasingly flat everywhere. Then posterior mean and covariance matrix of $\boldsymbol{\beta}$ have the following convergence,

$$\boldsymbol{\zeta}' \to \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}\right)^{-1} \mathbf{B}^\top \mathbf{K}^{-1} \boldsymbol{\psi}, \quad \boldsymbol{\Sigma}' \to \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}\right)^{-1}.$$

Consequently, the posterior mean and kernel converges to the Kriging predictor (BLUP) and covariance (cf. Eq (3.9) and (3.12)):

$$\widehat{Y}(\mathbf{x}) \to \left[\mathbf{k} - \mathbf{B}\left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}\right)^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k} - \mathbf{b}\right)\right]^\top \mathbf{K}^{-1} \boldsymbol{\psi}$$

$$k'(\mathbf{x}, \mathbf{x}') \to k(\mathbf{x}, \mathbf{x}') - \mathbf{k}^\top \mathbf{K}^{-1}\mathbf{k}' + \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k}\right)^\top \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B}\right)^{-1} \left(\mathbf{b} - \mathbf{B}^\top \mathbf{K}^{-1}\mathbf{k}'\right)$$

Because limiting the posterior mean results in the same expression as the Kriging predictor, we shall treat the Kriging predictor and the posterior mean interchangeably in this thesis.

3.1.4 Differentiation

The Kriging predictor and MSE play a central role in Efficient Global Optimization and their derivatives are frequently used in such algorithms. Thus, the gradients of the Kriging predictor (Eq. (3.9)) and MSE (Eq. (3.11)) w.r.t. the index variable are given below (using the *denominator layout*):

$$\frac{\partial \hat{f}}{\partial \mathbf{x}} = \frac{\partial \mathbf{b}}{\partial \mathbf{x}} \hat{\boldsymbol{\beta}} + \frac{\partial \mathbf{k}}{\partial \mathbf{x}} \mathbf{K}^{-1} (\mathbf{y} - \mathbf{B}\hat{\boldsymbol{\beta}}) \tag{3.34}$$

$$\frac{\partial s^2}{\partial \mathbf{x}} = 2\Big[\left(\frac{\partial \mathbf{k}}{\partial \mathbf{x}} \mathbf{K}^{-1} \mathbf{B} - \frac{\partial \mathbf{b}}{\partial \mathbf{x}} \right) (\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{B})^{-1} \left(\mathbf{B}^\top \mathbf{K}^{-1} \mathbf{k} - \mathbf{b}\right) - \frac{\partial \mathbf{k}}{\partial \mathbf{x}} \mathbf{K}^{-1} \mathbf{k} \Big], \tag{3.35}$$

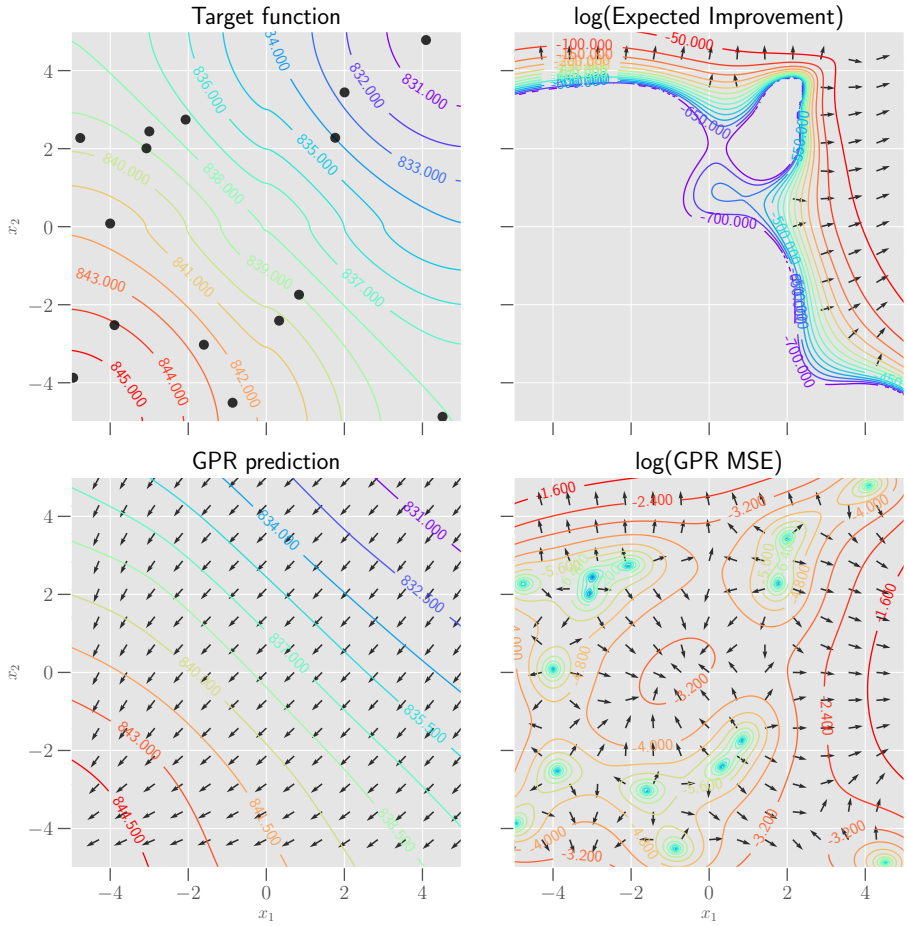


Figure 3.2: On the 2-D *Schwefel function* (**top-left**), several gradient fields and contour lines are depicted for the Kriging/GPR prediction (**bottom-left**), the Kriging/GPR MSE (**bottom-right**) and the so-called *Expected Improvement* criterion (**top-right**) defined on the Kriging prediction and MSE (cf. Eq. (4.5)). Ordinary Kriging with the Matérn 3/2 kernel is chosen for this illustration, which is trained on 15 uniformly generated locations (black dots in the **top-left** plot).

where

$$\frac{\partial \mathbf{k}}{\partial \mathbf{x}} = \left[\frac{\partial k(\mathbf{x}, \mathbf{x}_1)}{\partial \mathbf{x}}, \frac{\partial k(\mathbf{x}, \mathbf{x}_2)}{\partial \mathbf{x}}, \ldots, \frac{\partial k(\mathbf{x}, \mathbf{x}_n)}{\partial \mathbf{x}}\right].$$

For the Matérn 3/2 kernel (Eq. (3.5)), this derivative is given as:

$$\frac{\partial k(\mathbf{x}, \mathbf{x}')}{\partial x_i} = (-1)^s \frac{3\sigma^2 h_i}{\theta_i^2} \exp\left(-\sqrt{3}\frac{h_i}{\theta_i}\right), \quad h_i = |x_i - x_i'|, \quad s = \mathbb{1}_{[x_i', \infty)}(x_i).$$

In addition, in Fig. 3.2, the gradient calculation here is visualized on a 2-D *Schwefel function*.

3.2 Cluster Kriging

Despite the theoretically sound development of the Kriging model, it suffers from several issues when applied to large data sets. The major bottleneck is the high time and memory complexity of the model fitting process: The inverse of the covariance matrix $\mathbf{K}^{-1}$ needs to be computed for both the posterior mean and variance (Eq. (3.9) and (3.11)), which has roughly $O(n^3)$ time complexity (n is the number of data points)[1]. In addition, the likelihood function of hyper-parameters $\sigma, \boldsymbol{\theta}$ is expressed through $\mathbf{K}^{-1}(\sigma^2, \boldsymbol{\theta})$. In the *Maximum Likelihood Estimation* (MLE), $\mathbf{K}^{-1}$ needs to be calculated for each likelihood value, resulting in a $O(n^3)$ computational cost per hyper-parameter evaluation. Even if efficient numerical optimizers are used in MLE, e.g., the quasi-Newton method (Bonnans et al., 2006), this computational overhead is still extremely high for a large data set. This bottleneck hinders the practical usage of Kriging/GPR. Various attempts have been made to relax the computational complexity issue of Kriging (Rasmussen and Williams, 2006). The historical approaches on this topic are categorized as follows.

Subset Methods The first category of approximation algorithms uses only a subset of the complete data set to approximate a full Kriging model. The idea behind these methods is to get a realistic representation of the complete data set by taking only a small portion of the data points. The main issue with the subset approximation approach is to select a representative subset of the data set. Two major subset approximation algorithms are:

[1] There are asymptotically faster algorithms for matrix inversion, e.g., Strassen algorithm $O(n^{2.807})$ and Stothers $O(d^{2.373})$, but their practical performance is worse than some methods with $O(n^3)$ time complexity.

- *Subset of Data* (SoD) (Lawrence, 2004) is a naive approach in reducing complexity by taking a subset of $m < n$ data points. The points are usually taken at random. The obvious disadvantage of such an approach is that possible valuable information is lost in the process. Taking a representative subset of data points is a non-trivial task.
- *Subset of Regressors* (SoR) (Silverman, 1985) approximates Kriging by a linear combination of kernel functions on a set of basis points. The basis points are linearly weighted to construct the predictor. The choice of the basis points does influence the final outcome significantly. As noted also in Quiñonero-Candela and Rasmussen (2005), there are only m (number of basis points) degrees of freedom in the model because the model degenerates, which might be too restrictive.

Approximation using Sparsity In the second category, the sparsity of the covariance kernel is exploited for the approximation. Most of algorithms in this category also use a subset of the data as in the subset approximation method.

- *Sparse On-Line Gaussian Processes* (OGP) (Csató and Opper, 2002) uses a Bayesian on-line algorithm, together with a sequential construction of a subsample of the data that specifies the prediction of the GP model. The idea behind constructing a subsample of basis vectors is very similar to the Fully Independent Training Conditional mentioned below. The advantage of OGP is that additional data points can be added to the OGP model without always completely retraining the model.
- *Gaussian Markov Random Fields* (Hartman and Hössjer, 2008) uses an approximation of the covariance matrix with a sparse precision matrix. It uses *Gaussian Markov Random Fields* (GMRF) on a reasonably dense grid to exploit the computational benefits of a Markov field while keeping the formula of Kriging weights. This method reduces the complexity for simple and ordinary Kriging, but might not always be efficient with universal Kriging.
- *Fully Independent Training Conditional* (FITC) (Naish-Guzman and Holden, 2007; Snelson and Ghahramani, 2005) uses a more sophisticated likelihood approximation with a richer covariance structure. It is a non-degenerate version of the *SoR* algorithm. By providing a set of basis points (Pseudo inputs), the model is fitted and validated on the training data. As with *SoR*

the choice of basis points is a problem and it is usually either a subset of the training data or a uniform distribution over the input space.

Divide and Conquer Methods In this category, the time complexity issue is relaxed by partitioning a big data set into several smaller subsets (or clusters) and then constructing a Kriging/GPR model on each subset. Because such a partitioning is usually obtained via clustering techniques, the subset and the model trained on them only capture local properties of the target function. Despite of the construction of local models, typically a *global predictor* is obtained by combining the local Kriging/GPR models. In this thesis, a novel divide and conquer method, called **Cluster Kriging** is proposed.

- *Bayesian Committee Machines* (BCM) (Tresp, 2000) is an algorithm similar to the ones we propose, but developed from a completely different perspective. The basic motivation is to divide a huge training set into several relatively small subsets and then construct GPR models on each subset. The benefit of this approach is that the training time on each subset is satisfactory and the training task can be easily parallelized. After training, the prediction is made by a weighted combination of estimations from all the GPR models. In addition, the batch prediction is enabled to speed up the computation even further. However, when using independent hyper-parameters for each GPR model or some GPR models are badly fitted, BCM yields unsatisfactory performance in terms of accuracy.
- *Cluster Kriging* (CK) (van Stein, Wang, Kowalczyk, Emmerich, and Bäck, 2016) combines multiple local Kriging/GPR predictors that are constructed on several partitions of the data set, where the partitions are obtained from clustering algorithms. Loosely speaking, if the whole data set is partitioned into clusters of similar sizes, Cluster Kriging will reduce the time complexity by a factor of q^2 (where q is the number of clusters), resulting in n^3/q^2, if Kriging estimators are fitted sequentially. When exploiting q CPUs in parallel, the time complexity will be further reduced to n^3/q^3. Ideally, when scaling up q to be a linear function of n, the time complexity is reduced to a linear term of n and even becomes a constant in the parallelization mode. However, in practice, such a setting on q is not suggested because it is necessary to keep enough data points in each cluster, to ensure each local Kriging model is well-fitted. To estimate $f(\mathbf{x})$ at an unobserved data point $\mathbf{x}$,

each Kriging estimator provides a (local) estimation $\hat{f}$ and it is proposed to either combine all the Kriging estimations or select the most proper Kriging estimations for $f(\mathbf{x})$. There are many options for the data partitioning, e.g., *K-means* (MacQueen et al., 1967) and *Gaussian mixture models* (Reynolds, 2009) (GMM), and the Kriging model on clusters can also be combined in different manners. By varying the options in each step of the Cluster Kriging, many algorithms can be generated. Four of them will be explained in the next section. In this section, the options in each step of the algorithms are introduced step-by-step.

Several other attempts have been made to divide the Kriging model in sub-models (Chen and Ren, 2009; Nguyen-Tuong et al., 2009). In Chen and Ren (2009), a *Bagging* (Breiman, 1996) method is proposed to increase the robustness of the Kriging algorithm, rather than speeding up the algorithm's training time. In Nguyen-Tuong et al. (2009), a partitioning method is introduced to separate the data points into local Kriging models and combine the different models using a distance metric.

All of these approximation algorithms have their advantages and disadvantages and they are compared to our newly proposed Cluster Kriging algorithms. For the empirical study, three commonly applied algorithms: *SoD*, *FITC* and *BCM* are selected to compare with the proposed approaches in this thesis.

3.2.1 Clustering

Given some data points $\mathrm{X} = \{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n\} \subset \mathrm{S}$ and corresponding response values $\mathbf{y} = (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))^\top$, the first step in Cluster Kriging is to cluster the data set $(\mathrm{X}, \mathbf{y})$ into several smaller subsets. In general, the goal is to obtain a set $\mathcal{P}$ containing q clusters on the input data set X.

$$\mathcal{P} = \{\mathrm{X}_1, \mathrm{X}_2, \ldots, \mathrm{X}_q\}, \quad \text{where } \bigcup_{i=1}^{q} \mathrm{X}_i = \mathrm{X}. \tag{3.36}$$

As with the partition on X, the response values $\mathbf{y}$ are also grouped: $\mathbf{y} = (\mathbf{y}_1^\top, \mathbf{y}_2^\top, \ldots, \mathbf{y}_q^\top)^\top$. The clustering can be done in many ways, with the most simple and feasible approach being random clustering. For our framework, however, we introduce three more sophisticated partitioning methods that are used in the experiments later on.

Hard Clustering Hard clustering splits the data into k smaller *disjoint* data sets: $\mathrm{X}_i \bigcap \mathrm{X}_j = \emptyset$ $(i \neq j)$. This can be achieved by various methods, for instance the K-means algorithm (MacQueen et al., 1967). K-means clustering minimizes the within-cluster sum of squares, that is expressed as:

$$\arg\min_{\mathcal{P}} \sum_{i=1}^{q} \sum_{\mathbf{x} \in \mathrm{X}_i} \left\| \mathbf{x} - \boldsymbol{\mu}_i \right\|^2, \tag{3.37}$$

where $\boldsymbol{\mu}_i \in \mathbb{R}^d$ is the centroid of cluster i and is calculated as the mean of the points in X_i. The evaluation of the within-cluster sum of squares takes $O(nqd)$ execution time.

Soft Clustering Instead of using a hard clustering approach, a fuzzy clustering algorithm can be used to introduce slight overlap between the various smaller data sets, which might increase the final model accuracy. To incorporate fuzzy clustering, instead of directly applying cluster labels, the probabilities that a point belongs to a cluster are calculated (Eq. (3.39)). This probability is called the membership value of a point to a cluster. With ν a user defined setting that defines the overlap, $\lceil \nu n/q \rceil$ number of points with the highest membership values are assigned for each cluster. Here ν is set between 1 (no overlap) and 2 (completely overlapping clusters).

In principle, any fuzzy clustering algorithm can be used for the partitioning. In this thesis the *Fuzzy C-means* (FCM) (Dunn, 1973) clustering algorithm and the *Gaussian Mixture Models* (GMM) (Reynolds, 2009) are used. FCM is a clustering algorithm very similar to the well known *K-means*. The algorithm differs from K-means in that it has additional membership coefficients and a fuzzifier. The membership coefficients of a given point give the degrees that this point belongs to each cluster. These coefficients are normalized so they sum up to one. The algorithm can be fitted on a given data set and returns the coefficients for each data point to each cluster. The number of clusters is a user defined parameter. Fuzzy C-means optimizes the objective function given in Eq. (3.38) iteratively. In each iteration, the membership coefficients of each point being in the clusters are computed using Eq. (3.39). Subsequently, the centroid of each cluster $\boldsymbol{\mu}_j$ is computed as the center of mass of all data points, taking the membership coefficients as weights. The objective of fuzzy C-means is to find a set of centroids

that minimizes the following function:

$$\sum_{i=1}^{n}\sum_{j=1}^{q} w_{ij}^{m} \left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|^2, \tag{3.38}$$

where w_{ij} are the membership values (see Eq. 3.39) and m is the so-called fuzzifier ($m = 2$ in this thesis). The fuzzifier determines the level of cluster fuzziness as follows:

$$w_{ij}^{m} = \frac{1}{\sum_{k=1}^{q}\left(\frac{\left\|\mathbf{x}_i - \boldsymbol{\mu}_j\right\|}{\left\|\mathbf{x}_i - \boldsymbol{\mu}_k\right\|}\right)^{\frac{2}{m-1}}} \tag{3.39}$$

The other fuzzy clustering procedure used is the Gaussian Mixture Models. GMM are used together with the *expectation-maximization* (EM) (Sundberg, 1974) algorithm for fitting the Gaussian models. The mixture models are fitted on the training data and later used in the weighted combination of the Kriging models by estimating cluster membership probabilities of the unseen data points. The advantage of this clustering technique is that it is fairly robust and that the number of clusters can be specified by the user. For the GMM method one could use the full covariance matrix whenever the dimensionality of the input data is small. However, when working with high dimensional data a diagonal covariance matrix can be used instead. The time complexity of GMM depends on the underlying EM algorithm. In each iteration of EM, it takes $O(nq)$ operations to re-estimate the model parameters.

Regression Tree Partitioning The third method used is the partitioning by use of a Regression Tree (Breiman et al., 1984) on the complete training set. The regression tree splits the data set recursively at the best splitting point using the variance reduction criterion. Each leaf node of the Regression Tree represents a cluster of data points. The number of leaves (or the number of records per leave) can be set by the user. By reducing the variance in each leaf node and therefore the variance in each data set, the Kriging models can be fitted to the local data sets much better as will be presented later on. The time complexity of using a Regression Tree for the partitioning is $O(n)$, given that the depth of the tree or the number of leaf nodes is set by the user.

The partitioning done by the regression tree depends on the splitting criterion. For a faster execution of the Cluster Kriging algorithm we could choose to use

a splitting criterion that splits the data set in each node evenly, balancing the load for each of the local Kriging models attached to the leafs. From emprical experience we know that splitting using the standard variance reduction function generally results in better performing models than using such an evenly splitting criterion. This is likely due to the fact that data sets with a lower variance can be more easily fitted by a Kriging model.

3.2.2 Modeling

Technically, modeling the function f using Kriging/GPR implies using the stochastic process $\{Y(\mathbf{x}) : \mathbf{x} \in \mathrm{S}\}$ (cf. Eq. (3.1)) as the *statistical model* of f. Under this setting, the response values $\mathbf{y}$ are treated as the observations from Y. After partitioning the data set into several clusters, Kriging/GPR models are fitted on each of the smaller data sets. Consider the random vector $\boldsymbol{\psi} = (Y(\mathbf{x}_1), Y(\mathbf{x}_2), \ldots, Y(\mathbf{x}_n))^\top$. It is also partitioned according to the clustering on X: $\boldsymbol{\psi} = (\boldsymbol{\psi}_1^\top, \boldsymbol{\psi}_2^\top, \ldots, \boldsymbol{\psi}_q^\top)^\top$. For simplicity we assume the kernel functions used on each cluster to be the same and *Ordinary Kriging* is used in each cluster. Typically, each cluster only captures the local information about f and thus the Kriging model on each cluster shall be called *local Kriging/GPR model*. On each cluster, the (local) posterior distribution of the $Y(\mathbf{x})$ is:

$$Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i \sim \boldsymbol{\mathcal{N}}\left(\hat{f}_i(\mathbf{x}), s_i^2(\mathbf{x})\right), \quad i = 1, 2, \ldots, q, \tag{3.40}$$

where $\hat{f}_i$ and s_i^2 are the Kriging estimator and MSE in Eq. (3.9) and (3.11) except that the observations $\mathbf{y}_i$ is now only a fraction of the whole observations $\mathbf{y}$. Note that training the Kriging estimator can be easily parallelized, which gives an additional speedup to Cluster Kriging. Another benefit of building each model separately, is that each model has usually a much better local fit than a single global Kriging model would obtain.

3.2.3 Cluster Kriging Predictor

For the prediction, several approaches are proposed in the following. Depending on the partitioning method used before, the simplest approach to predict the unseen data point is by using a single local model. When the partitions are overlapping a combination of the different local models into one global model is then required.

Single Cluster Predictor The simplest method is to pick just one local Kriging model for each data point and use this local model for the prediction. This does require the partitioning used to create partitions based on locality like K-means clustering or a regression tree. First the partitioning method is used to predict which cluster the new data point belongs to, then the Kriging model trained using this particular cluster is used to predict the mean and variance at the new data point. In case of the Regression Tree procedure, the targets are predicted from new unseen data points by first deciding which model needs to be used, using the Regression Tree. The target is then predicted using the specific Kriging model assigned to the leaf node. The main advantage of this method is that there is no combination of different predictions and only one of the local Kriging models needs to provide a prediction. This results in a significant speed-up for the prediction task. Disadvantages of this method are 1) a potential inability of capturing the global trend of the target function and 2) artificial discontinuities at the boundary of partitions. In Fig. 3.3 (top row), we visualize a Cluster Kriging model using regression trees, in which the intersections between the different local models are marked by black dashed lines. It can be observed that the edges of the local models are not completely matching, meaning that the predictions near the border are not as smooth as they would be in a global Kriging model. It can also be observed that the area covered by each cluster is not the same, which is due to the splitting criterion of the regression tree. While the splitting criterion could be chosen in such a way that it balances the cluster sizes, using variance reduction as the splitting criterion generally gives better fitted local models.

Superposition of Posterior Processes Instead of using single model predictions, the multiple local models can be combined into one global model using various combination procedures. Some additional assumptions are necessary to give the following derivation. Assume an independent Gaussian process prior on each cluster:

$$\forall i \neq j \in \{1, 2, \ldots, q\}, \quad Y_i \perp\!\!\!\perp Y_j, \quad Y_i \sim t + \mathcal{GP}(0, k(\cdot, \cdot)).$$

After clustering (e.g., *K-means*) the data set $(\mathrm{X}, \mathbf{y})$, independent posterior Gaussian processes Y_i' are obtained on each cluster:

$$Y_i' \coloneqq Y_i \mid \boldsymbol{\psi}_i \sim \mathcal{GP}(\hat{f}_i, k_i'(\cdot, \cdot)),$$

where the posterior mean $\hat{f}_i$ and kernel k_i' are defined in Eq. (3.32) and (3.33). In this sense, it is possible to construct a "global" Gaussian process as the superposition of all posterior Gaussian processes. In addition, a weighting scheme $\{w_i\}_i$ is used to model how much "belief" should be put on each posterior process. Using positive weights whose sum is one, the posterior process is:

$$\mathcal{Y} \coloneqq \sum_{i=1}^{q} w_i Y_i' \sim \mathcal{GP}\left(\sum_{i=1}^{q} w_i \hat{f}_i, \sum_{i=1}^{q} w_i^2 k_i'(\cdot,\cdot)\right),$$

The posterior kernel is derived as follows: consider the covariance between $\mathcal{Y}(\mathbf{x}_1)$ and $\mathcal{Y}(\mathbf{x}_2)$:

$$\begin{aligned}\operatorname{Cov}\left\{\sum_{i=1}^{q} w_i Y_i'(\mathbf{x}_1), \sum_{j=1}^{q} w_j Y_j'(\mathbf{x}_2)\right\} &= \sum_{i=1}^{q}\sum_{j=1}^{q} w_i w_j \operatorname{Cov}\left\{Y_i'(\mathbf{x}_1), Y_j'(\mathbf{x}_2)\right\} \\ &= \sum_{i=1}^{q} w_i^2 k_i'(\mathbf{x}_1, \mathbf{x}_2).\end{aligned}$$

At an unobserved point $\mathbf{x}$, the following predictive distribution is obtained,

$$\mathcal{Y}(\mathbf{x}) \sim \mathcal{N}\left(\sum_{i=1}^{q} w_i \hat{f}_i(\mathbf{x}), \sum_{i=1}^{q} w_i^2 s_i^2(\mathbf{x})\right), \tag{3.41}$$

where $s_i^2(\mathbf{x}) = k_i'(\mathbf{x},\mathbf{x})$. The best linear unbiased predictor of $\mathcal{Y}$ is its mean function: $\widehat{\mathcal{Y}} = \sum_{i=1}^{q} w_i \hat{f}_i$ and its MSE is the variance $\sum_{i=1}^{q} w_i^2 s_i^2$. Note that the predictor and its MSE depend on the choice of weights. The optimal predictor is defined in the sense that the MSE is minimized with respect to the weight (van Stein, Wang, Kowalczyk, Bäck, and Emmerich, 2015):

$$\begin{aligned}\underset{\{w_1,\ldots,w_q\}}{\text{minimize}} \quad & \sum_{i=1}^{q} w_i^2 s_i^2(\mathbf{x}) \\ \text{subject to} \quad & \sum_{i=1}^{q} w_i = 1, \quad w_i \geq 0, \quad i = 1,\ldots,q.\end{aligned}$$

This convex optimization problem can be solved by using Lagrange Multipliers, resulting in:

$$w_i^*(\mathbf{x}) = \frac{1/s_i^2(\mathbf{x})}{\sum_{j=1}^{q} 1/s_j^2(\mathbf{x})}. \tag{3.42}$$

The optimal weights are then used to construct the optimal predictor, which is the inner product of the model predictions with the optimal weights.

Mixture of Posterior Processes As an alternative to the linear predictor given in Eq. (3.41) that arises from the superposition of posterior processes, we formulate another linear predictor here, resulting from the *mixture* of posterior processes. Firstly, the combination weights are motivated a bit differently: for the GMM and other soft clustering approaches, the membership probabilities can be used for unseen records to define the weights for the combination of predictions. For instance, given a point $\mathbf{x}$, the weights are defined as,

$$w_i \coloneqq \Pr(C = i \mid \mathbf{x}), \quad i = 1, \ldots, q, \tag{3.43}$$

where C is the cluster indicator variable ranging from 1 to q. Note that those weights can be given by the clustering algorithm or obtained by an optimization procedure (see below). Secondly, instead of considering an independent Gaussian process prior for each cluster, *a single and global Gaussian process prior is assumed for all clusters*. By applying the total probability with respect to the cluster indicator variable C, the conditional density of Y over $\boldsymbol{\psi}$ is (van Stein, Wang, Kowalczyk, Emmerich, and Bäck, 2016):

$$\begin{aligned} p(Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}) &= \sum_{i=1}^{q} p(Y(\mathbf{x}), C = i \mid \boldsymbol{\psi} = \mathbf{y}, \mathbf{x}) \\ &= \sum_{i=1}^{q} p(Y(\mathbf{x}) \mid C = i, \boldsymbol{\psi} = \mathbf{y}) \Pr(C = i \mid \mathbf{x}) \\ &\approx \sum_{i=1}^{q} p(Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i) \Pr(C = i \mid \mathbf{x}). \end{aligned} \tag{3.44}$$

Note that we approximate the density $p(Y(\mathbf{x}) \mid C = i, \boldsymbol{\psi} = \mathbf{y})$ by $p(Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i)$. Such an approximation is accurate when the amount of the overlap between clusters is small. In Eq. (3.44), the first term within the summation is the posterior density obtained from cluster i. The second term represents the probability that data point $\mathbf{x}$ belonging to a cluster, which is the weight in Eq. (3.43). Consequently, the overall predictive density $p(Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y})$ comes from the *mixture of posterior processes*. According to statistical decision theory (Hastie et al., 2009), the best predictor of Y when knowing the conditional density of Y on $\mathbf{y}$ is the conditional

expectation, i.e.,

$$\begin{aligned}\mathbb{E}\left\{Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}\right\} &= \int_{-\infty}^{\infty} y \sum_{i=1}^{q} p(Y(\mathbf{y}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i) \Pr(C = i \mid \mathbf{x}) \, \mathrm{d}y \\ &= \sum_{i=1}^{q} \Pr(C = i \mid \mathbf{x}) \mathbb{E}\left\{Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i\right\} \\ &= \sum_{i=1}^{q} w_i \hat{f}_i(\mathbf{x}). \end{aligned} \tag{3.45}$$

In contrast to Eq. (3.41), the predictor above is also a linear combination of Kriging predictors from all clusters. However, the differences are 1) the predictive density $p(Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y})$ is no longer Gaussian, 2) the weights in Eq. (3.41) are resulted from an optimization procedure while the weights in Eq. (3.45) are either given directly by the clustering algorithm or obtained from the optimization. To optimize the weights, please consider the MSE of this predictor, which is the variance of the mixture of posterior processes:

$$\begin{aligned}&\mathrm{Var}\left\{Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}\right\} \\ &= \mathbb{E}\left\{Y(\mathbf{x})^2 \mid \boldsymbol{\psi} = \mathbf{y}\right\} - \left(\mathbb{E}\left\{Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}\right\}\right)^2 \\ &= \sum_{i=1}^{q} w_i \left(\mathrm{Var}\{Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i\} + \left(\mathbb{E}\{Y(\mathbf{x}) \mid \boldsymbol{\psi}_i = \mathbf{y}_i\}\right)^2\right) - \left(\mathbb{E}\{Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}\}\right)^2 \\ &= \sum_{i=1}^{q} w_i \left(s_i^2(\mathbf{x}) + \hat{f}_i^2(\mathbf{x})\right) - \left(\sum_{i=1}^{q} w_i \hat{f}_i(\mathbf{x})\right)^2. \end{aligned} \tag{3.46}$$

Again, the weights are considered optimal in the sense that the MSE is minimized:

$$\begin{aligned} \underset{\{w_1,\ldots,w_q\}}{\text{minimize}} \quad & \mathrm{Var}\left\{Y(\mathbf{x}) \mid \boldsymbol{\psi} = \mathbf{y}\right\} \\ \text{subject to} \quad & \sum_{i=1}^{q} w_i = 1, \quad w_i \geq 0, \quad i = 1, \ldots, q. \end{aligned}$$

Cluster Kriging Variants By choosing different methods for the clustering and prediction, various Cluster Kriging variants are instantiated:

- *Optimally Weighted Cluster Kriging* (OWCK) uses a *K-means* clustering algorithm for the partitioning and the superposition of posterior processes to construct the predictor.

- *Optimally Weighted Fuzzy Cluster Kriging* (OWFCK) is similar to OWCK except that *K-means* is replaced by *Fuzzy C-means*.

- *Gaussian Mixture Model Cluster Kriging* (GMMCK) uses Gaussian Mixture Models to partition the data into q overlapping clusters and the membership probabilities are used as the combination weights. The mixture of posterior Gaussian processes (Eq. (3.45)) is used for the prediction.

- *Model Tree Cluster Kriging* (MTCK) uses a regression tree to partition the data in the objective space. The tree is generated from the root node by recursively splitting the training data using the target variable and the variance reduction criterion. Once a node contains less than the minimum samples needed to split or the node contains only one record, the splitting stops and the node is called a leaf. To control the number of clusters, the user can set the maximum number of leaves or the minimum leaf size. Next, each leaf node is assigned a unique index and each record belonging to the leaf is assigned to this index. For each leaf, a Kriging predictor is built using only those records assigned to this leaf. For the prediction, the regression tree decides which Kriging predictor should be used.

3.2.4 Experiments

A broad variety of experiments is conducted to compare Optimally Weighted Cluster Kriging and its Fuzzy and Model Tree variants, to a wide set of other Kriging approximation algorithms. The algorithms included in the test are: *Bayesian Committee Machines*, both with shared parameters (BCM sh.) and with individual parameters (BCM), *Subset of Data* (SoD), *Fully Independent Training Conditional* (FITC), *Optimally Weighted Cluster Kriging* (OWCK) using K-means clustering, Fuzzy Cluster Kriging using Fuzzy C-means (OWFCK), Fuzzy Cluster Kriging with Gaussian Mixture Models (GMMCK) and finally Model Tree Cluster Kriging (MTCK). The algorithms are evaluated on three different data sets from the *UCI machine learning repository* (Bache and Lichman, 2013):

- *Concrete Strength* (Yeh, 1998) is a data set with 1030 records, 8 attributes and one target attribute. The task is to predict the strength of concrete.

- *Combined Cycle Power Plant* (CCPP) (Kaya et al., 2012) is a data set of 9568 records, 3 attributes and one target attribute. The target is the hourly electrical energy output and the task is to predict this target.
- *SARCOS* (Vijayakumar et al., 2005) is a data set from *gaussianprocess.org* with a training set of 44484 records, 21 attributes and 7 target attributes. The task is to predict the joint torques of an anthropomorphic robot arm. All 21 attributes are used as training data but only the 1^{st} target attribute is used as target. The data set comes with a predefined test set of 4449 records.

For the *Concrete Strength* data set and all synthetic data sets: FITC is set to a range of inducing points starting from 32 and increasing in powers of 2 to 512. SoD is set to the same range as FITC but for SoD this means the number of data points. BCM, both shared and non-shared versions and all *Cluster Kriging* variants are set to a range from 2 to 32 clusters, increasing with powers of 2. For the *Combined Cycle Power Plant* data set: FITC is set to a range of inducing points starting from 64 and increasing in powers of 2 to 1024. *SoD* is set to a range from 256 to 4092 data points. BCM, both shared and non-shared versions and all *Cluster Kriging* variants are set to a range from 4 to 64 clusters. Finally, for the *SARCOS* data set, the range of FITC's inducing points stays the same as for the CCPP data set, for SoD the range is from 512 to 8184 data points, and for all cluster based algorithms and the model tree variant, the range is set from 8 to 128 clusters.

In addition, 8 synthetic data sets with each 10000 records, 20 attributes and one target attribute are used. The synthetic data sets are generated on common benchmark functions: *Ackley*, *Schaffer*, *Schwefel*, *Rastrigin*, *H1*, *Rosenbrock*, *Himmelblau* and *Diffpow*. The implementations of those functions are taken from the *Deap* Python Package (Fortin et al., 2012).

Hyper-parameter Optimization As mentioned before, Ordinary Kriging is chosen for all the clusters throughout this thesis. For each local Ordinary Kriging model, its constant trend β is estimated using the GLS (Generalized Least Squares) formula (Eq. (3.9)). Consequently, the so-called profile log-likelihood is adopted to estimate the hyper-parameter. In each local Kriging model, hyper-parameters $\sigma^2, \boldsymbol{\theta}$ of the kernel function are optimized using the Maximum Likelihood Estimation (MLE) method. As for the choice of numerical optimization algorithm, we use a quasi-Newton method (BFGS) (Fletcher, 2013) with restarting heuristic.

Table 3.2: Average R^2 score per data set for each algorithm

Data set	SOD	OWCK	GMMCK	OWFCK	FITC	BCM	BCM sh.	MTCK
concrete	0.784	0.826	0.839	0.696	0.675	−81.888	−242.459	**0.851**
CCPP	0.948	0.937	0.968	0.916	0.890	0.220	−24.602	**0.968**
sarcos	0.964	0.894	0.996	0.570	0.941	−627.280	0.448	**0.999**
ackley	0.952	0.957	0.951	0.954	0.260	0.921	−0.039	**0.981**
schaffer	0.321	0.388	0.369	0.406	0.208	0.452	−0.050	**0.672**
schwefel	0.990	0.973	0.977	0.947	0.006	0.969	−0.043	**0.999**
rast	0.973	0.947	0.948	0.932	0.322	0.914	−0.043	**0.998**
h1	0.676	−0.082	0.527	−1.125	0.165	0.657	−0.046	**0.977**
rosenbrock	0.999	0.997	0.997	0.981	0.000	0.994	−0.050	**1.000**
himmelblau	0.997	0.995	0.995	0.981	0.291	0.994	−0.044	**1.000**
diffpow	0.995	0.991	0.991	0.975	0.001	−0.001	−0.001	**1.000**

Table 3.3: Average MSLL score per data set for each algorithm

Data set	SOD	OWCK	GMMCK	OWFCK	FITC	BCM	BCM sh.	MTCK
concrete	−0.837	−0.946	−1.100	−0.692	−0.629	18.590	68.013	**−1.140**
CCPP	−0.089	−1.438	−1.525	−1.109	−1.165	7.826	69.346	−1.193
sarcos	−1.926	−1.371	−3.147	−0.302	−1.463	780.090	507.721	**−3.429**
ackley	−1.622	−1.516	−1.517	−1.462	−0.104	7.352	13.010	**−2.012**
schaffer	0.477	−0.073	0.081	−0.091	−0.107	16.872	11.707	**−0.514**
schwefel	−2.554	−2.013	−2.162	−1.944	−0.002	−0.144	12.034	**−3.278**
rast	−2.179	−1.686	−1.807	−1.642	−0.193	4.554	11.590	**−2.901**
h1	−0.766	−0.276	−0.540	−0.060	−0.059	9.018	17.393	**−1.967**
rosenbrock	−3.479	−2.915	−3.074	−2.738	high*	0.612	18.575	**−4.054**
himmelblau	−3.204	−2.646	−2.790	−2.553	−0.193	−1.422	12.826	**−3.739**
diffpow	−3.020	−2.548	−2.666	−2.438	high*	high*	high*	**−3.744**

Table 3.4: Average SMSE score per data set for each algorithm

Data set	SOD	OWCK	GMM-CK	FCM-CK	FITC	BCM	BCM sh.	MTCK
concrete	0.216	0.174	0.161	0.304	0.325	82.888	243.459	**0.149**
CCPP	0.052	0.063	0.032	0.084	0.110	0.780	25.602	**0.032**
sarcos	0.036	0.106	0.004	0.430	0.059	628.280	0.552	**0.001**
ackley	0.048	0.043	0.049	0.046	0.740	0.079	1.039	**0.019**
schaffer	0.679	0.612	0.631	0.594	0.792	0.548	1.050	**0.328**
schwefel	0.010	0.027	0.023	0.053	0.994	0.031	1.043	**0.001**
rast	0.027	0.053	0.052	0.068	0.678	0.086	1.043	**0.002**
h1	0.324	1.082	0.473	2.125	0.835	0.343	1.046	**0.023**
rosenbrock	0.001	0.003	0.003	0.019	1.000	0.006	1.050	**0.000**
himmelblau	0.003	0.005	0.005	0.019	0.709	0.006	1.044	**0.000**
diffpow	0.005	0.009	0.009	0.025	0.999	1.001	1.001	**0.000**

Whenever the fuzzy clustering algorithm is applied, the overlap rate ν is set to 10%, which is chosen based empirical investigations: although higher percentages (above 10%) usually increase the accuracy marginally, it also brings additional computational costs as each cluster becomes larger. For the Model Tree variant, the number of leaves is enforced by setting a minimum number of data points per leaf and an optional maximum number of leaves.

Quality Measurements The quality of the experiments is estimated with the help of 5-fold cross validation, except for the *SARCOS* data set, which uses its predefined test set. The experiments are performed in a test framework similar to the framework proposed in (Chalupka et al., 2013), i.e., several quality measurements are used to evaluate the performance of each algorithm. The *Coefficient of determination* R^2 score, *Mean Standardized Log Loss* (MSLL) (Rasmussen and Williams, 2006) and the *Standardized Mean Squared Error* (SMSE) are measured for each test run. The *Mean Standardized Log Loss* is a measurement that takes both the prediction and MSE (estimated by the model) into account, penalizing inaccurate predictions that have small estimated MSEs. For MSLL and SMSE lower scores are better, for R^2, 1.0 is the best possible score meaning a perfect fit and everything lower is worse.

Results On real-world data sets *Concrete Strength*, *CCPP* and *SARCOS*, the experiment results are summarized in the following tables. Two performance measures, time and accuracy (x and y axis respectively) are shown. The R^2 scores of each data set per algorithm, averaged over all folds, are shown in Table 3.2. The MSLL scores are provided in Tab. 3.3 and the SMSE scores in Tab. 3.4. The best results for each data set are indicated in bold face.

3.3 Cluster Kriging and EGO

When applying the EGO algorithm to a large initial data set (e.g., in the experiment design), typically the Kriging model is re-trained in every iteration and the CPU time spent on the hyper-parameter re-estimation becomes computationally infeasible. To relax this bottleneck, it is proposed to use the Cluster Kriging algorithm in an EGO algorithm. Specifically, the following three Cluster Kriging variants shall be used:

- *Cluster Kriging* (OWCK)
- *Gaussian Mixture Model Cluster Kriging* (GMMCK)
- *Model Tree Cluster Kriging* (MTCK)

When choosing the MTCK variant, it brings several other advantages than the time complexity reduction. 1) The search space is recursively divided into smaller hypercubes, in a manner that the variance of the target value on each node is greedily reduced. Such a reduced the variance of target values in each cluster potentially facilitates the numerical stability in the model training, because the covariance matrix $\mathbf{K}$ tends to become singular when the target value varies abruptly in the local scale. 2) For the infill criterion, multi-modality is artificially created as a by-product of applying the MTCK variant. Intuitively, as independent Kriging/GPR models are trained on each tree partitions, the prediction MSE increases rapidly around the boundary of the partition. Potentially, this behavior results in local optimality of the infill criterion on each partition. Using this artefact, multiple distinct and potentially well-performing points can be proposed for the evaluation. Essentially, this is an alternative approach to the *infill criterion parallelization* problem stated in Section 4.5. Our argument is visually validated in Fig. 3.3. Here 500 data points are sampled on the 2-D Ackley function using the *Halton sequence* (Niederreiter, 1992). It is important to observe that the prediction MSE shows basins of attraction on each partition. Consequently, the expected improvement criterion also exhibits basins of attraction on each partition and thus is highly multi-modal. For each partition, the local maximum of EI is indicated by the red star symbol.

3.3.1 The algorithm

Although various complexity reduction (or approximation) methods exist for Kriging (for instance, FITC (Naish-Guzman and Holden, 2007; Snelson and Ghahramani, 2005) and Bayesian Committee Machines (Tresp, 2000)), we state that Cluster Kriging is more suitable for the EGO algorithm for the following reasons (Wang et al., 2017):

1. Kriging predictors (posterior processes in Eq. (3.40)) on each cluster can be trained in parallel, which yields an additional linear speedup in practice.

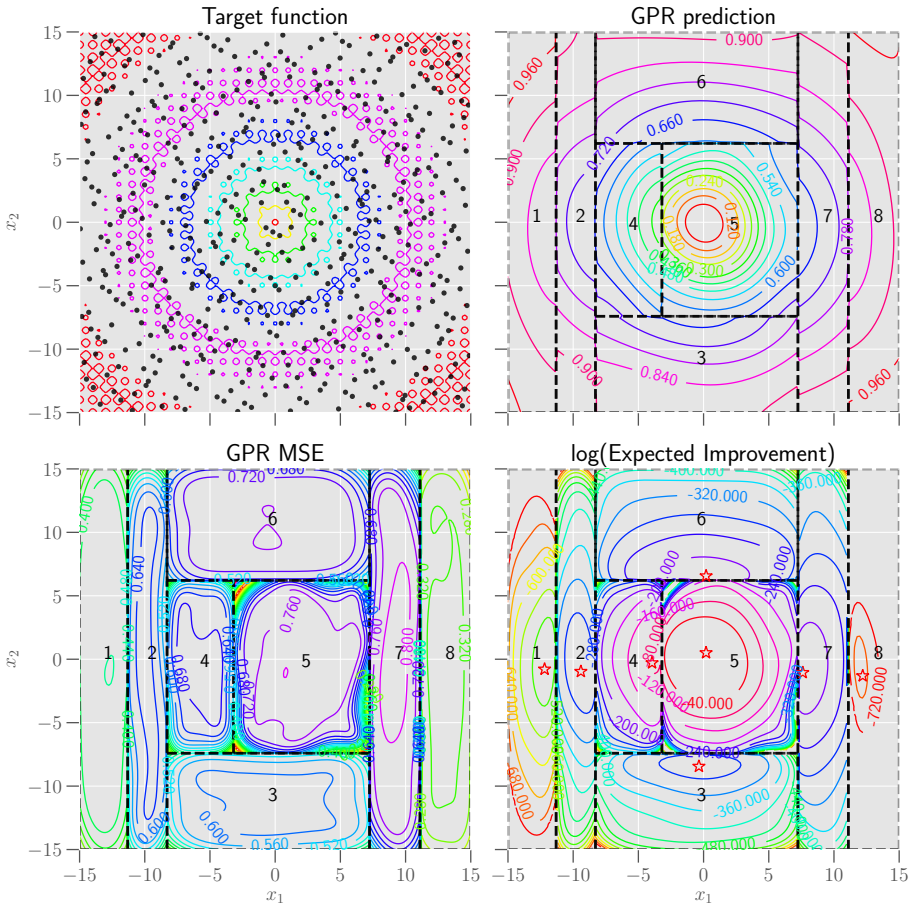


Figure 3.3: On the 2-D *Ackley function* (**top-left**), 500 random points (black dots in the top-left plot) are generated using the *Halton sequence* (Niederreiter, 1992). A Model Tree Cluster Kriging (MTCK) with the Gaussian kernel is trained on the data point, where the underlying tree clusters are indicated by dashed lines (except the top-left plot). Contour lines are depicted for the Kriging/GPR prediction (**top-right**), the Kriging/GPR MSE (**bottom-left**) and the so-called *Expected Improvement* (EI) criterion (**bottom-right**) defined on the Kriging prediction and MSE (cf. Eq. (4.5)). Multiple local maxima (☆ in the bottom-right plot) of EI are obtained by conducting the quasi-Newton search on each cluster.

2. After a new candidate solution is found via the optimization on the infill criterion, the hyper-parameters of Kriging need to be re-estimated. Taking the cluster information into account, it is proposed to only re-estimate the hyper-parameters on the clusters that this new solution belongs to. This operation leads to additional speedup in model training, as in the best scenario, only one local Kriging predictor is re-trained.

3. The infill criterion, e.g., the expected improvement is still well-defined over the Cluster Kriging because either the Gaussian posterior process (Eq. (3.41)) or the mean and variance function (Eq. (3.45) and (3.46)) are available.

The resulting algorithm is presented in Alg. 7. Note that, the training of the initial Kriging models can be parallelized (line 4). The counter c is used to keep track of the number of the recently evaluated data points. When c is bigger than 10% of the size of the data set, the clustering procedure is performed again to balance the size of clusters.

The commonly used infill criteria, e.g., Expected Improvement (Eq. (4.5)) remain well-defined on all the variants of Cluster Kriging in the following sense. Usually infill criteria are defined over the posterior process and take the Gaussian assumption on it. Some Cluster Kriging variants, e.g., superposition of posterior processes (Eq. (3.41)) admit a *Gaussian* posterior. For the others, e.g., mixture of the posterior processes (Eq. (3.45)), we argue that although the posterior is not Gaussian any longer, it is accurate enough to use the first- and second-order structure of the posterior for infill criteria.

For the optimization of the infill criterion (line 8 in Alg. 7), it is possible to exploit fast black-box optimization algorithms, for instance the well-known *Covariance Matrix Adaptation Evolution Strategy* (CMA-ES) (Hansen, 2006; Hansen and Ostermeier, 2001), because the evaluation of the infill criterion is not expensive compared to the Kriging fitting procedure. However, as most of infill criteria have a closed-form, it is straightforward to explore the gradient field of infill criteria. And the global optimization can be conducted by applying a quasi-Newton method with random restarts. To align with existing work (Roustant et al., 2012) on using gradient-based optimization techniques for the infill criteria, we give the gradient of the predictor and MSE in Cluster Kriging, as they are required to differentiate most of infill criteria. For the superposition of posterior processes (Eq. (3.41)), the

Algorithm 7 Cluster Kriging assisted Efficient Global Optimization

procedure CK-EGO($\mathrm{X}, \mathbf{y}, f, q, \mathscr{A}$) ▷ q: number of clusters
  $\{\mathrm{X}_i, \mathbf{y}_i\}_{i=1}^{q} \leftarrow$ CLUSTERING($\mathrm{X}, \mathbf{y}, q$)
  for $i = 1 \rightarrow q$ **do**
    $Y \mid \mathbf{y}_i \sim \mathcal{N}\left(\hat{f}_i, s_i^2\right)$ ▷ Train the Kriging predictor on each cluster
  end for
  $c \leftarrow 0$
  while the stop criteria are not fulfilled **do**
    $\mathbf{x}^* \leftarrow \arg\max_{\mathbf{x} \in \mathrm{S}} \mathscr{A}(\mathbf{x})$ ▷ Maximize the infill criterion
    $y^* \leftarrow f(\mathbf{x}^*)$ ▷ Evaluation
    $c \leftarrow c + 1$
    if $c/|\mathrm{X}| > 10\%$ **then** ▷ $|\mathrm{X}|$: cardinality of X
      $\mathrm{X}, \mathbf{y} \leftarrow$ MERGE($\{\mathrm{X}_i, \mathbf{y}_i\}_{i=1}^{q}$)
      $\{\mathrm{X}_i, \mathbf{y}_i\}_{i=1}^{q} \leftarrow$ CLUSTERING($\mathrm{X}, \mathbf{y}, q$) ▷ Re-clustering
      for $i = 1 \rightarrow q$ **do**
        $Y \mid \mathbf{y}_i \sim \mathcal{N}\left(\hat{f}_i, s_i^2\right)$
      end for
      $c \leftarrow 0$
    else
      for every cluster i that $\mathbf{x}^*$ belongs to **do**
        $\mathrm{X}_i \leftarrow \mathrm{X}_i \cup \{\mathbf{x}^*\},\ \mathbf{y}_i \leftarrow (\mathbf{y}_i^\top, y^*)^\top$ ▷ Extend the data set
        $Y \mid \mathbf{y}_i \sim \mathcal{N}\left(\hat{f}_i, s_i^2\right)$ ▷ Re-train the predictor on cluster i
      end for
    end if
  end while
  return $\mathbf{x}^*$
end procedure

gradients of its predictor and MSE are (cf. Eq. (3.41) and (3.42)):

$$\frac{\partial \hat{f}}{\partial \mathbf{x}} = \sum_{i=1}^{q} \left(w_i \frac{\partial \hat{f}_i}{\partial \mathbf{x}} + \hat{f}_i \frac{\partial w_i}{\partial \mathbf{x}} \right)$$
$$\frac{\partial s^2}{\partial \mathbf{x}} = \sum_{i=1}^{q} \left(w_i^2 \frac{\partial s_i^2}{\partial \mathbf{x}} + 2 w_i s_i^2 \frac{\partial w_i}{\partial \mathbf{x}} \right)$$
$$\frac{\partial w_i}{\partial \mathbf{x}} = \sum_{i=1}^{q} \left(\frac{1}{s_i^4 M} \frac{\partial s_i^2}{\partial \mathbf{x}} + \frac{1}{s_i^2 M^2} \sum_{i=1}^{q} \frac{1}{s_i^4} \frac{\partial s_i^2}{\partial \mathbf{x}} \right), \quad M = \sum_{j=1}^{q} \left(s_j^2 \right)^{-1}$$

The gradient of the Kriging predictor and MSE on each cluster, $\partial \hat{f}_i / \partial \mathbf{x}, \partial s_i^2 / \partial \mathbf{x}$, are given in Eq. (3.34) and (3.35). For the mixture of posterior processes, its gradient can be obtained in a similar way. This is omitted here for the sake of simplicity.

In addition, for the Tree-based local Kriging models (MTCK), it is shown (Fig. **??**) that each cluster (leaf node) can be treated as a sub-problem in the infill criteria optimization. Therefore, it might be more efficient to conduct independent searches in each leaf region of the Regression Tree and choose the best point from all these sub-problems.

3.3.2 Experiments

Several experiments are conducted to show both the empirical time complexity and convergence rate of the proposed Cluster Kriging based EGO, including all the variants of Cluster Kriging discussed in Section 3.2.3. The performance of the proposed algorithm is compared to the original EGO that uses *Ordinary Kriging* (OK). For our experiments, the benchmark functions chosen are *Ackley*, *Rastrigin* and *Schaffer*. These functions are chosen because they are used often in optimization experiments, are highly multi modal, and are of a relatively high complexity.

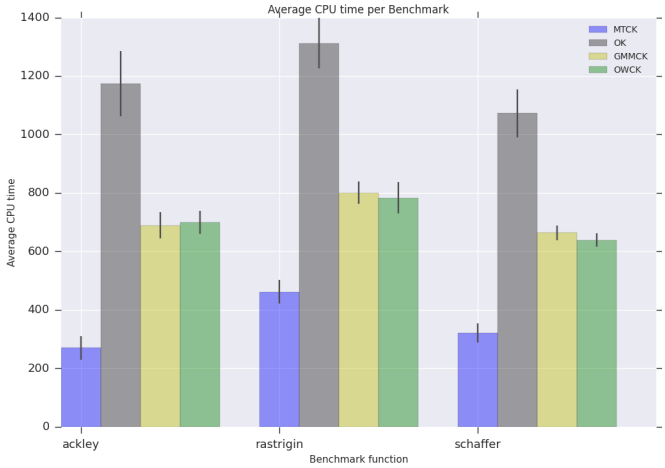
The algorithms under comparisons are: EGO with Ordinary Kriging (OK), Tree-based local Kriging models (MTCK), Superposition of Kriging models (OWCK) and the mixture of Kriging models (GMMCK). Each of the Cluster Kriging variants uses 5 clusters. Both execution time and convergence rate are being measured with a fixed set of EGO iterations and optimization budget. The convergence is measured by taking the absolute error between the real optimum of the benchmark

functions and the found optimum for each iteration of EGO. Each EGO run performs 10 iterations for the three benchmark functions in two dimensions. Three different initial sample sizes (500, 1000 and 5000) are used to train the surrogate models, in order to illustrate the growth of CPU time required per algorithm, when the size of the data available increases. For each different experimental setup, the average time and distance to the optimum is recorded over 20 runs with different random seed ($[0, 20]$).

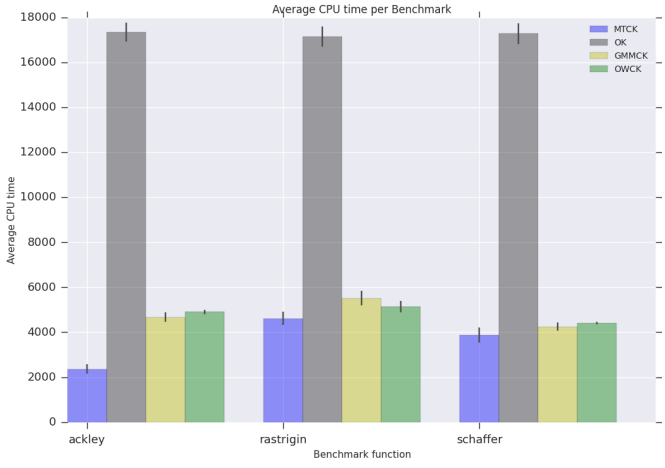
Results In Fig. 3.5 it can be observed that the Cluster Kriging based EGO variants perform very similar to OK, depending on the target function; a specific variant even outperforms Ordinary Kriging. Due to the relative large variance in the results it is difficult to judge which algorithm performs better. However, in terms of the CPU time (Fig. 3.4), it can be observed that Cluster Kriging and in particular MTCK takes only a fraction of the time that Ordinary Kriging requires. Using a sample size of 500 points this difference is mainly due to the re-fitting of only one local model at a time. This can be seen by comparing MTCK with GMMCK and OWCK, since all three Cluster Kriging variants use the same number of local models and only MTCK uses an adaptive local model strategy. When the number of points increases to 1000 and even 5000, the difference between the three Cluster Kriging variants decreases but the difference with Ordinary Kriging becomes enormous. This shows that using EGO with Ordinary Kriging quickly becomes infeasible when the number of data points grows.

3.4 Summary

This chapter addresses three aspects of the Kriging/GPR method. Firstly, a unified view of Kriging/GPR is provided: the same formulation can be derived independently from the theory of optimal linear predictors, Bayesian statistics and the optimization in Reproducing Kernel Hilbert Spaces (RKHS). We try to link those three approaches together and give a conceptual comparison among them. Secondly, the time complexity bottleneck of Kriging is discussed in detail, where several novel methods (Cluster Kriging variants) are derived, aiming at reducing the training time and increase the model quality. Finally, the proposed Cluster Kriging method is combined with the EGO algorithm to demonstrate its usefulness in improving the existing optimization algorithm.



(a) CPU time, sample size 1000



(b) CPU time, sample size 5000

Figure 3.4: Average CPU time in seconds per benchmark function for varying sample sizes.

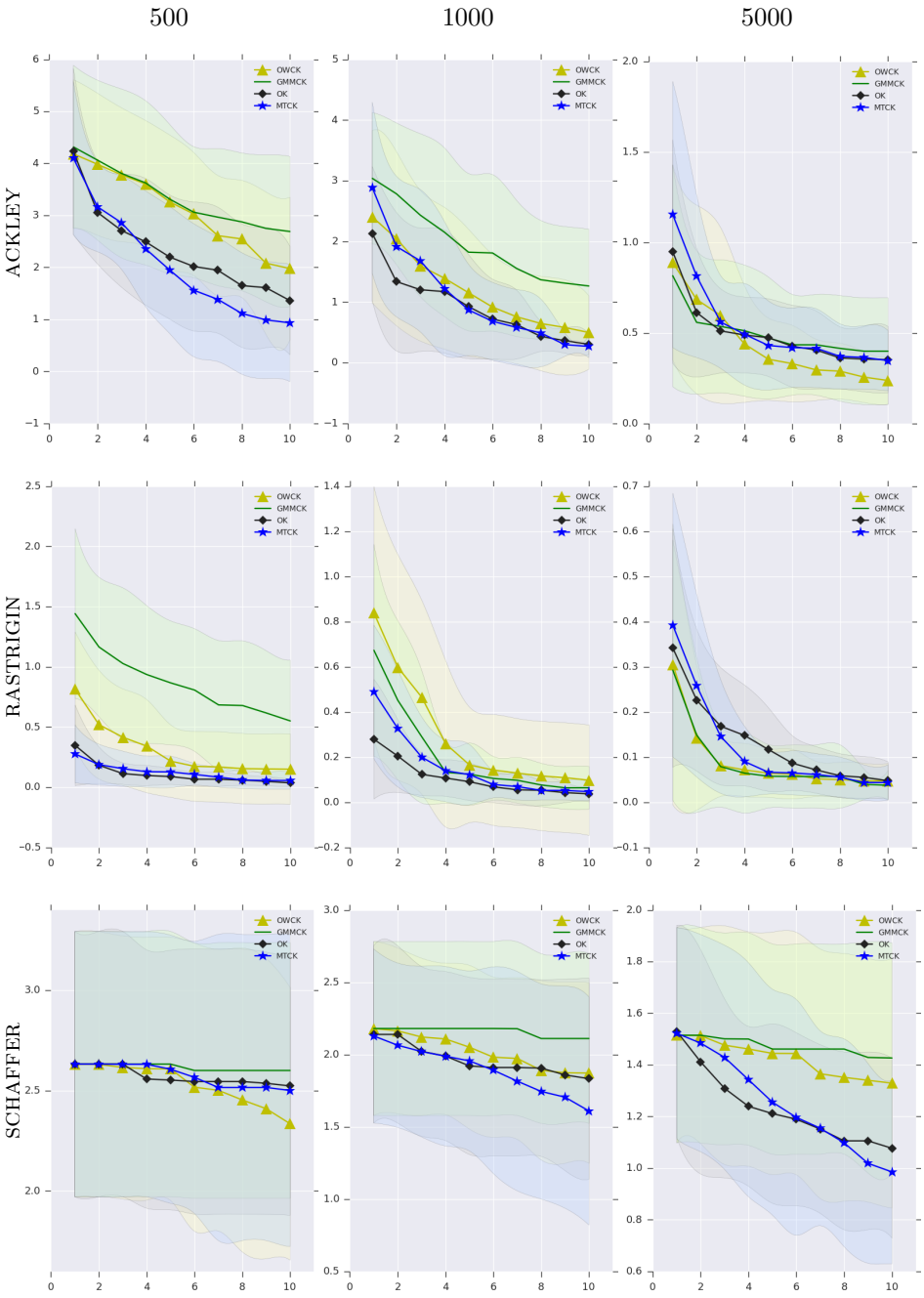


Figure 3.5: Average convergence of the absolute error of three benchmark functions in two dimensions, with varying training sample sizes n and 10 iterations of EGO. Shown is the average over 20 runs (lines) and one standard deviation (shaded areas). Legend: ◆: Ordinary Kriging, ▲: OWCK, ★: MTCK, –: GMMCK.

CHAPTER 4

Infill Criteria

When using surrogate modeling in combination with optimization techniques, it is crucial to determine how the model should be explored/exploited properly due to the fact that surrogate models give rise to errors in the prediction. Firstly, it is possible to define a "gain" function $G : \mathrm{S} \to \mathbb{R}$ to assess the unknown locations (S is again the search space), e.g., the potential *improvement* over the current best fitness value. Secondly, considering the randomness from the surrogate model $\mathscr{M}$ (usually a statistical model), the "gain" function becomes stochastic and it is necessary to use some statistical features from it, e.g., the expectation:

$$\mathscr{A}(\mathbf{x}) = \mathbb{E}\left\{G(\mathbf{x}) \mid \mathscr{M}\right\}.$$

Such a function $\mathscr{A} : \mathrm{S} \to \mathbb{R}$ is the so-called **infill criterion**. Note that in some literature, it is also called *acquisition function* (Snoek et al., 2012). The next location to evaluate is simply the maximum of the infill criteria:

$$\arg\max_{\mathbf{x}\in\mathrm{S}} \mathscr{A}(\mathbf{x}). \tag{4.1}$$

This formalism depends on two design choices: the statistical model $\mathscr{M}$ and the "gain" function G. As for the statistical model, Kriging/GPR is very commonly applied as it provides a theoretical quantification for the modeling error, through the Kriging MSE (cf. Eq. (3.11)). Some other popular models include *random forests* (Hutter et al., 2011; Bartz-Beielstein et al., 2005) and *support vector regression* (SVR) (Forrester et al., 2008). For those models, however, the theoretical prediction error is typically not available and the *empirical error* is used instead. Here, Kriging/GPR shall be assumed as the statistical model throughout all the discussions.

As for the "gain" function, one common choice is the potential improvement over the current best fitness value, achieved by evaluating an unobserved location. Maximizing the expected improvement leads to a greedy, stepwise optimization strategy. As an alternative, the so-called *cumulative regret* is considered in the multi-armed bandit: the regret in iteration t is $R_t = f(\mathbf{x}^*) - f(\mathbf{x}_t)$, where $\mathbf{x}^* = \arg\min_{\mathbf{x}\in\mathrm{S}} f(\mathbf{x})$ and $\mathbf{x}_t$ is the location chosen in iteration t. In practice, as $f(\mathbf{x}^*)$ is unknown, the regret R_t has to be estimated. It is possible to construct a non-stepwise gain by summing up all the regrets since the beginning of the optimization:

$$G = -\sum_{t=1}^{T} R_t.$$

Note that T stands for the current iteration and the minus sign is intended to convert the regret values to a gain function. For this gain function, the *Upper Confidence Bound* (UCB) criterion is proposed (Auer, 2002): given a surrogate model $\hat{f}$ (e.g., Kriging) of f and the MSE of the prediction $s^2(\mathbf{x})$, UCB is defined as:

$$\mathrm{LCB}(\mathbf{x};\beta) = \hat{f}(\mathbf{x}) - \sqrt{\beta s^2(\mathbf{x})}, \tag{4.2}$$

where β is a carefully chosen learning rate that explicitly controls the trade-off between exploitation and exploration. Note that this infill criterion is also known as *Lower Confidence Bound* (LCB) in terms of minimization. Obviously, a high value of β emphasizes more on the model uncertainty and thus tends to be more explorative. When Kriging/GPR is chosen as the statistical model, this infill criterion is called Gaussian Process Upper Confidence Bound (GP-UCB) (Srinivas et al., 2010). In addition, criteria with free parameters such as β shall be called *parameterized infill criteria* in this thesis. Other infill criteria have been proposed, based on different types of the gain function, e.g., *BayesGap* (Hoffman et al., 2014) and *UGap* (Gabillon et al., 2012), which are gap-based exploration approaches. For a survey and conceptual comparison among those infill criteria, please see Jones (2001); Forrester et al. (2008).

The infill criterion plays a vital role in many optimization paradigms, including the Efficient Global Optimization, Multi-armed Bandits, Monte-Carlo Tree Search (MCTS) and Multi-objective optimization (Emmerich, Yang, Deutz, Wang, and Fonseca, 2016). In this thesis, we shall focus on the Efficient Global Optimization algorithm, which is a sequential design strategy designed to solve expensive global optimization problems (Močkus, 1975, 2012; Jones et al., 1998).

It is important to point out that different infill criteria can give rise to conflicts on the promising location to evaluate, e.g., the probability of improvement favors low-risk locations while the expected improvement tends to find locations with a high gain (see Section 4.2). Multiple conflicting infill criteria can be considered as a multi-objective optimization problem (Bischl et al., 2014; Wang et al., 2016). Such a multi-objective treatment gives the decision makers the flexibility to choose among low-risk and/or high-gain solutions and possibly leads to parallelization of the Bayesian optimization. An alternative approach is proposed by Hutter et al. (2012), where multiple LCB criteria are created by sampling several β values from an exponential distribution with the unit mean. Furthermore, it is proposed to use portfolio strategies to select an infill criterion in each step of the Efficient Global Optimization (Hoffman et al., 2011; Ursem, 2014).

As for the optimization of the infill criteria, derivative-free Evolutionary algorithms (Bäck and Schwefel, 1993) and gradient-based method (e.g., quasi-Newton method) are often used/combined to search for the global optimum. As an alternative, Wang et al. (2018) propose to diversify the search by adapting the niching techniques to find multiple (local) optima of the acquisition function.

4.1 Improvement-based Infill Criteria

Over the last decades, much research has been put into finding a function $\mathscr{A}$ that provides a good balance between exploration and exploitation for various applications. One category of such functions, called *Improvement-based infill criteria*, is of particular interest for the following reasons: this category of functions has clear statistical meanings and those are widely applied in the field of efficient global optimization. Formally, the improvement is a function[1] defined on the stochastic process model Y of the objective function f. In terms of minimization, it is (Schonlau et al., 1998):

$$I(\mathbf{x}) = \begin{cases} f_{\min} - Y(\mathbf{x}) & \text{if } Y(\mathbf{x}) < f_{\min}, \\ 0 & \text{otherwise.} \end{cases} \tag{4.3}$$

Assume that we have observed the data set $(\mathrm{X}, \mathbf{y})$ on f and let $f_{\min} = \min\{\mathbf{y}\}$ stand for the best function value found so far. When choosing the Gaussian prior on Y, the posterior process $Y(\mathbf{x}) \mid \mathbf{y} \sim \mathcal{N}(\hat{f}(\mathbf{x}), s^2(\mathbf{x}))$ (cf. Eq. (3.29)) is taken

[1] Naturally, this function I is also a stochastic process over S.

in Eq. (4.3). In the following discussion, the posterior mean and variance shall be abbreviated as $\hat{f}$ and s^2 if there is no ambiguity on the location $\mathbf{x}$ under the consideration. The distribution of $I(\mathbf{x}) \mid \mathbf{y}$ is known as *Rectified Gaussian*[1], whose density function is written as (cf. Eq. (A.5)):

$$p_I(u;\mathbf{x}) = \begin{cases} \Phi\left(\dfrac{\hat{f} - f_{\min}}{s}\right)\delta(u) & u < f_{\min}, \\ s^{-1}(2\pi)^{-1/2}\exp\left(-\dfrac{(u-(f_{\min}-\hat{f}))^2}{2s^2}\right) & \text{otherwise.} \end{cases} \tag{4.4}$$

Here $\delta(\cdot)$ is the Dirac delta. Most of the improvement-based infill criteria are constructed to summarize the statistical properties of the improvement. A short review of the improvement-based infill criteria is given as follows.

- *Expected Improvement* (EI) is originally proposed by Močkus (1975) and utilized as the infill criterion in the standard *Efficient Global Optimization* (EGO) algorithm (Jones et al., 1998). It is defined as the first moment of the improvement:

$$\mathrm{EI}(\mathbf{x}) = \mathbb{E}\left\{I(\mathbf{x}) \mid \mathbf{y}\right\} = \left(f_{\min} - \hat{f}\right)\Phi\left(\frac{f_{\min} - \hat{f}}{s}\right) + s\phi\left(\frac{f_{\min} - \hat{f}}{s}\right). \tag{4.5}$$

  Here $\Phi(\cdot)$ and $\phi(\cdot)$ are the cumulative distribution function (c.d.f.) and probability density function (p.d.f.) of the standard normal random variable. As will be shown in the next section, the EI criterion is highly multi-modal and tries to balance between exploration and exploitation.

- *Bootstrapped Expected Improvement* (BEI) (Kleijnen et al., 2012) tries to correct the bias in EI due to the fact that the Kriging MSE derived in Eq. (3.11) is an *underestimate* of the true Kriging MSE when the hyper-parameters are estimated (den Hertog et al., 2006). Bootstrapped EI uses parametric bootstrapping to approximate the real Kriging MSE. Although BEI is shown as a more reliable alternative to EI, it also brings a large amount of computational cost.

- *Probability of Improvement* (PI) gives the probability of realizing an improvement (Žilinskas, 1992; Jones, 2001). This criterion is more biased towards exploitation than exploration since it rewards the solutions that are more

[1] Do not confuse the rectified Gaussian with the so-called truncated Gaussian distribution. See Appendix A for the clarification.

certain to yield an improvement over the current best solution, without taking the amount of the actual improvement into account:

$$\mathrm{PI}(\mathbf{x}) = \Pr(Y(\mathbf{x}) < f_{\min} \mid \mathbf{y}) = \Phi\left(\frac{f_{\min} - \hat{f}}{s}\right). \tag{4.6}$$

Loosely speaking, PI rewards low risk solutions that typically come with a relatively small amount of improvement while EI rewards solutions that give high improvement on average but could be risky to realize such an improvement. Therefore, the maximization of PI is considered as a risk minimization strategy and in contrast, the maximization of EI is considered as a gain maximization strategy. In Section 4.2, the trade-off between these two criteria is investigated by treating them as a bi-objective optimization problem.

- *Weighted Expected Improvement* (WEI) is an alternative approach to explicitly control the balance between exploration and exploitation. Consider Eq. (4.5): the first term calculates the difference between the current best $f_{\min}$ and the prediction $\hat{f}$, penalized by the probability of improvement. The second term is large when the RMSE s is large, meaning a large uncertainty about the prediction is preferred. Therefore, it is also straightforward to balance those two terms within EI (Sóbester et al., 2005):

$$\mathrm{WEI}(\mathbf{x}; w) = w\left(f_{\min} - \hat{f}\right)\Phi\left(\frac{f_{\min} - \hat{f}}{s}\right) + (1-w)s\phi\left(\frac{f_{\min} - \hat{f}}{s}\right), \tag{4.7}$$

  where $w \in [0, 1]$ is the balancing weight. Sóbester et al. (2005) argues that this additional parameter is problem-dependent.

- *Generalized Expected Improvement* (GEI) (Schonlau et al., 1998) is a generalization of the EI criterion where an additional parameter, g, is introduced to compute the gth-order moment of the improvement. The larger the value of g, the more explorative locations will be awarded by the criterion and vice versa. GEI is defined as follows:

$$\mathrm{GEI}(\mathbf{x}; g) = \mathbb{E}\left\{I(\mathbf{x})^g \mid \mathbf{y}\right\} = s^g \sum_{k=0}^{g} (-1)^k \binom{g}{k} \hat{f}^{g-k} T_k, \tag{4.8}$$

  where T_k is defined recursively for $k > 1$:

$$T_k = -u^{k-1}\phi(u) + (k-1)T_{k-2}, \tag{4.9}$$

with

$$u = \frac{f_{\min} - \hat{f}}{s}, \quad T_0 = \Phi(u), \quad T_1 = -\phi(u)$$

Note that as GEI is expressed recursively, additional computational costs are attached to it and moreover it becomes very difficult to differentiate GEI. The setting of the additional integer parameter g is entirely empirical. Sasena et al. (2002) propose a "Simulated Annealing"-like approach to decrease the value of g gradually, resulting in highly explorative behavior in the beginning of the optimization and more exploitative behavior after several iterations. The settings for g proposed are in the form of a look-up table where g starts at 20 and quickly goes down to 0 after iteration 35.

- *Multiple Generalized Expected Improvement* (MGEI) (Ponweiser et al., 2008) instantiates multiple normalized GEI criteria, using different g settings in parallel. They obtain k best local optima which are evaluated for the next iteration. The main disadvantage of this approach is the need of a large number of evaluations.

4.2 Balancing Risk and Gain

In either efficient global optimization or experimental design, a single infill criterion is maximized to generate the next promising location (solution) to evaluate. The choice of criteria has a significant impact on the performance because each criterion represents a different optimization strategy. When considering PI and EI, on the one hand, a low PI value can be viewed as risky evaluation and the maximization of PI can be seen as a *risk minimization strategy*. However, the risk minimization strategy can still lead to small improvement. On the other hand, the maximization of EI leads to high gains on average and thus it is a strategy of *expected gain maximization*. However, it does not minimize the rate of failure (no improvement is found). In Wang et al. (2016), we propose to consider the maximization of PI and EI simultaneously to find the trade-off between them, which naturally forms a *bi-objective* optimization problem. In this manner, it is possible to search for the best set of trade-off locations, namely the Pareto efficient set and candidate locations can be selected from the Pareto efficient set according to human preferences on risk/expected gain, or some pre-defined decision-making rules. Formally, the

following *vector-valued* infill criterion is considered:

$$\mathscr{A} : \mathrm{S} \to \mathbb{R}^2, \quad \mathbf{x} \mapsto (\mathrm{PI}(\mathbf{x}), \mathrm{EI}(\mathbf{x}))^\top. \tag{4.10}$$

The *set* of candidate locations is the Pareto efficient set of $\mathscr{A}$, namely:

$$\mathcal{X} = \arg\max_{\mathbf{x} \in \mathrm{S}} \mathscr{A}(\mathbf{x}). \tag{4.11}$$

Note that such a vector-valued infill criteria can be easily generalized to many objectives. Before solving Eq. (4.11) numerically, the following observations are seen in the objective space, i.e., the space formed by PI and EI values. As shown in the definition of PI and EI (Eq. (4.6) and (4.5)), both of them can be expressed completely by the Kriging predictor $\hat{f}$ and root mean squared error (RMSE) s. Therefore, both PI and EI can be considered in the space formed by $\hat{f}$ and s. Some properties of PI and EI can be inferred by investigating their behavior in such space with the benefit that such approach is independent of a specific Kriging model. The contour lines of two criteria are shown in Fig. 4.1. Without loss of generality, we put the current best function value $f_{\min}$ to zero and normalize $\hat{f}, s$ to $[-1, 1]$ and $[0, 1]$, respectively. In the figure, the maximum of PI and EI is illustrated by

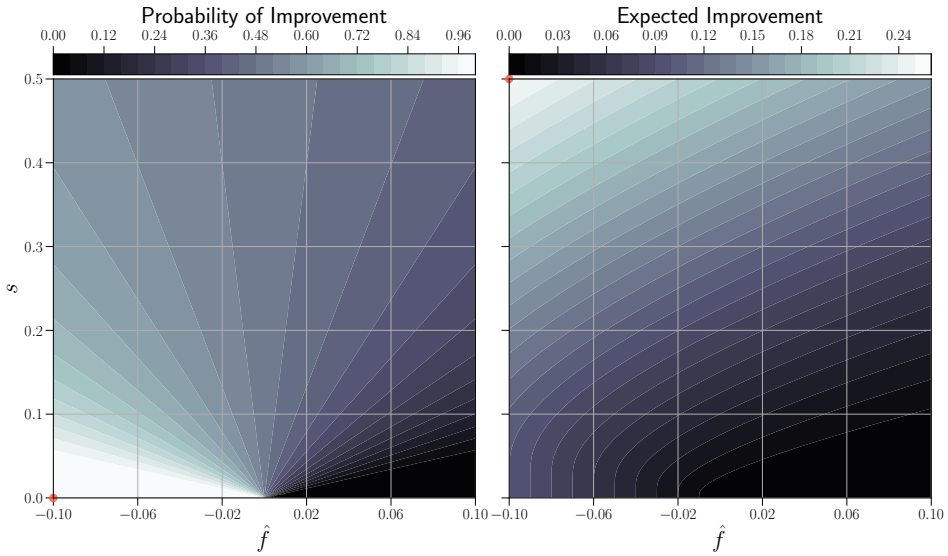


Figure 4.1: The contour lines of Probability of Improvement (Left) and Expected Improvement (Right) in the space of Kriging prediction and RMSE. The red points indicate the maximum of PI and EI, respectively.

the red point. Furthermore, $\hat{f} < 0$ indicates predictions that are smaller (better) than $f_{\min}$. In this case, PI tends to find the point with the minimal RMSE. According to the Kriging MSE expression (Eq. (3.11)), the RMSE s decreases to zero when approaching any observed location. Therefore, when better predictions exist in the Kriging model, PI prefers locations "next to" the observed ones. When Kriging predictions are worse than $f_{\min}$, PI prefers the point with the largest RMSE and thus is of high risk. In contrast, EI always increases with increasing

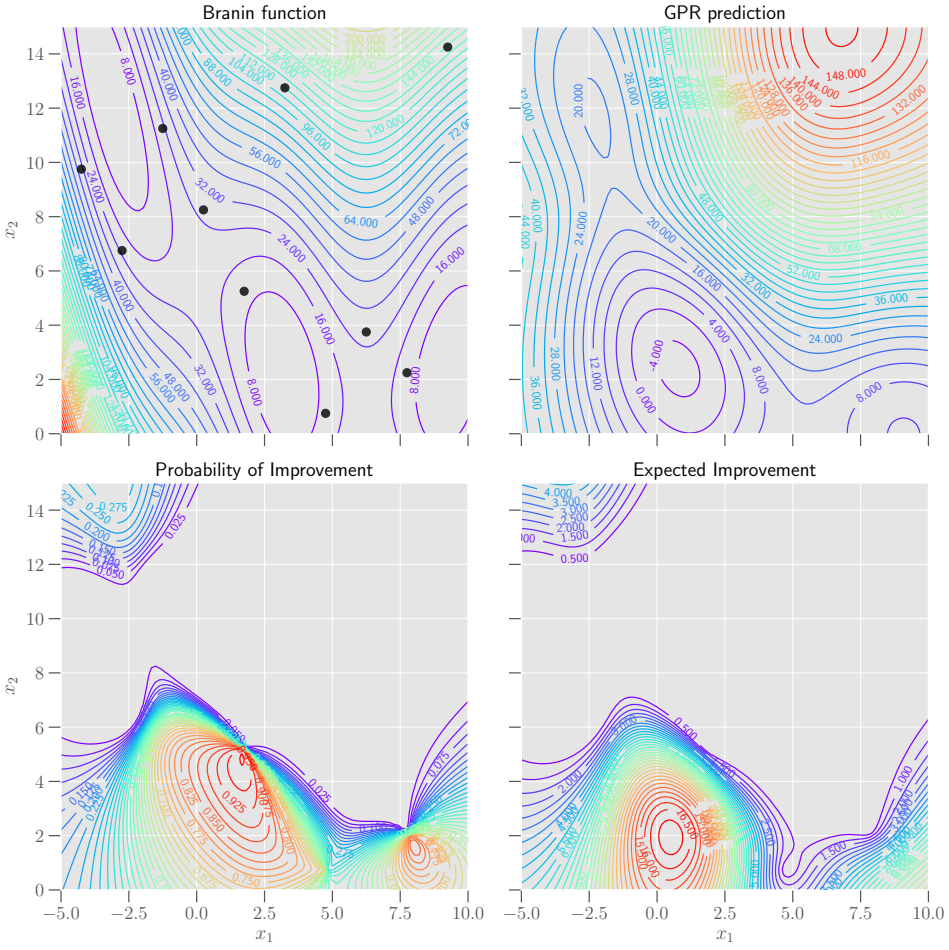


Figure 4.2: On the *Branin function* (**top-left**), 10 data points (black dots) are generated uniformly, on which an *Ordinary Kriging* model (with Matérn 3/2 kernel) is trained. Contour lines of the prediction (**top-right**), Expected Improvement (**bottom-left**) and Probability of Improvement (**bottom-right**) are depicted.

RMSE and decreases with increasing Kriging predictions. Note that, although it seems possible to compute the Pareto front directly from Fig. 4.1 (by looking for the points that satisfy the Karush-Kuhn-Tucker (KKT) conditions based on the contour line of PI and EI), the resulting Pareto front is not feasible due to the fact that not every point in the objective space is *attainable*: depending on the underlying Kriging/GPR model, the pre-image of a point $(\hat{f}, s) \in [-1, 1] \times [0, 1]$ does not necessarily exist in the search/decision space S.

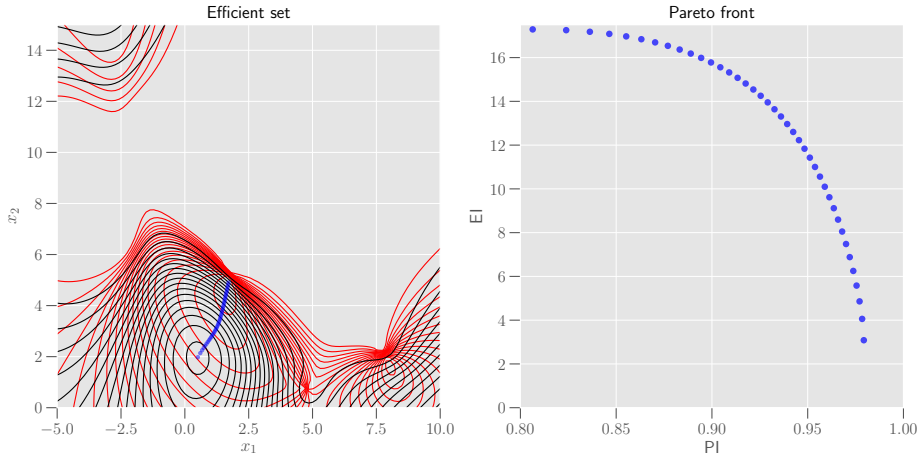


Figure 4.3: Pareto approximation sets of the bi-objective infill criterion formed by the PI and EI landscape (Fig. 4.2). The results are obtained from the HIGA-MO with population size 20. **Left**: Pareto efficient sets (blue dots) between PI (red contours) and EI (black contours). **Right**: The corresponding Pareto fronts.

In Wang et al. (2016), it is proposed to solve this problem numerically by a gradient-based multi-objective optimization algorithm, called *Hypervolume Indicator Gradient Ascent Multi-objective Optimization* (HIGA-MO) (Wang et al., 2017). The gradients of PI and EI are required for this algorithm, which are given as follows. By introducing the auxiliary variable $u = (f_{\min} - \hat{f})/s$, the gradients are:

$$\frac{\partial \,\mathrm{PI}}{\partial \mathbf{x}} = -\frac{\phi(u)}{s}\left(\frac{\partial \hat{f}}{\partial \mathbf{x}} + u\frac{\partial s}{\partial \mathbf{x}}\right), \quad \frac{\partial \,\mathrm{EI}}{\partial \mathbf{x}} = \phi(u)\frac{\partial s}{\partial \mathbf{x}} - \Phi(u)\frac{\partial \hat{f}}{\partial \mathbf{x}}.$$

By plugging those gradients into HIGA-MO (Alg. 10 in Section 5.2), the Pareto front/efficient set of $\mathscr{A}$ can be obtained. We illustrate the result of the bi-objective optimization on the well-known 2-D *Branin* function. Here 10 design locations are generated and evaluated on the Branin function using Latin Hypercube Sampling

and an Ordinary Kriging is built on those locations. The landscapes of PI and EI are shown in Fig. 4.2: the global structures of the landscapes are quite similar while the local landscape differs subtly. The following setting of HIGA-MO is used for this problem: the population size is set to 40 and the step-size is set to 0.004 multiplied by the maximal range of the search space. HIGA-MO is terminated after finishing 2000 generations. The results from the gradient-based algorithm are shown in Fig. 4.3, where the efficient set and the Pareto front are indicated by the blue points.

4.3 Moment-Generating Function of Improvement

Following the intuition on using the higher moments of the improvement, an novel infill criterion based on the *Moment-Generating Function* (MGF) of the improvement is introduced in (Wang et al., 2017), where all the moments are combined. Loosely speaking, given the existence of the moment-generating function, it can be expanded as a Taylor series, whose terms are proportional to all the moments (to the infinite order) of the improvement. Therefore, such a function is considered as combination of all the moments. Formally, the MGF of the improvement $I(\mathbf{x})$ is an alternative way to give its probability distribution and it is defined as:

$$\forall t \in \mathbb{R}, \quad M(\mathbf{x}, t) := \mathbb{E} \exp\left(tI(\mathbf{x})\right) = \int_{-\infty}^{\infty} e^{tu} p_I(u; \mathbf{x}) \, \mathrm{d}u.$$

Moreover, the moment-generating function can be calculated using the density function of $I(\mathbf{x})$:

$$\begin{aligned} M(\mathbf{x}, t) = 1 + \Phi\left(\frac{f_{\min} - \hat{f}'}{s}\right) \exp\left(\left(f_{\min} - \hat{f}\right) t + \frac{s^2 t^2}{2}\right) - \Phi\left(\frac{f_{\min} - \hat{f}}{s}\right), \\ \hat{f}' = \hat{f} - s^2 t. \end{aligned} \tag{4.12}$$

This function has a closed form and is well-defined for all $t \in \mathbb{R}$. From a different perspective, the Taylor expansion of the MGF is:

$$M(\mathbf{x}, t) = 1 + t\mathbb{E}I(\mathbf{x}) + \frac{t^2}{2!}\mathbb{E}I^2(\mathbf{x}) + \frac{t^3}{3!}\mathbb{E}I^3(\mathbf{x}) + \cdots = \sum_{n=0}^{\infty} \frac{t^n}{n!}\mathbb{E}I^n(\mathbf{x}). \tag{4.13}$$

Note that, for an arbitrary distribution, the above series might not converge for all the $t \in \mathbb{R}$, even if all the moments exist. When treating $t^n/n!$ as the weight for

each moment $\mathbb{E}I^n$, this function can also be considered as a linear combination of the moments, where the weights are controlled by variable t. In addition, it is possible to normalize the weights by observing the fact that: $\sum_{n=0}^{\infty} \frac{t^n}{n!} = e^t$, which converges for all $t \in \mathbb{R}$. Thus, the normalized MFG function[1] is obtained by dividing it by e^t. The additional parameter t controls the trade-off between exploration and exploitation of the search. To visualize this, multiple sets of weights are plotted in Fig. 4.4 by varying t. According to the figure, a low value of t (e.g., $t < 1$) assigns more weights to the lower moments (e.g., the expected improvement), rendering the search process mainly exploitative. As for the higher values of t, the "center" (mean) of the weight distribution is t and dispersion (variance) is also increasing with respect to t. This indicates that more higher moments of the improvement are taken into account when t increases and therefore the search tends to be more explorative.

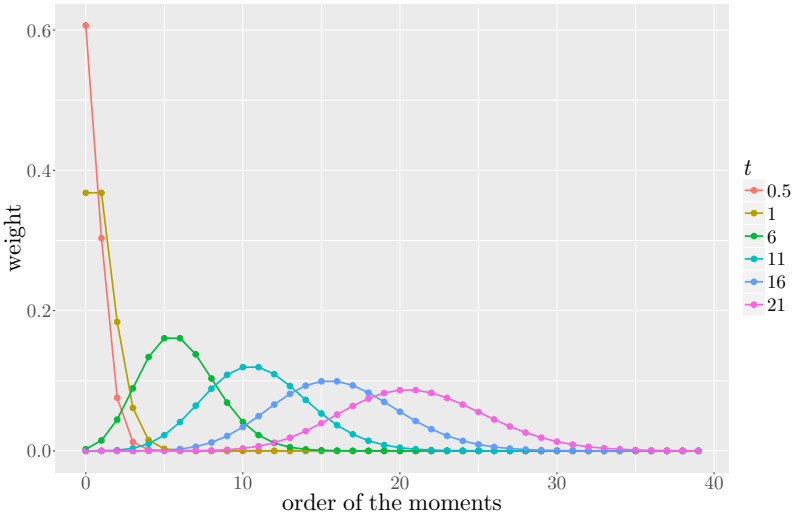


Figure 4.4: Distribution of the combination weights in the normalized moment-generating function by varying the t value from 0.5 to 21.

Finally, it is proposed to incorporate the Probability of Improvement (PI) in the proposed acquisition function. This is achieved by treating PI as the "zero-order" moment of $I(\mathbf{x})$ and replacing the constant 1 in Eq. (4.13). Putting all

[1] In fact, the normalized weight, $t^n/e^t n!$ is exactly the probability mass function of the Poisson distribution.

the considerations together, the proposed **Moment-Generating Function of Improvement** $\mathcal{M}$ (MGFI) is defined as:

$$\begin{aligned}\mathcal{M}(\mathbf{x};t) &= \frac{M(\mathbf{x},t) - 1 + \mathrm{PI}(\mathbf{x})}{e^t} \\ &= \mathrm{PI}(\mathbf{x}) + \frac{t}{e^t}\mathbb{E}I(\mathbf{x}) + \frac{t^2}{2!e^t}\mathbb{E}I^2(\mathbf{x}) + \frac{t^3}{3!e^t}\mathbb{E}I^3(\mathbf{x}) + \cdots \\ &= \Phi\left(\frac{f_{\min} - \hat{f}'}{s}\right)\exp\left(\left(f_{\min} - \hat{f} - 1\right)t + \frac{s^2t^2}{2}\right) \end{aligned} \tag{4.14}$$

where $\hat{f}'$ is defined in Eq. (4.12). In order to align with existing work (Wang et al., 2016) on using gradient-based optimization techniques for infill criteria, the gradient of MGFI is given as well:

$$\begin{aligned}\frac{\partial\mathcal{M}(\mathbf{x};t)}{\partial\mathbf{x}} = C\Bigg[&\Phi\left(\frac{f_{\min} - \hat{f}'}{s}\right)\left(t^2 s\frac{\partial s}{\partial\mathbf{x}} - t\frac{\partial\hat{f}}{\partial\mathbf{x}}\right) \\ &- \phi\left(\frac{f_{\min} - \hat{f}'}{s}\right)\left(\frac{1}{s}\frac{\partial\hat{f}'}{\partial\mathbf{x}} + \frac{f_{\min} - \hat{f}'}{s^2}\frac{\partial s}{\partial\mathbf{x}}\right)\Bigg],\end{aligned}$$

where

$$\frac{\partial\hat{f}'}{\partial\mathbf{x}} = \frac{\partial\hat{f}}{\partial\mathbf{x}} - 2ts\frac{\partial s}{\partial\mathbf{x}}, \quad C = \exp\left(\left(f_{\min} - \hat{f} - 1\right)t + \frac{s^2t^2}{2}\right).$$

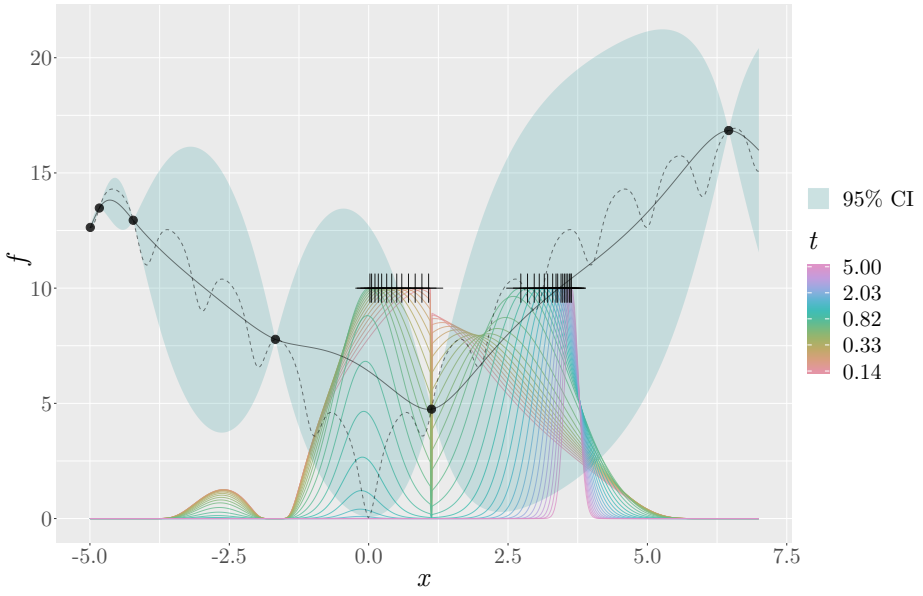
The gradients $\partial\hat{f}/\partial\mathbf{x}$ and $\partial s/\partial\mathbf{x}$ are expressed in Eq. (3.34) and (3.35).

Comparison to GEI As with the Generalized Expected Improvement, MGFI is designed to exploit the higher moments of the improvement. Compared to GEI, the main advantages of $\mathcal{M}$ are:

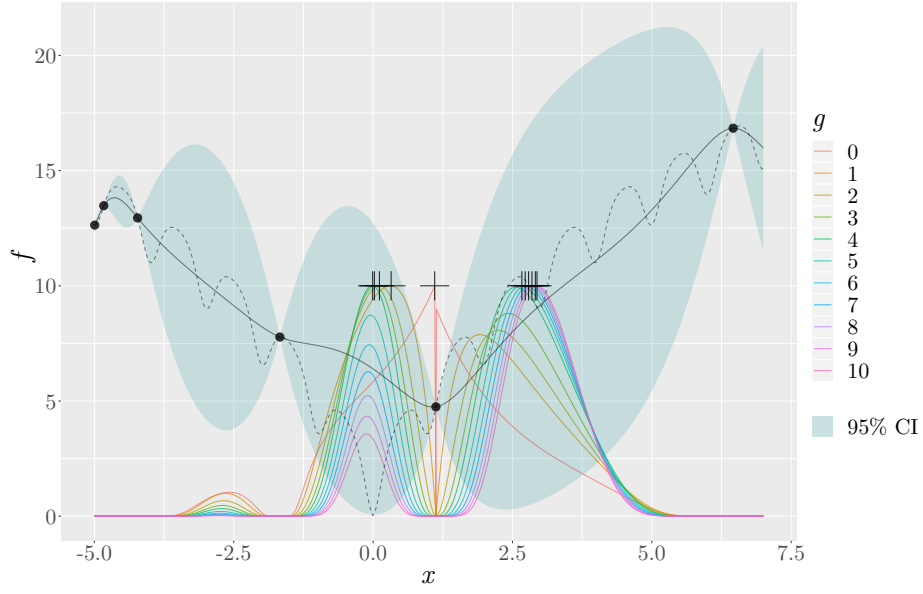
1. $\mathcal{M}$ combines all the moments using a weight distribution instead of using one moment each time in GEI. This leads to a much smaller "change" in the acquisition function when tuning the addition parameter t.

2. It has a simple closed-form expression, in contrast to a recursive formula (Eq. (4.9)) for GEI. As a result, it is obvious to see that $\mathcal{M}$ is computationally less expensive than GEI. The gradient of $\mathcal{M}$ can also be easily calculated.

3. The extra parameter t that balances the exploration and exploitation, takes continuous values while the parameter g in GEI is an integer variable. Consequently, when tuning this additional parameter, the effect can be more smooth: cooling schedules such as, e.g., the exponential decay, can be applied.



(a) MGF of Improvement (MGFI)



(b) Generalized Expected Improvement (GEI)

Figure 4.5: On the 1-D *Ackley function* (dashed curve), a GPR model (solid curve) is built on the black dots. MGFI and GEI (both normalized and rescaled to $[0, 10]$) are plotted by varying parameters t and g, whose maxima are indicated by black crosses. The shaded area shows the 95% confidence interval of the prediction.

The difference between MGFI and GEI is illustrated in Fig. 4.5. On the 1-D Ackley function, a GPR model is built on 6 uniformly distributed samples (black dots) in $[-5, 7]$. In addition, 11 MFGI functions are created using a log-scaled t value from roughly 0.1 to 3 while 11 GEI functions are depicted with g from 0 to 10. The maximum of the infill criteria (the black crosses) is evaluated in the next iteration. Comparing the spread of those maximum points, it is obvious that when g increases in GEI, those maxima of infill criteria are getting close to each other and thus become indistinguishable. This means the GEI functions are, in fact, indifferent when the parameter g increases. However, for MGFI, those maxima still show significant differences when t increases. Therefore, the *effective range* of the parameter g is much narrower than that of t.

4.4 Cooling Strategies for MGFI

As discussed in the last section, the functionality of parameter t is analogous to that of temperature in simulated annealing (Nourani and Andresen, 1998). Consequently it is straightforward to use the temperature cooling strategies to improve the optimization procedure. In Wang et al. (2018), we propose to adopt two most commonly applied cooling strategies in simulated annealing, namely (Nourani and Andresen, 1998):

- Exponential strategy: $t_{i+1} = \alpha t_i, \quad 0 < \alpha < 1.$
- Linear strategy: $t_{i+1} = t_i - \eta, \quad \eta > 0.$

In each cooling strategy, naturally, there are two parameters to set: the initial temperature t_0 and the cooling speed (α for exponential strategy and η for the linear one). As shown in the next subsection, the initial temperature t_0 is a free parameter, whose setting should be highly problem-dependent. As for the cooling speed, it should be determined with respect to the prescribed function evaluation budget. For example, a fast cooling speed for a short run length can be hazardous because the search will quickly become very exploitative and even ends up with stagnation. However, instead of setting the cooling speed directly, the temperature t_f at the final iteration of the algorithm[1] is of interest here due to the fact that the moment whose order is the closest to the current temperature

[1]Note that the algorithm might terminate before consuming all the budget if other termination criteria are implemented and satisfied.

has the biggest weight in MGFI (see Fig. 4.4). Therefore, t_f determines the major functioning order of moment in the final stage of the search. Let $N_{\max}$ be the maximal number of iterations. The cooling speed parameters are determined as follows: $\alpha = (t_f/t_0)^{1/N_{\max}}, \eta = (t_0 - t_f)/N_{\max}$.

4.4.1 Impact of Temperature Configurations

Intuitively, the optimal settings for the initial and final temperature t_0, t_f should depend on the specific problem. An experiment is performed to investigate how large the impact of temperature configurations can be on test problems. Here 10 different temperature configurations (shown in Tab. 4.1) are generated from the Latin hypercube sampling with design ranges: $t_0 \in [1, 5]$ and $t_f \in [10^{-3}, 0.5]$.

Table 4.1: Latin hypercube design of the temperature configuration.

No.	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
t_0	1.084	1.273	1.495	1.756	2.063	2.423	2.847	3.344	3.928	4.613
t_f	0.366	0.197	0.005	0.009	0.016	0.001	0.031	0.106	0.057	0.003

The following experiment is carried out on noiseless functions of the BBOB benchmark (Hansen et al., 2010, 2009) for the *exponential cooling strategy*. We only select the 10 multi-modal functions $f15$-$f24$ from BBOB. The unimodal functions are skipped because efficient global optimization is designed for multi-modal functions and using a high temperature (high explorative effect) usually leads to inefficient convergence on a unimodal one. In addition, 30 independent runs are conducted on each test function. The function evaluation budget is set to $50 \times D$ and the size of the initial LHS design is fixed to $10 \times D$. For $D = 2$, the performance of each temperature configuration is reported in Fig. 4.6, where *empirical cumulative distribution functions of the running length (ECDFs)* are shown. ECDF measures the success probability (if the algorithm reaches a given target fitness $f^* + \Delta f$ in one run) as a function of maximal number of function evaluations allowed. On some functions, ECDFs from all the temperature configurations differ largely, e.g., $f19$ and $f23$. However, on $f17$, ECDFs are quite similar to each other, implying the temperature configuration has relatively small impact for $f17$. Moreover, by checking the winning configuration on each function, it is obvious that there is no global winner for this multi-modal function class. This

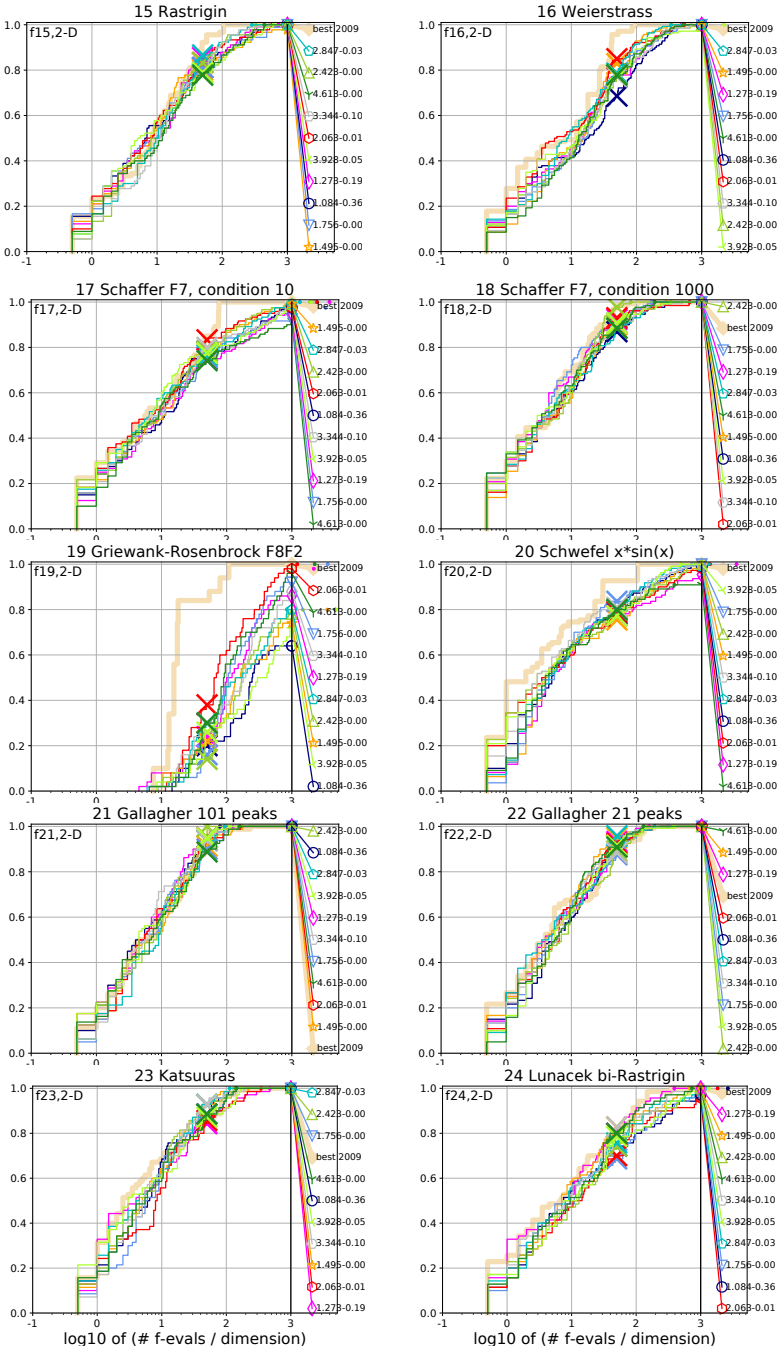


Figure 4.6: Bootstrapped empirical cumulative distribution of the number of function evaluations divided by dimension for all functions in 2-D. The targets are chosen from $10^{[-8..2]}$ such that the bestGECCO2009 artificial algorithm just did not reach them within a given budget of $k \times$ DIM, with $k \in \{0.5, 1.2, 3, 10, 50\}$.

observation suggests that the temperature settings are highly problem-dependent and should indeed be configured in practice.

4.4.2 Benchmarking the Cooling Strategies

For the experiments, only the multi-modal test functions $f15 - f24$ are picked for the same reason. Both exponential and linear cooling strategies are tested. Note that the temperature configuration is not investigated here due to the high computational time it takes. Instead, the following setting is used: $t_0 = 2, t_f = 0.1$. A relatively small function budget is chosen because it is the scenario on which Bayesian optimization should perform well: the size of the initial design is set to $10 \times D$ and the totally function evaluation is limited to $50 \times D$. The maximal function evaluation is set as the only termination criterion in the experiment. Therefore, $40 \times D$ iterations are executed in each run. For each test function, 30 instances are created for independent runs. For the settings of the Gaussian process, the Matérn 3/2 kernel is used throughout the experiment. The model is fitted via the maximum likelihood method, which is solved by the Limited-memory BFGS (L-BFGS) algorithm with the restarting heuristic. In addition, a small number of function evaluations $8 + \lfloor 40 \log D \rfloor$ is set for L-BFGS because the computational overhead on the likelihood function explodes quickly as the number of evaluations goes up.

On $D = 10$, the benchmark results are shown in Fig. 4.7. In the plot, "MGFI" indicates the application of MGFI criterion with a constant temperature setting of 1 (no cooling strategy in this case). "MGFI-exp" stands for MGFI with exponential cooling strategy while "MGFI-linear" is for the linear cooling. Note that all the infill-criteria fail completely on $f19$, indicating many more function evaluations are needed to solve this function. Using a linear or exponential cooling strategy, MGFI outperforms the commonly applied expected improvement criterion on seven functions out of $f15 - f24$. This comparison suggests that MGFI with a cooling strategy is preferable for BBOB multi-modal functions. In addition, there is no clear winner between the linear and exponential cooling strategies. However, on function $f21$, EI wins the competition. This situation might be caused by a very poor setting of the cooling temperature and requires further investigation.

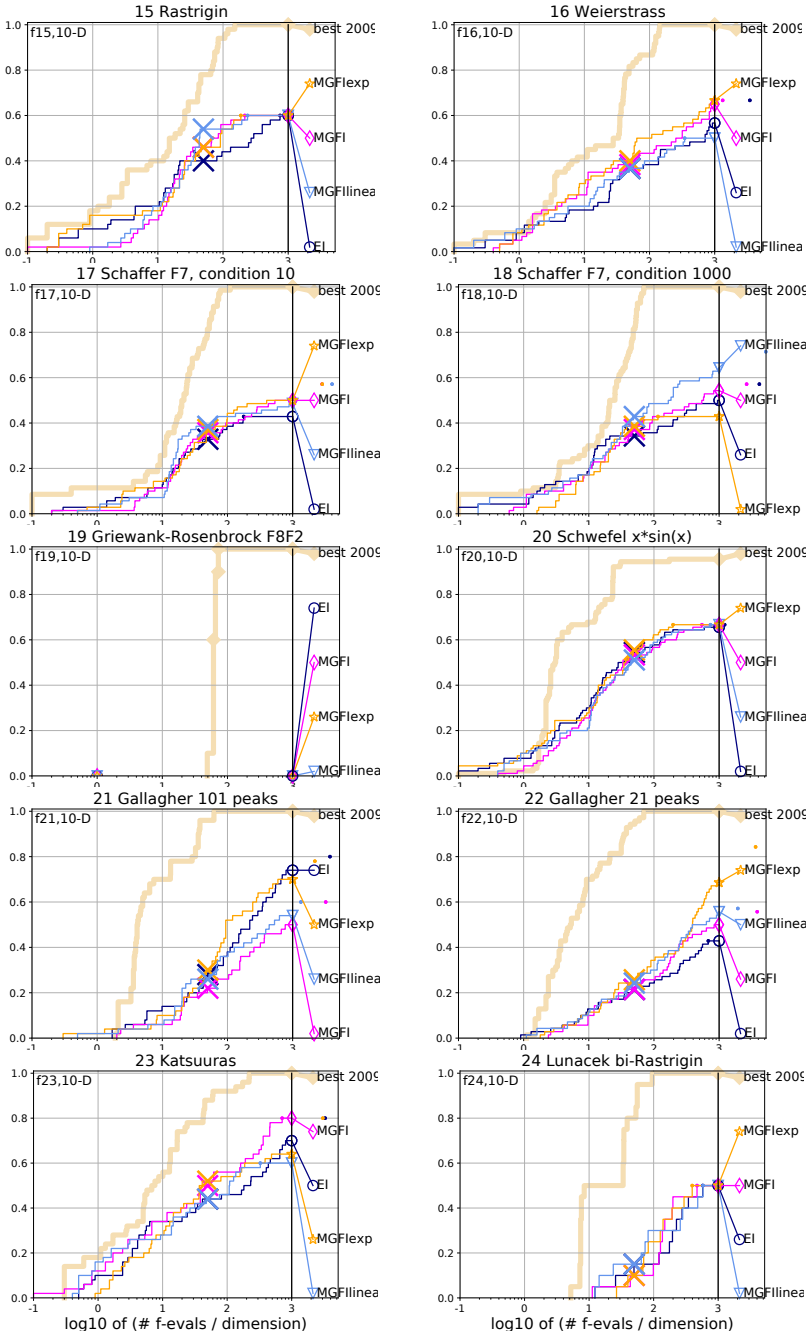


Figure 4.7: Bootstrapped empirical cumulative distribution of the number of function evaluations divided by dimension for all functions in 10-D. The targets are chosen from $10^{[-8..2]}$ such that the bestGECCO2009 artificial algorithm just did not reach them within a given budget of $k \times$ DIM, with $k \in \{0.5, 1.2, 3, 10, 50\}$.

4.5 Parallelization

In some applications, the target function f to optimize is actually represented by a time-consuming simulator run. Multiple simulators can be distributed over many CPUs or a grid of machines, allowing for the parallelization of function evaluations. In order to take advantage of this parallelism, several promising candidate locations are needed from infill criteria $\mathscr{A}$, in addition to the maximum of $\mathscr{A}$. For instance, naively, this can be done by randomly sampling q points ($q > 1$) in addition to $\mathscr{A}$'s maximum (Hutter et al., 2010). However, this method might be *inefficient* because many random samples would never be chosen for the evaluation if a sequential q-step maximization of infill criteria were performed. Although parallelization methods for infill criteria have been discussed extensively in the literature, there is a lack of a clear definition of the problem itself. Here the following formulation is introduced. Rigorously, considering a GP prior Y on f and the initial data set $(\mathrm{X}, \mathbf{y})$, each step in the sequential maximization of infill criteria is:

$$\mathbf{x}'_i = \arg\max_{\mathbf{x}\in\mathrm{S}} \mathscr{A}\left(\mathbf{x} \mid \mathbf{y}, \left\{ f(\mathbf{x}'_k) \mid \mathbf{x}'_k = \arg\max_{\mathbf{x}\in\mathrm{S}} \mathscr{A}\left(\mathbf{x} \mid \mathbf{y}, \{f(\mathbf{x}'_n) \mid \ldots\}_{n=1}^{k-1}\right)\right\}_{k=1}^{i-1}\right).$$

Note that the infill criterion is denoted as $\mathscr{A}(\mathbf{x} \mid \mathbf{y}, \{f(\mathbf{x}'_k)\}_k)$, to emphasize its dependence (from the underlying GP posterior) on evaluations $\{f(\mathbf{x}'_k)\}_k$ from all previous steps as well as the initial observations $\mathbf{y}$. In addition, the expression above is written in a recursive manner on purpose to show that it is not possible to decompose the maximization of infill criteria at a certain step from all previous function evaluations. Here, the goal of *infill criteria parallelization* is formulated as to obtain q points $\{\mathbf{x}_i\}_{i=1}^q \subset \mathrm{S}$ from an infill criterion $\mathscr{A}$ without any function evaluation, such that when comparing to points $\{\mathbf{x}'_i\}_{i=1}^q$ from the sequential q-step maximization of $\mathscr{A}$, the following condition holds:

$$\begin{aligned}\mathbb{E}\left\{\min\left(Y(\mathbf{x}_1), \ldots, Y(\mathbf{x}_q)\right) \mid \mathbf{y}\right\} \approx \min \Big\{ & \mathbb{E}\left\{Y(\mathbf{x}'_1) \mid \mathbf{y}\right\}, \\ & \mathbb{E}\left\{Y(\mathbf{x}'_2) \mid \mathbf{y}, f(\mathbf{x}'_1)\right\}, \\ & \ldots, \\ & \mathbb{E}\left\{Y(\mathbf{x}'_q) \mid \mathbf{y}, \{f(\mathbf{x}'_i)\}_{i=1}^{q-1}\right\} \Big\}.\end{aligned}$$

This condition regulates that the expectation of the best fitness from q locations is roughly the same as the minimal expected fitness value obtained from sequential q-step maximization of $\mathscr{A}$, or shortly the parallelization exhibits the same effect as

the sequential approach on average. Note that, although this condition gives a clear target when designing the parallelization method for infill criteria, it is difficult to validate parallelization methods on it. Thus, in the following, the approaches shall be discussed regardless of this condition.

4.5.1 Multi-point Infill Criteria

Multi-point Expected Improvement (q-EI) is proposed by Schonlau (1998) and computes the expectation of the smallest improvement among a set of correlated locations:

$$\mathrm{EI}^q = \mathbb{E}\left\{ f_{\min} - \min(Y(\mathbf{x}_1), Y(\mathbf{x}_2), \ldots, Y(\mathbf{x}_q)) \mid \mathbf{y} \right\}. \tag{4.15}$$

To give the exact formula of q-EI, it requires to integrate the smallest order statistic from q-correlated Gaussian random variables. The exact formula of 2-EI is derived in Ginsbourger et al. (2010). For an arbitrary number of points, the formula is given in Chevalier and Ginsbourger (2013). There are two heuristics, *Kriging Believer* and *Constant Liar*, proposed to approximate q-EI with fewer computations (Ginsbourger et al., 2010). In Kriging Believer, q points are obtained via the sequential q-step maximization of $\mathscr{A}$, where the real evaluation $f(\mathbf{x})$ is replaced by the Kriging prediction. In Constant Liar, a pre-defined fixed value is used for $f(\mathbf{x})$ and thus is called a "lie".

4.5.2 Multi-instance of Infill Criteria

Using the additional parameter β in LCB (Eq. (4.2)), Hutter et al. (2012) propose an alternative parallelization method, where q different β-values are sampled from the log-normal distribution $\mathit{Lognormal}(0, 1)$ and subsequently q different LCB criteria are instantiated using β samples:

$$\beta_i \sim \mathit{Lognormal}(0,1), \quad \mathbf{x}_i = \arg\min_{\mathbf{x}\in\mathrm{S}} \mathrm{LCB}(\mathbf{x}; \beta_i), \quad i = 1, 2, \ldots, q. \tag{4.16}$$

Compared to q-EI, this method brings no additional computational cost and it serves as reasonable between exploration and exploitation. On one hand, as the probability mass of the standard log-normal distribution concentrates around small values, most of the β samples will be relatively small and the corresponding LCB criteria are of low risk. On the other hand, the standard log-normal also possesses a long tail, meaning that it is possible to obtain a few large β samples with a small

probability. Note that such a trade-off is controlled by the mean and standard deviation of the log-normal distribution. At the time of writing, to the best of our knowledge, there is no theoretical work on investigating the impact of those parameters in the log-normal distribution, or a proof to show that the log-normal distribution is the optimal probability law for this purpose, in the first place. Regardless of this theoretical concern, this method can be easily applied to other *parameterized* infill criteria, e.g., MGFI and GEI, although its performance needs to be tested systematically.

4.5.3 Multi-objective Infill Criteria

The multi-objective treatment of the infill criteria (Section 4.2) also allows for the parallelization. As a natural extension to the bi-objective formulation in Eq. (4.10), the general vector-valued infill criteria is considered:

$$\boldsymbol{\mathscr{A}} : \mathrm{S} \to \mathbb{R}^m, \quad \mathbf{x} \mapsto \left(\mathscr{A}_1(\mathbf{x}), \mathscr{A}_2(\mathbf{x}), \ldots, \mathscr{A}_m(\mathbf{x})\right)^\top,$$

where $\mathscr{A}_1, \mathscr{A}_2, \ldots, \mathscr{A}_m : \mathrm{S} \to \mathbb{R}$ are real-valued infill criteria (subject to maximization) that are either selected from some well-defined infill criteria or generated from a parameterized one, e.g., using Eq. (4.16). To select q points based on this problem, for instance, we could adopt the so-called *decomposition-based multi-objective optimization* (Zhang and Li, 2007), in which q different linear scalarizations on infill criteria are generated uniformly and q points are obtained as the maximum of the linear scalarization, namely

$$\mathbf{x}_k = \underset{\mathbf{x} \in \mathrm{S}}{\arg\max} \sum_{i=1}^{m} w_i \mathscr{A}_i(\mathbf{x}), \quad w_1, w_2, \ldots, w_m \sim \mathcal{U}(0,1), \quad k = 1, \ldots, q.$$

Note that $\mathcal{U}(0,1)$ stands for the uniform distribution over $[0,1]$. Although this multi-objective proposal seems plausible, it remains untested as of the time of writing.

4.5.4 Niching-based Infill Criteria Maximization

Considering the landscape of EI, this approach is motivated by the observation that its landscape is usually highly multi-modal (Jones et al., 1998; Wang et al., 2018) (also see Fig. 4.2). The original EGO algorithm aims at finding the global

optimum of the EI landscape, which is another global optimization task and therefore is difficult to solve (the exhaustive branch-and-bound method has been proposed initially). Suppose $\mathbf{x}^*$ is the global maximum of EI and $\mathbf{x}'$ is a *local optimum*. After incorporating the fitness $f(\mathbf{x}^*)$, the Kriging formula are updated such that the Kriging MSE are largely reduced at $\mathbf{x}^*$ and a neighborhood around it. Consequently, the EI value drops locally in this neighborhood due the fact that EI decreases with deceasing Kriging MSE. As this is only a local change of the EI landscape, EI values around $\mathbf{x}'$ are normally not affected. Then, $\mathbf{x}'$ would possibly become new global maximum if EI is maximized again. Instead of using the sequential maximization of EI to "expose" local maxima, it is proposed to combine EGO with a so-called *niching evolution strategy* (Shir and Bäck, 2005b), where the niching method is intended to find multiple distinct local maxima simultaneously. In the evolutionary computation, niching refers to a collection of methods that aims at locating multiple distinct local optima, in order to improve the exploration on highly multi-modal function. In this manner, within one iteration, it is possible to locate the global maximum of EI as well as some local maxima, which would be explored using a few iterations, if the sequential maximization were performed. The resulting parallelization method is called **Niching-q-EI** (Wang et al., 2018). Interestingly, from the niching perspective, niching-q-EI can also be considered as a meta-model assisted niching algorithm where the niche formation is performed on the EI landscape instead of the real objective function. The niching-q-EI algorithm is summarized in Alg. 8.

Niching Evolution Strategies Herein the *niching evolution strategy* is briefly introduced. In general, Evolutionary Algorithms (EAs) have the tendency to converge quickly into a single solution in the search space. However, in many problem solving scenarios (e.g., global optimization), locating and maintaining multiple solutions/optima is required. Niching is developed to achieve this goal by forming sub-populations in order to maintain the population diversity of EAs. The most successful techniques are *fitness sharing* (Goldberg and Richardson, 1987) and *Crowding* (De Jong, 1975).

Although the niching technique is initially proposed mainly for Genetic Algorithms (GAs), it has also been introduced to classic $(1 \dot{+} \lambda)$ self-adaptive Evolution Strategies by Shir and Bäck (2005b). Later, it is further developed for the derandomized ES (Shir and Bäck, 2005a) including the well-known Covariance Matrix Adaptation

Algorithm 8 Niching-based Efficient Global Optimization

procedure NICHING-q-EI(q, S, f)
  Given: the number of points q for parallelization
  Sample the initial design $\mathrm{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\} \subset \mathrm{S}$
  Evaluate $\mathbf{y} \leftarrow (f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_n))^\top$
  Construct the Kriging/GPR model $\hat{f}$ on $\mathrm{X}, \mathbf{y}$.
  while the stop criteria are not fulfilled **do**
    $\{\mathbf{x}'_1, \ldots, \mathbf{x}'_q\} \leftarrow (1 \dot{+} \lambda)$-NICHING-ES($q, \mathrm{S}, \mathrm{EI}(\mathbf{x}; \hat{f})$) ▷ Alg. 9
    Parallel evaluation $y_1, \ldots, y_q \leftarrow f(\mathbf{x}'_1), \ldots, f(\mathbf{x}'_q)$
    $\mathrm{X} \leftarrow \mathrm{X} \cup \{\mathbf{x}'_1, \ldots, \mathbf{x}'_q\}$
    $\mathbf{y} \leftarrow (\mathbf{y}^\top, y_1, \ldots, y_q)^\top$
    Re-construct the Gaussian process model $\hat{f}$ on $\mathrm{X}, \mathbf{y}$
  end while
end procedure

Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001) and finally evolved into a self-adaptive approach which allows for the niche radius and the niche shape adaptation (Shir et al., 2007). The $(1 \dot{+} \lambda)$-niching evolution strategy with fixed niche radius and spherical niche shape (Alg. 9) is chosen for our purpose and it works as follows: given q optima expected to investigate, the niching procedure initializes $q + p$ so-called "D-sets" (Shir et al., 2008) which are evolution strategy kernels containing all the adapted strategy parameters (step-size, covariance matrix) as well as decision parameters (current search point/solution, mutation vectors). Each D-set defines the current search point and all the internal information regarding an evolution strategy in a given time during the evolution.

The q D-sets are meant for identifying q possible local optima/peaks while p D-sets are for a "non-peak" domain, which are randomly regenerated every κ generations. The purpose of p "non-peak" D-sets is to explore the search space so that new niches would emerge and the probability of finding undiscovered optima is increased. The niching procedure then proceeds to generate λ offspring for each D-set. The population of $\lambda(q + p)$ offspring is evaluated according to the fitness function. Using their corresponding fitness values, the selection of p search points is conducted based on the *dynamic peak identification* (DPI) algorithm (Miller and Shaw, 1996), using a prescribed niche radius ρ.

The functionality of DPI is to select a subset from the population in which each

search point has a good fitness value and is not within the radius of the remaining points. The selected p search points are considered as parental points for the next generation and their D-sets are inherited from their parents. Finally, the D-sets are updated based on the selected points. The process above is repeated until the termination criteria are satisfied.

Algorithm 9 Niching Evolution Strategy

procedure $(1 \overset{+}{,} \lambda)$-NICHING-ES$(q, \mathrm{S}, f)$▷ q: number of niches, S: search space, f: fitness function
  Initialize D-set $\{D_i\}_{i=1}^{q+p}$ in search space S
  Set generation counter $c \leftarrow 0$
  while the stop criteria are not fulfilled **do**
    for $i = 1 \rightarrow (q + p)$ **do**
      Generate λ mutations according to D_i
    end for
    Evaluate the population using fitness function f.
    Obtain the dynamic peaks set DPS by performing Dynamic Peak Identification.
    for p in DPS **do**
      Set p as a new search point
      Inherit the D-set from the parent of p and update the D-set accordingly.
    end for
    if $N =$ size of $DPS < q$ **then**
      Generate $N - q$ new search points and reset corresponding D-sets.
    end if
    $c \leftarrow c + 1$
    if $c \mod \kappa = 0$ **then**
      Randomly re-generate $(q + 1)^{th} \ldots (q + p)^{th}$ D-sets.
    end if
  end while
end procedure

For implementation details, we choose the niching-DR2 (Shir and Bäck, 2005b) among many other niching evolution strategies which could be introduced into EGO. It is the niching version of a so-called "second derandomized evolution strategy" (DR2) (Ostermeier et al., 1994). We choose niching-DR2 because it is both simple to implement and converges fast. The parameters of niching-DR2,

which are listed in Alg. 9, are set as following: the function evaluation budget is set the be $10^3(q+p)$. The parameter κ controls the frequency of sub-population resampling and is set to 10. The niche radius ρ required by DPI procedure is computed as (Shir and Bäck, 2005b):

$$\rho = \frac{r}{\sqrt[n]{q}}, \quad r = \frac{1}{2}\sqrt{\sum_{k=1}^{n}(x_{k,max} - x_{k,min})^2},$$

where n is the dimensionality and $x_{k,max}, x_{k,min}$ are the upper and lower bound of each coordinate in the search space. The termination criterion for the niching ES is currently given by the function evaluation budget.

Example As with the Constant Liar (CL) strategy (Ginsbourger et al., 2010), the niching approach is also expected to have a repulsive behavior between the points, due to the niching formation. The CL repulsive behavior is controlled with increasing lie value L. The setting $L = \max\{\mathbf{y}\}$ and $L = \text{mean}\{\mathbf{y}\}$ leads to a space filling behavior (Ginsbourger et al., 2010). The behavior of the niching approach and the CL strategy are further compared and visualized on the Himmelblau's function:

$$y_H(x_1, x_2) = (x_1^2 + x_2 + 11)^2 + (x_1 + x_2^2 - 7)^2$$

Himmelblau's function has four global optima located at $(3, 2)$, $(-2.81, 3.13)$, $(-3.78, -3.23)$ and $(3.58, -1.85)$ with global minimal value 0. In order to show that the niching approach can maintain multiple distinct points, we choose four points to be generated in each iteration, which are expected to identify four global optima on Himmelblau's function.

In Fig. 4.8, we compare the CL strategy with $L = \min\{\mathbf{y}\}$ (CL min) to niching-q-EI in four consecutive iterations (from top to bottom). Both of the two approaches locate the four basins of attraction. They also explore the search space while keeping track of all the points found, showing a trade-off between exploitation and exploration. The difference is that the niching approach is not likely to sample two points in one high performance region while the CL min strategy would result in two (or even more) points explore the same region (see step 3 and 4 in the figure). In addition, we also estimate the q-EI values of points found from a 4-point EGO with niching on Himmelblau's function. The estimation is conducted by Monte Carlo simulations of the q-EI formula (Ginsbourger et al., 2010).

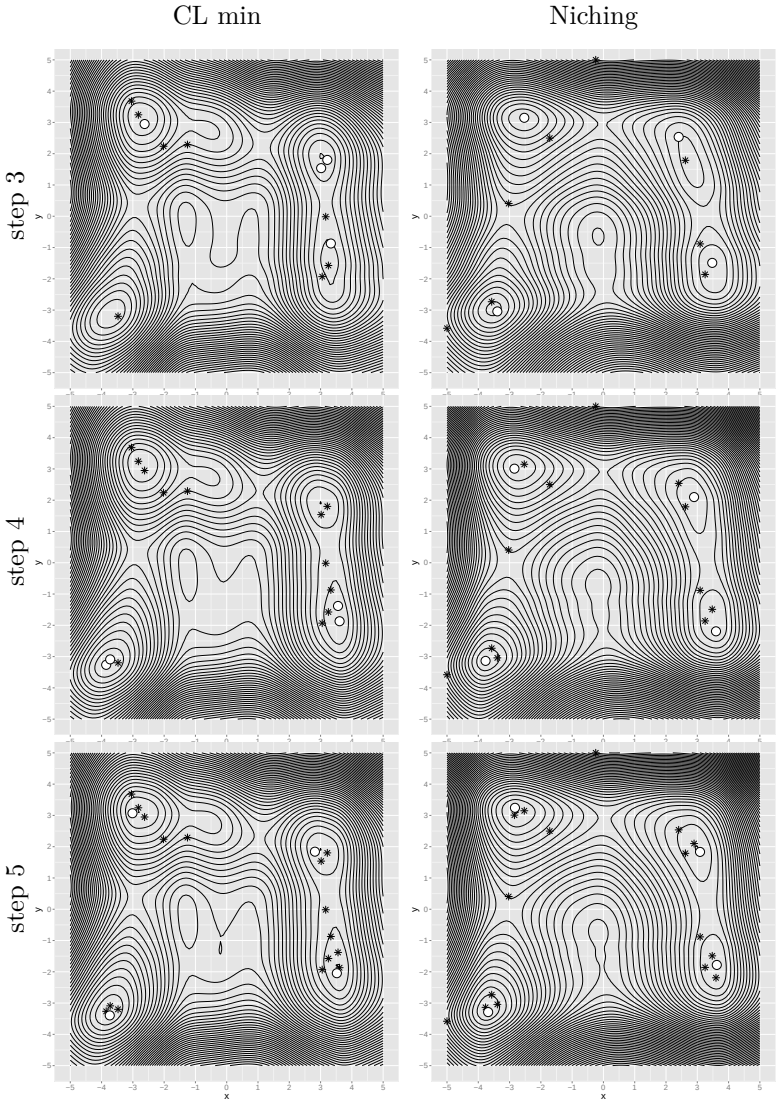


Figure 4.8: On *Himmelblau's function*: 4-point EGO in four iterations. The white circles indicates the current sites found in each iteration while black stars show the sites sampled in history. **Left**: Constant Liar Strategy using min{**y**}. **Right**: niching-q-EI.

4.6 Experimental Comparison

In this section, we test the niching-q-EI and three Constant Liar variants: CL min, CL max and CL mix on a collection of test functions. The CL mix strategy (Chevalier and Ginsbourger, 2013) is a mixture of the CL min and CL max in which two batches of points are generated from the CL min/max and the batch of better q-EI value is provided to the Kriging model. For all the tests we use the `DiceKriging` and `DiceOptim` packages (Roustant et al., 2012). The experiment is presented in three parts: First we list the test functions selected. Then, the global convergence is compared among all the tested algorithms and finally the q-EI values of the point found are compared. All the algorithms are tested in 10 iterations and all the results are averaged over 100 runs. The reason to choose a small number of iterations as test run length is simple: on one hand, the classic EGO algorithm is capable of locating all the optima on several test functions (Jones et al., 1998) using 10 to 20 iterations. Thus, longer runs are not necessary. On the other hand, due to our observations, most of the space explorations and the Kriging model updates happen in the first 10 to 15 iterations on the 2-D functions, in which the EGO algorithm makes large progress.

Test Functions We select 6 artificial multi-modal continuous functions:

- The transformed *Hartman6* function is defined on $[0,1]^6$ and is a unimodal function. The original function is transformed by $-\log(-\,\text{Hartman6}(\mathbf{x}))$. This is the test-case where niching-q-EI is expected to perform badly. We set $q = 3$ on this function.
- M is a hyper-grid multi-global function. Its global optima are uniformly distributed and have optimal value -1. The function expression is listed below. We test the algorithms in 2-D where 10 minima are located in $[0,1]^2$.

$$M(\mathbf{x}) = -\frac{1}{d}\sum_{i=1}^{d}\sin^{\alpha}(5\pi x_i).$$

  Note that d is the dimensionality and we choose $\alpha = 6$.

- The *Branin function* is a multi-global function and a classical test-case in global optimization (Jones et al., 1998; Schonlau, 1998). It is defined in 2-D with three global optima. The global minimal value is roughly 0.4.

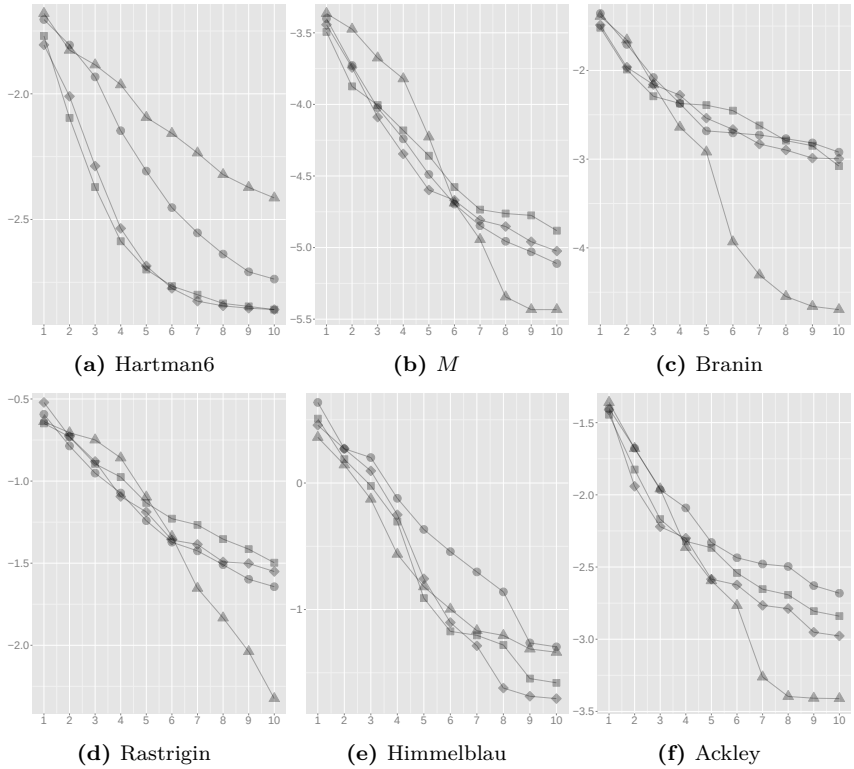
- The *Rastrigin function* (Torn and Zilinskas, 1989) is a multi-modal function, which has only one global optimum, surrounded by a number of local minima. The test is performed on 2-D. It has 6 optima in the space $[0, 1.5]^2$.
- *Himmelblau's function* is a multi-global function which is introduced previously in this thesis. The search space is $[-5, 5]^2$.
- The *Ackley function* is a multi-modal function and has only one global optimum. The global optimum has a much lower value than the local optima.

We choose the number of points generated in each iteration equal to the number of minima with two exceptions: on Hartman6, which is a unimodal function, we choose $q = 3$ and on the Ackley function where the local optima increases exponentially with the increasing distance to the global optimum, we choose $q = 9$. For Hartman6 function, it is intended to show that the niching-q-EI could perform quite badly with $q > 1$ setting. We thus choose a moderate value $q = 3$. For the Ackley function with range $[-5, 5]^2$, there are 8 sub-optimal locations whose function values are the same, inferior to the global optimal but superior to the remaining local optima. We would like to locate such sub-optima as well as the global one and thus choose $q = 9$.

Convergence results The *relative mean squared error* (Chevalier and Ginsbourger, 2013) is used to measure the convergence rate to the global optimum. It is defined as:

$$\mathrm{rMSE}_i = \frac{1}{p} \sum_{k=1}^{p} \left(\frac{f_i^k - f^*}{f^*} \right)^2 .$$

Here rMSE_i denotes the relative mean squared error at iteration i (rMSE should not be confuse the Kriging RMSE in Section 3.1.1). Furthermore, p is the number of runs performed while f_i^k is the minimum value observed at iteration i in run number k. Note that we translate optimal values of some test functions to prevent 0 when calculating the rMSE. The relative mean squared error on each test function is shown in Fig. 4.9. Note that the rMSE is scaled by $\log 10$. On the unimodal function Hartman6, the results of 3-points EGO show that the niching-q-EI performs much worse than any variants of CL strategy. This is the expected behavior because the niches formed on the EI landscape of Hartman6 do not map to any local optima and the niching method is performing space-filling using all three niches.



(a) Hartman6 **(b)** M **(c)** Branin

(d) Rastrigin **(e)** Himmelblau **(f)** Ackley

Figure 4.9: The relative mean squared error to the global optimal (y axis in $\log 10$) against iterations (x axis). Legend: ■: CL max, ●: CL min, ◆: CL mix, ▲: niching-q-EI.

On the multi-modal Rastrigin function, niching-q-EI actually outperforms all the CL variants. On the Branin function, niching-q-EI performs equally to the CL max in the first three iterations and makes a large acceleration from the fourth iteration. On the M function, the niching approach works much worse in the first 6 iterations and accelerates again in the later iterations. On the Rastrigin function, the same behavior is observed. We think that the reason is that initially the Kriging prediction response surface differs from the real landscapes drastically so that the niches formed on the EI landscape do not map to any high performance region. After updating the Kriging model, local optima on the objective function would possibly create local optima in the EI landscape. On Himmelblau's function, niching-q-EI is the worst method initially and finally catches up with the CL mix from iteration 8. On the Ackley function, niching-q-EI performs roughly the

same as the CL min strategy and outperforms both the CL min and CL mix after iteration 5. In general, convergence plots suggest that initially niching-q-EI performs worse than or equally to the CL variants and accelerates the convergence after updating the Kriging model for some iterations. Furthermore, such behavior may even suggest a possible mixture approach where the CL strategy is applied in the beginning and then the algorithm is switched to the niching method to gain from the acceleration.

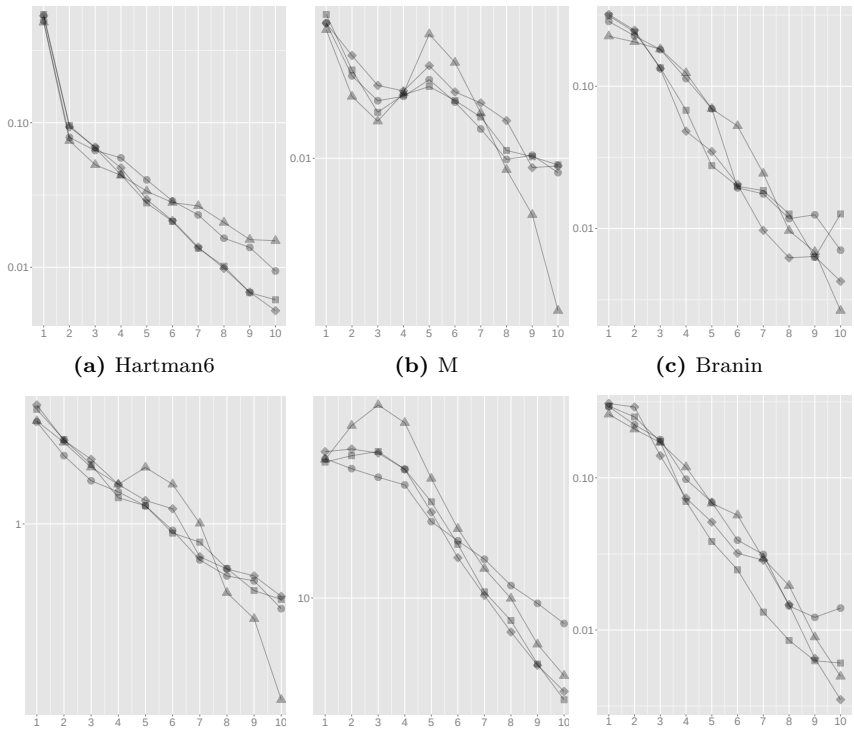


Figure 4.10: Average q-EI value of points from tested algorithms. Legend: ■: CL max, ●: CL min, ◆: CL mix, ▲: niching-q-EI. The x axis represents the iteration while the y axis is the averaged q-EI measured.

q-EI of search points The average q-EI values of the points generated in each iteration are computed by Monte Carlo simulations (Ginsbourger et al., 2010), which are shown in Fig. 4.10. The plotted values are scaled by $\log 10$. All the average q-EI values decrease with respect to increasing iterations. It is not clear

which method is the winner in general. Focusing on the first 4 iterations, q-EI values of the search points found by the niching approach are roughly smaller than q-values from the CL variants on the Hartman6, $\mathcal{M}$ and Ackley function. On the Rastrigin and Himmelblau's function, the proposed algorithm gives higher q-EI values in the middle (iteration $4, 5, 6$) of the test. On the Branin function, the differences are not significant. In general, niching-q-EI shows a much faster q-EI value reduction when the iteration goes upper than 8.

4.7 Summary

Infill criteria control the exploration-exploitation trade-off in the Efficient Global Optimization algorithm. In this chapter, we focus on the improvement-based infill criteria, that is a class of functions that calculates the potential improvement over the current best objective value. It is shown that the exploration-exploitation trade-off can be explicitly controlled by considering the risk and return of the infill criterion, resulting in a multi-objective infill criterion. Alternatively, such an exploration-exploitation control can also be realized by the novel infill-criterion, called Moment-Generating Function of Improvement (MGFI). MGFI introduces a real parameter, called "temperature" that tunes the exploration-exploitation trade-off smoothly. Furthermore, the cooling strategies, that are originally introduced in the Simulated Annealing, are applied to the temperature parameter of MGFI. The resulting infill criterion exhibits high exploration behaviors in the beginning (high temperature) and evolves towards exploitative behaviors as the temperature cools down. Finally, the parallelization problem of infill-criteria is investigated. The challenge here is how to obtain several well-performing candidate solutions based on the infill criterion. Several new methods are proposed for this purpose, including multi-objective infill criteria and niching-assisted infill criteria maximization.

CHAPTER 5

Numerical Multi-objective Optimization

Many multi-objective optimization (MOO) algorithms have been proposed and exploited in real-world problems over the years, e.g., NSGA-II (Deb et al., 2000), SPEA2 (Zitzler et al., 2001) and SMS-EMOA (Beume et al., 2007). These evolutionary multi-criteria optimization (EMO) algorithms employ heuristic operators (e.g., random variation and selection operators), instead of using the gradient information of the objective functions. For a large subclass of such problems, that is the **continuous** multi-objective optimization problem, gradient-based algorithms are of interest due to the fact that they are generally fast, precise and stable with respect to local convergence. Various gradient-based approaches have been proposed for the multi-objective optimization task (Fliege and Svaiter, 2000; López et al., 2012; Hillermeier, 2001; Schütze et al., 2011). A relatively new idea is proposed by (Emmerich et al., 2007; Emmerich and Deutz, 2014), in which the gradient of the hypervolume indicator with respect to a set of decision vectors is computed. In this chapter, we adopt the definition and the computation of the *hypervolume indicator gradient* to steer the search points within the decision space. By using the hypervolume indicator gradient (Emmerich and Deutz, 2014), the search points are moved into the direction of steepest ascent w.r.t. the hypervolume indicator. Therefore, the proposed numerical multi-objective optimization algorithm is termed *hypervolume indicator gradient ascent multi-objective optimization* (HIGA-MO). The major benefits of exploiting hypervolume gradients are 1) the points in the objective space will be well distributed on the Pareto front, 2) it is almost free of control parameters, and 3) the algorithm has a high precision of convergence to the Pareto front.

However, the first implementation of this idea showed numerical problems. As a remedy, ideas that were developed in the field of evolutionary multi-criterion

optimization are adopted in this thesis. Firstly, the hypervolume indicator may have zero gradient components at some decision vectors, e.g., the dominated points. The well-known non-dominated sorting technique is adopted and combined with the hypervolume indicator gradient computation, in order to equip each decision vector with a multi-layered gradient. Secondly, the normalization of the hypervolume indicator *sub*-gradient is used to overcome the "creepiness" phenomenon observed in earlier versions of hypervolume gradient ascent, and caused by an imbalance in the length of sub-gradients which leads to a slow convergence speed (Sosa Hernández et al., 2014). Thirdly, the usage of constant step-sizes is no longer appropriate if the precise convergence to the Pareto front is aimed for. Instead, a cumulative step-size control inspired by the optimal gradient ascent is proposed to dynamically adapt the step-size. Such a cumulative step-size control resembles the step-size adaptation mechanism in the well-known CMA-ES (Hansen and Ostermeier, 2001), an evolutionary algorithm for single objective continuous optimization. The resulting algorithm is tested on problems named ZDT1-4 and ZDT6 from (Zitzler et al., 2000). Its performance is compared to three evolutionary algorithms: NSGA-II (Deb et al., 2000), SPEA2 (Zitzler et al., 2001) and SMS-EMOA (Beume et al., 2007), as well as the other methods for steering the dominated points.

In addition, the hypervolume-based numerical MOO is extended by differentiating the hypervolume gradient again, yielding the *hypervolume indicator Hessian matrix*. We furthermore investigate the condition on which the Hessian matrix stays *non-singular*, showing that it is "safe" to apply the Hessian in general applications. Based on the Hessian matrix, the hypervolume indicator Newton method is proposed and validated.

In the following, the general settings/notations on *set-oriented numerics* are given first. In multi-objective optimization problems (MOPs), a collection of functions, represented as the m-tuple below, are optimized simultaneously:

$$(f_1 \colon \mathrm{S}_1 \to \mathbb{R}, f_2 \colon \mathrm{S}_2 \to \mathbb{R}, \ldots, f_m \colon \mathrm{S}_m \to \mathbb{R}), \quad \mathrm{S}_1, \mathrm{S}_2, \ldots, \mathrm{S}_m \subseteq \mathbb{R}^d.$$

where d denotes the dimension of the domain of each function and m denotes the number of objective functions. Without loss of generality, we assume all the functions above are to be minimized (maximization problems can be transformed into minimization problems). In this thesis, it is assumed that each objective function f_i is continuously differentiable almost everywhere in S_i. Thus, the MOP

can be formulated as follows:

$$\min_{\mathbf{x}\in \mathrm{S}} \mathbf{f}(\mathbf{x}), \quad \mathrm{S} = \bigcap_{i=1}^{m} \mathrm{S}_i \subseteq \mathbb{R}^d,$$

where $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))^\top$ is a vector-valued function composed of m objective functions: $\mathbf{f}\colon \mathrm{S} \to \mathbb{R}^m$. Note that the minimization of the vector-valued function $\mathbf{f}$ is understood with respect to the *Pareto order* $\prec$ as defined in Section 1.2. Let $\mathbf{a}, \mathbf{b}$ be two distinct points in $\mathbb{R}^m$. We say $\mathbf{a} \prec \mathbf{b}$ iff $a_i \leq b_i$, $i = 1, \ldots, m$, where $\leq$ is the natural total order on the real numbers. Because of the continuous differentiability assumption on each objective function, $\mathbf{f}$ is again continuously differentiable almost everywhere in S. The gradient information is expressed as transpose of the Jacobian matrix as follows:

$$\frac{\partial \mathbf{f}(\mathbf{x})}{\partial \mathbf{x}} = \left[\nabla f_1(\mathbf{x}), \nabla f_2(\mathbf{x}), \ldots, \nabla f_m(\mathbf{x})\right], \quad \nabla f_i(\mathbf{x})\colon \mathrm{S} \to \mathbb{R}^d, \quad i = 1, 2, \ldots, m.$$

In addition, it is assumed that each gradient vector above (column vector) can be computed either analytically or numerically. In MOPs, a set of decision vectors are moved in *decision space* S to approximate the Pareto efficient set, which is the so-called Pareto efficient set approximation:

$$X = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(\mu)}\right\}, \ \mathbf{x}^{(i)} \in \mathrm{S}, \ i = 1, 2, \ldots, \mu.$$

with corresponding Pareto front approximation set (objective vectors) in the *objective space*:

$$Y = \left\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(\mu)}\right\}, \ \mathbf{y}^{(i)} = \mathbf{f}(\mathbf{x}^{(i)}) \in \mathbb{R}^m, \ i = 1, 2, \ldots, \mu.$$

In order to measure and compare the quality among Pareto front approximation sets Y, one approach is to quantify the quality by constructing a proper indicator. The most common one is the hypervolume indicator H (Zitzler and Thiele, 1998; Zitzler et al., 2003). Given a reference point $\mathbf{r} \in \mathbb{R}^m$, the **hypervolume indicator** of the Pareto front approximation set Y can be expressed as:

$$H(Y;\ \mathbf{r}) = \lambda^m \left(\bigcup_{\mathbf{y}\in Y} [\mathbf{r}, \mathbf{y}] \right),$$

where λ^m denotes the Lebesgue measure on $\mathbb{R}^m$, which is the size of the hypervolume dominated by the approximation set Y with respect to the reference space. Note that the reference point $\mathbf{r}$ will be assumed to be a given constant and thus omitted

in the following notations for brevity. The hypervolume indicator gradient is defined as the gradient of the hypervolume indicator with respect to the approximation of the Pareto efficient set, which is proposed in Emmerich and Deutz (2014); Emmerich et al. (2007). In this thesis, the derivation of the hypervolume indicator gradient is reformulated and the notation is simplified. In the following, we shall use matrix calculus notations with denominator layout, meaning that the derivative of a vector/matrix is laid out according to the denominator.

5.1 Mixed-Peak Test Problem

Prior to the discussion of the numerical MOO algorithm, a bi-objective problem class, called *Mixed-Peak* problems are introduced for investigating the behavior of the proposed algorithms. Such a problem class is chosen over other standard benchmark problems, e.g., the so-called ZDT problems (Zitzler et al., 2000), because 1) it allows for controlling the problem difficulty of its instance, by varying the number of peaks in each objective function. 2) it is smooth and differentiable almost everywhere in its domain, which makes it a perfect test problem for the gradient and Hessian methods. As no analytical property is available on this problem, the detailed analysis is conducted on this problem and as a result, the expressions of the Pareto front and efficient set are derived.

5.1.1 Mixed-Peak Functions

In this this, a sophisticated problem generator, called *Multiple Peaks Model 2* (MPM2, Wessing (2015)), is adopted to illustrate the proposed topological definitions and further analyze the behavior of explorative algorithms. Such a function class is a mixture of similar unimodal functions, i.e., the peaks, that have *convex* local level sets, which is typically combined with the well-known Karush-Kuhn-Tucker theorem to identify local efficient points. In addition, the complexity of the problem can be easily controlled by the number of peaks. The mixed-peak function is defined as an unconstrained function $f : \mathbb{R}^d \to \mathbb{R}$ that is subject to

minimization:

$$f(\mathbf{x}) = 1 - \max_{1 \le i \le N} \{g_i(\mathbf{x})\}, \quad \mathbf{x} \in \mathbb{R}^d. \tag{5.1}$$

$$g_i(\mathbf{x}) = h_i \left(1 + \frac{\left(\sqrt{(\mathbf{x} - \mathbf{c}_i)^\top \mathbf{\Sigma}_i (\mathbf{x} - \mathbf{c}_i)} \right)^{s_i}}{r_i} \right)^{-1}, \quad i = 1, \ldots, N. \tag{5.2}$$

The function g above defines a parameterized *quasi-concave* unimodal peak, whose negative leads to *quasi-convex* valleys on function f. According to the `optproblems` package (Wessing, 2016), it has the following parameters: (1) number of peaks $N \in \mathbb{Z}_{>0}$, (2) center $\mathbf{c}_i \in \mathbb{R}^d$, height $h_i \in [0, 1]$ and radius $r_i \in [0.25\sqrt{d}, 0.5\sqrt{d}]$ per peak, with decision space dimension d, (3) "shape" $s_i \in [1.5, 2.5]$ per peak, controlling the landscape's steepness, (4) rotation of the elliptical level sets based on a *positive definite* matrix $\mathbf{\Sigma}_i$. In the following, we will use the norm notation $\|\mathbf{x} - \mathbf{c}_i\|_{\mathbf{\Sigma}_i} := \sqrt{(\mathbf{x} - \mathbf{c}_i)^\top \mathbf{\Sigma}_i (\mathbf{x} - \mathbf{c}_i)}$ as it can be considered as the Mahalanobis distance w.r.t. $\mathbf{\Sigma}_i$.

Ridges: As a result from the definition of f (Eq. (5.1)), the landscape can contain ridges. The set of all ridges of f can be represented by:

$$\mathcal{R} = \left\{ \mathbf{x} \in \mathbb{R}^d \mid \exists i \neq j \in \{1, 2, \ldots, N\}, g_i(\mathbf{x}) = g_j(\mathbf{x}) \text{ and } g_i(\mathbf{x}) = \max_{1 \le k \le N} \{g_k(\mathbf{x})\} \right\},$$

i.e., the set of all points on which the value of f is simultaneously attained by at least two peak functions. In the simple case, when the $\mathbf{\Sigma}_i$'s are identical and the peaks differ only in centers, the ridges actually form a Voronoi diagram in the decision space. According to Eq. (5.1), for any point that is not on the ridge, $\mathbf{x} \in \mathbb{R}^d \setminus \mathcal{R}$, there is only one peak function that is effective or *active*. From now on, the active peak function at $\mathbf{x}$ is denoted as g_τ w.r.t. $\tau = \arg\max_{1 \le i \le N} \{g_i(\mathbf{x})\}$. In fact, ridges separate the decision space into many *active regions*, on each of which only a single peak function g is active:

$$\mathcal{A}_i = \left\{ \mathbf{x} \in \mathbb{R}^d \mid \forall k \in \{1, 2, \ldots, N\} \setminus \{i\},\ g_i(\mathbf{x}) > g_k(\mathbf{x}) \right\}, \quad i = 1, 2, \ldots, N.$$

Note that the active regions $\mathcal{A}_i$'s are open and mutually disjoint and the union of all such active regions $\mathcal{A} = \cup_{1 \le i \le N} \mathcal{A}_i$ is equal to the set of non-ridge points.

Convex Local Level Sets: Given the quasi-concavity of each peak g_i, $1-g_i$ has local convex level sets in $\mathbb{R}^d$. If the function $1-g_i$ is restricted to an ε-Euclidean ball $B_\varepsilon(\mathbf{x}^*) = \left\{\mathbf{x} \in \mathbb{R}^d \mid \|\mathbf{x}-\mathbf{x}^*\| < \varepsilon\right\}$ for every $\mathbf{x}^* \in \mathbb{R}^d$ and every $\varepsilon > 0$, the resulting function $1-g_i\big|_{B_\varepsilon(\mathbf{x})} : B_\varepsilon(\mathbf{x}) \to \mathbb{R}$ also has local convex level sets. Also, due the fact that the active regions $\mathcal{A}_i$'s are disjoint and open, for every non-ridge point $\mathbf{x}^*$, it is possible to find a $\delta > 0$ (depending on $\mathbf{x}^*$) such that $B_\delta(\mathbf{x}^*) \subset \mathcal{A}_\tau$ and $B_\delta(\mathbf{x}^*) \cap \mathcal{A}_i = \emptyset$, $\forall i \neq \tau$ (τ is the unique index of the active peak function at $\mathbf{x}^*$). Then the restricted f to $B_\delta(\mathbf{x}^*)$, $f\big|_{B_\delta(\mathbf{x}^*)}$ equals $1-g_\tau\big|_{B_\delta(\mathbf{x}^*)}$ and thus it has local convex level sets. Therefore, we have the following conclusion:

$$\forall \mathbf{x}^* \in \left(\mathbb{R}^d \setminus \mathcal{R}\right)\ \exists \delta > 0,\ f\big|_{B_\delta(\mathbf{x}^*)} \text{ has local convex level sets.} \tag{5.3}$$

For the points on the ridge, $\mathbf{x}^* \in \mathcal{R}$, the conclusion above does not hold because it is not possible to find a δ such that $B_\delta(\mathbf{x}^*)$ has no intersection with all $\mathcal{A}_i$'s except $\mathcal{A}_\tau$.

As the gradient of the mixed-peak function is required to derive the Pareto front, we given it as follows:

$$\nabla f(\mathbf{x}) = \frac{h_\tau s_\tau}{r_\tau}\left(1 + \frac{\|\mathbf{x}-\mathbf{c}_\tau\|_{\mathbf{\Sigma}_\tau}^{s_\tau}}{r_\tau}\right)^{-2} \|\mathbf{x}-\mathbf{c}_\tau\|_{\mathbf{\Sigma}_\tau}^{s_\tau-2}\, \mathbf{\Sigma}_\tau(\mathbf{x}-\mathbf{c}_\tau). \tag{5.4}$$

5.1.2 Mixed-Peak Bi-objective Problem

By generating two different configurations for the parameters in Eq. (5.1), two different multimodal functions are constructed, naturally defining a bi-objective optimization problem:

$$f_1(\mathbf{x}) = 1 - \max_{1\le i\le N} g_i(\mathbf{x}) \to \min, \qquad f_2(\mathbf{x}) = 1 - \max_{1\le i\le N'} g'_i(\mathbf{x}) \to \min\,.$$

Note that the peak function g and g' (and its parameters N and N') are distinguished by the superscript. Next, the efficient set and Pareto front are derived analytically. In the following, the analytical efficient set and Pareto front are derived.

One Peak Scenario We first consider a simple case in which each objective function consists of one peak without any ridges in the domain. In this case, the

objective functions degenerate to:

$$f_1(\mathbf{x}) = 1 - h\left(1 + \frac{\|\mathbf{x}-\mathbf{c}\|_{\mathbf{\Sigma}}^{s}}{r}\right)^{-1}, \quad f_2(\mathbf{x}) = 1 - h'\left(1 + \frac{\|\mathbf{x}-\mathbf{c}'\|_{\mathbf{\Sigma}'}^{s'}}{r'}\right)^{-1}.$$

According to the Karush-Kuhn-Tucker (KKT) condition (Ehrgott, 2006) for multi-objective optimization problems, a *necessary condition* for $\mathbf{x}^* \in \mathbb{R}^d$ being efficient is:

$$\exists \lambda_1 > 0, \lambda_2 > 0,\ \lambda_1 \nabla f_1(\mathbf{x}^*) + \lambda_2 \nabla f_2(\mathbf{x}^*) = \mathbf{0}.$$

Substituting the condition above by the gradient expression (Eq. (5.4)) leads to:

$$\lambda_1 C(\mathbf{x}^*)\mathbf{\Sigma}(\mathbf{x}^* - \mathbf{c}) + \lambda_2 C'(\mathbf{x}^*)\mathbf{\Sigma}'(\mathbf{x}^* - \mathbf{c}') = \mathbf{0},$$
$$\text{with}\ \ C(\mathbf{x}^*) := \frac{hs}{r}\left(1 + \frac{\|\mathbf{x}^*-\mathbf{c}\|_{\mathbf{\Sigma}}^{s}}{r}\right)^{-2} \|\mathbf{x}^*-\mathbf{c}\|_{\mathbf{\Sigma}}^{s-2}.$$

And C' is defined similarly to C by adding prime superscripts to all parameters. As a result, the condition above can further be simplified to:

$$\exists \lambda_1 > 0, \lambda_2 > 0,\ \ \mathbf{\Sigma}(\mathbf{x}^* - \mathbf{c}) = -\frac{\lambda_2 C'(\mathbf{x}^*)}{\lambda_1 C(\mathbf{x}^*)}\mathbf{\Sigma}'(\mathbf{x}^* - \mathbf{c}'). \tag{5.5}$$

Let us denote $k := \lambda_2 C'(\mathbf{x}^*)/\lambda_1 C(\mathbf{x}^*)$. Thus, $\lambda_1, \lambda_2 > 0$ and $C, C' \geq 0$ result in $k \geq 0$. In addition, $C \to 0$ leads to $k \to \infty$, i.e., $\mathbf{x}^* \to \mathbf{c}$. Due to the fact that C and C' are continuous functions w.r.t. $\mathbf{x}^*$, k is also *continuous* in $\mathbb{R}^d$. Therefore, it must take any value between its minimum and maximum, resulting in $0 \leq k < \infty$. Taking the range of k into account, every point that satisfies Eq. (5.5) can be written as:

$$\forall k > 0,\ \ \mathbf{x}^* = \mathbf{c} - \left(\frac{\mathbf{\Sigma}}{k} + \mathbf{\Sigma}'\right)^{-1}\mathbf{\Sigma}'(\mathbf{c} - \mathbf{c}'). \tag{5.6}$$

Note that the points above are not necessarily local efficient points (as defined in Section 1.2). The sufficiency can be shown as follows: for any point $\mathbf{x}^* \in \mathbb{R}^d$ satisfying Eq. (5.6) – remember, there is no ridge in this scenario – there exists an $\varepsilon > 0$ such that the restricted objective function $f_1\big|_{B_\varepsilon(\mathbf{x}^*)}$ has local convex level sets according to Eq. (5.3). Similarly, there exists an $\varepsilon' > 0$ such that $f_2\big|_{B_{\varepsilon'}(\mathbf{x}^*)}$ has local convex level sets. It is then possible to construct a Euclidean ball with radius $\varepsilon^* := \min\{\varepsilon, \varepsilon'\}$ such that: $f_1\big|_{B_{\varepsilon^*}(\mathbf{x}^*)}$ and $f_2\big|_{B_{\varepsilon^*}(\mathbf{x}^*)}$ both have local **convex** level sets. This implies that it is always possible to find a neighborhood around a point where the local level sets of both objective functions are convex. Thus, it is sufficient

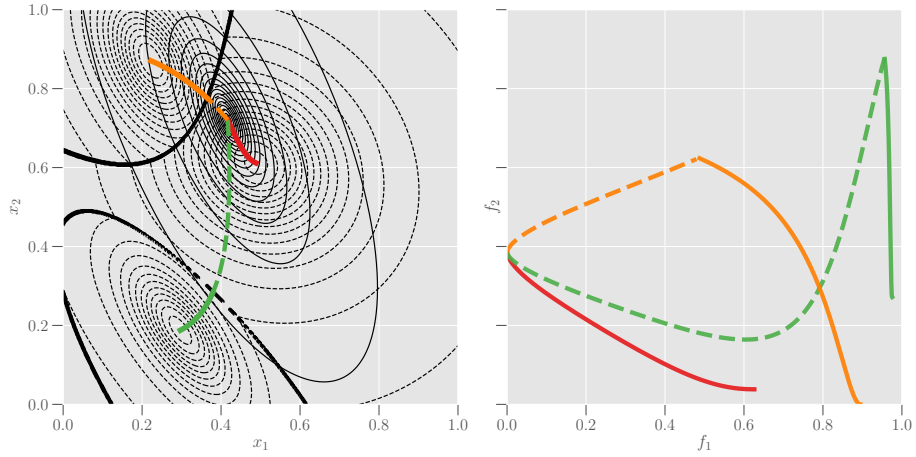


Figure 5.1: Example of analytical Pareto fronts and efficient sets: the contour lines of f_1 (solid curves, 1 peak) and f_2 (dashed curves, 3 peaks) are drawn in the decision space (left) with ridges shown as thick solid curves. Three local efficient sets are drawn in different colors while the dashed extensions of them represent the pseudo-efficient sets. The corresponding Pareto fronts are shown on the right.

to conclude that points satisfying Eq. (5.6) are *locally Pareto efficient* and the efficient set of the problem is expressed as:

$$\mathcal{X}_{LE} = \left\{ \mathbf{c} - \left(\frac{\boldsymbol{\Sigma}}{k} + \boldsymbol{\Sigma}' \right)^{-1} \boldsymbol{\Sigma}'(\mathbf{c} - \mathbf{c}') \;\middle|\; 0 \leq k < \infty \right\}. \tag{5.7}$$

Consequently, the Pareto front can implicitly be obtained by applying the objective functions to the efficient set from above. When the contour lines are spherical for both objective functions, the arguments here can be largely simplified. We omit such a special case, since it has already been discussed in detail in Kerschke et al. (2016).

Multiple Peaks If each of the objective functions consists of multiple peak functions, namely $N > 1$, the efficient set derived in Eq. (5.7) can be adapted in the following manner: suppose function f_1 and f_2 contain N and N' peaks, respectively. For each pair of peaks between two objective functions (e.g., g_i and g'_j), a *pseudo-efficient set* can be calculated according to Eq. (5.7) as if the rest of

the peaks in both objective functions were not existing:

$$\mathcal{P}_{ij} = \left\{ \mathbf{c}_i - \left(\frac{\mathbf{\Sigma}_i}{k} + \mathbf{\Sigma}'_j \right)^{-1} \mathbf{\Sigma}'_j (\mathbf{c}_i - \mathbf{c}'_j) \mid 0 \leq k < \infty \right\},$$

where $\mathbf{c}_i$ and $\mathbf{c}'_j$ are the centers of the i-th and j-th peak of function f_1 and f_2, respectively. Note that Eq. (5.7) requires that no ridge is present in the function domain and thus for the set defined above, it is not necessarily a local efficient set. Let us denote the active region of peak g_i and g'_j as $\mathcal{A}_i$ and $\mathcal{A}'_j$, respectively. Then the region on which g_i and g'_j are both active is $\mathcal{A}_i \cap \mathcal{A}'_j$. Consider the intersections of $\mathcal{P}_{ij}$ and the ridges $\mathcal{R}$ of f_1 for instance: at such points, any infinitesimal movement towards a different active region other than $\mathcal{A}_i \cap \mathcal{A}'_j$ will revert the direction of ∇f_1 and therefore this movement will improve both f_1 and f_2 values of the intersection points. This implies that the points in $\mathcal{P}_{ij}$ intersecting or crossing the ridges are not efficient for g_i and g'_j. In other words, the efficient set $\mathcal{X}^*_{ij} = \mathcal{P}_{ij} \cap \mathcal{A}_i \cap \mathcal{A}'_j$ associated with peak g_i and g'_j is the intersection of $\mathcal{P}_{ij}$ with the active regions of both peak functions. In addition, all local efficient sets can be enumerated by calculating the local efficient set associated with each pair of peaks between two objective functions: $\mathcal{X}^* = \bigcup_{i=1}^{N} \bigcup_{j=1}^{N'} \mathcal{X}^*_{ij}$. An example of this is illustrated in Fig. 5.1. Here, three pseudo-efficient sets are depicted in different colors (red, orange and green) and the orange and green sets are truncated by the ridges (thick black lines), where the valid local efficient sets are depicted as solid curves.

5.2 Hypervolume Indicator Gradient

Intuitively, the hypervolume indicator can be expressed as a function of the Pareto efficient set approximation X, which allows for the differentiation of hypervolume indicator with respect to decision vectors. More specifically, by concatenation of all the vectors in this set, we obtain a so-called $\mu \cdot d$-vector:

$$\mathbf{X} = \left[\mathbf{x}^{(1)^\top}, \mathbf{x}^{(2)^\top}, \ldots, \mathbf{x}^{(\mu)^\top} \right]^\top \in \mathrm{S}^\mu \subseteq \mathbb{R}^{\mu \cdot d}.$$

and its corresponding Pareto front approximation vector can be written as a $\mu \cdot m$-vector:

$$\mathbf{Y} = \left[\mathbf{y}^{(1)^\top}, \mathbf{y}^{(2)^\top}, \ldots, \mathbf{y}^{(\mu)^\top} \right]^\top \in \mathbb{R}^{\mu \cdot m}.$$

In order to establish a connection between $\mu \cdot d$-vectors and $\mu \cdot m$-vectors, we define a mapping $\mathbf{F}\colon \mathrm{S}^{\mu} \to \mathbb{R}^{\mu \cdot m}$,

$$\mathbf{F}(\mathbf{X}) := \left[\mathbf{f}(\mathbf{x}^{(1)})^{\top}, \mathbf{f}(\mathbf{x}^{(1)})^{\top}, \ldots, \mathbf{f}(\mathbf{x}^{(\mu)})^{\top}\right]^{\top}.$$

Now consider that the hypervolume indicator, that is normally defined in the objective space, can be re-written as a function of $\mu \cdot d$-vectors by composition:

$$\mathcal{H}_{\mathbf{F}}(\mathbf{X}) := H(\mathbf{F}(\mathbf{X})),$$

which is a continuous mapping from S^{μ} to $\mathbb{R}$, for which under certain regularity conditions the gradient is defined (in case of differentiable objective functions only for a zero measure subset of $\mathbb{R}^{\mu \cdot d}$ the gradient is undefined, in which case one-sided derivatives still exist). Given $\mathcal{H}_{\mathbf{F}}$, its derivatives (hypervolume indicator gradient) are defined (given they exist) by:

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}} = \left[\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}}^{\top}, \ldots, \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}}^{\top}\right]^{\top}, \tag{5.8}$$

where each of the term in the RHS of the equation above is called *sub-gradient*, which is the local hypervolume change rate by moving each decision vector infinitesimally. It has been shown in Emmerich and Deutz (2014) that the hypervolume indicator gradient is the concatenation of the hypervolume contribution gradients. Moreover, the sub-gradients can be calculated by applying the chain rule:

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}} = \frac{\partial \mathbf{y}^{(i)}}{\partial \mathbf{x}^{(i)}} \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{y}^{(i)}} \tag{5.9}$$

$$= \sum_{k=1}^{m} \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(i)})} \nabla f_k(\mathbf{x}^{(i)}). \tag{5.10}$$

The first partial derivative in Eq. (5.9) is the gradient of $\mathcal{H}_{\mathbf{F}}$ in the objective space while the second one is the transpose of the Jacobian matrix of the mapping $\mathbf{F}$. Eq. (5.10) is the detailed form. From it, it is clear that the hypervolume indicator gradient is a *linear combination of gradient vectors of objective functions*, where the weight for an objective function is the partial derivative of the hypervolume indicator at this objective value. We omit the calculation for gradients of $\mathcal{H}_{\mathbf{F}}$ in the objective space for simplicity, noting that in the bi-objective case they correspond to the length of the steps of the attainment curve. For the high dimensional case and efficient computation, see Emmerich and Deutz (2014).

Note that in practice the length of the sub-gradients usually differs by orders of magnitude, leading to the "creepiness" behavior (Sosa Hernández et al., 2014) that some decision vectors move much faster than the rest, Such a behavior results in a very slow convergence speed and points might get dominated by others. As a remedy, it is suggested to normalize all the sub-gradients.

5.2.1 Steering Dominated Points

The difficulty increases when applying the hypervolume indicator gradient direction for steering the decision vectors: the hypervolume indicator can either be zero or only one-sided at decision vectors. For example, at every strictly dominated search point, the hypervolume indicator sub-gradient is zero, because the Pareto front and thus the hypervolume indicator remain unchanged if it is moved locally in an infinitesimally small neighborhood. For every weakly dominated point, the hypervolume indicator sub-gradient at this point, even does not exist due to the fact that only one-sided partial derivatives exist. Consequently, such decision vectors will become stationary in the gradient ascent method. One obvious solution to such a problem is to apply evolutionary operators (mutation and crossover) on those search points (decision vectors) until they become non-dominated. However, as we are aiming for a fully deterministic multi-objective optimization algorithm, randomized operators are not adopted in this thesis.

Some methods have been proposed to steer dominated points (Ren et al., 2015; Wang et al., 2017; López et al., 2012). The most prominent one, proposed in López et al. (2012), computes the gradient at dominated points as follows (for bi-objective problems):

$$-\left(\frac{\nabla f_1(\mathbf{x}^{(i)})}{\left\|\nabla f_1(\mathbf{x}^{(i)})\right\|}+\frac{\nabla f_2(\mathbf{x}^{(i)})}{\left\|\nabla f_2(\mathbf{x}^{(i)})\right\|}\right), \quad \mathbf{x}^{(i)} \text{ is dominated.}$$

which is a sum of normalized gradients of each objective function (the minus symbol is for the minimization problem). It guarantees that dominated decision vectors move into the *dominance cone* (Wang et al., 2017). However, such a method only considers the movement of single points, instead of a set of search points and it does not generalize to more than two dimensions. We shall call this method **Lara's direction** in the following experiments, where it is compared with the method proposed in this thesis. Another method for steering the dominated points is

proposed by the authors in Wang et al. (2017). It steers dominated points towards the nearest gap on the non-dominated set. The search direction is determined as the gradient of the distance of the dominated objective vector to the center of its nearest gap. Again, this method steers dominated points independently and is termed **gap-filling** in this thesis. In the above methods, dominated points are steered widely independent of each other, which might result in a diversity loss.

In this thesis, we propose to use the *non-dominated sorting* technique that is developed in the NSGA-II algorithm (Srinivas and Deb, 1994), in order to compute the hypervolume indicator gradients of multiple layers of non-dominated sets. In detail, the decision and objective vectors are partitioned into q subsets, or *layers* according to their dominance rank in the objective space:

$$\begin{aligned} \mathbf{X} &\to \left\{\mathbf{X}^1, \mathbf{X}^2, \ldots, \mathbf{X}^q\right\}, \\ \mathbf{X}^i &= \left[\mathbf{x}^{(i_1)^\top}, \mathbf{x}^{(i_2)^\top}, \ldots, \mathbf{x}^{(i_\mu)^\top}\right]^\top, \end{aligned}$$

where X^i indicates a layer of order i and i_μ indexes decision vectors in the ith rank layer. The layers can be recursively defined as (given ND as the procedure

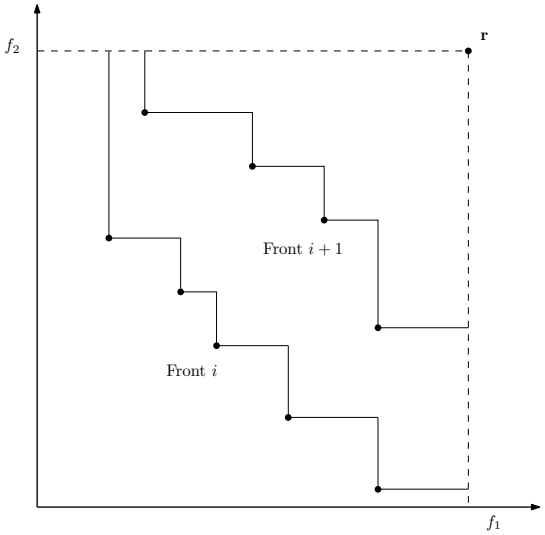


Figure 5.2: Schematic graph showing the partition of the objective vectors using non-dominated sorting. For each partition (layer), a hypervolume indicator is defined and thus its gradient can be computed.

that selects the non-dominated subset from an approximation set):

$$X^1 = \text{ND}(X), \quad X^{i+1} = \text{ND}\left(X - \bigcup_{j=1}^{i} X^j\right),$$

where q is the highest index i such that $X^i \neq \emptyset$. Note that the $\mu \cdot m$-vector is also partitioned as above. In principle, it is possible to compute the hypervolume indicator gradient for any layer by ignoring all the layers that dominate it (have a lower rank) temporarily. This partition is illustrated in Fig. 5.2. The hypervolume volume indicator gradient on the approximation set $\mathbf{X}$ can be (re-)written as the concatenation of hypervolume indicator gradients on each layer:

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}} := \left[\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X}^1)}{\partial \mathbf{X}^1}^\top, \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X}^2)}{\partial \mathbf{X}^2}^\top, \ldots, \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X}^q)}{\partial \mathbf{X}^q}^\top\right]^\top. \tag{5.11}$$

Note that again q is the number of layers obtained from non-dominated sorting techniques. The gradient computation given in Eq. (5.10) can be used to compute each gradient term above. Thus, each decision vector is associated with a steepest ascent direction that maximizes its hypervolume contribution on each layer.

There are two advantages of using the non-dominated sorting procedure. Firstly, maximizing the hypervolume will not only steer the points towards the Pareto front, but also spread out the points across the intermediate Pareto front approximation. By applying the hypervolume indicator gradient direction on each layer, the decision vectors on each layer will be well distributed before a dominated layer merges into the global Pareto front and thus the additional cost to spread out points after merging is small. Moreover, when the Pareto efficient set is disconnected in the decision space, the proposed approach will increase the convergence speed due to the fact that each connected efficient set is treated as one layer and the decision vectors on it are spread quickly over the efficient sets. This effect can be shown by visualizing the trajectories of the approximation set on a simple objective landscape. In Fig. 5.3, trajectories of the approximation set are illustrated in both decision and objective space, on MPM2 functions (from the R `smoof` package[1]). In the decision space, it is clear that our layering approach (Fig. 5.3) manages to approximate five disconnected efficient sets with a good distribution of points. Secondly, on the real landscape, it is possible that local Pareto fronts exist (e.g., consider the well-known ZDT4 problem (Zitzler et al., 2000)). Using the non-dominated sorting, it is more

[1] https://github.com/jakobbossek/smoof

likely to identify those local Pareto fronts, which could be helpful to balance global and local search. This advantage of the proposed approach is exploited by the authors in multi-objective multi-modal landscape analysis (Kerschke et al., 2016).

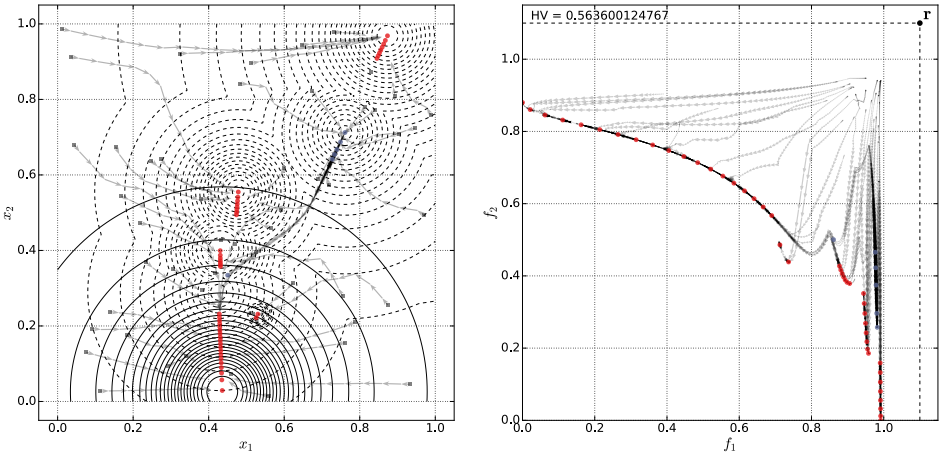


Figure 5.3: Trajectories of 50 points under hypervolume indicator gradient direction to approximate the Pareto front using 10^3 function evaluations. The experiment is conducted on a bi-objective problem MPM2 (from the R `smoof` package) in the 2-D decision space. All five disconnected components of the Pareto front are obtained with well distributed points. **Left**: the decision space. **Right**: the objective space.

5.2.2 Step-size adaptation

The constant step-size setting that is common in gradient descent (ascent) for the single objective optimization task, is no longer appropriate. Usually, the length of the gradient vector (in the gradient field) gradually goes to zero when approaching the local optimum. In this case, a properly set constant step-size will lead to the local optimum in a stable manner. However, in our case, due to the normalization, the length of the search steps is always 1 when decision vectors are approaching the Pareto efficient set. If a constant step-size is applied here, the decision vector will *overshoot* its optimal position and begin to oscillate (even diverge). In order to tackle this issue, the step-size of the decision vectors needs to 1) gradually decrease when approaching the Pareto efficient set and 2) increase quickly when

the decision vectors are far away from the efficient set. In addition, it is reasonable to use individual step-sizes that are controlled independently for each decision vector because their optimal step-size differs largely.

A cumulative step-size adaptation mechanism is proposed to approximate the optimal step-size in the optimization process. It is inspired by the following observation: in single objective gradient optimization, if the step-size is set optimally, then consecutive search directions are perpendicular to each other. In order to approximate the optimal step-size setting, the inner product of consecutive normalized hypervolume indicator gradients is calculated. If such an inner product is positive, it indicates the current step-size is smaller than the optimal one and vice versa:

$$I_t^{(i)} = \left\langle \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}^{(i)}}^{(t-1)}, \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}^{(i)}}^{(t)} \right\rangle, \quad i = 1, \ldots, \mu, \quad t = 1, 2, \ldots.$$

Note that superscripts $(t), (t-1)$ are iteration indices. In addition, such an inner product computed in each iteration fluctuates hugely and direct use of it leads to unstable adaptation behavior. Therefore, the inner product is cumulated using exponentially decreasing weights through the iterations to get a more stable indicator for the step-size adaptation. The cumulative rule for the inner product is written as follows:

$$p_t^{(i)} \leftarrow (1-c)p_{t-1}^{(i)} + cI_t^{(i)}, \quad i = 1, \ldots, \mu, \quad t = 1, 2, \ldots. \tag{5.12}$$

Note that $p_t^{(i)}$ denotes the cumulated inner product for search point i at iteration t and c $(0 < c < 1)$ is the accumulation coefficient. Such an inner product accumulation rule is similar to the cumulative step-size adaptation mechanism in the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen and Ostermeier, 2001), where consecutive mutation steps are accumulated for step-size adaptation. Based on the cumulated inner product, a simple control rule is designed to adapt the step-size online:

$$\sigma_{t+1}^{(i)} = \begin{cases} \alpha\sigma_t^{(i)} & \text{if } p_t^{(i)} < 0, \\ \sigma_t^{(i)} & \text{if } p_t^{(i)} = 0, \\ \sigma_t^{(i)}/\alpha & \text{if } p_t^{(i)} > 0. \end{cases} \quad 0 < \alpha < 1. \tag{5.13}$$

where $\sigma_t^{(i)}$ is the individual step-size for search point i at iteration t. In this thesis, the settings of $c = 0.7, \alpha = 0.8$ are suggested by tuning the algorithmic performance on MPM2 functions.

The backtracking line search (Nocedal and Wright, 2000), which is a common technique to approximate the optimal step-size in single objective gradient ascent, is not suitable for the proposed algorithm. It requires additional function evaluations for each search point to estimate the optimal step-size setting. Such additional costs are no longer acceptable for the set-based algorithm. In contrast, the proposed cumulative step-size adaptation mechanism does not bring any additional overheads.

5.2.3 Hypervolume Indicator Gradient Ascent Algorithm

In this section, the algorithmic components developed in the previous sections are combined into the *Hypervolume Indicator Gradient Ascent Multi-objective Optimization* (HIGA-MO) algorithm.

In practice, the continuous objective function can be non-differentiable at some points, even if the function is almost everywhere differentiable (e.g., on the constraint boundary of the ZDT1 problem). To overcome this issue, it is suggested to *mutate* those points in the decision space. Given a point $\mathbf{x} \in \mathbb{R}^d$, it is mutated in the decision space S when the gradient of objective functions at $\mathbf{x}$ contains invalid values (e.g., the derivative becomes infinite when approaching the origin, on function $f = 1/x$). The mutation of $\mathbf{x}$ should be local but large enough to escape from the non-differentiable regions. For this purpose, the mutation operator in Differential Evolution (Storn and Price, 1997) is adopted here because it is adaptive and only contains a single parameter. Suppose $\mathbf{x}$ is in the ith ranked layer ($\mathbf{x} \in \mathbf{X}^i$), then the following mutation operation is applied on $\mathbf{x}$:

$$\mathbf{x} \leftarrow \mathbf{x} + F(\mathbf{x}^{(a)} - \mathbf{x}^{(b)}), \tag{5.14}$$

where $\mathbf{x}^{(a)}, \mathbf{x}^{(b)}$ are randomly picked from $\mathbf{X}^i$. Furthermore, $F \in [0, 2]$ is the differential weight that is set according to the literature. It is necessary to compute the differential vector within the same layer of $\mathbf{x}$ because the Pareto efficient set is possibly disconnected in the decision space and differential vectors computed across layers possibly create non-local mutations.

The resulting algorithm is presented in Alg. 10. In line 4, the non-dominated sorting procedure is called to partition the approximation set. In line 7 the hypervolume indicator gradient is computed for every decision vector on each layer. If a decision vector has either zero gradient or is not differentiable, it is

mutated in line 9 according to Eq. (5.14). In line 11, the hypervolume indicator sub-gradient is normalized before decision vectors are moved in the steepest ascent manner (line 12). The cumulative step-size adaptation (Eq. (5.12) and (5.13)) is then applied in line 13. In addition to the common usage of the function evaluation budget for the termination criterion, it is suggested here to check stationarity of search points: a decision vector is considered stationary if the norm of its

Algorithm 10 Hypervolume Indicator Gradient Ascent Multi-Objective Optimization

1: **procedure** HIGA-MO($\mu, \mathrm{S}, \mathbf{f}, \nabla\mathbf{f}$) ▷ $\nabla\mathbf{f}$: Jacobian of the objective function
2: $c \leftarrow 0.7,\ \alpha \leftarrow 0.8,\ F \in [0, 2]$. ▷ endogenous parameters
3: Initialize μ search points $X = \left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(\mu)}\right\} \subset \mathrm{S}$ uniformly.
4: **while** the termination criteria are not satisfied **do**
5: $Y \leftarrow \left\{\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots, \mathbf{y}^{(\mu)}\right\} \leftarrow \left\{\mathbf{f}(\mathbf{x}^{(1)}), \mathbf{f}(\mathbf{x}^{(2)}), \ldots, \mathbf{f}(\mathbf{x}^{(\mu)})\right\}$
6: $\left\{X^1, X^2, \ldots, X^q\right\} \leftarrow$ NON-DOMINATED-SORTING(X, Y)
7: **for** $k = 1$ to q **do**
8: **for** every $\mathbf{x}^{(i)}$ in X^k **do**
9: Compute the sub-gradient (Eq. (5.9)):

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}} \leftarrow \nabla\mathbf{f}(\mathbf{x}^{(i)}) \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{y}^{(i)}}$$

10: **if** $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}}$ is undefined **then**
11: Randomly pick $\mathbf{x}^{(a)} \neq \mathbf{x}^{(b)}$ from X^k
12: $\mathbf{x}^{(i)} \leftarrow \mathbf{x}^{(i)} + F(\mathbf{x}^{(a)} - \mathbf{x}^{(b)})$
13: **else**
14: $\mathbf{g}^{(i)} \leftarrow \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}} \Big/ \left\| \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(i)}} \right\|$ ▷ sub-gradient normalization
15: $\mathbf{x}^{(i)} \leftarrow \mathbf{x}^{(i)} + \sigma^{(i)} \mathbf{g}^{(i)}$ ▷ gradient ascending
16: $p^{(i)} \leftarrow (1 - c)p^{(i)} + c\langle \mathbf{g}^{(i)}, \mathbf{g}^{(i)}_{\mathrm{old}} \rangle$ ▷ cumulation
17: $\sigma^{(i)}_{t+1} = \begin{cases} \alpha\sigma^{(i)}_t & \text{if } p^{(i)}_t < 0, \\ \sigma^{(i)}_t & \text{if } p^{(i)}_t = 0, \\ \sigma^{(i)}_t / \alpha & \text{if } p^{(i)}_t > 0. \end{cases}$ ▷ step-size control
18: $\mathbf{g}^{(i)}_{\mathrm{old}} \leftarrow \mathbf{g}^{(i)}$
19: **end if**
20: **end for**
21: **end for**
22: **end while**
23: **return** X, Y
24: **end procedure**

sub-gradient multiplied by the step-size is close to zero ($\leq 10^{-8}$).

5.2.4 Experiments

Experiment settings To test the performance of HIGA-MO, the well-known ZDT problems (Deb et al., 2000) are selected as benchmark problem set. The proposed algorithm is compared to three well-established evolutionary multi-objective optimization algorithm: NSGA-II, SPEA2 and SMS-EMOA. The parameters in those two algorithms are set according to the literature (Deb et al., 2000; Beume et al., 2007; Zitzler et al., 2001). In addition, other methods for steering the dominated point (Section 4), Lara's direction and Gap-filling, are tested against HIGA-MO. For these two methods, the non-dominated points are moved using the hypervolume indicator gradient.

The hypervolume indicator and convergence measure used in Beume et al. (2007), are adopted here as the performance metrics. The convergence measure is calculated numerically by discretizing the Pareto front into 1000 points. For the hypervolume indicator computation, the reference point $[11, 11]^\top$ is used for the test problems $\mathrm{ZDT1} - 4$ and ZDT6. Two experiments are conducted: one with a relatively small population setting $\mu = 40$ while the other uses a large population, $\mu = 100$. A relatively small function evaluation budget, 100μ, is chosen here due to the reason that in long runs, all deterministic methods stagnate to local optima. All the algorithms terminate if the maximal function evaluation budget is reached. For each algorithm, 15 independent runs are conducted to obtain average performance measures. The initial step-size of the proposed HIGA-MO algorithm is set to 0.05 multiplied by the maximum range of the decision space. The internal reference point to compute the hypervolume indicator gradient is set to $[11, 11]^\top$ to ensure every objective vector is within the reference space.

Results The test results are shown in Tab. 5.1 for $\mu = 40$ and Tab. 5.2 for $\mu = 100$. The hypervolume of the non-dominated set after termination is used to compute the performance measures. For the small population setting, HIGA-MO outperforms the evolutionary algorithms (NSGA-II, SPEA2 and SMS-EMOA) on ZDT1-3 and ZDT6 problems, both in terms of hypervolume indicator and convergence measure. By checking the standard deviation, it is obvious that HIGA-MO generates more stable results compared to evolutionary algorithms and

such deviations are only affected by the initialization of the approximation set and the technique to handle the non-differentiable points (Eq. (5.14)). Comparing it to the other two methods, namely, Lara's direction and Gap-filling, that steer the dominated points independently, HIGA-MO gives a higher hypervolume indicator value on ZDT1-3 while Lara's method performs better on ZDT6.

Table 5.1: $\mu = 40$: performance measures on ZDT1-4 and ZDT6 problems.

Test-function	Algorithm	Convergence measure			Hypervolume indicator		
		Average	Std. dev.	Rank	Average	Std. dev.	Rank
ZDT1	HIGA-MO	**0.00500490**	1.3075e-02	1	**120.62948062**	4.0750e-03	1
	Lara's direction	0.07747718	6.4031e-02	3	120.33761711	1.2309e-01	2
	Gap-filling	0.06061863	1.2352e-01	2	120.22307239	4.6840e-01	3
	NSGA-II	0.10960371	3.2542e-02	5	119.33541376	3.7345e-01	4
	SMS-EMOA	0.09376444	3.5934e-02	4	119.20965862	4.8101e-01	5
	SPEA2	0.32006024	5.9788e-02	6	116.27370195	1.6826e+00	6
ZDT2	HIGA-MO	0.00036082	3.6233e-05	3	**120.31634691**	9.8307e-04	1
	Lara's direction	**0.00011253**	5.0289e-05	1	118.92812930	3.5019e+00	3
	Gap-filling	0.00015973	2.0645e-04	2	119.45871166	2.5324e+00	2
	NSGA-II	0.16511979	7.7092e-02	4	114.03423180	3.7806e+00	4
	SMS-EMOA	0.24929199	8.4178e-02	5	109.17629732	3.2584e+00	5
	SPEA2	0.67688451	1.5708e-01	6	104.54506810	3.3537e+00	6
ZDT3	HIGA-MO	0.00031903	5.0492e-05	2	128.55259300	7.9970e-01	2
	Lara's direction	**0.00028076**	5.0842e-05	1	125.78304061	3.5114e+00	6
	Gap-filling	0.00034568	5.4557e-05	3	**128.75911576**	9.2658e-03	1
	NSGA-II	0.00228282	5.9689e-03	4	126.56081625	2.8857e+00	3
	SMS-EMOA	0.00405046	5.7238e-03	5	125.88966563	2.9289e+00	5
	SPEA2	0.00635668	1.0852e-02	6	126.55026001	2.5895e+00	4
ZDT4	HIGA-MO	38.13060527	7.6780e+00	4	0.00000000	0.0000e+00	6
	Lara's direction	43.19742796	1.1544e+01	5	0.00000000	0.0000e+00	5
	Gap-filling	52.35972878	1.2465e+01	6	1.16325406	4.3525e+00	4
	NSGA-II	4.07411956	1.6869e+00	2	75.28344930	1.8038e+01	2
	SMS-EMOA	**3.52099683**	1.7386e+00	1	**78.04608227**	1.8555e+01	1
	SPEA2	11.17677922	4.9514e+00	3	19.34577362	2.2000e+01	3
ZDT6	HIGA-MO	3.83694298	1.3668e+00	6	113.28359226	1.3577e+00	2
	Lara's direction	**0.00010409**	4.3909e-05	1	**116.86127498**	1.6820e+00	1
	Gap-filling	3.02249489	2.7090e+00	5	106.81768735	2.0573e+01	3
	NSGA-II	1.28139859	3.0071e-01	2	97.53535725	3.8143e+00	4
	SMS-EMOA	1.36426329	3.1163e-01	3	96.84386232	4.2309e+00	5
	SPEA2	2.22799304	7.2398e-01	4	86.25780584	7.9570e+00	6

In terms of the convergence measure, Lara's direction always outperforms HIGA-MO on ZDT1-3 and 6. Lara's direction moves the dominated points toward the Pareto front without considering their distribution while HIGA-MO is designed to achieve both. Thus, HIGA-MO requires more efforts to approach the Pareto front than Lara's direction, in terms of the convergence measure. On ZDT4, which

has a highly multi-modal landscape, none of the gradient-based methods (HIGA-MO, Lara's direction and Gap-filling) achieves comparable results to evolutionary algorithms. The gradient-based methods easily stagnate in the local Pareto-front and fail to move towards the global one. For such a highly multi-modal optimization problem, a restart heuristic could improve the performance of gradient-based algorithms. For the large population setting, Tab. 5.2 shows roughly the same results for algorithm comparisons as for the small population setting.

Table 5.2: $\mu = 100$: performance measures on ZDT1-4 and ZDT6 problems.

Test-function	Algorithm	Convergence measure			Hypervolume indicator		
		Average	Std. dev.	Rank	Average	Std. dev.	Rank
ZDT1	HIGA-MO	**0.00031201**	4.1269e-05	1	**120.64580412**	1.7718e-03	1
	Lara's direction	0.02103585	4.7314e-02	5	120.48926778	5.2474e-02	2
	Gap-filling	0.02091304	6.1387e-02	4	120.42616648	2.7937e-01	5
	NSGA-II	0.01769266	4.6048e-03	3	120.45030137	4.5135e-02	4
	SMS-EMOA	0.01234011	2.6377e-03	2	120.48071780	3.6130e-02	3
	SPEA2	0.06017346	1.7966e-02	6	119.86686583	2.1615e-01	6
ZDT2	HIGA-MO	0.00028335	3.3303e-05	3	**120.31710222**	2.3560e-03	1
	Lara's direction	**0.00005498**	1.2085e-05	1	120.30338190	2.9998e-03	2
	Gap-filling	0.00007857	8.7094e-05	2	120.14758158	1.5778e-01	3
	NSGA-II	0.02834448	4.4153e-03	5	119.16220851	1.0985e+00	4
	SMS-EMOA	0.02338094	7.0938e-03	4	118.40070248	2.7352e+00	5
	SPEA2	0.08566545	4.8472e-02	6	114.48551919	4.4285e+00	6
ZDT3	HIGA-MO	0.00047505	7.5997e-05	3	128.77154126	8.5828e-03	3
	Lara' direction	0.00046485	5.9553e-05	2	128.77257561	5.2596e-03	2
	Gap-filling	**0.00039660**	4.9392e-05	1	128.77099724	3.3611e-03	4
	NSGA-II	0.00063823	5.1880e-05	5	**128.77436195**	1.1318e-03	1
	SMS-EMOA	0.00055256	3.5594e-05	4	128.34841609	1.0889e+00	6
	SPEA2	0.00243258	6.6391e-03	6	128.55447469	7.9741e-01	5
ZDT4	HIGA-MO	31.34155544	3.9090e+00	4	0.00000000	0.0000e+00	6
	Lara's direction	40.35930710	1.1041e+01	5	0.00000000	0.0000e+00	5
	Gap-filling	43.47103886	1.5933e+01	6	5.23444012	1.5425e+01	4
	NSGA-II	**0.80498648**	5.0038e-01	1	**109.60569075**	5.4368e+00	1
	SMS-EMOA	1.01209147	6.3095e-01	2	107.14186469	7.1460e+00	2
	SPEA2	2.80155378	1.3959e+00	3	83.82023960	1.5461e+01	3
ZDT6	HIGA-MO	3.54689504	1.2985e+00	5	113.79978098	8.8488e-01	2
	Lara's direction	**0.00004369**	1.2553e-05	1	**116.49314419**	1.4990e+00	1
	Gap-filling	4.12388484	2.9230e+00	6	86.58598768	3.4123e+01	6
	NSGA-II	0.43202530	7.1773e-02	3	109.28079070	1.2513e+00	4
	SMS-EMOA	0.40028650	1.1394e-01	2	109.87049482	1.8951e+00	3
	SPEA2	0.49692387	1.2882e-01	4	108.17997611	1.9177e+00	5

As shown in the experimental results on ZDT4, the proposed algorithm fails to approach the global Pareto front and gets stuck in local ones instead. In practice, such an issue can be tackled by using restart heuristics to re-sample the stagnated

points. In addition, it is possible to hybridize HIGA-MO with an evolutionary multi-objective (EMO) algorithm, where the global search ability of an EMO helps the algorithm to escape from a deceptive, local Pareto front and HIGA-MO could achieve fast convergence speed when approaching the global Pareto front. Such an approach has been proposed in López et al. (2012) and the optimal way to combine HIGA-MO with EMOs should be investigated.

The experiments conducted in this thesis are on a small number of problems. In future research, the proposed algorithm should be investigated on more multi-objective problems. When using a large number of search points, the objective vectors on the Pareto front are close to each other, which might result in relatively slow movement. In this case, its performance needs to be further tested. In addition, it is of interest to compare HIGA-MO empirically to other set-based scalarization method (Schütze et al., 2016).

5.3 Hypervolume Indicator Hessian

In this section, we first derive the Hessian matrix of the hypervolume indicator for the general *multi-objective* optimization scenario. The Hessian matrix in *bi-objective* cases is treated in Section 5.3.1. For conciseness, matrix calculus notations are used in the following derivation, which helps to understand the structure of the Hessian matrix. The hypervolume Hessian matrix is the "Jacobian" of the hypervolume gradient defined as follows:

$$
\begin{aligned}
\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) &= \frac{\partial}{\partial \mathbf{X}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{X}} \right) \qquad (5.15) \\
&= \left(\underbrace{\frac{\partial}{\partial \mathbf{X}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}} \right)}_{\mu \cdot d \times d}, \ldots, \frac{\partial}{\partial \mathbf{X}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}} \right) \right) \\
&= \begin{pmatrix} \frac{\partial}{\partial \mathbf{x}^{(1)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}} \right) & \cdots & \frac{\partial}{\partial \mathbf{x}^{(1)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}} \right) \\ \vdots & \ddots & \vdots \\ \frac{\partial}{\partial \mathbf{x}^{(\mu)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(1)}} \right) & \cdots & \frac{\partial}{\partial \mathbf{x}^{(\mu)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(\mu)}} \right) \end{pmatrix},
\end{aligned}
$$

where each *sub-gradient* is differentiated with respect to $\mathbf{X}$. This results in μ^2 block partitions $(d \times d)$ of the Hessian matrix. The (i, j)-block matrix can be

further expressed as follows:

$$\begin{aligned}
&\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{x}^{(j)}}\right) = \frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathbf{y}^{(j)}}{\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial \mathbf{y}^{(j)}}\right) \\
&= \sum_{k=1}^{m} \frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial f_k(\mathbf{x}^{(j)})}{\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(j)})}\right) \\
&= \underbrace{\sum_{k=1}^{m} \frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(j)})}\right)\nabla f_k(\mathbf{x}^{(j)})^{\top}}_{\mathbf{A}_{ij}} + \underbrace{\sum_{k=1}^{m} \frac{\partial^2 f_k(\mathbf{x}^{(j)})}{\partial \mathbf{x}^{(i)}\partial \mathbf{x}^{(j)}}\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(j)})}}_{\mathbf{B}_{ij}}.
\end{aligned} \tag{5.16}$$

According to the differentiation above, each (i,j)-block matrix is a combination of two components: $\mathbf{A}_{ij}$ and $\mathbf{B}_{ij}$. Note that matrix $\mathbf{A}_{ij}$, $\frac{\partial}{\partial \mathbf{x}^{(i)}}\left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(j)})}\right)$ is a column vector of size n and stands for the sub-gradient of $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k(\mathbf{x}^{(j)})}$ at $\mathbf{x}^{(j)}$. In the following, we abbreviate $f_k(\mathbf{x}^{(i)})$ as $f_k^{(i)}$ and its gradient $\nabla f_k(\mathbf{x}^{(i)})$ as $\nabla f_k^{(i)}$.

The first component: $\mathbf{A}_{ij}$ Due the fact that the matrix $\mathbf{a}_{ij}$ is a sum of m outer products, this term has *at most* rank m. It is possible to make $\mathbf{A}_{ij}$ to have full rank only if $m \geq n$. In other cases, $\mathbf{A}_{ij}$ is always rank deficient ($rank(\mathbf{A}_{ij}) \leq m < d$). This indicates that in the "usual" multi-objective optimization case, where the number of objective functions is smaller than the number of decision variables, such a matrix $\mathbf{A}_{ij}$ is *always singular*.

In the following lemma, a detailed expression of $\mathbf{A}_{ij}$ is given for the bi-objective case ($m = 2$). Without loss of generality, we assume that the objective vectors (and corresponding decision vectors) are arranged according to the ascending order of the first objective values.

Lemma 5.1. *Let $m = 2$, $i = 1, \ldots, \mu$ and $j = 1, \ldots, \mu$. Assume that all vectors $\mathbf{x}^{(i)}$ are mutually non-dominated, then the first component $\mathbf{A}_{ij}$ is non-zero only if the block matrix is located on the main diagonal ($i = j$) or the first diagonal above/below the main diagonal ($|i - j| = 1$), and it can be written as:*

$$\mathbf{A}_{ij} = \begin{cases} \nabla f_2^{(j)} \nabla f_1^{(j)\top} + \nabla f_1^{(j)} \nabla f_2^{(j)\top} & \text{if } i = j \\ -\nabla f_1^{(j+1)} \nabla f_2^{(j)\top} & \text{if } i = j+1 \\ -\nabla f_2^{(j-1)} \nabla f_1^{(j)\top} & \text{if } i = j-1 \\ \mathbf{0} & \text{otherwise.} \end{cases} \tag{5.17}$$

Proof. Assume a fixed reference point $\mathbf{r} = (r_1, r_2)^{\top}$. To simplify the formulation, we denote $f_1^{(\mu+1)} := r_1$ and $f_2^{(0)} := r_2$. The partial derivative of the hypervolume

indicator w.r.t. the objective value is derived in Emmerich and Deutz (2014), which corresponds to the length of the steps of the attainment curve:

$$\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_1^{(j)}} = f_2^{(j)} - f_2^{(j-1)}, \ \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_2^{(j)}} = f_1^{(j)} - f_1^{(j+1)}. \tag{5.18}$$

It is clear that $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_1^{(j)}}$ is a function of only $\mathbf{x}^{(j)}$ and $\mathbf{x}^{(j-1)}$ (similar argument holds for $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_2^{(j)}}$). The gradient of the partial derivatives can be given, for example: $\frac{\partial}{\partial \mathbf{x}^{(j)}} \left(\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k^{(j)}} \right) = \nabla f_2^{(j)}$. Such a gradient is nonzero for at least one objective function, when $i = j$, $i = j+1$ or $i = j-1$. By substituting the required gradients into Eq. (5.16), the expression of $\mathbf{A}_{ij}$ can be obtained. □

The second component: $\mathbf{B}_{ij}$ $\mathbf{B}_{ij}$ is a weighted sum of second order derivatives of the objective functions, where the weights are partial derivatives of the hypervolume indicator at each objective value (cf. Eq. (5.10)). Note that the second order derivative $\frac{\partial^2 f_k^{(j)}}{\partial \mathbf{x}^{(i)} \partial \mathbf{x}^{(j)}}$ is not zero if and only if $i = j$:

$$\mathbf{H}_k^{(j)} \coloneqq \frac{\partial^2 f_k^{(j)}}{\partial {\mathbf{x}^{(j)}}^2},$$

is the Hessian matrix of objective function f_k at point $\mathbf{x}^{(j)}$. Consequently, matrix $\mathbf{B}_{ij}$ can be written as:

$$\mathbf{B}_{ij} = \begin{cases} \sum_{k=1}^{m} \frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k^{(j)}} \mathbf{H}_k^{(j)} & \text{if } i = j \\ \mathbf{0} & \text{if } i \neq j. \end{cases} \tag{5.19}$$

Note that $\frac{\partial \mathcal{H}_{\mathbf{F}}(\mathbf{X})}{\partial f_k^{(j)}}$ can be obtained from Eq. (5.18). The singularity of matrix $\mathbf{B}_{ij}$ depends on the properties of the Hessian matrices of the objective functions. Under the assumption that all objective functions are convex (the objective-wise Hessian matrices are positive-definite), matrix $\mathbf{B}_{ij}$ is also positive-definite, under the condition that all objective functions are subject to maximization (for minimization, $\mathbf{B}_{ij}$ is negative-definite). In general, if each objective function has non-singular Hessian matrix almost everywhere, it is obvious that the matrix $\mathbf{B}_{ij}$ is non-singular.

5.3.1 The Bi-objective Case

For a bi-objective optimization problem, the hypervolume Hessian matrix has the following structure:

$$\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X}) = \begin{pmatrix} \mathbf{D}_1 & \tilde{\mathbf{A}}_1 & & & \\ \tilde{\mathbf{A}}_1^\top & \mathbf{D}_2 & \tilde{\mathbf{A}}_2 & & \\ & \tilde{\mathbf{A}}_2^\top & \ddots & \ddots & \\ & & \ddots & \ddots & \tilde{\mathbf{A}}_{\mu-1} \\ & & & \tilde{\mathbf{A}}_{\mu-1}^\top & \mathbf{D}_\mu \end{pmatrix},$$

where $\mathbf{D}_i = \mathbf{A}_{ii} + \mathbf{B}_{ii}$ and $\tilde{\mathbf{A}}_i = \mathbf{A}_{i(i+1)} = -\nabla f_2^{(i)} \nabla f_1^{(i+1)^\top}$ according to Eq. (5.17). Note that the Hessian matrix $\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})$ is a *tridiagonal block* matrix. It is important to investigate when the diagonal block matrix is singular. Due to the difficulty of the investigation, we start to discuss the invertibility of the Hessian matrix in two special cases: single-point system, where only a single decision vector is moved and two-point system where the interactions between two points need to be considered.

One-point system In this case, the hypervolume Hessian matrix degenerates to the diagonal block matrix $\mathbf{D}_i$, that can be expressed using Eq. (5.19):[1]

$$\mathbf{D}_i = \underbrace{\left(f_2^{(i)} - f_2^{(i-1)}\right)\mathbf{H}_1^{(i)} + \left(f_1^{(i)} - f_1^{(i+1)}\right)\mathbf{H}_2^{(i)}}_{\mathbf{B}_{ii}} + \underbrace{\nabla f_2^{(i)} \nabla f_1^{(i)^\top} + \nabla f_1^{(i)} \nabla f_2^{(i)^\top}}_{\mathbf{A}_{ii}}.$$

To investigate the invertibility of such a matrix, we assume that each objective function is **convex**, in addition to the differentiability assumption.

Theorem 5.1. *If the decision vector belongs to the efficient set, $\mathbf{x}^{(i)} \in P_{\mathcal{X}}$ and its two neighbors $\mathbf{x}^{(i-1)}$ and $\mathbf{x}^{(i+1)}$ are not weakly dominated simultaneously, then the diagonal block matrix $\mathbf{D}_i$ is non-singular and negative definite.*

Proof. Note the following facts: 1) Due the assumption that both objective functions are convex, the objective-wise Hessian matrices $\mathbf{H}_1^{(i)}$ and $\mathbf{H}_2^{(i)}$ are positive definite. 2) Since the minimization task is assumed and $\mathbf{x}^{(i-1)}, \mathbf{x}^{(i+1)}$ are not weakly dominated simultaneously, the coefficients $\left(f_2^{(i)} - f_2^{(i-1)}\right)$ and $\left(f_1^{(i)} - f_1^{(i+1)}\right)$ are non-positive but do not take zero value at the same time (Emmerich and Deutz,

[1] As only a single decision vector is considered here, the script index i can be removed. We still keep it because the discussion of invertibility here holds for every diagonal block matrix.

2014). 3) If $\mathbf{x}^{(i)} \in P_{\mathcal{X}}$, then the two objective-wise gradients are anti-parallel to each other, namely $\exists \beta > 0,\ \nabla f_1^{(i)} = -\beta \nabla f_2^{(i)}$, due the Karush-Kuhn-Tucker theorem (Ehrgott, 2006). Then, $\forall \mathbf{y} \in \mathbb{R}^d \setminus \{\mathbf{0}\}$, the quadratic form associated with $\mathbf{D}_i$ is:

$$\mathbf{y}^\top \mathbf{D}_i \mathbf{y} = \left(f_2^{(i)} - f_2^{(i-1)}\right) \mathbf{y}^\top \mathbf{H}_1^{(i)} \mathbf{y} + \left(f_1^{(i)} - f_1^{(i+1)}\right) \mathbf{y}^\top \mathbf{H}_2^{(i)} \mathbf{y} - 2\beta \left(\mathbf{y}^\top \nabla f_2^{(i)}\right)^2.$$

Each term on the right-hand-side of the equation above is negative according to the facts listed above and therefore their sum, $\mathbf{y}^\top \mathbf{D}_i \mathbf{y} < 0$. Consequently, $\mathbf{D}_i$ is negative definite and thus non-singular. □

Theorem 5.2. *If the objective functions are convex and the decision vector $\mathbf{x}^{(i)}$ does not belong to the efficient set $\mathcal{X}$, the matrix $\mathbf{D}_i$ is non-singular if and only if the following condition holds:*

$$\left(\nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)} + 1\right)^2 \neq \left(\nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_2^{(i)}\right) \left(\nabla {f_1^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)}\right).$$

Proof. We introduce the following notations:

$$\mathbf{P}_i \coloneqq \left(\nabla f_1^{(i)}, \nabla f_2^{(i)}\right), \quad \mathbf{Q}_i \coloneqq \begin{pmatrix} \nabla {f_2^{(i)}}^\top \\ \nabla {f_1^{(i)}}^\top \end{pmatrix}.$$

Then $\mathbf{D}_i$ can be re-written as: $\mathbf{D}_i = \mathbf{B}_{ii} + \mathbf{P}_i \mathbf{Q}_i$. Note that $\mathbf{B}_{ii} = \left(f_2^{(i)} - f_2^{(i-1)}\right) \mathbf{H}_1^{(i)} + \left(f_1^{(i)} - f_1^{(i+1)}\right) \mathbf{H}_2^{(i)}$ is a combination of objective-wise Hessian matrices. Since both of the objective function are *convex*, $\mathbf{H}_1^{(i)}$ and $\mathbf{H}_2^{(i)}$ are positive definite. In addition, the coefficients $\left(f_2^{(i)} - f_2^{(i-1)}\right)$ and $\left(f_1^{(i)} - f_1^{(i+1)}\right)$ are negative in case of minimization. Consequently, $\mathbf{B}_{ii}$ is negative definite and thus *non-singular*. According to the matrix inversion lemma (Woodbury matrix identity), $\mathbf{C}_i$ is invertible if and only if $\mathbf{T}_i = \mathbf{I}_{2\times 2} + \mathbf{Q}_i \mathbf{B}_{ii}^{-1} \mathbf{P}_i$ is invertible:

$$\mathbf{T}_i = \begin{pmatrix} \nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)} + 1 & \nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_2^{(i)} \\ \nabla {f_1^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)} & \nabla {f_1^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_2^{(i)} + 1 \end{pmatrix}.$$

As matrix $\mathbf{T}_i$ is always of size 2×2, its determinant is much easier to compute than that of $\mathbf{C}_i$:

$$\det(\mathbf{T}_i) = \left(\nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)} + 1\right)^2 - \left(\nabla {f_2^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_2^{(i)}\right) \left(\nabla {f_1^{(i)}}^\top \mathbf{B}_{ii}^{-1} \nabla f_1^{(i)}\right).$$

The matrix $\mathbf{D}_i$ is non-singular if and only if the determinant above is non-zero. □

Note that the analysis of the convergence of a single point to the maximal hypervolume is structurally similar to the maximization of the hypervolume contribution of a single point in a set. The only difference is that the reference point is provided by coordinates of the neighboring non-dominated points in the objective space. Therefore, Theorem 5.2 also holds for maximizing the hypervolume contribution of a single point, as long as the neighboring points in the objective space are kept fixed.

Two-point system In this case, the hypervolume Hessian matrix looks as follows:

$$\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) = \begin{pmatrix} \mathbf{D}_1 & \tilde{\mathbf{A}}_1 \\ \tilde{\mathbf{A}}_1^\top & \mathbf{D}_2 \end{pmatrix}.$$

Again, we assume that two decision points $\mathbf{x}^{(1)}$ and $\mathbf{x}^{(2)}$ are mutually non-dominated. For this 2×2 block matrix, its invertibility is given by the following theorem.

Theorem 5.3. *If the diagonal block matrices $\mathbf{D}_1, \mathbf{D}_2$ are non-singular, then the hypervolume indicator Hessian matrix is non-singular if and only if:*

$$\left(\nabla {f_1^{(1)}}^\top \mathbf{D}_1^{-1} \nabla f_1^{(1)}\right)\left(\nabla {f_2^{(2)}}^\top \mathbf{D}_2^{-1} \nabla f_2^{(2)}\right) \neq 1.$$

Proof. See appendix B.1. □

Note that the non-singularity condition above does not hold even when both of the search points are in the efficient set. Such a situation can be depicted using a simple bi-objective optimization problem with $d = 1$ and $m = 2$:

$$f_1 = x^2, \quad f_2 = (1 - x)^2, \quad x \in \mathbb{R}.$$

To illustrate the singularity scenario, only two decision points are used, namely $x^{(1)}$ and $x^{(2)}$. For such a problem, the efficient set is the interval $[0, 1]$ and thus the box $[0, 1]^2$ is the region where $x^{(1)}$ and $x^{(2)}$ are both efficient. In Fig. 5.4, the set where the Hessian matrix is singular is depicted by the curved boundary of the shaded area. As shown in this example, in the two-point system, the Hessian matrix is not always invertible, even if all the search points belong to the Pareto efficient set. Moreover, in the shaded area (in $[0, 1]^2$), the hypervolume Hessian matrix is even *indefinite*, which would make it more difficult for Newton method to converge to the optimum.

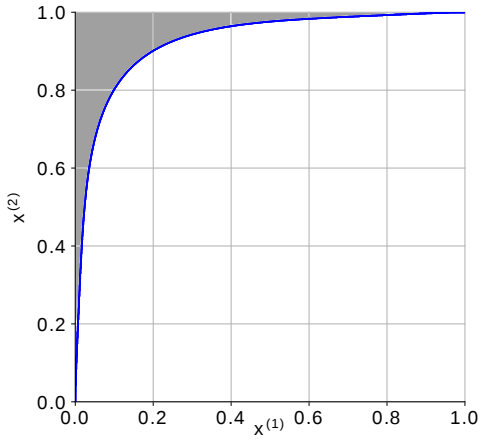


Figure 5.4: An example of two-point case in bi-objective optimization problem: The box $[0,1]^2$ is the region where $x^{(1)}, x^{(2)}$ are both Pareto efficient. The hypervolume Hessian matrix is singular when $(x^{(1)}, x^{(2)})$ is on the blue curve.

Remark. As illustrated in this example, the hypervolume Hessian matrix is only singular on a set of zero measure and therefore the applicability of the Newton method is not affected by the singularity because the probability of entering into such a set is zero. However, in general additional caution is needed if the set where the hypervolume Hessian is singular has nonzero measure, or in case there are regions where the Hessian is indefinite (which happens in our example).

5.3.2 Hypervolume Indicator Newton Method

After having stated the hypervolume gradient and Hessian matrix for a $\mu \cdot n$-vector $\mathbf{X}$ for a given MOP, we are now in the position to address the population based Newton method for hypervolume maximization. For this, we will first consider the unconstrained case and later on discuss first attempts to treat constrained problems. Given an unconstrained MOP and a population of μ points, the Newton step (or Newton function) is defined as follows:

$$\Delta\mathbf{X} := -\sigma \left[\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})\right]^{-1} \nabla \mathcal{H}_{\mathbf{F}}(\mathbf{X}). \tag{5.20}$$

In practice, a small step size $\sigma \in (0,1]$ is introduced to ensure the so-called Wolfe conditions (Wright and Nocedal, 1999) are met after each Newton step. As with the treatment in Section 5.2, the Newton step for decision point $\mathbf{x}^{(i)}$ is denoted by $\Delta\mathbf{X}^{(i)} \in \mathbb{R}^d, i = 1, \ldots, \mu$. Since the hypervolume indicator sub-gradient (Eq. (5.9))

for the *strictly dominated point* $\mathbf{x}^{(i)}$ of $\mathbf{X}$ is zero, its corresponding Newton direction is also zero. Consequently, such a point will remain stationary when applying the set-based Newton method. For the sake of simplicity, it is assumed that all the points contained in $\mathbf{X}$ are *mutually non-dominated* by each other. In case any point is dominated, it is always possible to apply the non-dominated-sorting approach proposed in Section 5.2.1. The **Hypervolume Newton Method** (HNM) is thus defined as

$$\mathbf{X}_0 \in \mathbb{R}^{\mu \cdot d}, \quad \mathbf{X}_{i+1} = \mathbf{X}_i + \Delta\mathbf{X}, \quad i = 0, 1, 2, \ldots. \tag{5.21}$$

The pseudo code for HNM is shown in Alg. 11. For the step size control, we suggest to choose the initial step size $\sigma_0 = 1$ and adjust it online using the backtracking line search. If automatic differentiation is used to evaluate the (exact) hypervolume indicator gradient and the Hessian matrix at each iteration, the cost for each Newton step is given by $5\mu + (4 + 6d)\mu$ function evaluations.

Algorithm 11 Hypervolume Newton Method

procedure HNM($\mathbf{X}, N, \varepsilon$) ▷ $\mathbf{X}$: initial approximation set, N: maximal iteration, ε: tolerance on the length of hypervolume gradient
  for $i = 1 \to N$ **do**
    Compute $\nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X})$, $\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})$
    Compute step size σ by backtracking line search
    $\mathbf{X} \leftarrow \mathbf{X} - \sigma \left[\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})\right]^{-1} \nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X})$ ▷ Newton step
    if $\|\nabla\mathcal{H}_{\mathbf{F}}(\mathbf{X})\| < \varepsilon$ **then**
      return X
    end if
  end for
  return X
end procedure

Example. In order to demonstrate the performance of the HNM we consider the following bi-objective optimization problem (also known as the MOP1 problem):

$$\begin{aligned} f_1 &= (x_1 - 1)^2 + (x_2 - 1)^2 \\ f_2 &= (x_1 + 1)^2 + (x_2 + 1)^2, \end{aligned} \tag{5.22}$$

where we choose as reference point $\mathbf{r} = (20, 20)^\top$.

(a) We choose $\mu = 5$ and the initial approximation set X as

$$\begin{aligned}&\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(5)}\right\} \\ &= \left\{\begin{pmatrix} 0 \\ -2 \end{pmatrix}, \begin{pmatrix} 0.5 \\ -1.5 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1.5 \\ -0.5 \end{pmatrix}, \begin{pmatrix} 2 \\ 2 \end{pmatrix}\right\}.\end{aligned} \tag{5.23}$$

Fig. 5.5 shows the performance of HNM both in the decision and objective space. As it can be seen, the iterations quickly approach the optimal solution for $\mu = 5$ and a given reference point. This observation is confirmed in Tab. 5.3a, where the hypervolume values, the norm of the gradients, and the error measured in terms of the Hausdorff distance (Schütze et al., 2012) of $\mathbf{X}$ and the optimal solution are displayed for each iteration. The values indicate quadratic convergence.

(b) Next, we consider the same setting but using a different initial approximation set:

$$\begin{aligned}&\left\{\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \mathbf{x}^{(3)}, \mathbf{x}^{(4)}, \mathbf{x}^{(5)}\right\} \\ &= \left\{\begin{pmatrix} -0.12 \\ -1.57 \end{pmatrix}, \begin{pmatrix} 0.48 \\ -1.24 \end{pmatrix}, \begin{pmatrix} 1 \\ -1 \end{pmatrix}, \begin{pmatrix} 1.32 \\ -0.26 \end{pmatrix}, \begin{pmatrix} 1.89 \\ -0.11 \end{pmatrix}\right\}.\end{aligned} \tag{5.24}$$

Fig. 5.6 and Tab. 5.3b show the numerical results of HNM. In step 2, $\mathbf{x}^{(1)}$ gets dominated by $\mathbf{x}^{(3)}$. The iteration thus continues with the remaining 4 point excluding $\mathbf{x}^{(1)}$. HNM converges (again quadratically) toward the optimal hypervolume population, albeit for population size $\mu = 4$.

5.4 Summary

The multi-objective optimization problem is investigated in this chapter. The general goal here is to generalize the first- and second-order optimization method from the single-objective scenario to the multi-objective scenario. In order to achieve this goal, the notion on the gradient is extended to the multi-objective problem: the partial derivatives of the hypervolume indicator is taken w.r.t. to the decision points and the so-called hypervolume indicator gradient is defined as the concatenation of such partial derivatives at all decision points. Based on this extension, a gradient ascent algorithm, called hypervolume indicator gradient ascent multi-objective optimization (HIGA-MO) is proposed to maximize the hypervolume in the steepest manner. Following this treatment, the second-order

Table 5.3: On the MOP1 Problem (Eq. (5.22)), Alg. 11 is executed for seven iterations and the following values are listed iteratively: the hypervolume value, the size of the approximation set μ, the error in the Hausdorff distance to the optimal approximation set and the norm of the hypervolume indicator gradient.

(a) Using Eq. (5.23) for $\mathbf{X}_0$ (cf. Fig. 5.5).

Iter	μ	$\mathcal{H}_{\mathbf{F}}$	Error	$\|\nabla\mathcal{H}_{\mathbf{F}}\|$
0	5	306.5000	76.5695	48.8262
1	5	369.5622	13.5072	21.0628
2	5	379.0652	4.0042	13.9973
3	5	382.7340	0.3355	2.8800
4	5	383.0680	0.0015	0.2000
5	5	383.0695	0.0000	0.0013
6	5	383.0695	0.0000	0.0000
7	5	383.0695	0.0000	0.0000

(b) Using Eq. (5.24) for $\mathbf{X}_0$ (cf. Fig. 5.6)

Iter	μ	$\mathcal{H}_{\mathbf{F}}$	Error	$\|\nabla\mathcal{H}_{\mathbf{F}}\|$
0	5	321.5483	61.5212	52.9006
1	5	376.6161	6.4534	14.6855
2	4	373.5446	9.5249	2.0132
3	4	380.6982	2.3713	0.1104
4	4	380.6985	2.3710	0.0002
5	4	380.6985	2.3710	0.0000
6	4	380.6985	2.3710	0.0000
7	4	380.6985	2.3710	0.0000

derivatives of the hypervolume indicator is formulated. In addition, we investigate the condition on which the resulting Hessian matrix is regular (non-singular). This is an essential prerequisite for using the hypervolume indicator Hessian matrix correctly. In addition, to investigate the proposed algorithms, a bi-objective problem class, called Mixed-Peak problems are introduced. This problem class allows for directly controlling the problem difficulty of its instance.

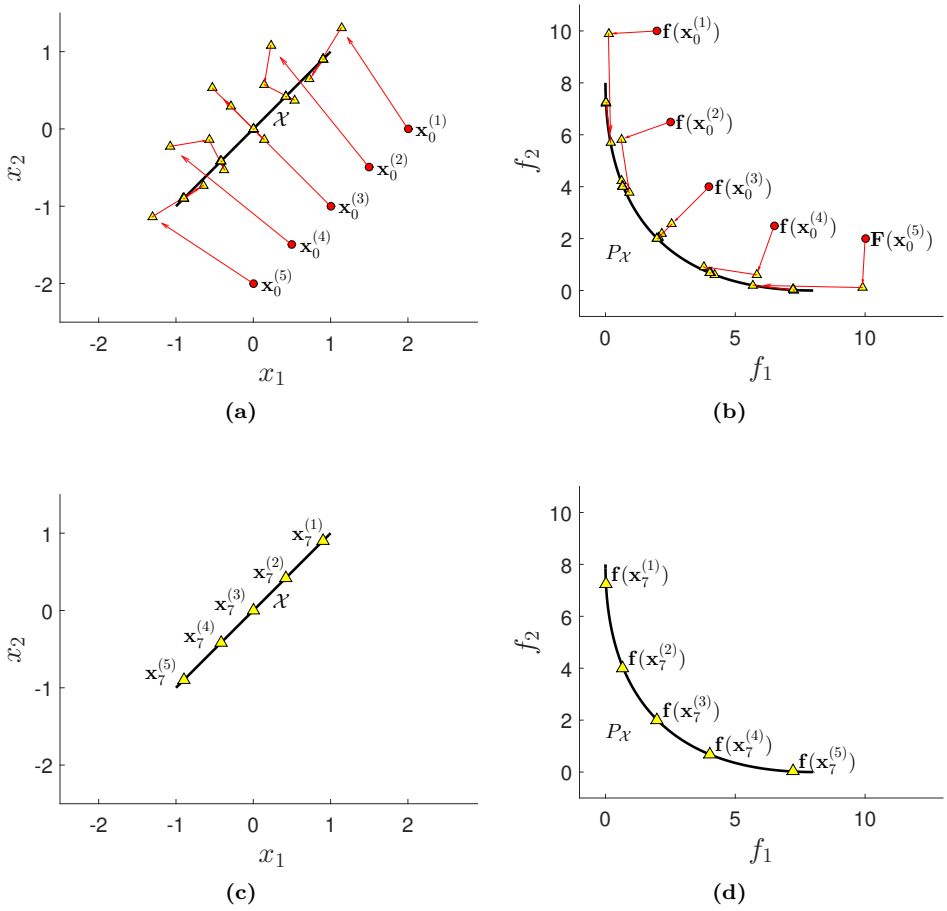


Figure 5.5: Numerical result of HNM on the MOP1 problem using Eq. (5.23) as the initial approximation set. **Top**: the iterations in decision and objective space. **Bottom**: the optimal solution and its image for $\mu = 5$ and $\mathbf{r} = (20, 20)^\top$.

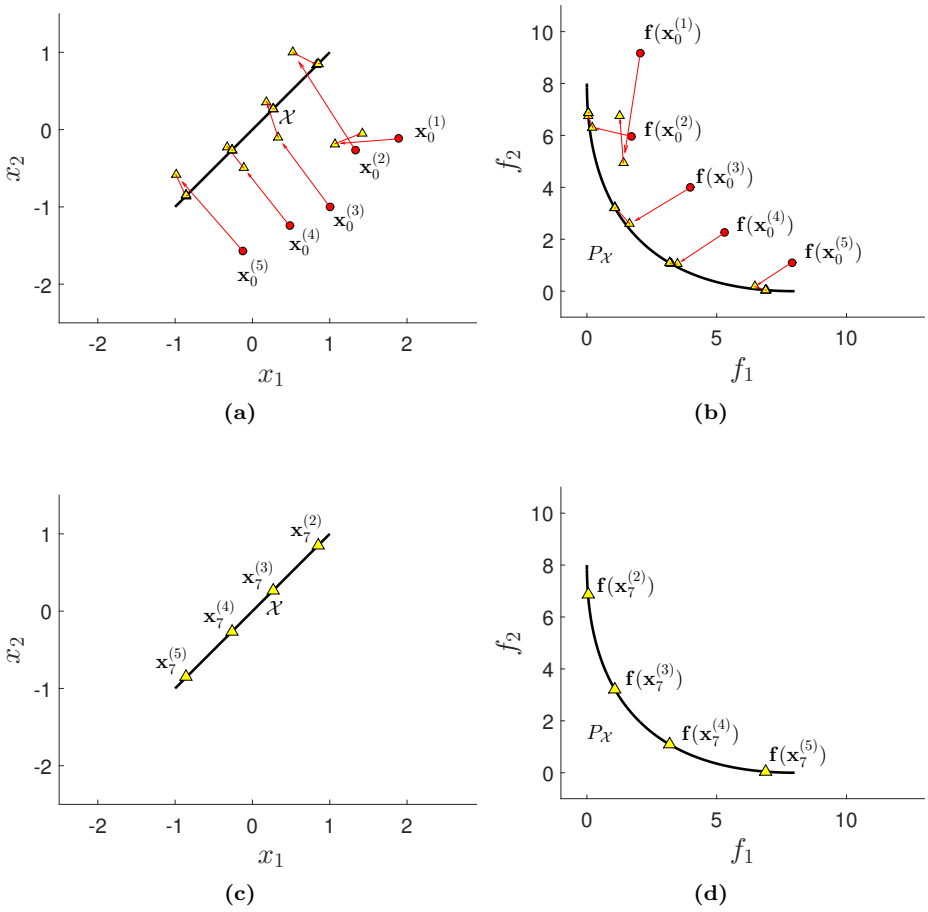


Figure 5.6: Numerical result of HNM on the MOP1 problem using Eq. (5.24) as the initial approximation set. **Top**: the iterations in decision and objective space. **Bottom**: the optimal solution and its image for $\mu = 4$ and $\mathbf{r} = (20, 20)^\top$.

CHAPTER 6

Conclusion

In this thesis, several important aspects of the stochastic optimization are investigated in depth. In **Chapter 1**, the discussion begins with the fundamental definitions on single- and multi-objective optimality (Section 1.1). It is attempted to give rigorous definitions on the procedure of *stochastic optimization*. In addition, a mathematical prerequisite, called *Matrix Calculus* is also described. The research question in this chapter is:

> *Can we generalize the notion of local optimality in the single objective case to multi-objective problems?*

We give a generalization by revisiting the local efficient points and defining the local efficient set (Def. 1.8). Basically, a local efficient set should consist of local efficient points that collectively forms a connected component.

Chapter 2 discusses several issues on generating *stochastic variations* in $\mathbb{R}^d$. Herein, the major concerns are:

> *When using a small sample size in stochastic variation, what is the drawback of the so-called sampling error? How to mitigate such drawbacks? Is it possible to justify our approach both theoretically and empirically?*

Random sampling from the multivariate Gaussian distribution is the most commonly applied stochastic variation. It suffers from the well-known sampling error when the sample size is small. This issue is re-visited in Chapter 2 and an improved sampling method, called *mirrored orthogonal sampling* is proposed to relax the drawback. Both theoretical analysis and empirical study are conducted on the proposed sampling method. After plugged into evolution strategies, this sampling

method is, in addition, tested on the well-known BBOB benchmark. The second research question is:

> *The well-known Efficient Global Optimization (EGO) algorithm differs largely from the canonical stochastic optimization algorithms (e.g., evolutionary algorithms). Is it possible to find a unified treatment to incorporate EGO into stochastic optimization algorithms?*

In EGO, new candidate locations are generated by optimizing the so-called infill criterion. This procedure is considered as a special stochastic variation method, called *mutation by optimization*.

Chapter 3 resorts to a different topic, the surrogate modeling. The Kriging/Gaussian Process Regression (GPR) is discussed in depth, where an unified treatment is presented from three different perspectives: 1) the theory on the best linear predictor, 2) reproducing kernel Hilbert Space and 3) Bayesian inference. Our major theoretical concerns are:

> *The most prominent feature of Kriging/GPR is the prediction uncertainty. What does this uncertainty truly characterize? How does the uncertainty related to the inaccuracy in the function approximation?*

Our consideration starts with a clear specification of the modeling assumption: the target function f is a *sample function* of a prescribed stochastic process Y. An optimal linear predictor $\widehat{Y}$ is derived for Y based on the partial information (finite locations) on Y. Then the function approximation is obtained by taking a realization $\hat{f}$ from $\widehat{Y}$. Based on this treatment, it is clearly seen that the Kriging uncertainty is the MSE of the predictor: $s^2 = \mathbb{E}\{Y - \widehat{Y}\}^2$. Moreover, the accuracy of function approximation is related to s^2 (theorem 3.1). When it comes to the application of the Kriging/GPR method, the following question is of our interest:

> *The Kriging/GPR method suffers from a cubic time complexity when dealing with large data sets. It is then crucial to propose methods that reduce the time complexity.*

In the reminder of this chapter, a novel algorithmic framework, called *Cluster Kriging* is proposed to tackle this issue. Cluster Kriging is tested on some selected functions and data sets, exhibiting an acceleration of the modeling speed as well

as an improved modeling precision. Despite the successful experiments on Cluster Kriging, there is still a lack of theoretical treatment (using one of the three aforementioned perspectives) to illustrate its asymptotic convergence property to the target function. This part should be investigated in future work.

Chapter 4 focuses on the following problem: given a thorough study on the surrogate model, naturally the next step is to investigate how the surrogate model can be utilized in a reasonable manner. The utility of each location on a surrogate model is quantified by a well-defined function, called *infill criterion*. The research question in this chapter are:

> *Since some infill criteria are more explorative and some others are more exploitative, does it make sense to look for the optimal trade-off among infill criteria? Can we design an infill criterion that controls this trade-off explicitly and smoothly?*

Prior to the investigation on the exploration-exploitation trade-off, most of the improvement-based infill criteria is discussed. Among them, two infill criteria, Probability of Improvement (PI) and Expected Improvement (EI) are selected for the investigation, where PI and EI are treated as a bi-objective optimization task. This task is solved using a gradient-based multi-objective optimization algorithm that is proposed later in this thesis (Section 5.2). In addition, to explicitly control such trade-offs, a novel infill criterion, *Moment-Generate Function of Improvement* (MGFI) is proposed as the an extension of all improvement-based criteria. It is equipped with an continuous control parameter t (called temperature), which is enable to scale the exploration behavior smoothly. The other research question discussed is:

> *How to parallelize the EGO algorithm, namely proposing multiple candidate locations in each iteration?*

The goal of parallelizing infill criteria is formulated in the first place. Then several well-known parallelization methods are discussed: multi-point Expected Improvement and Multi-Instance of infill criteria. In addition, we proposed two nee approached: 1) packing several infill criteria as a multi-objective function and thus multiple point could be obtained via the multi-objective optimization. 2) when using a single infill criteria, multiple distinct locations can be found by applying the so-called *niching* technique. Although the proposed parallelization methods

seem plausible, some of them remains untested as of the time of writing. It is important to perform systematic experiments on them in the future.

Chapter 5 discusses the numerical multi-objective optimization (MOO). The demands on this topic originate from many numerical multi-objective tasks that arise in the study of stochastic optimization, e.g., the multi-objective treatment of infill criteria in Chapter 4. The research question in this chapter is:

> *How to design a deterministic optimization algorithm for multi-objective problems using either the gradient or the Hessian matrix of the objective function, such that both the convergence rate to the Pareto front and the distribution of non-dominated points are considered simultaneously?*

The contribution in this chapter are three-fold: firstly, we mathematically analyze the so-called *Mixed-Peak* bi-objective test problem. Secondly, the gradient field and Hessian matrix of the hypervolume indicator are derived and studied in depth. Thirdly, two novel numerical MOO algorithms, namely the hypervolume-based first- (gradient) and second-order (Hessian) methods are proposed and tested. Differentiating the hypervolume indicator resolves our research question due to the fact that maximizing the hypervolume leads to the convergence to the Pareto front and a set of well-distributed efficient points. In parallel to the algorithmic development, we also investigate the condition on which the hypervolume indicator Hessian would be singular. The singularity would make the Hessian inapplicable for the optimization. The preliminary proof/results show that the subset (of the search space S) where the Hessian is singular is of measure zero in S, meaning that it is generally safe to apply the Hypervolume Hessian method. However, this theoretical study is conducted in an idealized setting: 1) two points in the approximation set and 2) the objective function is set to the simple sphere model. As the next step, it is necessary to investigate the singularity condition furthermore on more general functions with more approximation points.

Many research directions/proposals can be given by combining the techniques in multiple chapters. For example, the novel infill criteria MGFI (Section 4.3) can be combined with Kriging-based multi-objective optimization algorithm, e.g., SMS-EGO (Ponweiser et al., 2008), replacing the EI criteria. In addition, as the surrogate model is required in SMS-EGO, it is also recommended to use the

proposed Cluster Kriging (Section 3.2) with this algorithm, aiming at improving the algorithm running time and convergence rate.

In addition, the theoretical study in this thesis can also be continued. In Chapter 2, although the mirrored orthogonal sampling is analyzed in depth, the distribution of the proposed *uniform random orthogonal vectors* (Def. 2.1) is yet to discover. In Chapter 3, despite the successful experimental result of Cluster Kriging, the theoretical aspects are not treated. Thus, it is quite important to check its modeling ability from the perspective of either Bayesian inference or RKHS. In Chapter 4, when combining the EGO algorithm and the proposed MGFI criteria, the convergence rate of the resulting optimizer is theoretically unknown, although the faster empirical convergence is validated from benchmarking.

APPENDIX A

Gaussian Distribution

Assume the probability space $(\Omega, \mathscr{F}, \mathbb{P})$ and a measurable space $(\mathbb{R}, \mathscr{B})$, where $\mathscr{B}$ is the Borel algebra on $\mathbb{R}$. A random variable $X : \Omega \to \mathbb{R}$ is said to be normally distributed if and only if its probability distribution $\mathbb{P}_X : \mathscr{B} \to [0, 1]$, defined as a push-forward measure, $\forall B \in \mathscr{B},\ \mathbb{P}_X(B) \coloneqq \mathbb{P}(X^{-1}[B])$, admits the following form:

$$\mathbb{P}_X(B) = \int_B \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right) \mathrm{d}\lambda, \tag{A.1}$$

where λ is the Lebesgue measure on $\mathbb{R}$ and m, σ^2 are the mean and variance of X, respectively. We typically use the notation $X \sim \mathcal{N}(m, \sigma^2)$. This distribution $\mathbb{P}_X$ is called *Gaussian measure* and the notation $\mathcal{G}_{m,\sigma^2}$ is assigned to it. The *cumulative distribution function* (c.d.f.) of X is

$$\Phi_{m,\sigma^2}(x) = \mathcal{G}_{m,\sigma^2}\left(\{X \in \mathbb{R} \ : \ X \leq x\}\right).$$

In addition, the *probability density function* (p.d.f.) of X is the Radon-Nikodym derivative of $\mathcal{G}_{m,\sigma^2}$ w.r.t. λ:

$$\phi_{m,\sigma^2}(x) = \frac{\mathrm{d}\mathcal{G}_{m,\sigma^2}}{\mathrm{d}\lambda} = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-m)^2}{2\sigma^2}\right), \tag{A.2}$$

that is, by definition, the integrand in Eq. (A.1). In the multivariate case, consider the measurable space $(\mathbb{R}^n, \mathscr{B}^n)$ where $\mathscr{B}^n$ is the Borel algebra on $\mathbb{R}^n$. A random vector $\mathbf{x} = (X_1, X_2, \ldots, X_n)^\top : \Omega \to \mathbb{R}^n$ is said to follow the multivariate Gaussian distribution, if and only if any linear combination $\mathbf{c}^\top \mathbf{x},\ \mathbf{c} \in \mathbb{R}^n$ admits the distribution as in Eq. (A.1). In addition, the distribution of $\mathbf{x}$ is

$$\forall B \in \mathscr{B}^n,\ \mathcal{G}^n_{\mathbf{m},\mathbf{K}}(B) = \int_B (2\pi)^{-\frac{n}{2}} \det(\mathbf{K})^{\frac{1}{2}} \exp\left(-\frac{1}{2}(\mathbf{y}-\mathbf{m})^\top \mathbf{K}^{-1}(\mathbf{y}-\mathbf{m})\right) \mathrm{d}\lambda^n,$$

where λ^n is the n-dimensional Lebesgue measure on $(\mathbb{R}^n, \mathscr{B}^n)$ and $\mathbf{m}, \mathbf{K}$ are the mean and covariance matrix of $\mathbf{x}$. As with the univariate case, we shall take the notation $\mathbf{x} \sim \boldsymbol{\mathcal{N}}(\mathbf{m}, \mathbf{K})$ and its cumulative distribution function is,

$$\Phi^n_{\mathbf{m},\mathbf{K}}(\boldsymbol{\zeta}) = \mathcal{G}^n_{\mathbf{m},\mathbf{K}}\left(\{\mathbf{x} \in \mathbb{R}^n \ : \ \mathbf{x} \leq \boldsymbol{\zeta}\}\right).$$

Given an arbitrary partition on $\mathbf{x} = (\mathbf{x}_1^\top, \mathbf{x}_2^\top)^\top$, in which $\mathbf{x}_1$, $\mathbf{x}_2$ have n_1 and n_2 components, respectively. The distribution of $\mathbf{x}$ can be re-written as

$$\begin{bmatrix}\mathbf{x}_1 \\ \mathbf{x}_2\end{bmatrix} \sim \boldsymbol{\mathcal{N}}\left(\begin{bmatrix}\mathbf{m}_1 \\ \mathbf{m}_2\end{bmatrix}, \begin{bmatrix}\mathbf{K}_{11} & \mathbf{K}_{12} \\ \mathbf{K}_{21} & \mathbf{K}_{22}\end{bmatrix}\right),$$

where all sub-mean vectors and sub-covariance matrices are obtained by applying the same partition on $\mathbf{m}$ and $\mathbf{K}$. The *marginal distribution* of $\mathbf{x}_1$ is Gaussian:

$$\mathbf{x}_1 \sim \boldsymbol{\mathcal{N}}\left(\mathbf{m}_1, \mathbf{K}_{11}\right). \tag{A.3}$$

The result holds for $\mathbf{x}_2$ in the same manner. In addition, the *conditional distribution* of $\mathbf{x}_1$ on $\mathbf{x}_2 = \mathbf{v}$ is Gaussian (Tong, 2012):

$$\mathbf{x}_1 \mid \mathbf{x}_2 = \mathbf{v} \sim \boldsymbol{\mathcal{N}}\left(\mathbf{m}_1 + \mathbf{K}_{12}\mathbf{K}_{22}^{-1}(\mathbf{v} - \mathbf{m}_2),\ \mathbf{K}_{11} - \mathbf{K}_{12}\mathbf{K}_{22}^{-1}\mathbf{K}_{21}\right). \tag{A.4}$$

Often, the value of Gaussian random variables is restricted:

$$X \sim \mathcal{N}(m, \sigma^2), \quad X_R = \max\{0, X\}.$$

The random variable X_R is known as the *Rectified Gaussian* and its distribution shall be denoted as $\mathcal{N}_R(m, \sigma^2)$. Note that the rectification "concentrates" all the probability measure in $(-\infty, 0)$ to the rectification point 0, leading to an infinite impulse at this point. Thus, the p.d.f. of X_R is:

$$p_{X_R}(x) = \Phi_{m,\sigma^2}(0)\delta(x) + \phi_{m,\sigma^2}(x)H(x), \tag{A.5}$$

where δ is the Dirac delta (distribution)[1] and H is the step function:

$$\delta(x) = \begin{cases}\infty & x = 0, \\ 0 & x \neq 0.\end{cases}, \quad H(x) = \begin{cases}0 & x \leq 0, \\ 1 & x > 0.\end{cases}$$

The rectification is sometimes confused with the so-called *truncated Gaussian*, which is the distribution of a Gaussian variable $X \sim \mathcal{N}(m, \sigma^2)$ conditioning on an interval $(a, b) \subset \mathbb{R}$:

$$p(x \mid a < X < b) = \frac{\phi_{m,\sigma^2}(x)}{\Phi_{m,\sigma^2}(b) - \Phi_{m,\sigma^2}(a)}.$$

[1] Formally, the Dirac delta should be defined either as a distribution or measure. We use the heuristic characterization here for the sake of simplicity.

APPENDIX B

Proof

B.1 Theorem 5.3

Proof. Let us define $\mathbf{a} \coloneqq -\nabla f_1^{(2)}$ and $\mathbf{b} \coloneqq \nabla f_2^{(1)}$, such that $\tilde{\mathbf{A}}_1 = \mathbf{b}\mathbf{a}^\top$ and

$$\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X}) = \begin{pmatrix} \mathbf{D}_1 & \mathbf{b}\mathbf{a}^\top \\ \mathbf{a}\mathbf{b}^\top & \mathbf{D}_2 \end{pmatrix}.$$

For two block matrices, their column vectors are denoted as: $\mathbf{D}_1 = (\mathbf{d}_1, \ldots, \mathbf{d}_n)$ and $\mathbf{D}_2 = (\mathbf{d}'_1, \ldots, \mathbf{d}'_n)$. The hypervolume Hessian is of size $2n \times 2n$ and its determinant can be simplified using the Laplace expansion along **the first** n **rows** of the $\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})$. To achieve this, n distinct columns need to be selected out of $2n$ rows. Let S be the set of the n-element subsets of $\{1, 2, \ldots, 2n\}$:

$$S = \{\{1, 2, \ldots, n\}, \{1, 2, \ldots, n-1, n+1\}, \ldots\}$$

For every $L \in S$, we define its complement $L' \coloneqq \{1, 2, \ldots, 2n\} \setminus L$. Note that a permutation is defined on $\{1, 2, \ldots, 2n\}$, by appending L' to L: $\{L, L'\}$ and we shall use $N(L)$ to denote the number of inversions in $\{L, L'\}$. According to the Laplace expansion, such a determinant can be expressed as:

$$\det\left(\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})\right) = \sum_{L \in S} (-1)^{N(L)} b_L c_{L'},$$

where b_L is the cofactor of the hypervolume Hessian, which is the determinant of the minor matrix obtained by keeping the first n rows and n columns given in L. Similarly, $c_{L'}$ is the *complementary* cofactor of b_L, obtained by removing the first n rows and n columns given in L. For example, if $L = \{1, 2, \ldots, n\}$, then $b_L = \det(\mathbf{D}_1)$ and $c_{L'} = \det(\mathbf{D}_2)$. In particular, when L contains two or more elements from $\{2n+1, 2n+2, \ldots, 2n\}$, meaning that at least two columns from

$\mathbf{b}\mathbf{a}^\top$ are chosen to compute b_L, it is obvious that the cofactor b_L is zeros because all the columns from $\mathbf{b}\mathbf{a}^\top$ are linear dependent. Using this observation, the expansion can be simplified:

$$\begin{aligned}
\det\left(\nabla^2\mathcal{H}_{\mathbf{F}}(\mathbf{X})\right) &= \underbrace{\det\left(\mathbf{D}_1\right)\det\left(\mathbf{D}_2\right)}_{L=\{1,2,\ldots,n\}} \\
&+ \underbrace{(-1)^1\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},a_1\mathbf{b})\right)\det\left((b_n\mathbf{a},\mathbf{d}'_2\ldots,\mathbf{d}'_n)\right)}_{L=\{1,2,\ldots,n-1,n+1\}} \\
&+ \underbrace{(-1)^2\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},a_2\mathbf{b})\right)\det\left((b_n\mathbf{a},\mathbf{d}'_1,\mathbf{d}'_3,\ldots,\mathbf{d}'_n)\right)}_{L=\{1,2,\ldots,n-1,n+2\}}+\cdots \\
&+ \underbrace{(-1)^n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},a_2\mathbf{b})\right)\det\left((b_n\mathbf{a},\mathbf{d}'_1,\mathbf{d}'_2,\ldots,\mathbf{d}'_{n-1})\right)}_{L=\{1,2,\ldots,n-1,2n\}}+\cdots
\end{aligned}$$

There are n terms shown in the equation above, resulting from choosing the first $n-1$ columns and one column from $\{2n+1,2n+2,\ldots,2n\}$. Those terms can also be simplified:

$$\begin{aligned}
&(-1)^1 a_1 b_n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},\mathbf{b})\right)\det\left((\mathbf{a},\mathbf{d}'_2\ldots,\mathbf{d}'_n)\right) \\
&+(-1)^3 a_2 b_n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},\mathbf{b})\right)\det\left((\mathbf{d}'_1,\mathbf{a},\mathbf{d}'_3\ldots,\mathbf{d}'_n)\right)+\cdots \\
&+(-1)^{2i-1} a_i b_n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},\mathbf{b})\right)\underbrace{\det\left((\mathbf{d}'_1,\ldots,\mathbf{d}'_{i-1},\mathbf{a},\mathbf{d}'_{i+1},\ldots,\mathbf{d}'_n)\right)}_{\text{move }\mathbf{a}\text{ to the }i\text{-th column}} \\
&=-b_n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},\mathbf{b})\right)\det\left(\mathbf{D}_2\right)\sum_{i=1}^{n}a_i\frac{\det\left((\mathbf{d}'_1,\ldots,\mathbf{d}'_{i-1},\mathbf{a},\mathbf{d}'_{i+1},\ldots,\mathbf{d}'_n)\right)}{\det\left((\mathbf{d}'_1,\mathbf{d}'_2,\ldots,\mathbf{d}'_n)\right)} \qquad \text{(B.1)} \\
&=-b_n\det\left((\mathbf{d}_1,\ldots,\mathbf{d}_{n-1},\mathbf{b})\right)\det\left(\mathbf{D}_2\right)\mathbf{a}^\top\mathbf{D}_2^{-1}\mathbf{a}
\end{aligned}$$

Note that the last step above is according to Cramer's rule for the equation $\mathbf{D}_2\mathbf{x}=\mathbf{a}$ ($\mathbf{D}_1$ and $\mathbf{D}_2$ are assumed to be nonsingular):

$$x_i=\frac{\det\left((\mathbf{d}'_1,\ldots,\mathbf{d}'_{i-1},\mathbf{a},\mathbf{d}'_{i+1},\ldots,\mathbf{d}'_n)\right)}{\det\left((\mathbf{d}'_1,\mathbf{d}'_2,\ldots,\mathbf{d}'_n)\right)}$$

In principle, the same simplification here can be applied to other terms in the hypervolume Hessian determinant:

$$
\begin{aligned}
&\det\left(\nabla^2 \mathcal{H}_{\mathbf{F}}(\mathbf{X})\right) = \det\left(\mathbf{D}_1\right) \det\left(\mathbf{D}_2\right) \\
&- \underbrace{b_n \det\left((\mathbf{d}_1, \ldots, \mathbf{d}_{n-1}, \mathbf{b})\right) \det\left(\mathbf{D}_2\right) \mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a}}_{\text{drop column } n \text{ from } \mathbf{D}_1} \\
&- \underbrace{b_{n-1} \det\left((\mathbf{d}_1, \ldots, \mathbf{d}_{n-2}, \mathbf{b}, \mathbf{d}_n)\right) \det\left(\mathbf{D}_2\right) \mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a}}_{\text{drop column } n-1 \text{ from } \mathbf{D}_1} - \cdots \\
&- \underbrace{b_1 \det\left((\mathbf{b}, \mathbf{d}_2, \ldots, \mathbf{d}_n)\right) \det\left(\mathbf{D}_2\right) \mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a}}_{\text{drop column 1 from } \mathbf{D}_1} \\
&= \det\left(\mathbf{D}_1\right) \det\left(\mathbf{D}_2\right) \left[1 - \mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a} \sum_{i=1}^{n} b_i \frac{\det\left((\mathbf{d}_1, \ldots, \mathbf{d}_{i-1}, \mathbf{b}, \mathbf{d}_{i+1}, \ldots, \mathbf{d}_n)\right)}{\det\left((\mathbf{d}_1, \mathbf{d}_2, \ldots, \mathbf{d}_n)\right)}\right] \\
&= \left(1 - \left(\mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a}\right) \left(\mathbf{b}^\top \mathbf{D}_1^{-1} \mathbf{b}\right)\right) \det\left(\mathbf{D}_1\right) \det\left(\mathbf{D}_2\right)
\end{aligned}
$$

Again, in the last step above the same argument as in Eq. (B.1) is applied. Because matrices $\mathbf{D}_1$ and $\mathbf{D}_2$ are nonsingular, the hypervolume Hessian matrix is nonsingular as long as $1 - \left(\mathbf{a}^\top \mathbf{D}_2^{-1} \mathbf{a}\right) \left(\mathbf{b}^\top \mathbf{D}_1^{-1} \mathbf{b}\right)$ is not zero. □

Bibliography

Ababou, R., A. C. Bagtzoglou, and E. F. Wood (1994). On the Condition Number of Covariance Matrices in Kriging, Estimation, and Simulation of Random Fields. *Mathematical Geology 26*(1), 99–133.

Agrawal, R. B., K. Deb, and R. B. Agrawal (1995). Simulated Binary Crossover for Continuous Search Space. *Complex systems 9*(2), 115–148.

Andrianakis, I. and P. G. Challenor (2012). The Effect of the Nugget on Gaussian Process Emulators of Computer Models. *Computational Statistics & Data Analysis 56*(12), 4215–4228.

Aronszajn, N. (1950). Theory of Reproducing Kernels. *Transactions of the American mathematical society 68*(3), 337–404.

Auer, P. (2002). Using Confidence Bounds for Exploitation-exploration Trade-offs. *Journal of Machine Learning Research 3*(Nov), 397–422.

Auer, P., N. Cesa-Bianchi, and P. Fischer (2002). Finite-time Analysis of the Multiarmed Bandit Problem. *Machine learning 47*(2-3), 235–256.

Auger, A., D. Brockhoff, and N. Hansen (2010). Mirrored Variants of the $(1, 2)$-CMA-ES Compared on the Noisy BBOB-2010 Testbed. In *Proceedings of the 12th annual conference companion on Genetic and evolutionary computation*, GECCO '10, New York, NY, USA, pp. 1575–1582. ACM.

Auger, A., D. Brockhoff, and N. Hansen (2011a). Analyzing the Impact of Mirrored Sampling and Sequential Selection in Elitist Evolution Strategies. In H.-G. Beyer and W. B. Langdon (Eds.), *Proceedings of the 11th workshop proceedings on Foundations of genetic algorithms*, FOGA '11, ACM, New York, pp. 127–138.

Auger, A., D. Brockhoff, and N. Hansen (2011b). Mirrored Sampling in Evolution Strategies with Weighted Recombination. In *Proceedings of the 13th Annual*

Conference on Genetic and Evolutionary Computation, GECCO '11, New York, NY, USA, pp. 861–868. ACM.

Auger, A. and N. Hansen (2011). Theory of Evolution Strategies: A New Perspective. *Theory of Randomized Search Heuristics: Foundations and Recent Developments 1*, 289–325.

Bache, K. and M. Lichman (2013). UCI Machine Learning Repository. `http://archive.ics.uci.edu/ml`.

Bäck, T. (1995). Order Statistics for Convergence Velocity Analysis of Simplified Evolutionary Algorithms. Volume 3 of *Foundations of Genetic Algorithms*, pp. 91 – 102. Elsevier.

Bäck, T. (1996). *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford, UK: Oxford University Press.

Bäck, T., C. Foussette, and P. Krause (2013). *Contemporary Evolution Strategies* (1 ed.). Natural Computing Series. Springer-Verlag Berlin Heidelberg.

Bäck, T. and H.-P. Schwefel (1993, March). An Overview of Evolutionary Algorithms for Parameter Optimization. *Evol. Comput. 1*(1), 1–23.

Bartz-Beielstein, T., C. W. G. Lasarczyk, and M. Preuss (2005, Sept). Sequential Parameter Optimization. In *2005 IEEE Congress on Evolutionary Computation*, Volume 1, pp. 773–780 Vol.1.

Beume, N., B. Naujoks, and M. Emmerich (2007). SMS-EMOA Multiobjective Selection Based on Dominated Hypervolume. *European Journal of Operational Research 181*(3), 1653–1669.

Beyer, H.-G. (1993, June). Toward a Theory of 'Evolution Strategies': Some Asymptotical Results from the $(1,^+\lambda)$-Theory. *Evolution Computation 1*(2), 165–188.

Beyer, H.-G. (2013). *The Theory of Evolution Strategies*. Springer Science & Business Media.

Bischl, B., S. Wessing, N. Bauer, K. Friedrichs, and C. Weihs (2014). MOI-MBO: Multiobjective Infill for Parallel Model-Based Optimization. In *International Conference on Learning and Intelligent Optimization*, pp. 173–186. Springer.

Björck, A. (1994, Jan). Numerics of Gram-Schmidt Orthogonalization. *Linear Algebra and its Applications 197-198*, 297–316.

Bonabeau, E., M. Dorigo, and G. Theraulaz (1999). *Swarm Intelligence: from Natural to Artificial Systems*. Number 1. Oxford university press.

Bonnans, J.-F., J. C. Gilbert, C. Lemaréchal, and C. A. Sagastizábal (2006). *Numerical Optimization: Theoretical and Practical Aspects*. Springer Science & Business Media.

Boyd, S. and L. Vandenberghe (2004). *Convex Optimization*. New York, NY, USA: Cambridge University Press.

Breiman, L. (1996). Bagging Predictors. *Machine Learning 24*(2), 123–140.

Breiman, L., J. Friedman, C. J. Stone, and R. A. Olshen (1984). *Classification and Regression Trees*. CRC press.

Brockhoff, D., A. Auger, N. Hansen, D. V. Arnold, and T. Hohm (2010). Mirrored Sampling and Sequential Selection for Evolution Strategies. In R. Schaefer, C. Cotta, J. Kolodziej, and G. Rudolph (Eds.), *Proceedings of the 11th international conference on Parallel problem solving from nature: Part I*, PPSN'10, Springer-Verlag, Berlin, pp. 11–21.

Buhmann, M. D. (2003). *Radial Basis Functions: Theory and Implementations*, Volume 12. Cambridge university press.

Chalupka, K., C. K. I. Williams, and I. Murray (2013, February). A Framework for Evaluating Approximation Methods for Gaussian Process Regression. *J. Mach. Learn. Res. 14*(1), 333–350.

Chen, T. and J. Ren (2009, March). Bagging for Gaussian Process Regression. *Neurocomput. 72*(7-9), 1605–1610.

Chevalier, C. and D. Ginsbourger (2013). Fast Computation of the Multi-points Expected Improvement with Applications in Batch Selection. In G. Nicosia and P. Pardalos (Eds.), *Learning and Intelligent Optimization*, Berlin, Heidelberg, pp. 59–69. Springer Berlin Heidelberg.

Cressie, N. (1990). The Origins of Kriging. *Mathematical geology 22*(3), 239–252.

Cressie, N. (2015). *Statistics for spatial data*. John Wiley & Sons.

Csató, L. and M. Opper (2002). Sparse On-line Gaussian Processes. *Neural computation 14*(3), 641–668.

De Jong, K. A. (1975). *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. Ph. D. thesis, Ann Arbor, MI, USA. AAI7609381.

Deb, K., S. Agrawal, A. Pratap, and T. Meyarivan (2000). A Fast Elitist Non-dominated Sorting Genetic Algorithm for Multi-objective Optimization: NSGA-II. In *Parallel Problem Solving from Nature, 2000 Conference on*, pp. 849–858. Berlin, Heidelberg: Springer.

den Hertog, D., J. P. C. Kleijnen, and a. Y. D. Siem (2006). The Correct Kriging Variance Estimated by Bootstrapping. *Journal of the Operational Research Society 57*(4), 400–409.

Dick, J. and F. Pillichshammer (2010). *Digital Nets and Sequences: Discrepancy Theory and Quasi-Monte Carlo Integration*. New York, NY, USA: Cambridge University Press.

Dunn, J. C. (1973). A Fuzzy Relative of the ISODATA Process and Its Use in Detecting Compact Well-Separated Clusters. *Journal of Cybernetics 3*(3), 32–57.

Eaton, M. L. (1983). *Multivariate Statistics: A Vector Space Approach*. Wiley, New York.

Ehrgott, M. (2006). *Multicriteria Optimization*. Springer Science & Business Media.

Emmerich, M. (2005). *Single-and Multi-Objective Evolutionary Design Optimization assisted by Gaussian Random Field Metamodels*. Ph. D. thesis, FB Informatik, TU Dortmund.

Emmerich, M. and A. Deutz (2014). Time Complexity and Zeros of the Hypervolume Indicator Gradient Field. In A. Tantar et al. (Eds.), *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation III*, Volume 500 of *Studies in Computational Intelligence*, pp. 169–193. Springer.

Emmerich, M., A. Deutz, and N. Beume (2007). Gradient-Based/Evolutionary Relay Hybrid for Computing Pareto Front Approximations Maximizing the $\mathcal{S}$-metric. In *Proceedings of the 4th International Conference on Hybrid Metaheuristics*, HM'07, pp. 140–156. Springer.

Emmerich, M., O. M. Shir, and H. Wang (2018). *Evolution Strategies*, pp. 1–31. Cham: Springer International Publishing.

Emmerich, M., K. Yang, A. Deutz, H. Wang, and C. M. Fonseca (2016). *A Multicriteria Generalization of Bayesian Global Optimization*, pp. 229–242. Cham: Springer International Publishing.

Fletcher, R. (2013). *Practical Methods of Optimization*. John Wiley & Sons.

Fliege, J. and B. F. Svaiter (2000). Steepest Descent Methods for Multicriteria Optimization. *Mathematical Methods of Operations Research 51*(3), 479–494.

Forrester, A., A. Keane, et al. (2008). *Engineering Design Via Surrogate Modelling: A Practical Guide*. John Wiley & Sons.

Fortin, F., F. Michel, M. A. Gardner, M. Parizeau, and C. Gagné (2012, jul). DEAP: Evolutionary Algorithms Made Easy. *Journal of Machine Learning Research 13*, 2171–2175.

Gabillon, V., M. Ghavamzadeh, and A. Lazaric (2012). Best Arm Identification: A Unified Approach to Fixed Budget and Fixed Confidence. In *Advances in Neural Information Processing Systems*, pp. 3212–3220.

Ginsbourger, D., R. Le Riche, and L. Carraro (2010). *Kriging Is Well-Suited to Parallelize Optimization*, pp. 131–162. Berlin, Heidelberg: Springer Berlin Heidelberg.

Goldberg, D. E. (1989). *Genetic Algorithms in Search, Optimization and Machine Learning* (1st ed.). Boston, MA, USA: Addison-Wesley Longman Publishing Co., Inc.

Goldberg, D. E. and J. Richardson (1987). Genetic Algorithms with Sharing for Multimodal Function Optimization. In *Genetic algorithms and their applications: Proceedings of the Second International Conference on Genetic Algorithms*, pp. 41–49. Hillsdale, NJ: Lawrence Erlbaum.

Hansen, N. (2006). *The CMA Evolution Strategy: A Comparing Review*, pp. 75–102. Berlin, Heidelberg: Springer Berlin Heidelberg.

Hansen, N. (2008, September). Adaptive Encoding: How to Render Search Coordinate System Invariant. In *Parallel Problem Solving from Nature - PPSN X*, Dortmund, Germany, pp. 205–214.

Hansen, N., A. Auger, S. Finck, and R. Ros (2010, March). Real-Parameter Black-Box Optimization Benchmarking 2010: Experimental Setup. Research Report RR-7215, INRIA.

Hansen, N., S. Finck, R. Ros, and A. Auger (2009). Real-Parameter Black-Box Optimization Benchmarking 2009: Noiseless Functions Definitions. Research Report RR-6829, INRIA.

Hansen, N., S. D. Müller, and P. Koumoutsakos (2003, March). Reducing the Time Complexity of the Derandomized Evolution Strategy with Covariance Matrix Adaptation (CMA-ES). *Evol. Comput. 11*(1), 1–18.

Hansen, N. and A. Ostermeier (2001, June). Completely Derandomized Self-Adaptation in Evolution Strategies. *Evol. Comput. 9*(2), 159–195.

Hartman, L. and O. Hössjer (2008). Fast Kriging of Large Data Sets with Gaussian Markov Random Fields. *Computational Statistics & Data Analysis 52*(5), 2331–2349.

Hastie, T., R. Tibshirani, and J. Friedman (2009). *The Elements of Statistical Learning* (2 ed.). Springer Series in Statistics. New York, NY, USA: Springer-Verlag New York.

Hillermeier, C. (2001). Generalized Homotopy Approach to Multiobjective Optimization. *Journal of Optimization Theory and Applications 110*(3), 557–583.

Hoffman, M., E. Brochu, and N. D. Freitas (2011). Portfolio Allocation for Bayesian Optimization. *Conference on Uncertainty in Artificial Intelligence*, 327–336.

Hoffman, M. W., B. Shahriari, and N. de Freitas (2014). On Correlation and Budget Constraints in Model-Based Bandit Optimization with Application to Automatic Machine Learning. *Proceedings of the Seventeenth International Conference on Artificial Intelligence and Statistics 33*, 365–374.

Hutter, F., H. H. Hoos, and K. Leyton-Brown (2011). *Sequential Model-Based Optimization for General Algorithm Configuration*, pp. 507–523. Berlin, Heidelberg: Springer Berlin Heidelberg.

Hutter, F., H. H. Hoos, and K. Leyton-Brown (2012). Parallel Algorithm Configuration. *LION 6*, 55–70.

Hutter, F., H. H. Hoos, K. Leyton-Brown, and K. Murphy (2010). Time-Bounded Sequential Parameter Optimization. In *International Conference on Learning and Intelligent Optimization*, pp. 281–298. Springer.

Jin, Y. (2011). Surrogate-assisted Evolutionary Computation: Recent Advances and Future Challenges. *Swarm and Evolutionary Computation 1*(2), 61–70.

Jones, D. R. (2001). A Taxonomy of Global Optimization Methods Based on Response Surfaces. *Journal of global optimization 21*(4), 345–383.

Jones, D. R., M. Schonlau, and W. J. Welch (1998). Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global optimization 13*(4), 455–492.

Journel, A. G. and M. Rossi (1989). When do we need a trend model in Kriging? *Mathematical Geology 21*(7), 715–739.

Kaya, H., P. Tüfekci, and S. F. Gürgen (2012). Local and Global Learning Methods for Predicting Power of a Combined Gas & Steam Turbine. *International Conference on Emerging Trends in Computer and Electronics Engineering (ICETCEE 2012)*, 13–18.

Kennedy, J. and R. Eberhart (1995, Nov). Particle Swarm Optimization. In *Neural Networks, 1995. Proceedings., IEEE International Conference on*, Volume 4, pp. 1942–1948 vol.4.

Kerschke, P., M. Preuss, S. Wessing, and H. Trautmann (2016). Low-Budget Exploratory Landscape Analysis on Multiple Peaks Models. In *Proceedings of the 18th Annual Conference on Genetic and Evolutionary Computation*, GECCO '16. ACM.

Kerschke, P., H. Wang, M. Preuss, C. Grimme, A. H. Deutz, H. Trautmann, and M. Emmerich (2016). Towards Analyzing Multimodality of Continuous Multiobjective Landscapes. In J. Handl, E. Hart, P. R. Lewis, M. López-Ibáñez, G. Ochoa, and B. Paechter (Eds.), *Proceedings of the 14th International Conference on Parallel Problem Solving from Nature (PPSN XIV)*, Volume 9921 of *Lecture Notes in Computer Science*, pp. 962–972. Springer.

Kimura, S. and K. Matsumura (2005). Genetic Algorithms Using Low-discrepancy Sequences. In *Proceedings of the 7th Annual Conference on Genetic and Evolutionary Computation*, GECCO '05, New York, NY, USA, pp. 1341–1346. ACM.

Kirkpatrick, S., C. D. Gelatt, and M. P. Vecchi (1983). Optimization by Simulated Annealing. *science 220*(4598), 671–680.

Kleijnen, J. P. C., W. van Beers, and I. van Nieuwenhuyse (2012). Expected Improvement in Efficient Global Optimization through Bootstrapped Kriging. *Journal of Global Optimization 54*(1), 59–73.

Kollo, T. o. and D. von Rosen (2005). *Advanced Multivariate Statistics with Matrices* (1 ed.), Volume 579 of *Mathematics and Its Applications*. Springer Netherlands.

Krige, D. G. (1951, December). A Statistical Approach to Some Basic Mine Valuation Problems on the Witwatersrand. *Journal of the Chemical, Metallurgical and Mining Society of South Africa 52*(6), 119–139.

Lawrence, N. D. (2004). Gaussian Process Latent Variable Models for Visualisation of High Dimensional Data. *Advances in neural information processing systems 16*(3), 329–336.

López, A. L., C. A. C. Coello, and O. Schütze (2012, May). *Using Gradient Based Information to Build Hybrid Multi-objective Evolutionary Algorithms*. Ph. D. thesis, CINVESTAV-IPN, Mexico City.

Lophaven, S. N., H. B. Nielsen, and J. Søndergaard (2002). *DACE: a Matlab Kriging Toolbox*, Volume 2. Citeseer.

Loshchilov, I., M. Schoenauer, and M. Sebag (2011). Adaptive Coordinate Descent. In *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 885–892. ACM.

Luke, S. (2009). *Essentials of Metaheuristics*, Volume 113. Lulu Raleigh.

MacQueen, J. et al. (1967). Some Methods for Classification and Analysis of Multivariate Observations. In *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*, Volume 1, pp. 281–297. Oakland, CA, USA.

Martinez-Cantin, R. (2014). Bayesopt: A Bayesian Optimization Library for Nonlinear Optimization, Experimental Design and Bandits. *The Journal of Machine Learning Research 15*(1), 3735–3739.

Miller, B. L. and M. J. Shaw (1996). Genetic Algorithms with Dynamic Niche Sharing for Multimodal Function Optimization. In *Evolutionary Computation, 1996., Proceedings of IEEE International Conference on*, pp. 786–791. IEEE.

Močkus, J. (1975). On Bayesian Methods for Seeking the Extremum. In *Optimization Techniques IFIP Technical Conference*, pp. 400–404. Springer.

Močkus, J. (2012). *Bayesian Approach to Global Optimization: Theory and Applications*, Volume 37. Springer Science & Business Media.

Moscato, P. et al. (1989). On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms. *Caltech concurrent computation program, C3P Report 826*, 1989.

Naish-Guzman, A. and S. Holden (2007). The Generalized FITC Approximation. In *Advances in Neural Information Processing Systems*, pp. 1057–1064.

Nguyen-Tuong, D., M. Seeger, and J. Peters (2009). Model Learning with Local Gaussian Process Regression. *Advanced Robotics 23*(15), 2015–2034.

Niederreiter, H. (1992). *Random Number Generation and quasi-Monte Carlo Methods*. Philadelphia, PA, USA: Society for Industrial and Applied Mathematics.

Nocedal, J. and S. Wright (2000). *Numerical Optimization* (2 ed.). Springer Series in Operations Research and Financial Engineering. Springer-Verlag New York.

Nourani, Y. and B. Andresen (1998). A Comparison of Simulated Annealing Cooling Strategies. *Journal of Physics A: Mathematical and General 31*(41), 8373.

O'Hagan, A. and J. Kingman (1978). Curve Fitting and Optimal Design for Prediction. *Journal of the Royal Statistical Society. Series B (Methodological)*, 1–42.

Øksendal, B. (2003). *Stochastic Differential Equations*. Universitext. Springer-Verlag Berlin Heidelberg.

Omre, H. (1987). Bayesian Kriging — Merging Observations and Qualified Guesses in Kriging. *Mathematical Geology 19*(1), 25–39.

Ostermeier, A., A. Gawelczyk, and N. Hansen (1994). Step-Size Adaption Based on Non-Local Use of Selection Information. In *Proceedings of the International Conference on Evolutionary Computation. The Third Conference on Parallel*

Problem Solving from Nature: Parallel Problem Solving from Nature, PPSN III, London, UK, UK, pp. 189–198. Springer-Verlag.

Ponweiser, W., T. Wagner, D. Biermann, and M. Vincze (2008). Multiobjective Optimization on a Limited Budget of Evaluations Using Model-Assisted $\mathcal{S}$-Metric Selection. In G. Rudolph, T. Jansen, N. Beume, S. Lucas, and C. Poloni (Eds.), *Parallel Problem Solving from Nature – PPSN X*, Berlin, Heidelberg, pp. 784–794. Springer Berlin Heidelberg.

Ponweiser, W., T. Wagner, and M. Vincze (2008, June). Clustered Multiple Generalized Expected Improvement: A Novel Infill Sampling Criterion for Surrogate Models. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 3515–3522.

Quiñonero-Candela, J. and C. E. Rasmussen (2005). A Unifying View of Sparse Approximate Gaussian Process Regression. *The Journal of Machine Learning Research 6*(1), 1939–1959.

Rao, C. R., H. Toutenburg, Shalabh, and C. Heumann. *Linear Models and Generalizations* (3rd ed.). Springer Series in Statistics. Springer-Verlag Berlin Heidelberg.

Rasmussen, C. and C. Williams (2006). *Gaussian Processes for Machine Learning*. Adaptative computation and machine learning series. University Press Group Limited.

Ren, Y., A. Deutz, and M. Emmerich (2015). On Steering Dominated Points in Hypervolume Gradient Ascent for Bicriteria Continuous Optimization (extended abstract). In *Numerical and Evolutionary Optimization, NEO 2015, Tijuana, Mexico (Book of abstracts)*.

Reynolds, D. (2009). Gaussian Mixture Models. In *Encyclopedia of Biometrics*, pp. 659–663. Springer.

Rosenbrock, H. H. (1960, January). An Automatic Method for Finding the Greatest or Least Value of a Function. *The Computer Journal 3*(3), 175–184.

Roustant, O., D. Ginsbourger, and Y. Deville (2012, 10). DiceKriging, DiceOptim: Two R Packages for the Analysis of Computer Experiments by Kriging-Based Metamodeling and Optimization. *Journal of Statistical Software 51*(1), 1–55.

Sacks, J., W. J. Welch, T. J. Mitchell, and H. P. Wynn (1989). Design and Analysis of Computer Experiments. *Statistical Science 4*(4), 409–423.

Santner, T., B. Williams, and W. Notz (2003). *The Design and Analysis of Computer Experiments*. Springer Series in Statistics. Springer.

Sasena, M. J., P. Papalambros, and P. Goovaerts (2002). Exploration of Metamodeling Sampling Criteria for Constrained Global Optimization. *Engineering optimization 34*(3), 263–278.

Schölkopf, B., R. Herbrich, and A. J. Smola (2001). A Generalized Representer Theorem. In *International conference on computational learning theory*, pp. 416–426. Springer.

Schonlau, M. (1998). *Computer Experiments and Global Optimization*. University of Waterloo.

Schonlau, M., W. J. Welch, and D. R. Jones (1998). Global versus Local Search in Constrained Optimization of Computer Models. *Lecture Notes-Monograph Series*, 11–25.

Schütze, O., C. Domínguez-Medina, N. Cruz-Cortés, L. Gerardo de la Fraga, J.-Q. Sun, G. Toscano, and R. Landa (2016). A Scalar Optimization Approach for Averaged Hausdorff Approximations of the Pareto front. *Engineering Optimization 48*(9), 1593–1617.

Schütze, O., X. Esquivel, A. Lara, and C. A. C. Coello (2012, Aug). Using the Averaged Hausdorff Distance as a Performance Measure in Evolutionary Multiobjective Optimization. *IEEE Transactions on Evolutionary Computation 16*(4), 504–522.

Schütze, O., A. Lara, and C. Coello Coello (2011). The Directed Search Method for Unconstrained Multi-objective Optimization Problems. *Proceedings of the EVOLVE–A Bridge Between Probability, Set Oriented Numerics, and Evolutionary Computation*.

Schwefel, H.-P. (1993). *Evolution and Optimum Seeking: The Sixth Generation*. New York, NY, USA: John Wiley & Sons, Inc.

Shir, O. M. et al. (2008). *Niching in Derandomized Evolution Strategies and its Applications in Quantum Control*. Natural Computing Group, LIACS, Faculty of Science, Leiden University.

Shir, O. M. and T. Bäck (2005a). Dynamic Niching in Evolution Strategies with Covariance Matrix Adaptation. In *Evolutionary Computation, 2005. The 2005 IEEE Congress on*, Volume 3, pp. 2584–2591. IEEE.

Shir, O. M. and T. Bäck (2005b). Niching in Evolution Strategies. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pp. 915–916. ACM.

Shir, O. M., M. Emmerich, and T. Bäck (2007). Self-Adaptive Niching CMA-ES with Mahalanobis Metric. In *Evolutionary Computation, 2007. CEC 2007. IEEE Congress on*, pp. 820–827. IEEE.

Silver, D., A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al. (2016). Mastering the Game of Go with Deep Neural Networks and Tree Search. *nature 529*(7587), 484–489.

Silverman, B. W. (1985). Some Aspects of the Spline Smoothing Approach to Non-parametric Regression Curve Fitting. *Journal of the Royal Statistical Society. Series B (Methodological) 47*(1), 1–52.

Slater, M. (2014). Lagrange Multipliers Revisited. In *Traces and Emergence of Nonlinear Programming*, pp. 293–306. Springer.

Snelson, E. and Z. Ghahramani (2005). Sparse Gaussian Processes using Pseudo-inputs. In *Advances in neural information processing systems*, pp. 1257–1264.

Snoek, J., H. Larochelle, and R. P. Adams (2012). Practical Bayesian Optimization of Machine Learning Algorithms. In *Advances in neural information processing systems*, pp. 2951–2959.

Sóbester, A., S. J. Leary, and A. J. Keane (2005). On the Design of Optimization Strategies Based on Global Response Surface Approximation Models. *Journal of Global Optimization 33*(1), 31–59.

Sosa Hernández, V. A., O. Schütze, and M. Emmerich (2014). *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation V*, Chapter Hypervolume Maximization via Set Based Newton's Method, pp. 15–28. Cham, Switzerland: Springer International Publishing.

Srinivas, N. and K. Deb (1994). Muiltiobjective Optimization using Nondominated Sorting in Genetic Algorithms. *Evolutionary computation 2*(3), 221–248.

Srinivas, N., A. Krause, S. Kakade, and M. Seeger (2010). Gaussian Process Optimization in the Bandit Setting: No Regret and Experimental Design. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, USA, pp. 1015–1022. Omnipress.

Stein, M. L. (1999). *Interpolation of Spatial Data: Some Theory for Kriging* (1 ed.). Springer Series in Statistics. Springer-Verlag New York.

Storn, R. and K. Price (1997, Dec). Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of global optimization 11*(4), 341–359.

Sundberg, R. (1974). Maximum Likelihood Theory for Incomplete Data from an Exponential Family. *Scandinavian Journal of Statistics*, 49–58.

Talbi, E.-G. (2009). *Metaheuristics: from Design to Implementation*, Volume 74. John Wiley & Sons.

Teytaud, O. and S. Gelly (2007). DCMA: yet another derandomization in Covariance-Matrix-Adaptation. In *Proceedings of the 9th annual conference on Genetic and evolutionary computation*, pp. 955–963. ACM.

Tong, Y. L. (2012). *The Multivariate Normal Distribution*. Springer Science & Business Media.

Torn, A. and A. Zilinskas (1989). Global Optimization. *Lecture Notes in Computer Science 350*.

Tresp, V. (2000). A Bayesian Committee Machine. *Neural computation 12*(11), 2719–2741.

Ursem, R. K. (2014). From Expected Improvement to Investment Portfolio Improvement: Spreading the Risk in Kriging-Based Optimization. In *International Conference on Parallel Problem Solving from Nature*, pp. 362–372. Springer.

van der Vaart, A. W. and J. H. van Zanten (2008). Rates of Contraction of Posterior Distributions Based on Gaussian Process Priors. *The Annals of Statistics*, 1435–1463.

van Stein, B., H. Wang, W. Kowalczyk, T. Bäck, and M. Emmerich (2015). Optimally Weighted Cluster Kriging for Big Data Regression. In E. Fromont, T. De Bie, and M. van Leeuwen (Eds.), *Advances in Intelligent Data Analysis XIV*, pp. 310–321. Cham: Springer International Publishing.

van Stein, B., H. Wang, W. Kowalczyk, M. Emmerich, and T. Bäck (2016). Fuzzy clustering for Optimally Weighted Cluster Kriging. In *Fuzzy Systems (FUZZ-IEEE), 2016 IEEE International Conference on*, pp. 939–945. IEEE.

Vijayakumar, S., A. D'souza, and S. Schaal (2005). Incremental Online Learning in High Dimensions. *Neural computation 17*(12), 2602–2634.

Wang, H., T. Bäck, and M. T. M. Emmerich (2018). Multi-point Efficient Global Optimization Using Niching Evolution Strategy. In A.-A. Tantar, E. Tantar, M. Emmerich, P. Legrand, L. Alboaie, and H. Luchian (Eds.), *EVOLVE - A Bridge between Probability, Set Oriented Numerics, and Evolutionary Computation VI*, Cham, pp. 146–162. Springer International Publishing.

Wang, H., A. Deutz, T. Bäck, and M. Emmerich (2017). Hypervolume Indicator Gradient Ascent Multi-objective Optimization. In H. Trautmann, G. Rudolph, K. Klamroth, O. Schütze, M. Wiecek, Y. Jin, and C. Grimme (Eds.), *Evolutionary Multi-Criterion Optimization*, Cham, pp. 654–669. Springer International Publishing.

Wang, H., M. Emmerich, and T. Bäck (2014). Mirrored Orthogonal Sampling with Pairwise Selection in Evolution Strategies. In *Proceedings of the 29th Annual ACM Symposium on Applied Computing*, SAC '14, New York, NY, USA, pp. 154–156. ACM.

Wang, H., M. Emmerich, and T. Bäck (2016). Balancing Risk and Expected Gain in Kriging-Based Global Optimization. In *Evolutionary Computation (CEC), 2016 IEEE Congress on*, pp. 719–727. IEEE.

Wang, H., M. Emmerich, and T. Bäck (2018). Cooling Strategies for the Moment-Generating Function in Bayesian Global Optimization. In *Evolutionary Computation (CEC), 2018 IEEE Congress on*, pp. to appear. IEEE.

Wang, H., Y. Ren, A. Deutz, and M. Emmerich (2017). *On Steering Dominated Points in Hypervolume Indicator Gradient Ascent for Bi-Objective Optimization*, pp. 175–203. Cham: Springer International Publishing.

Wang, H., B. van Stein, M. Emmerich, and T. Bäck (2017). A New Acquisition Function for Bayesian Optimization Based on the Moment-Generating Function. In *Systems, Man, and Cybernetics (SMC), 2017 IEEE International Conference on*, pp. 507–512. IEEE.

Wang, H., B. van Stein, M. Emmerich, and T. Bäck (2017). Time Complexity Reduction in Efficient Global Optimization using Cluster Kriging. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '17, New York, NY, USA, pp. 889–896. ACM.

Webster, R. and M. A. Oliver (2007). *Geostatistics for Environmental Scientists*. John Wiley & Sons.

Wessing, S. (2015). *Two-Stage Methods for Multimodal Optimization*. Ph. D. thesis, Technische Universität Dortmund.

Wessing, S. (2016). *optproblems: Infrastructure to Define Optimization Problems and Some Test Problems for Black-Box Optimization*. python package version 0.9.

Wright, S. and J. Nocedal (1999). Numerical Optimization. *Springer Science 35*(67-68), 7.

Yeh, I.-C. (1998). Modeling of Strength of High-performance Concrete using Artificial Neural Networks. *Cement and Concrete research 28*(12), 1797–1808.

Zhang, Q. and H. Li (2007). MOEA/D: A Multiobjective Evolutionary Algorithm Based on Decomposition. *IEEE Transactions on evolutionary computation 11*(6), 712–731.

Žilinskas, A. (1992). A Review of Statistical Models for Global Optimization. *Journal of Global Optimization 2*(2), 145–153.

Zimmerman, D., C. Pavlik, A. Ruggles, and M. P. Armstrong (1999). An Experimental Comparison of Ordinary and Universal Kriging and Inverse Distance Weighting. *Mathematical Geology 31*(4), 375–390.

Zitzler, E., K. Deb, and L. Thiele (2000, June). Comparison of Multiobjective Evolutionary Algorithms: Empirical Results. *Evolutionary Computation* (2), 173 – 195.

Zitzler, E., M. Laumanns, L. Thiele, et al. (2001). SPEA2: Improving the Strength Pareto Evolutionary Algorithm. In *Eurogen*, Volume 3242, pp. 95–100.

Zitzler, E. and L. Thiele (1998). Multiobjective Optimization Using Evolutionary Algorithms — A Comparative Case Study. In A. E. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel (Eds.), *Parallel Problem Solving from Nature — PPSN V*, Berlin, Heidelberg, pp. 292–301. Springer Berlin Heidelberg.

Zitzler, E., L. Thiele, M. Laumanns, C. M. Fonseca, and V. G. da Fonseca (2003, April). Performance Assessment of Multiobjective Optimizers: An Analysis and Review. *IEEE Transactions on Evolutionary Computation 7*(2), 117–132.

Acronyms

BLP Best Linear Estimator. 50

BLUE Best Linear Unbiased Estimator. 48

BLUP Best Linear Unbiased Predictor. 48

ECDF Empirical Cumulative Distribution Function. 36, 101

EGO Efficient Global Optimization. 37, 90, 113, 114

EI Expected Improvement. 78, 90–93, 96, 103, 108, 114, 154

GEI Generalized Expected Improvement. 91, 92, 98, 100

GLS Generalized Least Squares. 48

GPR Gaussian Process Regression. 39, 43, 44, 56, 59, 63, 69, 87, 88, 95, 100, 152

KKT Karush-Kuhn-Tucker conditions. 57, 95, 125

LHS Latin Hypercube Sampling. 57, 101

LUP Linear Unbiased Predictor. 47

MAP Maximum a Posterior. 59

MGF Moment-Generating Function. 96

MGFI Moment-Generating Function of Improvement. 98, 100, 101, 103, 107, 154, 155

MSE Mean Squared Error. 37, 39, 49, 50, 69, 71, 73, 80, 82, 94, 108

RBF Radial Basis Functions. 46

RKHS Reproducing Kernel Hilbert Space. 54, 56, 57, 155

Index

Summary

The black-box optimization problem is frequently encountered in many applications. For example, the tuning task of a machine learning algorithm or fitting a curve to some experimental data. In the PROMIMOOC project (**PRO**cess **MI**ning for **M**ulti-**O**bjective **O**nline **C**ontrol with industrial partners Tata Steel and BMW group), the optimization problem we are facing is to search for proper control parameters of production processes (for both partners), such that the number of defects generated during the production would be largely reduced. Such a problem is typically referred as an "black-box", as we don't directly model the physical mechanism behind the production process and there is no additional information about its mathematical characteristics (e.g., convexity and continuity) that would be very useful for the optimization. Therefore, the black-box problem is also considered very challenging. Another difficulty arises in the extremely high cost of making trials on the production line: suppose a candidate setting of control parameters (or candidate solution) is proposed by an optimization algorithm. The quality of this setting can only be assessed by applying it to the actual production line and then measuring defect rate in the output. This is typically very costly and risky: when the candidate setting doesn't actually perform well, many defects will be generated, resulting in extra production costs for industrial partners. To solve this problem efficiently and carefully, several fundamental optimization techniques have to be combined in a reasonable way.

First of all, as there is not much mathematical assumptions on the problem, we have to resort to the so-called *stochastic optimization algorithm*, instead of using the traditional optimization techniques from mathematics/operational research. The stochastic optimization algorithm is a class of methods that directly optimize the objective function by solely using the assessment (evaluation) of the candidate solution. Stochastic optimization algorithms are underpinned by the so-called

stochastic variation, which generates (local) random perturbations to modify the current search point. In evolutionary computation, this is typically called the mutation operator. Intuitively, the efficiency of a stochastic variation method greatly affects the performance of the corresponding optimization algorithm. This is the reason why we investigate the efficiency issue of such methods in depth (Chapter 2). As a result of the investigation, we propose a novel stochastic variation method, called *mirrored orthogonal sampling*, which aims at generating random perturbations that cover the search space (subset of $\mathbb{R}^d$) evenly. Both theoretical analysis and empirical study are conducted on the proposed method.

Secondly, because it is very costly to assess candidate solutions, it is common to replace an actual expensive assessment by a machine learning model, which is trained on the historical assessments. Then an optimization algorithm can query the quality of a candidate solution from the model, instead of running the real production process with this solution. Such a technique is called *surrogate modeling*. One big challenge in surrogate modeling is to give a reliable quantification about the uncertainty in model prediction due to the fact that data-driven models usually yield significant errors in prediction. In Chapter 3, we study the well-known *Kriging/Gaussian Process Regression* (GPR) model, that is capable of quantifying the uncertainty. The quantification approach in Kriging/GPR is discussed in detail. When it comes to the application of the Kriging/GPR method to real-world data, we are confronted with the following obstacle: The Kriging/GPR method suffers from a cubic time complexity when dealing with large data sets, limiting its applicability for big data sets. In the reminder of this chapter, a novel algorithmic framework, called *Cluster Kriging* is proposed to tackle this issue. Cluster Kriging is tested on some selected functions and data sets, exhibiting an acceleration of the modeling speed as well as an improved modeling precision.

Naturally, once a good surrogate model is obtained from the previous discussion, the next question is how to use such a model in a reasonable manner such that the uncertainty quantification is taken into account. It is possible to select the most trustworthy solution based on the surrogate model, or alternatively the point that possesses the highest potential to help the optimization procedure if the actual assessment were conducted on it. Such decisions are usually determined through an utility function on the surrogate model, called *infill criterion*. This is the topic of Chapter 4. The difficulty in designing the infill criterion is how to balance the trade-off between the model prediction (exploitation) and the model uncertainty

(exploration). In this chapter, we summarize the existing infill criterion and propose a novel infill criteria, called *Moment-Generating Function of Improvement* that allows for controlling this trade-off explicitly and smoothly. Furthermore, the parallelization issue of infill criteria is also considered thoroughly and several new parallelization methods are proposed and tested.

Lastly, we discuss the so-called multi-objective optimization problem: suppose we want to minimize the number of defects generated in the production and maximize the throughout simultaneously. In this case, it is typical not possible to find a setting of control parameters that satisfies both objectives in the same time and thus we have to adopt multi-objective optimization algorithms. In Chapter 5, we aims at designing a multi-objective optimization algorithm that is able to use either the gradient or the Hessian matrix of the objective function. To achieve this goal, the gradient field and Hessian matrix of the so-called hypervolume indicator are derived and studied in depth. As a result, two novel algorithms, namely the hypervolume-based first- (gradient) and second-order (Hessian) methods are proposed and tested.

Samenvatting

Het black-box-optimalisatieprobleem komt in veel toepassingen voor. Bijvoorbeeld, de afstemmingstaak van een machine-learning algoritme voor het leren van een curve op experimentele gegevens.

In het PROMIMOOC-project (**PRO**cess **MI**ning voor **M**ulti-**O**bjective **O**nline **C**ontrol met industriële partners Tata Steel en BMW groep), is het optimalisatieprobleem waarmee we geconfronteerd worden, het zoeken naar de juiste controleparameters van productieprocessen (voor beide partners), zodanig, dat het aantal defecten tijdens deze productie processen grotendeels gereduceerd word. Een dergelijk probleem wordt meestal een "black-box" genoemd, omdat we niet direct het fysieke mechanisme achter het productieproces modelleren en omdat er geen aanvullende informatie over de wiskundige kenmerken (bijvoorbeeld convexiteit en continuïteit) bekend zijn die zouden helpen bij de optimalisatie. Daarom wordt het Black-Box-probleem ook als zeer uitdagend beschouwd. Een andere moeilijkheid doet zich voor bij de extreem hoge kosten van het uitvoeren van proeven op de productielijn: veronderstel dat een kandidaat-instelling van controleparameters (of kandidaat-oplossing) wordt voorgesteld door een optimalisatie-algoritme. De kwaliteit van deze instelling kan alleen worden beoordeeld door deze toe te passen op de daadwerkelijke productielijn en vervolgens de defectfrequentie in de uitvoer te meten. Dit is vaak erg duur en riskant: wanneer de kandidaat-instelling niet goed presteert, zullen veel defecten worden gegenereerd en dit resulteert in extra productiekosten voor de industriële partners. Om dit probleem efficiënt en zorgvuldig op te lossen, moeten verschillende fundamentele optimalisatietechnieken op een redelijke manier worden gecombineerd.

Allereerst, omdat er niet veel wiskundige veronderstellingen over het probleem zijn, moeten we onze toevlucht nemen tot het zogenaamde *stochastische optimalisatie-algoritme*, in plaats van de traditionele optimalisatietechnieken uit de wiskunde en

operationeel onderzoek te gebruiken. Het stochastische optimalisatie-algoritme is een klasse van methoden die de doelfunctie direct optimaliseert door uitsluitend de beoordeling (evaluatie) van de kandidaat-oplossing te gebruiken. Stochastische optimalisatie-algoritmen worden onderbouwd door de zogenaamde *Stochastische variatie*, die (lokale) willekeurige verstoringen genereert om het huidige zoekpunt te wijzigen. In evolutionaire methoden wordt dit meestal de mutatie-operator genoemd. Intuïtief beïnvloedt de efficiëntie van een stochastische variatiemethode de prestaties van het bijbehorende optimalisatie-algoritme enorm. Dit is de reden waarom we de efficiëntie van dergelijke methoden grondig onderzoeken (Hoofdstuk 2). Als resultaat van het onderzoek stellen we een nieuwe stochastische variatiemethode voor, genaamd *mirrored orthogonal sampling*, die gericht is op het gelijkmatig genereren van willekeurige verstoringen die de zoekruimte (deelverzameling van $\mathbb{R}^d$) dekken. Zowel theoretische analyse als empirisch onderzoek zijn uitgevoerd op de voorgestelde methode.

Ten tweede, omdat het erg duur is om kandidaat-oplossingen te beoordelen, is het gebruikelijk om een dure beoordeling in de praktijk te vervangen door een machine learning model, dat is getraind met behulp van de historische beoordelingen. Dan kan een optimalisatie-algoritme de kwaliteit van een kandidaat-oplossing bepalen aan de hand van het model, in plaats van het echte productieproces met deze oplossing uit te voeren. Z'on techniek heet *Surrogaatmodellering*. Een grote uitdaging bij surrogaatmodellering is het geven van een betrouwbare kwantificering over de onzekerheid van de voorspelling van een model, omdat data-gestuurde modellen meestal significante voorspellingsfouten opleveren. In hoofdstuk 3 bestuderen we het bekende *Kriging/Gaussian Process Regression* (GPR) model, dat in staat is om de onzekerheid te kwantificeren. De kwantificeringsbenadering in Kriging/GPR wordt in detail besproken. Bij de toepassing van de Kriging/GPR-methode op reële gegevens, worden we geconfronteerd met het volgende obstakel: de Kriging/GPR-methode heeft een kubieke tijd complexiteit, waardoor de toepasbaarheid van de methode bij grote datasets wordt beperkt. In dit hoofdstuk wordt een nieuw algoritmisch raamwerk, genaamd *Cluster Kriging*, voorgesteld om dit probleem aan te pakken. Cluster Kriging wordt getest op een aantal geselecteerde functies en datasets, met een versnelling van de model leersnelheid en een verbeterde model precisie.

Vanzelfsprekend is, wanneer eenmaal een goed surrogaatmodel is verkregen, de vraag hoe een dergelijk model op een redelijke manier moet worden gebruikt, zodat

de kwantificering van de onzekerheid in aanmerking wordt genomen. Het is mogelijk om de meest betrouwbare oplossing te bepalen op basis van het surrogaatmodel, of als alternatief, welk punt het grootste potentieel heeft om de optimalisatieprocedure te helpen als de feitelijke beoordeling daarop is uitgevoerd. Dergelijke beslissingen worden meestal bepaald door een nutsfunctie op het surrogaatmodel, genaamd *invulcriterium*. Dit is het onderwerp van hoofdstuk 4. De moeilijkheid bij het ontwerpen van het invulcriterium is het in balans brengen van de afweging tussen de modelvoorspelling (exploitatie) en de modelonzekerheid (verkenning). In dit hoofdstuk vatten we het bestaande invulcriterium samen en stellen we een nieuw invulcriterium voor, genaamd *Moment-Generating Function of Improvement* dat het mogelijk maakt om deze afweging expliciet en soepel te regelen. Bovendien wordt de parallellisatie mogelijkheden van invulcriteriums ook grondig overwogen en worden verschillende nieuwe parallellisatie methoden voorgesteld en getest.

Ten slotte, bespreken we het zogenaamde optimaliseringsprobleem met meerdere doelstellingen: stel dat we het aantal defecten in de productie willen minimaliseren en het hele proces tegelijkertijd willen maximaliseren. In dit geval is het typisch niet mogelijk om een instelling van parameters te vinden die beide doelen op hetzelfde moment behaalt, daarom moeten we speciale algoritmen voor de optimalisatie van meerdere doeleinden gebruiken. In hoofdstuk 5 willen we een optimaliseringsalgoritme met meerdere doeleinden ontwerpen dat de gradiënt of de Hessiaanse matrix van de doelfunctie kan gebruiken. Om dit doel te bereiken, worden het gradiëntveld en de Hessiaanse matrix van de zogenaamde hypervolume-indicator afgeleid en grondig bestudeerd. Dientengevolge worden twee nieuwe algoritmen, namelijk de hypervolume-gebaseerde eerste- (gradiënt) en tweede-orde (Hessiaan) methoden voorgesteld en getest.

About the Author

Hao Wang born 1989 in Baoji, China, received his Master degree of Computer Science at Leiden University, The Netherlands in 2013. Hao worked as a PhD since May, 2014 in Leiden Institute of Advanced Computer Science (LIACS), supervised by Prof. Thomas Bäck and Associate Prof. Michael Emmerich. His research interests are proposing, improving and analyzing optimization algorithms, especially Evolutionary Strategies, as well as developing statistical machine learning methods for big and complex industrial data. He also aims at combining state-of-the-art optimization algorithm with data mining techniques to make efficient and robust optimizers for industry processes.

王昊者，己巳年正月十五生，秦地陈仓人也。自幼生性好动，非钻研之才，亦无格物致知之心。及至中学，心性渐变，随倾心于数字之妙，万物变化之理。至中学结业时，已立科学研究之志。岁在丁亥，初赴京师，研习算法之术。至大学结业之时，学业已成，然则胸中希冀未满，随而出西洋，至荷兰，为术业之精进，师洋人，专攻优化之术。乃至术业精通，忆大学往昔，岁月如梭，竟已有十一载。今已近而立之年，东眺中土之地，百业兴盛，视诸夷域，若如虚无缥缈之间。而我之云帆高涨，昼夜星驰，非只因吾之奋进，更赖国家培养之道，父母哺育之恩也。念此道此恩，吾必兢兢业业，用吾之所学，回馈父母，助我国家科学之盛，绵薄之力，在所不辞。

戊戌八月十一于荷兰莱顿