

# Tailoring of the Efficient Global Optimization (EGO) Algorithm - Applications for Crashworthiness Problems

**Koushyar Komeilizadeh**

Vollständiger Abdruck der von der TUM School of Engineering and Design der Technischen Universität München zur Erlangung eines

**Doktors der Ingenieurwissenschaften (Dr.-Ing.)**

genehmigten Dissertation.

Vorsitz: Prof. Dr.-Ing. Michael Manhart  
Prüfende der Dissertation: 1. Prof. Dr.-Ing. habil. Fabian Duddeck  
2. Assistant Prof. Dr. Elena Raponi

Die Dissertation wurde am 18.04.2024 bei der Technischen Universität München eingereicht und durch die TUM School of Engineering and Design am 01.10.2024 angenommen.





## Abstract

Numerical simulations have become an integral part of automotive design processes. Accordingly, optimization algorithms to improve the efficiency of this process or its outcomes have been continuously developed. However, each simulation can be expensive and furthermore, derivatives are not reliable. These factors make optimization algorithms based on response surface methods a promising choice. Among these algorithms, "Efficient Global Optimization (EGO)" is a favorite (adaptive) one.

In this thesis, we propose multiple improvements for EGO with crashworthiness as the main application. This is achieved by considering a lesson from the "no free lunch theorem", i.e., integrating information into the optimization procedure. The proposed enhancements are based on:

- Observing EGO tendencies, such as placing samples around boundaries.
- Observing how the fitted surface in EGO is affected, for example, by complementary kernels or alternative exclusion of the least contributing inputs.
- The impact of the hyperparameter bound on reducing overfitting.
- Following a desired value or a (narrow) range of known values efficiently, i.e., optimizing while knowing the active constraint or finding multiple good solutions instead of just the best one.

The proposed methods are first compared with standard EGO or other optimizers over a variety of test functions and in the end, they are always applied to a crashworthiness case. This main body of work is developed and tested for low-dimensional problems. However, the possibility to extend it (largely) to higher dimensions and other types of EGO (multi-disciplinary or multi-fidelity) have been taken into consideration as a background goal. Hence, while improving EGO by information inclusion is at the center of this work, the long-term goal is beyond it and seeks higher efficiency in larger dimensions. The results show that the intended enhancements have been achieved and suggestions for further improvements are provided accordingly.

## Zusammenfassung

Numerische Simulationen sind zu einem integralen Bestandteil von Fahrzeugauslegungsprozessen geworden. Dementsprechend wurden Optimierungsansätze zur effizienten Verbesserung dieses Prozesses oder seiner Ergebnisse kontinuierlich weiterentwickelt. Ansätze, die auf den Response-Surface-Methoden basieren, sind vielversprechend, wenn numerische Lösungen teuer sind und es keine zuverlässigen Gradienten gibt. Unter diesen Ansätzen ist "Efficient Global Optimization (EGO)" ein beliebter adaptiver Algorithmus.

In dieser Arbeit schlagen wir mehrere Verbesserungen für EGO vor, wobei die automobilen Crashsicherheit die Hauptanwendung ist. Dies wird erreicht, indem eine Lehre aus dem "no free lunch theorem" berücksichtigt wird, d. h. Informationen in das Optimierungsverfahren integriert werden müssen. Die vorgeschlagenen Verbesserungen basieren auf:

- Beobachten von EGO-Tendenzen, wie z. B. das Platzieren von Punkten in Grenzbereichen.
- Effekte von zwei eher komplementären Kernels auf das angepasste Modell, ebenso der alternierende Ausschluss der am wenigsten einflussreichen Eingangsgröße.
- Der Einfluss der definierten Hyperparametergrenzen auf die Reduzierung von Overfitting.
- Verfolgung von einem gewünschten Wert oder einem (engen) Bereich bekannter Werte, d. h. Optimieren, wenn die aktive Nebenbedingung bekannt ist, oder das Finden mehrerer guter Lösungen, anstatt nur der besten.

Die vorgeschlagenen Methoden werden zunächst mit Standard-EGO und anderen Optimierern anhand verschiedener Arten von Testfunktionen verglichen und am Ende jeweils auf einen Crashfall angewandt. Der Hauptteil der Arbeit wird für niedrigdimensionale Probleme entwickelt und getestet. Die Möglichkeit, sie (weitgehend) auf höhere Dimensionen und andere Arten von EGO zu erweitern, wurde als Hintergrundziel in Betracht gezogen. Die Ergebnisse zeigen, dass die beabsichtigten Ziele erreicht wurden. Ideen für darüber hinausgehende Verbesserungen werden präsentiert.

# Contents

<b>Abstract</b>	<b>i</b>
<b>Zusammenfassung</b>	<b>ii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation and aims . . . . .	1
1.1.1 Structure of chapters . . . . .	5
<b>2 Preliminaries</b>	<b>7</b>
2.1 Nonlinear optimization . . . . .	8
2.2 Response Surface Methods . . . . .	12
2.3 Design of Experiments (DOE) . . . . .	17
2.3.1 Quasi Monte Carlo . . . . .	20
2.3.2 Latin Hypercube . . . . .	20
2.3.3 Comparison of uniform DOE methods . . . . .	22
2.4 Curse of dimensionality . . . . .	25
2.5 Gaussian Process Regression . . . . .	27
2.5.1 Definition . . . . .	27
2.5.3 Model selection and adaptation of hyperparameters . . . . .	31
<b>3 Efficient global optimization (EGO)</b>	<b>36</b>
3.1 Introduction . . . . .	37
3.2 Infill criterion . . . . .	39
3.2.1 Lower confidence bound (LCB) . . . . .	40
3.2.2 Probability of improvement (PI) . . . . .	40
3.2.3 Expected improvement family . . . . .	40
3.3 Handling constraints . . . . .	42
3.3.1 Penalty methods and constrained infill criterion . . . . .	43
3.3.2 Expected Violation (EV) . . . . .	43
3.3.3 Expected Improvement with Feasibility Probability . . . . .	44
3.3.4 Conversion into a multi-objective problem . . . . .	46
3.3.5 Following part of the (feasible) design space . . . . .	48
3.3.6 Other constraint handling methods . . . . .	49
3.4 Handling noisy simulations . . . . .	50
<b>Dissertation</b>	<b>iii</b>

3.5	Handling EGO more efficiently - larger space treatments . . . . .	51
<b>4</b>	<b>Enhancements of EGO - Part I: Dimension adaptation and reduction, shape modification, sample translation and hybridization</b>	<b>54</b>
4.1	Introduction . . . . .	55
4.2	Suggested modifications and methods . . . . .	57
4.2.1	Dimension adaptation and shape modification . . . . .	57
4.2.2	Dimension reduction with the help of the MVMO algorithm . . . . .	57
4.2.3	Sample translation . . . . .	59
4.2.4	A hybrid meta-model optimizer . . . . .	60
4.3	Problems and results . . . . .	61
4.3.1	Prerequisites and settings . . . . .	62
4.3.2	Hartman-6 . . . . .	63
4.3.3	Himmelblau . . . . .	66
4.3.4	Five-bar truss . . . . .	69
4.3.5	Side sill impact . . . . .	71
4.3.6	Discussion . . . . .	74
<b>5</b>	<b>Enhancements of EGO - Part II: GPR fitting, EI</b>	<b>78</b>
5.1	GPR overfitting . . . . .	79
5.1.1	A new proposed procedure to set the hyperparameter bounds . . . . .	85
5.2	Problems and results . . . . .	87
5.2.1	Prerequisites and settings . . . . .	87
5.2.2	Results . . . . .	88
5.2.3	Discussion . . . . .	94
5.3	Proposed modification for (W)EI . . . . .	97
5.3.1	Test functions as unconstrained problems . . . . .	99
5.3.2	Mechanical design problems . . . . .	104
5.3.3	Shape optimization of a crash box . . . . .	108
5.3.4	Crash box results and discussion . . . . .	110
<b>6</b>	<b>Enhancement of EGO - Part III: constrained problems</b>	<b>114</b>
6.1	Motivations for following certain function values . . . . .	115
6.2	A new criterion to follow a certain value . . . . .	115
6.3	Problems and results . . . . .	117
6.3.1	BraninHoo . . . . .	118
6.3.2	Gomez3 . . . . .	119
6.4	Constraint handling via EI – FP . . . . .	122
6.5	A new constraint handling algorithm - following a desired range . . . . .	124
6.6	Problems and results . . . . .	127
6.6.1	A welded beam design problem . . . . .	127

6.6.2	A pressure vessel design problem . . . . .	133
6.6.3	An I-beam design problem . . . . .	136
6.6.4	Crash simulation - Solution space optimization . . . . .	139
6.7	Discussion . . . . .	145
<b>7</b>	<b>Conclusion</b>	<b>148</b>
7.1	Critical review and outlook . . . . .	149
<b>A</b>	<b>Setting the hyperparameter bounds</b>	<b>154</b>
<b>B</b>	<b>Mechanical design problems</b>	<b>159</b>
B.1	A speed reducer design problem . . . . .	160
B.2	A rolling element bearing design problem . . . . .	161
B.3	A Belleville disc spring design problem . . . . .	162
B.4	A hydrostatic thrust bearing design problem . . . . .	163
<b>C</b>	<b>Solution space method</b>	<b>164</b>
C.1	Solution space method . . . . .	165
	<b>List of Figures</b>	<b>166</b>
	<b>List of Tables</b>	<b>176</b>
	<b>Acronyms</b>	<b>178</b>
	<b>Bibliography</b>	<b>180</b>



# Chapter 1

## Introduction

### 1.1. Motivation and aims

From decision making and planning to systems design, optimization is a powerful means and even indispensable in cases such as large-scale logistics. Depending on the characteristics of the problem at hand, various types of optimization algorithms have been developed. The main considered application in this thesis is crashworthiness, in which an impacted structure should provide sufficient protection, i.e., there are constraints on intrusion and transferred forces to occupants. In this regard, safety can be achieved by adding components or weights, but at the cost of more energy consumption. Hence, optimization can be helpful to make a compromise among conflicting demands. Each crash simulation based on the explicit finite element method (FEM) can take a long time, the gradients are not reliably available and one mostly does not have access to the code, i.e., optimization has to handle a black box case. For such problems, algorithms based on response surface (RS) methods are good candidates, especially when design variables are continuous. Among the RS family, in this thesis we consider the Efficient Global Optimization (EGO) algorithm, which is adaptive and represents the underlying functions (i.e., objective and constraints) based on a Kriging or Gaussian process regression (GPR). EGO has two sub-optimization steps itself; their quality is essential for EGO success. This is in addition to the common case where the range and number of design variables have a large impact on the performance of any nonlinear optimization algorithm. EGO is efficient, but there is still room for more improvements, which are investigated in this thesis.

Although EGO is an established method, under the topic of crashworthiness one cannot find many realizations in the literature. For example, a search in the "abstract, title and keywords" section of

the Scopus database for the following keywords: "efficient global optimization, crash" or "EGO, crash, optimization" brings not even two pages of references as of January 2022. On the contrary, for example, in aerospace applications, various developments of EGO have been proposed and the method is quite popular. This is mainly due to the fact that multi-objective, multi-fidelity and multi-disciplinary optimization are considerably more common in aerospace engineering, see, e.g., Arsenyev (2016). In this work, crashworthiness is the main area of application, not aerospace. Nevertheless, we consider a condition for our proposed enhancements that they should not be too specific but rather general, in a way that later they can be (potentially) extended to some other applications as well. First, we review this limited EGO literature for crashworthiness.

One of the first publications using EGO in crashworthiness is Lee et al. (2002), where the energy absorption of a cylindrical tube is maximized by changing its radius and thickness. According to the authors, good results are obtained with only a few simulations and EGO is relatively independent of the noise (which one may expect from an explicit FEM simulation). In another direct application, EGO is used to optimize a low-speed bumper system crashworthiness problem (Kamel et al., 2008) and the authors found it to be better than a traditional surrogate-based approach. In Hamza and Shalaby (2014) EGO with multiple points per iteration (known as parallel EGO) is employed, which is obtained by optimizing different infill criteria. Here, the objective is to minimize the weight of selected structural components, while there are constraints on intrusion and peak force. A more demanding problem with the use of parallel EGO is solved in Sun et al. (2021), where for a bumper system, energy absorption is maximized and intrusion is minimized in three loading cases. In Raponi et al. (2021) optimization is used for two problems. One for defining the material properties of a composite component and the other one for optimizing a trigger using EGO from LS-OPT (Stander et al., 2019). EGO is relatively new in LS-OPT and according to the authors, has never been applied to such mechanical problems after their thorough search of the relevant literature. In the end, they conclude that EGO is a more convenient option with a lower cost. EGO has also been coupled with the CMA-ES (covariance matrix adaptation evolution strategy) algorithm to be used in the Level Set method (Raponi et al., 2019). This method is a topology optimization that represents a structure by the assembly of beams that can change center, direction, length, or be removed. Here, EGO brings efficient exploration and CMA-ES convergence to promising regions, and together they accelerate optimization. The state-of-the-art crashworthiness including the application of EGO and its variants such as parallel EGO is provided in Fang et al. (2017). The reviews so far do not address the enhancements of EGO proposed in this work. Other relevant reviews are rather given within the description of the EGO itself (Chapter 3), and partially in each of Chapters 4 to 6 when the main focus is on improving a specific feature or characteristic of EGO. Before going further, this main idea of improving EGO (for not very specific cases) raises a more fundamental question. Is it possible to have an optimization algorithm that is always among the best, regardless of the problem? In order to have a better insight, we have an introductory look at the "no free lunch theorem (NFLT)" next.

Wolpert and Macready introduced the NFLT for optimization (Wolpert and Macready, 1997). This



theory states that, by averaging all optimization problems, no optimization algorithm outperforms others (including random search) under any performance measure (Joyce and Herrmann, 2018). Some assumptions behind this theory are:

- Search space is finite (as do the outputs) but quite large.
- The visited points during the solution are distinct.
- There is no prior knowledge about the cost function and uniform admissibility of any cost function is assumed. Uniformity is not essential for the NFLT, as the theory holds for a range of non-uniform priors.

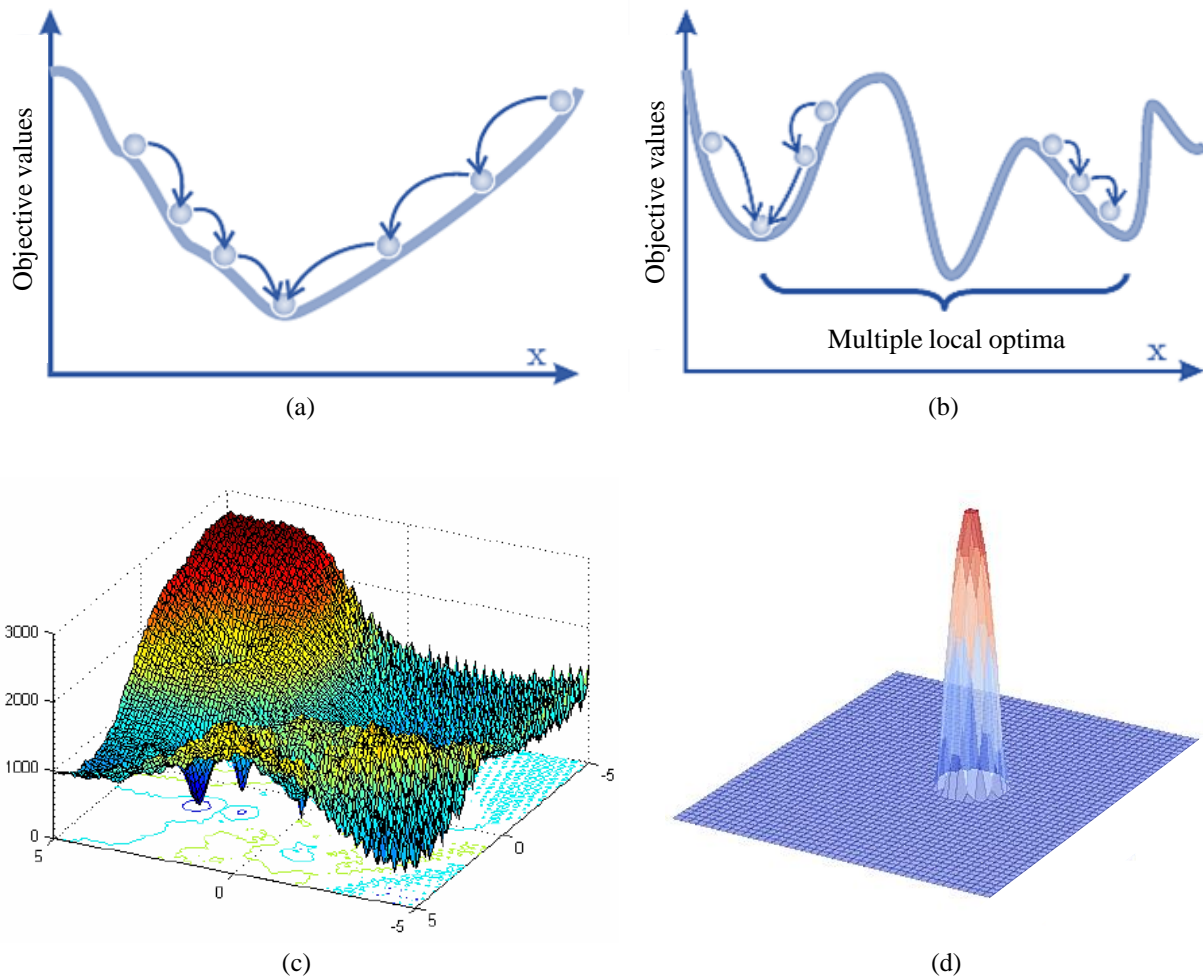
If we consider 'm' visited samples, the timely-ordered representation of the corresponding outputs (costs) is given by  $d_m^y = \{d_m^y(1), \dots, d_m^y(m)\}$ . So  $d_m^y(i)$  is the output of the  $i^{\text{th}}$  distinct visited sample. The authors further explain it by employing probability theory. Suppose that a certain algorithm 'a<sub>1</sub>' is repeated 'm' times for a certain cost function 'f', the cost of all samples ('d<sub>m</sub><sup>y</sup>') has the probability of  $P(d_m^y | f, m, a_1)$ . Then, they prove that if one sums all possible cost functions:

$$\sum_f P(d_m^y | f, m, a_1) = \sum_f P(d_m^y | f, m, a_2), \quad (1.1)$$

which means, for any performance measure, the considered algorithm has no influence on the result  $P(d_m^y | f, m, a)$  when averaging all possible cost functions. Extensions of this theory have been reviewed in Joyce and Herrmann (2018), among which there is a necessary and sufficient condition for NFL, where functions are sampled from a probability distribution called "block uniform". As mentioned above, an NFL assumption is the finite search space. Although many problems are continuous, a solution with computers involves a finite representation of the numbers, so according to the authors, the restriction of being finite is automatically met when using computers. However, for a continuous space, NFL does not hold (Auger and Teytaud, 2007). Yang reviews some arguments about the NFL (Yang, 2012), including the NFL assumption that samples are distinct and visited only once. He argues that this assumption is an oversimplification, since revisiting happens for almost all metaheuristic algorithms (except, e.g., for Tabu algorithms). He also refers to Wolpert and Macready (2006) that NFL does not hold for coevolution. Here, the aim is not to go into detail, but rather to understand the implications of the NFL for this work. Therefore, let us continue with the following example.

For a given optimization problem, which steps should be taken to find the optimum as fast as possible? Obviously, on the one hand, the selection of the next sample points depends on the structure of the optimization algorithm, e.g., on how it handles at least exploration and exploitation. On the other hand, it depends on the problem itself, e.g., the type of its landscape. In this regard, consider the cost functions depicted in Figure 1.1; plot (d) has a very sharp minimum surrounded by flat regions, and plot (c) has several shallow minima, but with a large basin of attraction and to some extent noise. Strategies to solve these problems efficiently can vary significantly. For example, (a) requires a local algorithm that converges very fast (highly exploitative), while (d) requires a global

algorithm that due to the lack of any exploitable structure should explore initially to a large degree to even find the basin. Contrary to (a) and (c), which require high bias to either exploration or exploitation, (b) is multi-modal and requires a balanced approach. (c) is similar to (b), but with noise and hence a successful algorithm on (b) may get stuck in (c). This example is by no means complete, but it is a showcase for a key concept, i.e., the importance of incorporating knowledge about the cost function in optimization algorithms. As Wolpert and Macready (Wolpert and Macready, 1997) put it, the lack of such incorporation leaves no formal assurance that the algorithm will be effective. Based on the above example, our aim of improving EGO is more refined now. We



**Figure 1.1** Four different types of cost functions. (a) and (b) are taken from Weise (2011) with modifications and (c) from (Liang et al., 2005). (d) is an Expected Improvement function from an adaptive stage of EGO.

achieve our goal by including information. Such information is either already available or can be deduced and employed for various aspects of optimization with EGO; from algorithm tendency and weaknesses itself (Chapter 4), to availability of analytical formulae for GPR and infill criteria in EGO (Chapter 5) to the demands from the problem, such as the presence of an active constraint (Chapter 6). In some cases, we combine information, e.g., the tendency of EGO to unnecessarily visit boundary regions is improved by comparing two approximations, one from the common GPR in EGO and the other one from a GPR with a different kernel (see Chapter 4). More details on the

proposed enhancements can be found in the next section, which describes the structure of chapters and especially, in the introduction of each of Chapters 4 to 6. In this work, we treat all problems as a black box, even if their analytical formula is available. We develop the proposed modifications and enhancements by trying them out not just on certain (similar) cases, but on various types of problem and at the end of each chapter, the results are applied to a crash simulation problem. Hence, the proposed modifications are developed and tested rather generally. In this way, we seek improvements for a group of problems and then apply the developed methods to a crash example and consequently avoid heavy tunings for a very specific crash problem. By considering various types of problem, it becomes also clear how a proposed improvement can adversely affect other cases. This potentially provides some insights for future developments.

### 1.1.1. Structure of chapters

In the next chapter, the preliminaries required for the rest of this work are provided. It starts with a concise review of nonlinear optimization and introduces the place of EGO within the corresponding classification of nonlinear algorithms. The following section introduces RS methods and some relevant concepts for EGO such as overfitting and uniform sampling, which are used in the first phase of EGO. GPR is a fundamental part of EGO and hence is reviewed in the last section of this chapter.

Chapter 3 presents the EGO algorithm and reviews various aspects of it. Several infill criteria are discussed and methods for handling constrained problems are compared. This review makes it more clear why EGO with certain settings is considered in the following chapters and paves the way for the proposed enhancements.

Chapter 4 is the first chapter that describes the methods developed in this work. There are several proposed enhancements based on how influential the design variables are, replacing the Gaussian kernel and proposing hybridization of methods. Some knowledge of EGO tendencies, such as its inclination to visit boundary regions, is also considered. Since there are several modifications, we test only a subset of all of their possible combinations to show general trends. The testbed includes various mathematical test functions, a linear truss structure and a side sill crash model. At the end of the chapter, the results are discussed and possible further improvements are provided.

In the first part of Chapter 5, the overfitting problem of GPR is examined more closely and a remedy is offered for isotropic Gaussian kernels. Since GPR is at the heart of EGO, a bad fitting leads to a wrong representation of the underlying function and hence, it degrades the efficiency of EGO in most cases. In the second part of this chapter, we propose to modify a common infill criterion called expected improvement (EI). The newly proposed EI is also suitable for both unconstrained and constrained problems and is tested over mathematical test functions similar to those of the first part of this chapter and in addition, on some mechanical design problems. In the end, the developed methods in this chapter are applied to shape optimization of a crash box. A discussion follows after each part to explain the results in more detail.

Chapter 6 also has two parts. In the first part, a new criterion is introduced for placing samples

along a certain value of a function, e.g., when a constraint is active. This criterion complements an existing one by being more exploratory. In the second part, the idea of following a certain value is expanded to a certain range. Here, multiple promising feasible solutions can be obtained instead of just looking for the optimum. The results are tested on three mechanical design problems to represent various behavior of the proposed method, not just in terms of obtaining multiple promising solutions, but also to compare the quality of the solution to that of the original EGO. At the end of the chapter, the developed method is tested on two crash models within the context of the solution space method. This method is not the focus of this thesis. However, it involves assuming a certain value for the displacement constraint, and hence, it is suitable for testing our proposed method. At the end of the chapter, the obtained results are discussed and the reasons behind the taken steps are explained.

Chapter 7 summarizes important concepts, methods and results of this work, especially from Chapters 4 to 6 and provides a critical review, discusses some issues, promising directions and good practices. This final chapter ends with raising some topics for future investigation.

## Chapter 2

### Preliminaries

This chapter introduces concepts and methods required for later sections of this work. It starts with nonlinear optimization, providing a concise classification of it and describing where EGO falls within. Several highly performant optimization algorithms are mentioned along with our selected one for optimization within EGO itself. The following section deals with RS methods. Relevant concepts such as the bias-variance dilemma are discussed and then an overview of the Design of Experiments (DOE) is provided. Various uniform sampling methods are reviewed, as the first phase of EGO is the evaluation of uniformly distributed samples. This chapter ends with a concise review of the curse of dimensionality and its consequences.

## 2.1. Nonlinear optimization

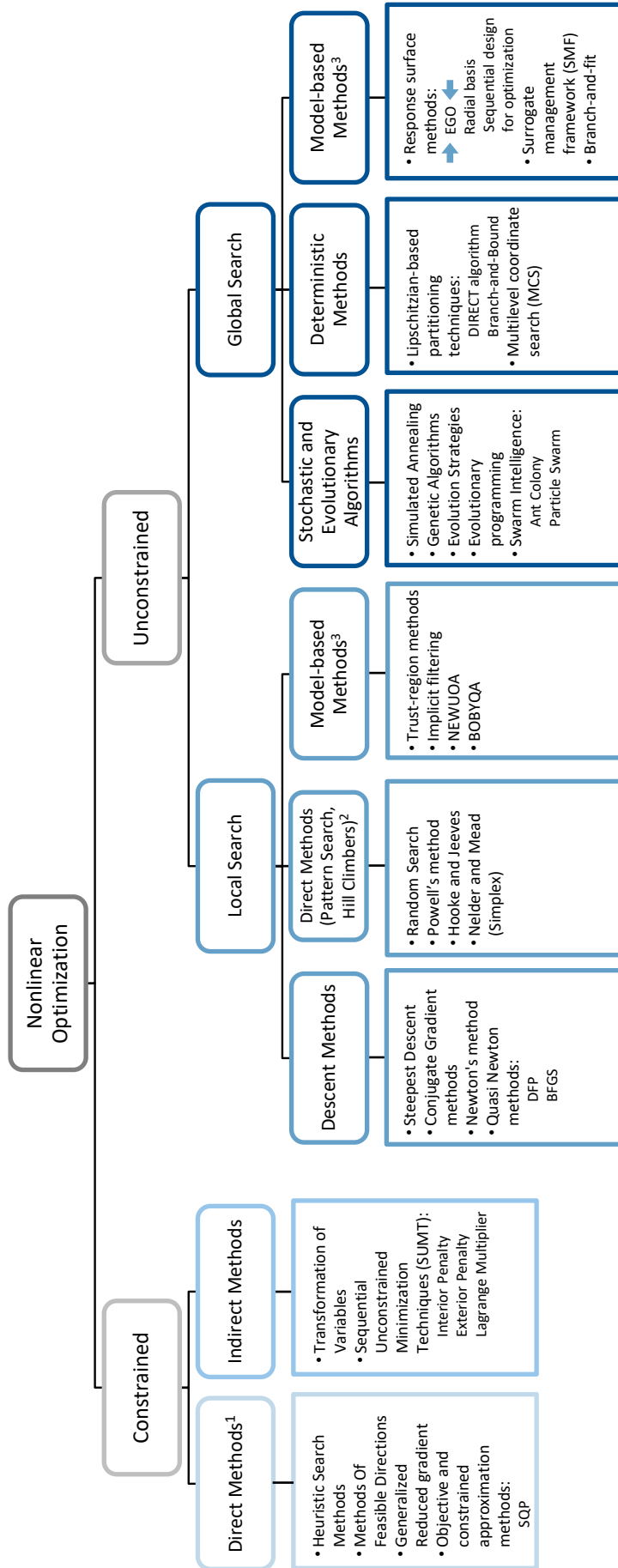
In a nonlinear optimization, at least one of the constraints or objectives is nonlinear. In this thesis, we consider only nonlinear single-objective problems with continuous variables. A typical problem is:

$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) \\ & \text{subject to} && c_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m. \\ & && h_j(\mathbf{x}) = 0, \quad j = 1, \dots, n. \\ & && a_k \leq x_k \leq b_k, \quad a_k, b_k \text{ and } x_k \in \mathbb{R}, \end{aligned}$$

where  $f(\mathbf{x})$  is the objective function,  $c_i(\mathbf{x})$  are the inequality constraints and  $h_j(\mathbf{x})$  the equality constraints. Each design variable  $x_k$  is also limited to the range  $[a_k, b_k]$ . Figure 2.1 classifies methods for solving nonlinear optimization problems based on the presence of constraints. A constrained problem can be solved by methods that directly consider constraints in their solution process. Sequential Quadratic Programming (SQP) is one of the most famous algorithms in this class that requires derivatives. Other ways of tackling constraints are mentioned under the label of "Indirect Methods" (Figure 2.1), which convert constrained problems to unconstrained ones, and hence, one of the unconstrained methods (right branch of Figure 2.1) can be used to find their optimum. For example, penalty methods add constraints to the objective function as a penalty term, when constraints are violated, and we use this approach in Chapter 4 of this work. Methods for solving unconstrained problems can be divided into two main categories of "Local Search" and "Global Search". Algorithms under Local Search tend to attract to the closest basin (local optima), so they are suitable for uni-modal optimization. The most efficient local search algorithms are categorized under "Descent" methods which require the availability of the first- or second-order derivatives of the underlying function. The highest convergence rate in "Descent Methods" belongs to the Newton family, but as it requires the computation of the Hessian, which can be quite expensive or non-reliable, other methods such as "Quasi-Newton Methods" have been developed, see Parkinson et al. (2018). The famous BFGS (Broyden–Fletcher–Goldfarb–Shanno) is from this category. A hybrid of this algorithm and DIRECT (a global search, deterministic method) is used to optimize the likelihood function in GPR (Butler et al., 2014). Another subtype of local search is the "Direct Methods" family with a lower convergence rate than descent methods. However, direct algorithms work only with function values, which is beneficial when gradients are not available or reliable. Compared to local search algorithms, global ones are better at finding optima in a multi-modal problem. Any local search can be made global by starting the algorithm from several points and selecting the best result at the end (i.e., multi-start strategy). Genetic algorithms and evolutionary strategies are among the most famous global search methods. Numerous stochastic and/or bio-inspired algorithms such as "particle swarm" have been developed in the last couple of decades. Some algorithms in both local search and global search may also use a surrogate model to approximate the underlying function. EGO belongs to the global branch (pointed to by two arrows

in Figure 2.1). Model-based derivative-free algorithms typically need at least "dimension + 1" samples to set up the first model. However, in subsequent iterations, only some of the samples are updated and therefore the evaluation cost is significantly reduced (Sahin et al., 2019). The classification here does not cover hybrid types, such as memetic algorithms (MAs) (Moscato, 1989) and coevolutionary algorithms (Hillis, 1990). MAs are hybrids between evolutionary algorithms (EAs) and local search techniques, which often exhibit both the robustness of EAs and the speed of local search methods at the same time (Pošík and Huyer, 2012). In coevolutionary algorithms, there are two or more populations instead of a commonly single population EAs, which constantly interact and adapt to each other's change (Boussaïd et al., 2013). In addition to hybridization, many attempts have also been made to improve global search algorithms by borrowing some features from other methods. For example, evolutionary strategies (ESs) were initially based on mutation, but they were then enhanced by taking the crossover concept from the genetic algorithm (GA). Other enhancements include making global search methods self-adaptive, so that they can learn the landscape and have a better convergence rate, see, e.g., Harrison et al. (2018).

As mentioned earlier, within EGO two optimization problems are solved in each iteration. The landscape of these two problems could be quite challenging for any optimizer, as will be discussed later. Here, we mention a couple of promising global and local algorithms that one can use for such problems. The local algorithms require strategies like multi-start to handle multi-modal cases and therefore become suitable for optimizing an infill criterion. Consequently, one or multiple infill points can be obtained. The Nelder-Mead simplex method is one of the most successful in low dimensions ( $d = 5$ ). However, it becomes very inefficient quite before  $d = 20$ , see Pošík and Huyer (2012) and Rowan (1990). This degradation emerges after some iterations, where the simplex enters a subspace and subsequent operations, i.e., expansion or reflection, simply change the location of the new proposed sample within this lower dimension, while there is a little chance of recovery according to Chu et al. (2011b). The authors in this article also suggested using principal component analysis (PCA) to identify these spuriously eliminated dimensions and bring them back into the search space again. Other strategies are to restart with a new simplex from the best available point or the sublex method, which decomposes the problem into subspaces, where the simplex is efficient, see Rowan (1990). For larger dimensions, BOBYQA (bound optimization by quadratic approximation, (Powell, 2009)) is a highly capable local search algorithm. Categorically, it comes from decent model-based optimizers like EGO, but from the local branch. Py-BOBYQA is a Python implementation of it, which improves handling noisy cases by a multi-start strategy. An adaptive restart strategy (changing the size of the trust-region) has also been suggested to make Py-BOBYQA a global optimizer, see Cartis et al. (2018) and Cartis et al. (2019). Now we introduce some (highly) performant global algorithms, along with mentioning some relevant points for this work. Differential evolution (DE) from the SciPy package (Virtanen et al., 2020) is an adaptive algorithm that is extensively used in this thesis. Even the original DE itself works well on low-dimensional problems ( $d \sim 10$ ) (Chu et al., 2011a). The reasons why DE is a successful algorithm are summarized in Feoktistov (2006). Even at international competitions, a family of



- Stochastic: requires random search steps.
  - Deterministic: does not require random search steps.
  - Model-based Methods: a surrogate model is used to guide the search process.
  - This table is by no means complete. E.g., there are hybrid algorithms based on the local and global search classes.
1. Directly considering constraints in the solution process.
  2. Values of the function is used not its derivative.
  3. A surrogate model is used to guide the search process.

**Figure 2.1** Methods for solving a nonlinear optimization (Rao, 2009), (Keane and Nair, 2005), (Rios and Sahmidis, 2013), (Kumar et al., 2016). EGO is pointed to by two arrows.



winning algorithms has been developed from DE. However, in recent competitions, other successful algorithms have emerged, e.g., from the MVMO family (mean variance mapping optimization) (Molina et al., 2018). We borrowed some features from MVMO to hybridize it with EGO in Chapter 4, where MVMO is also briefly described. In addition to DE and MVMO, another main influential algorithm in international competitions has been CMA-ES (Molina et al., 2018). This algorithm is introduced in Hansen and Ostermeier (2001) based on generating random samples from a multivariate normal distribution and then moving and reshaping the distribution, i.e., by updating its mean and covariance. These updates are set skillfully and the reason behind each step is very well explained in Hansen (2006). CMA-ES is highly efficient (as a derivative-free method) in converging to the optimum of a basin it finds. However, on the other hand, it may completely get lost, as the experience of the author of this work shows. Due to this behavior, CMA-ES is not among the most reliable algorithms to be used within EGO, especially given the nature of the corresponding fit and infill functions. CMA-ES was later enhanced to BIPOP-CMA-ES (bi-population covariance matrix adaptation evolution strategy) to improve exploitation (even more) and more importantly exploration by adding two interlaced restart strategies. One with an increasing population size and the other with a varying small population size (Hansen, 2009). This algorithm became the winner of the CEC-2009 competition (Hansen et al., 2010). One important point to mention is the fact that BIPOP-CMA-ES can be completely outperformed over some test functions in this competition (Hansen, 2009). These functions are from separable and multi-modal classes. This is an example of the NFLT in practice. The difficulty that this algorithm with multi-modality has, is described above and its problem with separability of inputs is to some extent clear from the way that BIPOP-CMA-ES is set up, i.e., the existence of the covariance matrix, which defines the interaction between design variables. Hence, the CMA-ES family is by nature expected to be more efficient on non-separable problems. For example, CMA-ES outperforms the standard PSO (particle swarm optimization) algorithm by orders of magnitude, while the table turns in the case of separable problems and PSO outperforms CMA-ES up to five times (considering the number of function evaluations) (Hansen et al., 2011). This promising performance by PSO on separable problems has also roots in the way this algorithm is designed. PSO updates each sample based on the difference between the personal best and also with the population best and there is no interaction term involved. For a more detailed comparison, see Hansen et al. (2011). This makes PSO potentially a good candidate for engineering problems where there are at least some weak relations between some of the inputs. In conclusion, CMA-ES and PSO are complementary in certain types of problems. Enabling an algorithm to deal with complementary cases is one way of improvement and the idea behind some proposed modifications in Chapters 4, 5, and 6. In this thesis, like any other work, some relevant aspects remain uncovered. Therefore, we now provide some references for the interested readers. A discussion on the selection of algorithms and comparison of various methods can be found in Rios and Sahinidis (2013), Kumar et al. (2016), Singh and Jana (2017) and Molina et al. (2017). Large-scale global optimization algorithms are reviewed in Mahdavi

et al. (2015) and dealing with multi-modal cases is covered in Preuss (2015).

## 2.2. Response Surface Methods

The evaluation of a black-box system is a procedure in which inputs are passed to the system, processed by it and outputs are measured at the end. In this thesis, such a system is a numerical model based on the FEM. In case of optimization, this procedure has to be performed many times, but in practice, the system can be computationally expensive to evaluate. Therefore, one solution is to obtain the outputs ( $\mathbf{y}$ ) only at certain inputs ( $\mathbf{X}$ ) and then use the obtained samples ( $\mathbf{X}, \mathbf{y}$ ) to approximate the response of the system at unknown locations. Here, the term response surface (RS) is used to refer to this approximation of the response, based on current samples. There are various methods to do such fitting and utilize it. These methods are collectively gathered under the name of response surface methods (RSMs). One can consider them as supervised learning methods in machine learning. For this reason and the fact that machine learning literature is more developed, we will refer to machine learning literature in this work. A response surface is also called a surrogate model or a meta-model. But, it is perhaps more precise to consider RSMs as a certain type of surrogate models. As Iuliano (2011) puts it, surrogate modeling approaches can be divided into three groups:

### 1. Multi-fidelity models

A high-fidelity model refers to a system that one would like to evaluate (e.g., crash simulation), but it is (usually) expensive to do so. Therefore, a simplified model of the system (in terms of physics or mathematics) is developed and called a low-fidelity model. This model is cheap to evaluate many times and the intention is to be representative of the main high-fidelity model well enough around points at which a high-fidelity system has been evaluated so far and hopefully even further away. In multi-fidelity optimization, both models are utilized. It is common to have many evaluations of the low-fidelity model with limited evaluations of the high-fidelity one, so in the end, better results are obtained (under a given time) compared to just optimizing the high-fidelity model on its own. Low-fidelity models can be mathematical or physical, see Duddeck and Wehrle (2015). In physical models, a component can be replaced by a simpler one(s), e.g., the frontal part of a car by a system of mass, springs and dampers as proposed in Marzbanrad and Pahlavani (2011). Thin-walled structures in a car (usually represented by shell elements) can be replaced by a combination of concentrated masses and rigid parts with joints (basically springs) in between. These simplified models can be found in Cornette et al. (1999) and Brell (2005). Cars' cross-members have been represented by super-elements and equivalent beams (having the same masses and moments of inertia as the original model) in Liu (2005). Multibody dynamics works with these simplified elements and hence can be considered as a framework for simulating with low-fidelity models, see Sousa et al. (2008) and Ambrósio (2005). Although the replacements mentioned above can reduce the number of degrees of freedom and hence evaluation time, they cannot completely replace the high-fidelity nonlinear crash models, especially away from points where the system has been evaluated. The

properties of these simplified elements can be obtained, for example, by finding the unknowns of a mathematical model representing the stiffness (via an RS), see Lee et al. (2002), or generally speaking, by fitting an RS to the component's characteristic force-displacement, e.g., see Carvalho et al. (2011). In both cases, the unknowns are found by minimizing the difference between the low- and high-fidelity models.

Another type of low-fidelity models is used in substructuring. Here, there are some important parts in the problem at hand and the aim is to keep them unchanged, while the rest of the structure will be removed or represented by boundary conditions. Optimization can be done on substructures (low-fidelity) and validated by the original model (high-fidelity), see Chase et al. (2012). It should be noted that if substructuring is applied to the system of equations, it is a type of reduced-order model approach (next surrogate group). Equivalent static load methods are another way of achieving physical low-fidelity models. The dynamic load in the transient analysis is replaced by these equivalent static forces; either a set of equivalent forces is applied globally and their value changes based on crash phases (when changes occur, for example, the engine starts contacting the firewall) as in Duddeck and Volz (2012), or these equivalent loads are applied on nodes for each time step as in Shin et al. (2007). In addition to low-fidelity models, high-fidelity ones can also be represented by RSMs, e.g., in Co-Kriging (see Forrester et al. (2007)), in which a Kriging RS represents both low- and high-fidelity models.

## 2. Reduced order modeling

The aim is to project the previously analyzed full-order simulations to a reduced space and therefore make faster calculations while keeping the loss of accuracy acceptable. Hence, in a model order reduction (MOR) method, the resultant vectors of the training simulations at the selected time steps are placed in a so-called snapshot matrix and then, a new basis for projection into a smaller space is calculated using a proper orthogonal decomposition or similar technique. However, there will still be some characteristic quantities, such as the evaluation of internal force vectors that depend on the full order space. To tackle this bottleneck, hyper-reduction techniques were introduced to further improve efficiency gains. One should note that MOR methods are economically viable if saving in calculations by using the reduced order model is high enough to surpass the cost of the reduction and related steps. Hence, it is usually meaningful to call the reduced model more than just a handful of times (e.g., in optimization or uncertainty evaluations). If the projection is linear, it can preserve the physics of the system well enough, while in a nonlinear projection (e.g., kernel PCA), this information can get lost, so as in data-fitting (i.e., RS) methods. This preservation of physics is helpful, e.g., in reducing the model based on keeping only a couple of first natural modes of the underlying system and eliminating the rest, see Koutsovasilis (2009) and Eid (2009) for more information. In nonlinear cases, projections can be linear, piecewise linear or nonlinear (e.g., kernel PCA). Piece-wise linear is conducted by dividing the space and applying a certain linear transformation for each section, see Chapter 2 of Bach (2019) for nonlinear MOR. Another way of reduction is to

separate the nonlinear part from the linear one at the equation level (e.g., linear and explicit formulations) but also couple the regions to keep the required continuities, see Faucher and Combescure (2003).

### 3. Data-fitting surrogates

This type was described above as RSMs; fitting over a set of data to use it for prediction at untried inputs.

Note that there are various ways of categorizing, e.g., data-fitting surrogates may be considered as a category of reduced-order modeling, and what was described above as reduced-order modeling can be referred to as projection-based ones.

As mentioned before, RMSs are not only making one of the surrogate model groups, but can also contribute to the other two groups. Some common response surface methods are briefly described below:

- **Polynomial regression**  
Employed mostly up to second-order; with increasing order, the required number of samples increases rapidly and there is a possibility of overfitting. Due to its inflexible shape, accuracy can be significantly sacrificed, but it is suitable for noisy cases. In RS-based optimization, a polynomial surface is usually fitted over a trust-region-like subspace and it is iteratively updated using samples partially from previous iterations and partially from the new one, e.g., see Wang (2003). This approach used to be more common in RS-based optimization, but newer methods have gained more popularity.
- **Support vector machine (SVM)**  
SVM is used for regression and classification, while several kernels are available. It is especially useful when the ratio of the number of samples to dimension is low, as it is less vulnerable to overfitting; but that comes at the cost of a simpler representation of the underlying function. Good performance of SV depends on the proper settings of hyperparameters (Basudhar et al., 2012). Based on the author's experience, fitting time is not reliable and it can be long in some cases (relative to the size of the problem). Although it has been used for RS optimization (see, e.g., Pan et al. (2010)), it is currently not a common method.
- **Radial basis function (RBF)**  
This family has different kernels with different degrees of smoothness, i.e., it is suitable for various data sets. It fits fast and based on the author's experience, has a good balance between accuracy and overfitting. RBF has been used in RS optimization, for example, in Holmström et al. (2008) and Bhosekar and Ierapetritou (2018). Some promising results in surrogate modeling or surrogate-based optimization are mentioned in Fang et al. (2017).
- **Gaussian process (GP) and Kriging**  
In general, these are powerful methods to represent various functions and hence they are more prone to overfitting. Since they provide not only a prediction but also its variance, a natural criterion for estimating confidence interval and exploration is already available. This has made

these methods versatile in different fields, including RS-based optimization. We describe GP and Kriging later in this chapter in detail.

- Artificial neural networks (ANN)

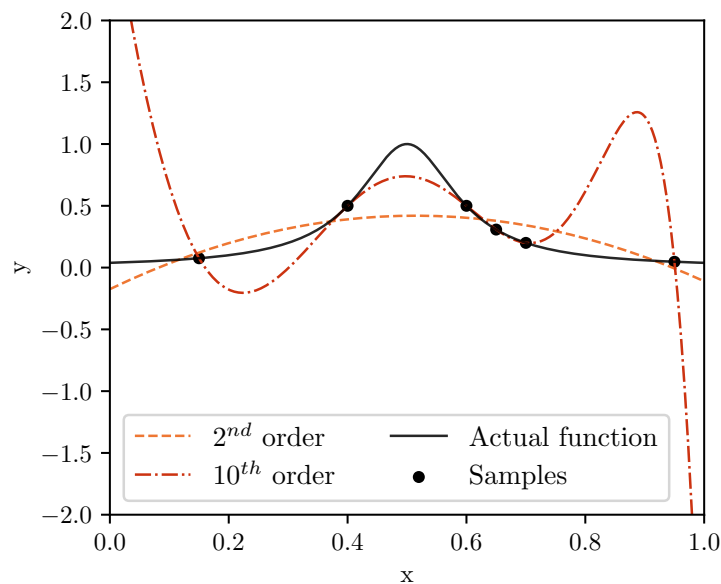
The most powerful method. Due to its multi-layered structure and how these layers are connected, ANN can represent complicated nonlinear outputs that other methods cannot even approach, e.g., image processing. However, here we are only interested in representing the output of a function by a response surface and not a classification of complicated patterns. So, GP is still powerful enough, with the advantage that it can be both interpolative and regressive. One ANN problem is the selection of the number of layers and neurons, which is not clear in advance and one needs to go into a procedure to balance accuracy and generalization (i.e., avoiding considerable overfitting). Given the settings, fitting can take a while and results can change to some extent due to the high multi-modality of the fitting criterion. ANN is more suitable for complicated tasks that need to be fitted once and used many times. Therefore, it is not that common for RS-based optimization, especially for adaptive ones. However, it has been used in an ensemble of RS methods to improve predictability.

- Ensemble (hybrid) of methods

It can be deduced from the "no free lunch theorem (NFLT)" that there is no single RS which is always the best and can represent all underlying functions in the most efficient way (Garbo and German, 2017). Hence, an ensemble of RS may be employed to be more successful in various underlying functions. This ensemble can be defined over various forms of a certain RS, like "random forest" which is a hybrid of decision trees, or over different RSMs. Consequently, optimization algorithms based on the hybrid of RSMs have been developed, which make use of the ensemble in two ways; either a hybrid model is constructed by a weighted sum of the considered RSs as reviewed in Ye et al. (2018), or each RSM makes its own suggestion for the next samples as in Viana et al. (2013). It is common to combine several RSs, for example, polynomial RS, RBF, and Kriging were combined in Goel et al. (2007), while the weight of each surrogate was defined based on its error. Pointwise cross-validation was used as a local measure among others to define the weights in Acar (2010). Zhang and colleagues employed second-order regression, radial basis, extended radial basis and Kriging (Zhang et al., 2012). They used sample density to define a trust-region within which the weight of each surrogate model is determined based on a local accuracy measure and hence an adaptive hybrid surrogate model is developed to capture local and global accuracy. It should be noted that some of the ensemble methods that are used for optimization employ the uncertainty from Kriging, to select several sample points, see Viana et al. (2013) and Cai et al. (2018). In case of the latter, based on the difference in the prediction of meta-models, the next sample points are generated from the better-performing meta-models gradually. In the end, when it is decided that the global optimum is close, a local search strategy is used to further refine the solution.

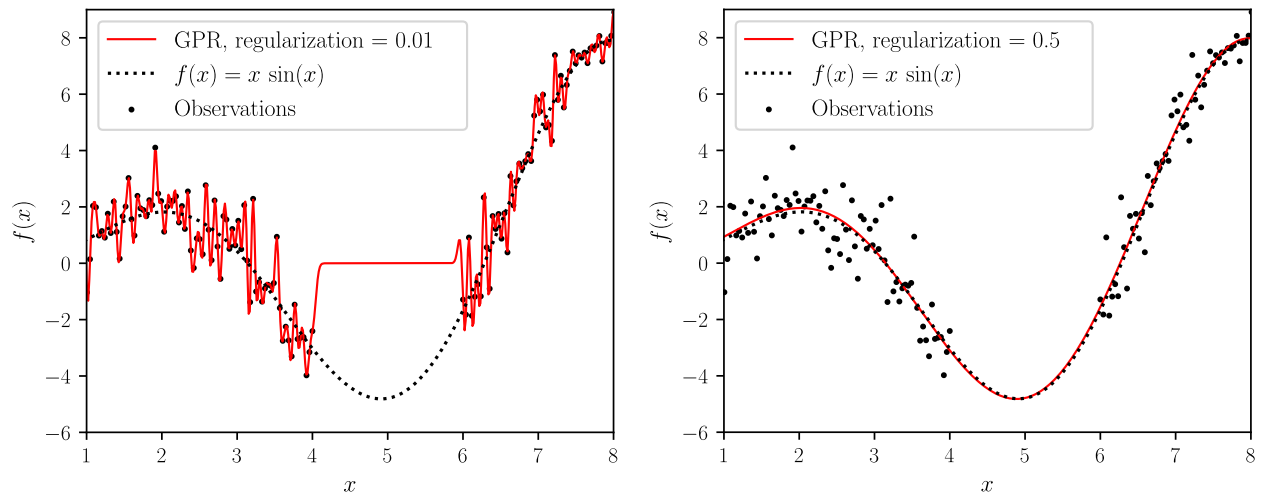
An important concept when dealing with RSMs is the evaluation of the model error due to bias

and variance. Bias is a lack of accuracy or, better to say, the difference with a target, i.e., RS approximation is different from the actual output at the already evaluated points. For example, a second-order polynomial regression usually has a large bias and underfits data (unless the underlying function is indeed simple and close to a second-order regression). In addition to this, high variance means, roughly speaking, that the model is able to fit various forms over the given data. For example, a tenth-order polynomial regression or GPR can fit different shapes over a certain data set, but a second-order polynomial regression cannot. Therefore, there is a possibility of selecting the form that learns (fits) current data well, but makes errors when it predicts at unsampled locations, i.e., generalizes inappropriately. This error is called overfitting (see Kuhn and Johnson (2013) and Figure 2.2). Noise can worsen overfitting, but severe overfitting exists in noiseless cases too (Domingos, 2012). We saw that GPR can have a large variance but, on the other hand,



**Figure 2.2** Polynomial of  $2^{nd}$  order is underfitting and of  $10^{th}$  order is overfitting.

a low bias (fits the current data well). In conclusion, either the accuracy can be high (low bias) along with the risk of overfitting (high variance) or vice versa; the risk of overfitting can be kept low at the cost of accuracy. In machine learning, the proper way to perform supervised learning (e.g., fitting an RS) is to divide the data set into two parts, one for fitting and one for final testing of the quality of the fitted model. The fitting part may also be divided again into two fitting/testing parts on its own to find the unknowns of the RS (i.e., fitting the RS) by minimizing the error on this second testing part. After fitting the model, it should be validated against the first testing part of the samples (called final testing above). If the error due to this final validation is significantly larger than the error during the fitting, then overfitting is happening and if it is significantly less, then underfitting exists. In either case, modifications must be carried out and the procedure is usually repeated until error levels are close enough. In other words, we try to have a trade-off between bias and variance, see Géron (2017). However, this proper fitting procedure is not commonly practiced in RS-based optimization, and the entire data is used for fitting. Although one may argue that



**Figure 2.3** The figure on the left shows an example of fitting noisy data very well, but making poor predictions (within (4,6)) while there is a regularization. By increasing the regularization to a sufficient level, predictions become much better, but noisy data is now regressed, so accuracy is lost (not a bad feature here). The left figure is the preferred one.

in RS-based optimization, one cannot afford a lot of data, let alone setting aside some of it for testing; but, on the other hand, it should not be forgotten that using the entire samples for fitting will increase the chance of overfitting for complex methods like GPR. One important implication of the bias/variance trade-off is that a more complex model is not always better than a simpler one. This may be overlooked easily, as a more complex model is able to fit the current data better and one can measure this and hence may automatically assume that the complex model also makes better predictions at untried points. However, this is not always true. There should be enough evidence (i.e., samples) to set the unknowns of an RS to proper values. Having a high number of samples relative to the RS unknowns can reduce the risk of overfitting to a large degree, but does not necessarily eliminate it. An ensemble of RSMs is a way to improve the bias-variance trade-off. One way of explanation is that RSMs with similar bias are combined (e.g., by averaging) to have (hopefully) less variance. The combined model is not necessarily better than the best model in the ensemble, but usually reduces the possibility of having an RS with weak performance (Zhang et al., 2012).

### 2.3. Design of Experiments (DOE)

DOE deals with the ways to define the location of the samples. Historically, it predates the computer era and was developed for physical experiments and consequently, a limited number of evaluations. With the advent of computers and consequently numerical simulations, the need for different and more flexible ways of sampling has emerged and as the result, the DOE literature has expanded.

In classical DOE, sampling is deterministic, i.e., the location of samples is predefined. In many classical designs, such as full factorial, samples are placed at the boundaries of the design space, so the variation of the response can be estimated over the entire range of the inputs. Some designs also consider central points to increase the order of the fitted polynomial to two. But, since physical

experiments are expensive, fitting a complete second-order model may be unaffordable. Therefore, some fractional designs were developed to allow keeping only important terms, yet having the possibility of using some second-order terms. The number of samples required for classical designs grows very quickly with the dimensionality of the problem, making these designs impractical in higher dimensions. Another characteristic of classical designs is that they place a lot of samples in boundaries and fail to adequately cover the interior of the design space, an attribute that may not be desirable to users in lower dimensions. For more information on classical DOE, see Montgomery (2012).

The need for more flexible sampling plans has led to new sets of methods, some of which try to fill the space as uniformly as possible. In this regard, two major criteria help to measure the space-filling properties of a sampling method. One is based on the discrepancy and the other on distance. The following summary is taken from Garud et al. (2017). First, we define discrepancy; the aim is to fill a hyper-rectangular design space called  $S$ . Consider  $H$  as a subspace of  $S$  with the volume  $V(H) = \Delta x_1 \Delta x_2 \dots \Delta x_N$ . If '#' represents the number of samples, then the discrepancy is defined as:

$$D = \sup_H \left| \frac{\#(\mathbf{x} \in H)}{\#S} - V(H) \right|. \quad (2.1)$$

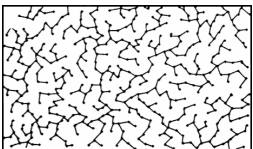
The lower the discrepancy, the more uniformly the samples fill the design space. The formula basically means that the ratio of the number of samples in a subspace should be proportional to the volume of the subspace. This formula is not straightforward for numerical calculations, so a version of it based on the  $L_2$  norm is developed (see Garud et al. (2017)).

Some distance-based criteria are given in Table 2.1. Among the listed criteria, all except Minimax try to put samples further from each other. Some developed methods for obtaining a uniform design include: Monte Carlo, Quasi-Monte Carlo, stratified sampling, Latin hypercube, orthogonal array and so on. Uniformity can be enhanced even more by optimizing a design using one of the above-mentioned criteria. Logically, the most efficient way of sampling is the one that is tailored to the problem at hand, i.e., there should be more samples in the regions where output changes fast. However, this does not happen in practice, as the underlying function is mostly unknown to us. Therefore, it is better to put samples uniformly in the design space until more information about the space is obtained, e.g., through adaptive sampling. For more information, see Garbo and German (2017).



**Table 2.1** Some distance-based criteria for uniformly distributing samples in the design space.  $\mathbf{x}^{(j)}$  is the  $j^{\text{th}}$  sample.  $N$  is dimension and  $M$  is the number of samples. Here,

$d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)}) = \sum_{i=1}^N |\mathbf{x}_i^{(j)} - \mathbf{x}_i^{(k)}|$  and  $d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})^p$  is a distance function where  $p > 0$  defines the distance norm.

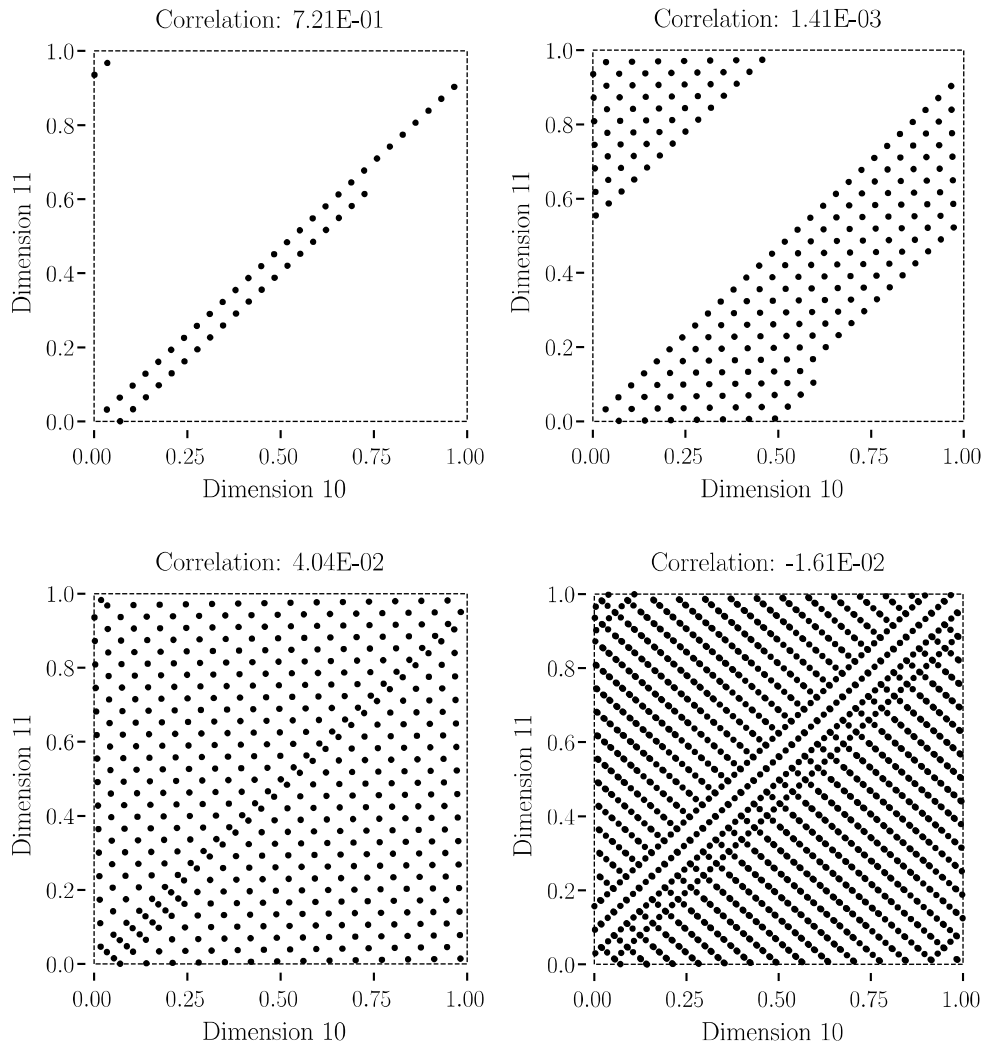
Method name	Formula	Explanation
Morris and Mitchell $\phi_p$	$\left[ \sum_{k=1}^{M-1} \sum_{j=k+1}^M d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})^{-p} \right]^{1/p}$	(Morris and Mitchell, 1995), the lower the $\phi_p$ , the more uniform the design.
Audze and Eglajs potential energy	$\sum_{k=1}^{M-1} \sum_{j=k+1}^M d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})^{-2}$	Special case of Morris and Mitchell $\phi_p$ criterion with Euclidean distance. The square root of the result is not considered as it does not change the uniformity order among the considered cases.
Maximin	$\max \left[ \min_{j \neq k} [ d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})^p ] \right]$	$d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})$ is defined as the distance function in Johnson et al. (1990), but it is redefined here to be consistent with other definitions.
Minimax	$\min \left[ \max_{j=1,2,\dots,n} [ \min_{j \neq k} [ d(\mathbf{x}, \mathbf{x}^{(j)})^p ] ] \right]$	Minimizing the maximin criterion (Johnson et al., 1990). Optimizing this criterion is more difficult than optimizing the Maximin, see Pronzato (2017) which also provides some optimization methods.
Minimal spanning tree (MST)		Samples are considered as nodes in a graph and they are connected while the sum of distances (i.e., edges' lengths) is minimized, involving all samples and without making closed loops. This is called a tree and it is made by starting from one sample and adding others step-wisely (Dusser et al., 1987). In the end, if the mean of all edge lengths is large and the corresponding standard deviation is low, the current set of samples has a good space-filling character.

### 2.3.1. Quasi Monte Carlo

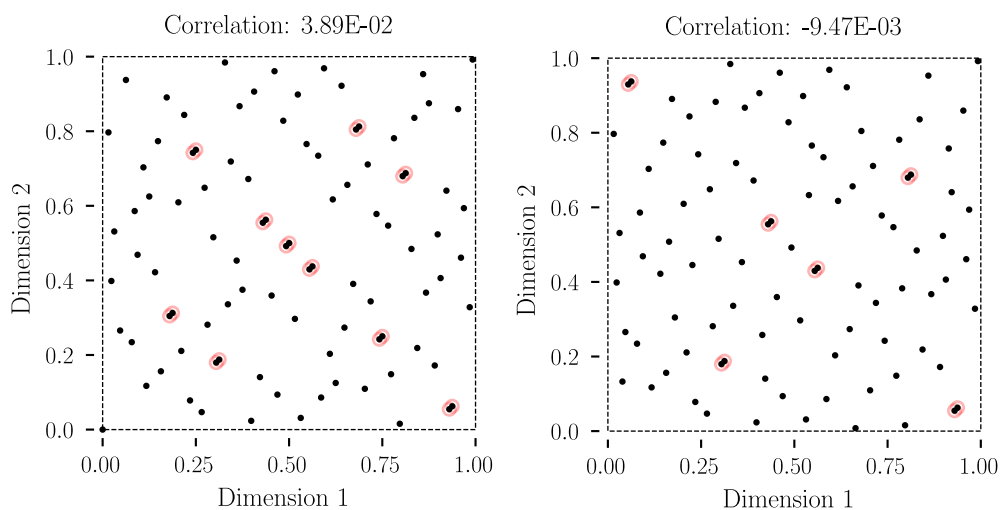
It is known that sampling from a uniform Monte Carlo scheme does not lead to a uniform distribution of samples in the design space unless the number of samples is (impractically) high. Therefore, Quasi-Monte Carlo methods (QMCMs) were developed to have better space-filling properties over a unit hypercube with the initial intention of calculating integrals in higher dimensions. Hammersley, Halton, Faure and Sobol sequences belong to this family. These methods create samples based on a sequence of prime numbers. In Halton, numbers are generated from basis functions, which are only prime numbers. In each dimension, a sequence is generated from a certain but unique prime number, e.g, the sequence  $\{1/2, 1/2^2, 3/2^2, 1/2^3, 3/2^3, 5/2^3, 7/2^3, \dots\}$  is made from prime number 2. Then, the weighted sum of the terms in the sequence can create any integer number for that dimension. If two prime numbers are close to each other and the number of samples is not high enough, it leads to samples lining up and hence correlations between corresponding dimensions, see Antinori (2017) for more information. Increasing the number of samples will reduce correlations. This problem becomes more and more significant in higher dimensions and especially for 2D projections of samples, when both projected dimensions are closer to the sample dimension. According to Bratley and Fox (1988), this can be improved by an advanced initialization using a Faure sequence, which can be even better than Halton. Bratley and Fox also provided the implementation of Sobol for up to 40 dimensions (here called Sobol1), which was later increased to 1111 dimensions (Sobol2) in Joe and Kuo (2003). Figure 2.5 left shows that some samples are much closer to each other than to the rest. This non-uniformity can be alleviated by skipping several first samples (right figure). Another common pattern also appears in the 2D projection of Sobol in higher dimensions. In some of the projections, a concentration of samples similar to what is shown in Figure 2.6 can occur. If more samples are generated from Sobol, the gaps in the figure will be occupied by them. However, even for an affordable number of samples, these patterns can still occur and indicate that sampling uniformity should be improved. Joe and Kuo (Joe and Kuo, 2008) fixed some of these poor 2D projections while keeping Sobol still a fast method (this version is called Sobol3). Similarly to other QMCMs, the original Halton has also improved over time. As mentioned earlier, the larger the prime number, the more pronounced the lining-up and hence the correlation problem. This is improved by the generalized Halton approach as mentioned in De Rainville et al. (2012), together with providing references to further modifications. The authors went one step further by employing an evolutionary algorithm to tune the possible permutations of the Halton sequence to achieve a lower discrepancy set of samples.

### 2.3.2. Latin Hypercube

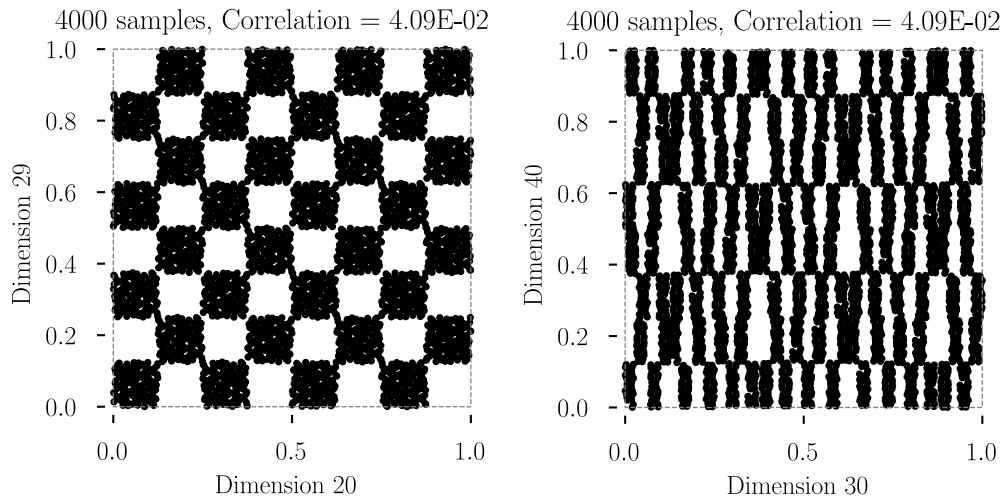
In a Latin Hypercube design (LHD), each dimension is divided into  $n$  equal levels, where  $n$  is the number of samples to be generated. Therefore, the space is divided into  $d^n$  cells, where  $d$  is the dimension. Then,  $n$  cells are randomly selected under the condition that there is only one sample at each level. This additional condition is what makes LHD different from uniform stratified sampling. One advantage of LH is that, if some dimensions are removed, the remaining design will still be an LH. It may be less uniform and more correlated, but it is still an LHD and hence data



**Figure 2.4** Halton in 15-dimensional space. The line-up of samples is shown in a 2D projection on dimensions 10 and 11. The correlation between these two dimensions decreases as the number of samples increases.



**Figure 2.5** Some samples can get very close to each other compared to their distances to other samples, see the highlighted sample pairs. The right figure shows that this non-uniformity can be alleviated by skipping some initial samples (here, 10).



**Figure 2.6** Two common Sobol patterns in which samples are concentrated in 2D projections.

points do not collapse on each other by removing dimensions (Viana, 2013). However, LHD can suffer from correlated dimensions and unsuitable uniform space-filling. Therefore, modifications have been proposed to improve the method. Ye introduced an orthogonal LHD in which the correlation is zero between all dimensions (Ye, 1998). However, he also mentions that although the correlation problem is solved, this does not necessarily translate into good space-filling properties. By slightly relaxing the orthogonality requirement and hence allowing small correlations (e.g., within  $(-0.03, 0.03)$ ), a strong improvement in space-filling can be achieved (Cioppa and Lucas, 2007). Optimal LHD (OLHD) is perhaps the most powerful improvement over LHD, providing good space-filling properties. The considered objective function is one of the uniformity criteria discussed earlier (Table 2.1). One disadvantage of LHD is the exponential growth of possible designs as fast as  $(\text{number of samples!})^{\text{dimension}}$  (Viana, 2013). Therefore, creating a high number of LH designs and comparing them based on an optimality criterion is a highly inefficient way of optimizing and hence a couple of methods are proposed instead. For more information see Morris and Mitchell (1995) and Jin et al. (2005). Optimization has also been employed to create designs with the desired correlation among inputs (Vořechovský and Novák, 2009). It should be noted that, even using the mentioned references, OLHD can be quite expensive. Putting aside its large computation cost, the performance of OLHD itself in high dimensions becomes questionable, as the results in the next section show.

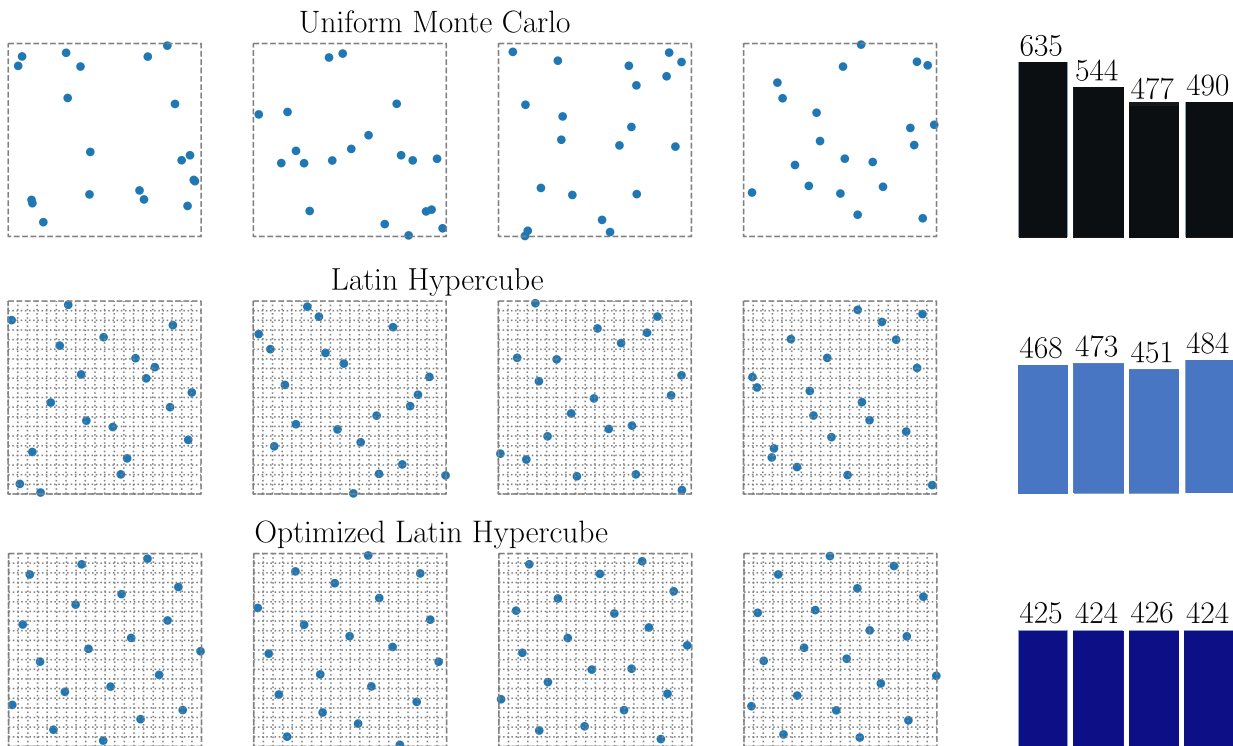
### 2.3.3. Comparison of uniform DOE methods

Figure 2.7 compares the uniformity of MC, LH and OLH based on the potential energy (PE) criterion in two dimensions<sup>1</sup>. It can be seen that MC has the highest PE and therefore is the worst in terms of uniformity and OLH is the best. The variation among the four exemplary cases is the lowest in OLH and the highest in MC, indicating that OLH is also the most robust and trustworthy. Next, Figure 2.8 shows three sets of designs, each repeated 20 times; one is two-dimensional with 20 samples and the other two are 20-dimensional with 400 and 2000 samples, respectively.

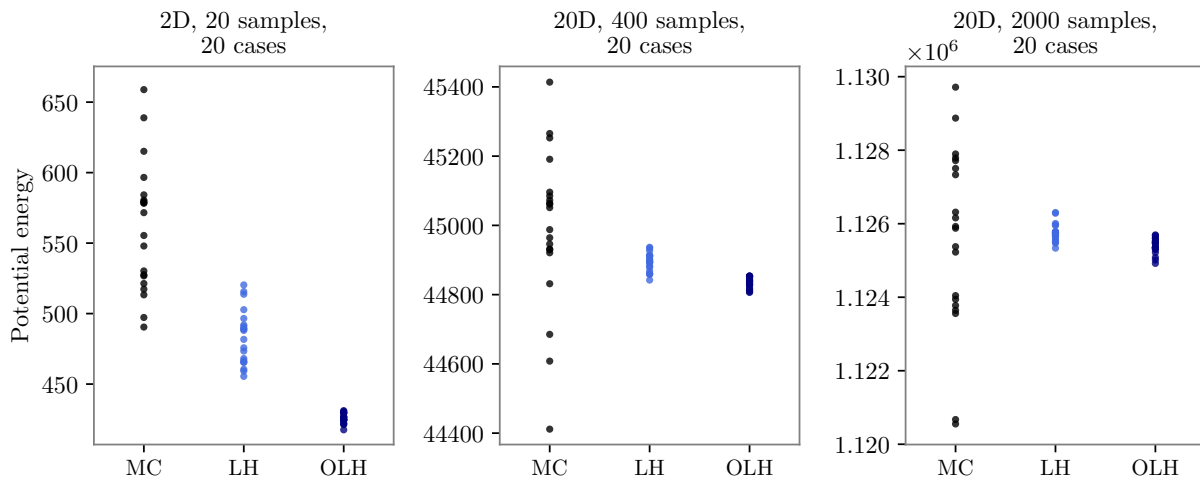
<sup>1</sup> Results are based on an in-house code, developed by Ilya Arsenyev and modified by the author of this work.

LH designs that are optimized to OLH here are different from those considered for LH designs. The outcome of the two-dimensional designs (the left figure) is the same as in the previous example, with a clear improvement from MC to LH and from LH to OLH. However, in 20D with 400 samples, not only OLH loses its advantage and ends up being worse than some MC designs, but it also takes several minutes for each case. OLH is still better than LH. However, not always and the improvement is marginal and becomes even less by increasing the number of samples to 2000. In addition, the range of the OLH's potential energy values for 20 cases is within that of the MC designs, and no large improvement is obtained compared to LH (contrary to 2D cases). So, a possible explanation is that OLH's optimizer converges consistently to non-global optima. If this is the case, it is not due to the lack of iterations, since the optimizer converged before reaching the maximum number of iterations in all cases. Simulated annealing was the optimizer, not perhaps the first choice, but capable of handling multi-modal functions. ESE (enhanced stochastic evolutionary) is a better optimizer and was specifically developed to obtain optimal (uniform) designs (Jin et al., 2005). Our experiment with ESE shows that the comments mentioned above for simulated annealing still hold. This implies that the current optimized LH is ineffective in higher dimensions and other methods should be investigated to have a better uniform design. Another observation from Figure 2.8 for 20D cases is that all LH designs' potential energies are within that of the Monte Carlo designs, contrary to 2D cases. So, in higher dimensions, it is possible that some MC designs are more uniform than the most uniform LH. This contrasts our observation from the 2D examples and is counter-intuitive; one may think that by stratifying the space and placing only one sample in each direction (i.e., LH), a more uniform design will be obtained compared to just randomly placing the samples (i.e., MC). But in higher dimensions, it seems that this is not the case anymore. Here, we just compared 20 sets of designs, because OLH takes a long time. However, we have repeated the procedure for 100 cases (except for OLH) and the results had the same trend (not shown here). One important question that remains is, in higher dimensions, how good the potential energy criterion is in identifying uniformity in the first place. The next section introduces more peculiar observations from higher dimensions and some challenges that it brings.

In Figure 2.9, we compare the uniformity of MC, Sobol family, Hammersley, Halton and LH over a range of dimensions using the normalized MST (Minimal spanning tree) criterion (see Table 2.1). The higher the MST, the more uniform the design. It can be seen that Hammersley is the best at low dimensions, but it degrades quickly and ends up being the worst at higher dimensions. Halton also degrades relative to others, and its uniformity gets as bad as Hammersley around dimension 14. Similarly, according to Antinori (2017), in more than 14 dimensions, both Halton and Hammersley show spurious correlations. On the other hand, interestingly, LHD and MC that were the worst in lower dimensions become comparable to others around dimension 14 and even end up being better than Sobol1 and Sobol3 in higher dimensions (35-40<sup>th</sup>). The difference between LHD and Monte Carlo is also quite small, similar to the result of the potential energy criterion shown in Figure 2.8. In addition to what is shown here, Garud also compared these DOE methods with the Maximin uniformity criterion and found a similar trend (Garud et al., 2017). It should be noted

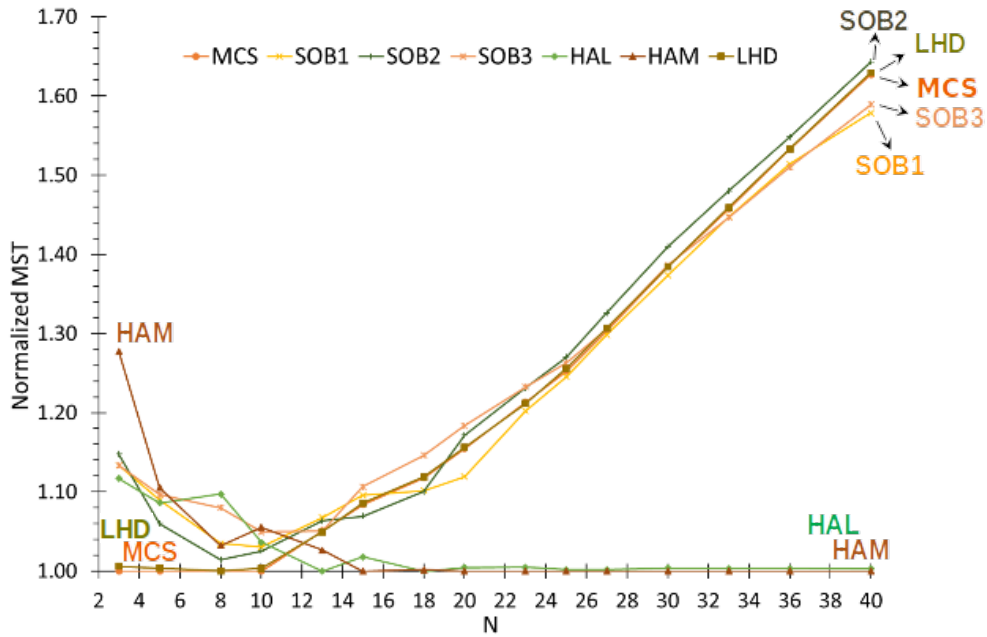


**Figure 2.7** Four cases of two-dimensional samples generated by uniform MC on the top, LH in the middle, and OLH on the bottom are shown on the left side. On the right, the value of the potential energy criterion for each case is depicted in bar plots correspondingly.



**Figure 2.8** Comparison of the value of the potential energy criterion in three different cases and in DOE methods. LH designs that are optimized in OLH are different from LH designs.

that current results are derived for up to dimension 40 and DOE methods that are close to each other in their uniformity rankings may change places, depending on the uniformity criterion, the number of samples and dimension. For example, Sobol3 has been shown to be better than Sobol2 in financial applications of several hundred dimensions (Harase, 2019).



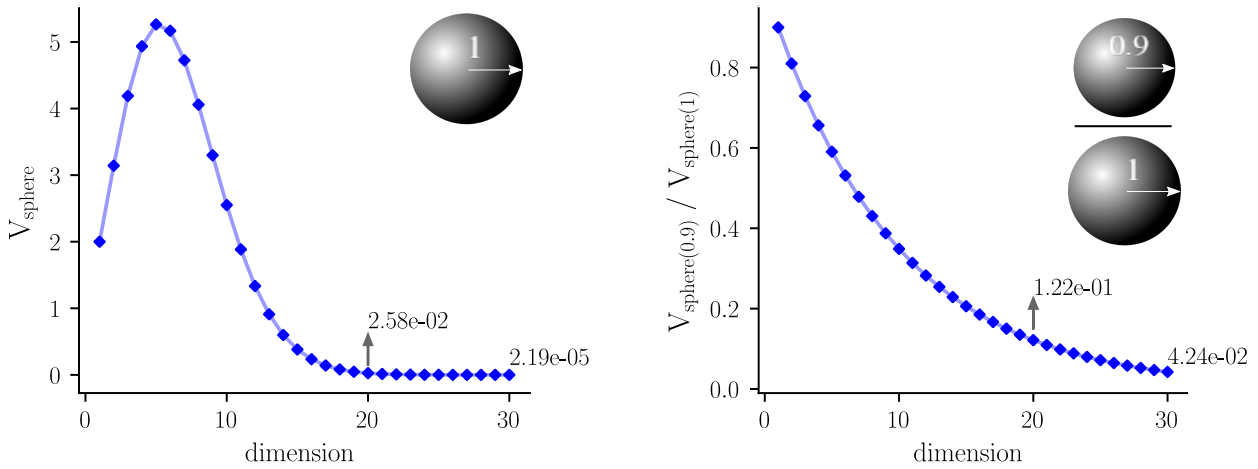
**Figure 2.9** Effects of dimensionality on the uniformity of the design using the normalized MST criterion for 1000 samples (Garud et al., 2017). Larger MST values are considered to have better space-filling. MCS: Monte Carlo sampling, SOB: Sobol, HAL: Halton, HAM: Hammersley and LHD: Latin Hypercube design.

## 2.4. Curse of dimensionality

With increasing the number of inputs, i.e., dimensionality, more samples are expected to be needed to adequately represent the output. What is less obvious is the fact that higher dimensions act as a vacuum and a huge amount of samples may be required up to exponential growth. In addition, intuition can easily fail. This implies that if a pattern is observed in low dimensions, it should not automatically be generalized to higher dimensions. For example, the uniformity of both LHD and MC get better in higher dimensions, while that of Hammersley and Halton gets worse.

Obviously, high-dimensionality does not happen by going from one dimension to the next, but it is rather more gradual. Below, examples from Verleysen (2003) are reproduced to illustrate some relevant aspects. Figure 2.10 shows that the volume of a hypersphere increases with dimension ( $d$ ) at the beginning, as expected, but after  $d = 5$ , volume decreases continuously to such an extent that even at  $d = 30$ , the volume of the hypersphere with unit radius is merely  $2.19e-5$ . Another interesting fact is that the ratio of the volume of a hypersphere with a radius of 0.9 to the volume of a unit hypersphere will approach zero as the dimension increases. This indicates that most of the volume will be in the outer shell of a high-dimensional hypersphere and not inside it. Therefore, most of the samples in the space will be closer to the boundaries than to the center. Figure 2.11 shows the same trend, but for a hypercube. Another quantity that takes effect from high-dimensionality is the distance among samples. First, let us emphasize the importance of distance. In numerical methods, distance is a key factor in defining the approximation function. In machine learning, distance is a so-called similarity measure that helps with detecting patterns and consequently prediction (the closer the points are to each other, the more similar the patterns). However, this similarity can collapse in higher dimensions when the closest and furthest distances to any point become less

and less distinguishable from each other. But how come? Assume that we are using a distance norm  $L_k$ ; the furthest distance of  $N$  points to the origin using  $L_k$  is named  $\max\{\|X_d\|_k\}$ , where  $X_d$  represents  $d$ -dimensional data points and the nearest distance to the origin is called  $\min\{\|X_d\|_k\}$ . Then, in Aggarwal et al. (2001) it is shown that the difference  $\max\{\|X_d\|_k\} - \min\{\|X_d\|_k\}$  changes with the rate of  $d^{1/k - 1/2}$  independently of the sample distribution. Therefore, by increasing the dimension, this rate goes to infinity for the Manhattan distance ( $k = 1$ ), converges to a constant for the Euclidean norm ( $k = 2$ ), and converges to zero for  $k \geq 3$ , i.e., minimum and maximum distance are not distinguishable and all distances are basically the same.

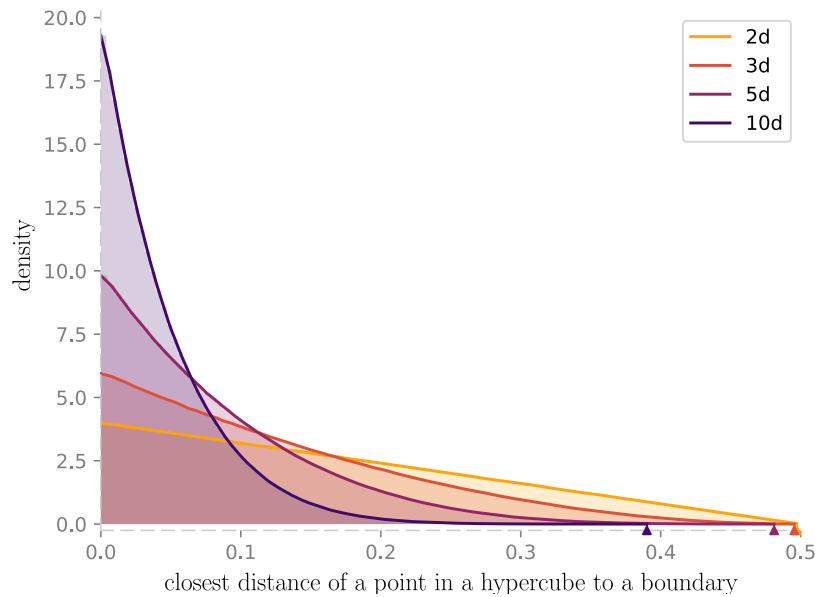


**Figure 2.10** Left: Change in the volume of a hypersphere with unit radius in terms of dimensionality. Right: relative change of volume between hyperspheres with radii 0.9 and 1, respectively. Based on Verleysen (2003).

These results show that the Manhattan distance is potentially a better candidate for higher dimensions. Aggarwal et al. went further and tried out fractional ( $0 < k < 1$ ) norms and showed that for a 20-dimensional classification problem, fractional norms are more successful than traditional ones (Aggarwal et al., 2001). It should be noted that any sampling that does not grow exponentially with increasing dimensions will eventually suffer from this convergence of minimum to maximum distance (Morgan and Gallagher, 2014) and hence its consequences.

Based on what has been discussed so far, it is questionable whether the LHD that divides the space uniformly has actually a suitable structure for higher dimensions, where the density of the space is not uniform at all. LHD is commonly used as the first part of EGO. However, investigating this method is not among the focus points of this work and remains for future research. Here, we do not use higher dimensions, but as the above review shows, the effect of higher dimension may already start at least before dimension 10, as Figures 2.9 and 2.10 indicate. We borrow some ideas from methods that deal with high dimensions, and some of our proposed modifications are also developed by having in mind the possibility of their expansion to higher dimensions in the future. See Chapter 4 for more information.





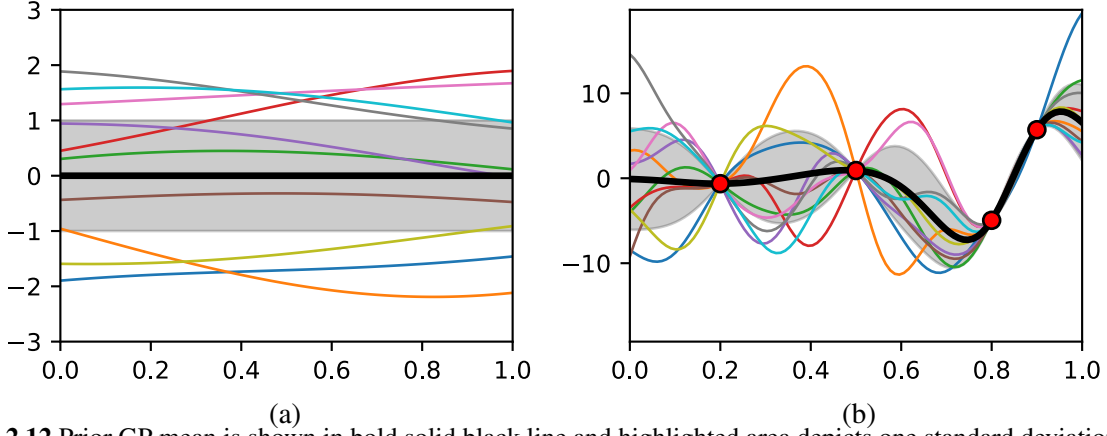
**Figure 2.11** 10 millions samples are generated from the uniform MC within hypercubes of different dimensions (2, 3, 5 and 10). The closest distance of each sample to a boundary (of the hypercube) is calculated and its histogram's density (i.e., dividing the number of samples in each histogram's bin by 10 millions) is plotted for the considered dimensions. For each dimension, a trend line is also fitted at the top center of the bins. Here, due to the high number of samples, the bins are quite narrow and hence difficult to be seen. This figure shows that, by increasing the number of dimensions, samples get closer to the boundaries and away from the center of the design space.

## 2.5. Gaussian Process Regression

Gaussian process (GP) is a stochastic process that can be used to model quantities in space and/or time. The simplicity of calculation (relative to alternatives) and at the same time, its power to represent a variety of complex functions make it popular in various fields, see, e.g., McBride and Sundmacher (2019), Wang et al. (2007), and Bichon et al. (2008). EGO is based on GP as a surrogate model, which is a type of supervised learning and also known as Gaussian process regression (GPR). Note that the term "regression" is used here to differentiate it from the classification and should not be interpreted as a conventional regressive response surface (RS) in which the surface passes between samples and not through them. However, a GPR surface can be interpolative or regressive. In the next sections, we explain the GPR based on Rasmussen and Williams (2006), therefore, we do not repeat mentioning this reference for every relevant formula or explanation.

### 2.5.1. Definition

GP assumes that each point in the space is a random variable with a normal distribution and there is a multivariate joint Gaussian distribution among any finite number of points of the space. It requires only the mean  $\mu(X)$  and the covariance matrix  $K(X, X')$  to define the joint normal distribution  $Y \sim N(\mu(X), K(X, X'))$ . The covariance matrix is at the heart of the method, which defines how points in the space are related to each other. Here, we are not interested in describing a GP or generating samples from it, but the aim is to use it within a procedure to fit a surface to a set of data (i.e., GPR). The underlying data that should be fitted has nothing to do with a stochastic process and it can be (is mostly) deterministic. We are just employing the capability of GP to represent a



**Figure 2.12** Prior GP mean is shown in bold solid black line and highlighted area depicts one standard deviation around each point. Figure (a) shows prior GP with zero mean and standard deviation of one (a common choice when there is no default information about data). The posterior distribution obtained based on four samples (red dots) is shown in Figure (b). In both figures, 10 sample functions are drawn from each case.

variety of shapes via sample functions, each of which is a realization of a GP (when every point in a GP has its value, then a sample function is formed). Therefore, here we try to find a suitable curve/surface that fits available data reasonably, i.e., neither under-represents it nor overfits it. This implies that the mean and covariance of the process should be defined properly. After that, we can use them to predict unsampled parts of the space (via the conditional distribution concept, see the explanation before Equation (2.3)). In other words, the general procedure is as follows:

1. Assuming a prior distribution, i.e., a certain mean and covariance. If we already have any default information, it can be considered here. Figure 2.12 (a) depicts some sample functions from a prior distribution.
2. Update the prior GP using the given data (that is, find appropriate hyperparameters that define mean and covariance). Usually, a specific model to represent the covariance matrix is selected by the user and its parameters are determined by the available samples while trying to fit the current samples very well and yet avoiding overfitting.
3. After finding the hyperparameters, we can use them to predict the output in other parts of the space that have not been sampled yet (i.e.,  $\mathbf{x}_*$ ).

Now, we obtain the prediction formula. Suppose that  $(\mathbf{x}, y)$  is a sample from an underlying function to be fitted, where  $\mathbf{x}$  is a  $d$ -dimensional input and  $y$  the corresponding scalar output. We can write the output as the function of all inputs  $y = Z(\mathbf{X})$ . We would like to fit a GP surface to the set of samples  $(\mathbf{X}, \mathbf{y})$ . Assume that we already know the proper values for the mean and covariance matrix (it will be discussed later how to find them). As mentioned before, there is a joint normal distribution between any subset of points in the space, so it is also true for training points (evaluated points so far,  $\mathbf{X}$ ) and test points (desired points for prediction,  $\mathbf{X}_*$ ):

$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}_* \end{bmatrix} \sim \mathcal{N} \left( \begin{bmatrix} \boldsymbol{\mu}(\mathbf{X}) \\ \boldsymbol{\mu}(\mathbf{X}_*) \end{bmatrix}, \begin{bmatrix} K(\mathbf{X}, \mathbf{X}) & K(\mathbf{X}, \mathbf{X}_*) \\ K(\mathbf{X}_*, \mathbf{X}) & K(\mathbf{X}_*, \mathbf{X}_*) \end{bmatrix} \right), \quad (2.2)$$

where  $\mathbf{y}_*$  are outputs at points  $\mathbf{X}_*$ , i.e., the unknown random variables we are looking for (i.e., prediction). We already know the GP values ( $\mathbf{y}$ ) at  $\mathbf{X}$ , therefore we can use them in a conditional distribution to obtain  $\mathbf{y}_*$ . Fortunately, the conditional distribution of a joint normal distribution is also a (joint) normal distribution and one can obtain the predicted mean ( $\bar{\mathbf{y}}_*$ ) and the corresponding covariance using the following analytical formula:

$$\bar{\mathbf{y}}_* = \boldsymbol{\mu}(\mathbf{X}_*) + K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X})]^{-1}(\mathbf{y} - \boldsymbol{\mu}(\mathbf{X})), \quad (2.3)$$

$$\text{Cov}(\mathbf{y}_*) = K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X})]^{-1}K(\mathbf{X}, \mathbf{X}_*). \quad (2.4)$$

In case of a single prediction point,  $\text{Cov}(y_*)$  will be the variance  $\sigma(\mathbf{x}_*)^2$  at that point. In some cases, it is necessary to represent a noisy data set; therefore  $\mathbf{y} = Z(\mathbf{X}) + \varepsilon$ . One assumption would be additive, independent and identically distributed Gaussian noise  $\varepsilon$  with variance  $\sigma_n^2$ , which leads to adding  $\sigma_n^2$  to the diagonal of the covariance matrix of the training points. That is, the only change is to replace  $K(\mathbf{X}, \mathbf{X})$  with  $K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I$ , while  $I$  is the identity matrix. Now, the prediction for the noisy case is:

$$\bar{\mathbf{y}}_* = \boldsymbol{\mu}(\mathbf{X}_*) + K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1}(\mathbf{y} - \boldsymbol{\mu}(\mathbf{X})), \quad (2.5)$$

$$\text{Cov}(\mathbf{y}_*) = K(\mathbf{X}_*, \mathbf{X}_*) - K(\mathbf{X}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1}K(\mathbf{X}, \mathbf{X}_*). \quad (2.6)$$

There is a more general formula:

$$y(\mathbf{x}) = \boldsymbol{\mu}(\mathbf{x}) + Z(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T \boldsymbol{\beta} + Z(\mathbf{x}). \quad (2.7)$$

In which a term called trend ( $\boldsymbol{\mu}(\mathbf{x}) = \mathbf{f}(\mathbf{x})^T \boldsymbol{\beta}$ ) is added to a GP with zero mean ( $Z(\mathbf{x})$ ). Basis functions  $\mathbf{f}(\mathbf{x})$  can be nonlinear, but since  $\boldsymbol{\beta}$ s are unknown, the trend is called a linear model. Here, according to Rasmussen and Williams (2006) the idea is that the underlying data is close to this global linear model with residuals dealt with by the GP term. What is widely used as the basis functions are polynomial regressions, i.e.,  $\mathbf{f}(\mathbf{x}) = (1, x, x^2, \dots)$ . This addition of a term to a GP is discussed in the Kriging literature. Based on the trend, the name is more distinct:

- Simple Kriging:  $\boldsymbol{\mu}(\mathbf{x})$  is a known constant;
- Ordinary Kriging:  $\boldsymbol{\mu}(\mathbf{x})$  is an unknown constant;
- Universal Kriging:  $\boldsymbol{\mu}(\mathbf{x})$  is a deterministic non-zero term and trend ( $\mathbf{f}(\mathbf{x})^T \boldsymbol{\beta}$ ) is a linear model.

For further information, consult Schonlau (1997) and Lichtenstern (2013). Usually, a model for the covariance matrix is considered that has unknown parameters. These unknowns plus  $\boldsymbol{\beta}$  are called hyperparameters and should be determined during the so-called fitting of GP to be able to use the model for prediction. The predicted mean and variance for universal Kriging are (see Arsenyev (2016)):

$$\bar{\mathbf{y}}_* = \mathbf{f}(\mathbf{x}_*)^T \hat{\boldsymbol{\beta}} + \mathbf{r}(\mathbf{x}_*)^T \mathbf{R}^{-1}(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}), \quad (2.8)$$

$$\sigma_*^2 = \sigma_z^2 \left( 1 - \begin{bmatrix} \mathbf{f}(\mathbf{x}_*)^T & \mathbf{r}(\mathbf{x}_*)^T \end{bmatrix} \begin{bmatrix} 0 & \mathbf{F}^T \\ \mathbf{F} & \mathbf{R} \end{bmatrix}^{-1} \begin{bmatrix} \mathbf{f}(\mathbf{x}_*) \\ \mathbf{r}(\mathbf{x}_*) \end{bmatrix} \right), \quad (2.9)$$

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^T \mathbf{R}^{-1} \mathbf{F})^{-1} \mathbf{F}^T \mathbf{R}^{-1} \mathbf{y}. \quad (2.10)$$

Here, the covariance  $K(\mathbf{X}, \mathbf{X})$  (see Table 2.2) is given by  $K(\mathbf{X}, \mathbf{X}) = \sigma_f^2 R(\mathbf{X}, \mathbf{X}')$ , where  $R(\mathbf{X}, \mathbf{X}')$  is the correlation function.  $\mathbf{r}(\mathbf{x}^*)$  is the correlation between training ( $\mathbf{X}$ ) and a test point ( $\mathbf{x}^*$ ) (i.e.,  $K(\mathbf{x}^*, \mathbf{X})$  in previous formulae when factoring out  $\sigma_f^2$ ) and  $\mathbf{F}(\mathbf{X})$  is a matrix formed by vectors  $\mathbf{f}(\mathbf{x})$  ( $F_{ij} = f_j(\mathbf{x}^{(i)})$ ). The ordinary Kriging formula can be found in Forrester et al. (2008). One apparent point from this Kriging formula is that its predicted mean and variance involve much more matrix multiplications compared to their GP counterparts, i.e., Equations (2.3 and 2.4) respectively.

### 2.5.2. Covariance functions

The covariance matrix gives GP its flexibility and the power to represent a variety of functions. A valid covariance matrix is a positive definite one, so it does not lead to negative variances. Covariance can be represented by functions each having its unknown parameter (so-called hyperparameters). These unknowns are determined during the fitting of the available data. Table 2.2 lists some common functions.

Note that in the literature, correlation models are also discussed, which are basically covariance functions in Table 2.2 without considering  $\sigma_f$  (representative of variance). The basic idea behind these covariance/correlation functions is to define how points are related to each other. If correlation reduces fast with distance, then points in the vicinity of each other can take significantly different values and hence a more rough function can be expected. Other patterns, such as repetitive surfaces, can be reproduced more efficiently using periodic covariance functions. One of the most common types of covariance models is called stationary, which is only a function of the distances between points in the space  $|\mathbf{x} - \mathbf{x}'|$ . In other words, it is invariant with respect to translation. Two famous stationary covariance functions are squared exponential (SE) and Matèrn. In an anisotropic SE, each dimension is scaled by its length scale  $l_i$  (see Table 2.2). The larger the  $l_i$ , the smoother the fitted surface along that dimension (i.e., the output will be less dependent on that dimension). Therefore, the value of the length scale can tell how much an output depends on the corresponding input. This can be used as a way to screen design variables and is known as automatic relevance determination (ARD), see, e.g., Section 2.4.1 in Forrester et al. (2008). Figure 2.13 shows the SE kernel with different length scales and the corresponding fitted GP curves. Note that the hyperparameter for Kriging is often designated by  $\theta$ , see e.g. Forrester and Keane (2009) and contrary to the length scale, it is in the numerator, directly multiplying to the distance measure as the only factor. So, for a SE kernel,  $\theta$  is equivalent to  $1/2l^2$ . In this work, we use GPR within EGO. However, we refer to the corresponding hyperparameter as  $\theta$  instead of  $l$ . In addition to the length scale, the Matèrn family has another parameter,  $\nu$  which is positive and allows Matèrn to represent different degrees of smoothness.  $\nu = 0.5$  results in a kernel similar to the exponential

covariance function; tends to have rough sample functions (see Figure 2.14). When  $\nu \rightarrow \infty$ , Matèrn converges to the SE kernel. In practice, for  $\nu \geq 3.5$  it is difficult to distinguish among different  $\nu$ .

**Table 2.2** List of some common covariance functions.  $d$  is dimension and  $r = |\mathbf{x} - \mathbf{x}'|$  is a distance measure. The column "S" defines whether the covariance functions are stationary or not. Based on Table 4.1 from Rasmussen and Williams (2006). In this work, we also refer to the hyperparameter  $l$  as  $\theta$ .

\*  $T = \text{diag}\left(\frac{1}{2l_1^p}, \dots, \frac{1}{2l_D^p}\right)$ ,  $p > 0$ ,  $p = 2$  is the most common.

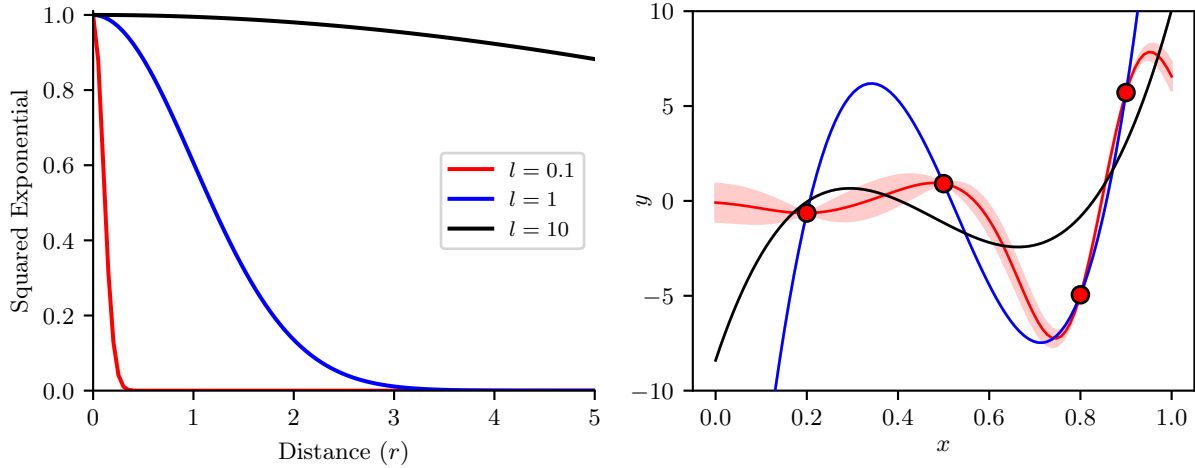
\*\*  $\Gamma(\nu)$  is the gamma function and  $K_\nu$  is the modified Bessel function.

Covariance function	Formula	Hyperparameters	S
Linear	$\sum_{d=1}^D \sigma_d^2 x_d x'_d$	$\sigma_d = \{\sigma_1 \dots \sigma_D\}$	
Polynomial	$\sigma_f(x \cdot x' + a)$	$\sigma_f, a$	
Exponential	$\sigma_f \exp\left(-\frac{r}{l}\right)$	$\sigma_f, l$	✓
Rational Quadratic	$\left(1 + \frac{r^2}{2\alpha l^2}\right)^{-\alpha}$	$\alpha, l$	✓
Isotropic squared exponential	$\sigma_f \exp\left(-\frac{r^p}{2l^p}\right)$ , $p > 0$	$\sigma_f, l$	✓
Anisotropic squared exponential*	$\sigma_f \exp\left(-(\mathbf{x} - \mathbf{x}')T(\mathbf{x} - \mathbf{x}')\right)$	$\sigma_f, \{l_1 \dots l_D\}$	✓
Periodic	$\sigma_f \exp\left(-\frac{2\sin^2(\pi r/t)}{l^2}\right)$	$\sigma_f, l, t$ (period)	✓
Matèrn**	$\frac{\sigma_f}{2^{\nu-1}\Gamma(\nu)} \left(\frac{\sqrt{2\nu}}{l}r\right)^\nu K_\nu\left(\frac{\sqrt{2\nu}}{l}r\right)$	$\sigma_f, l, \nu$	✓

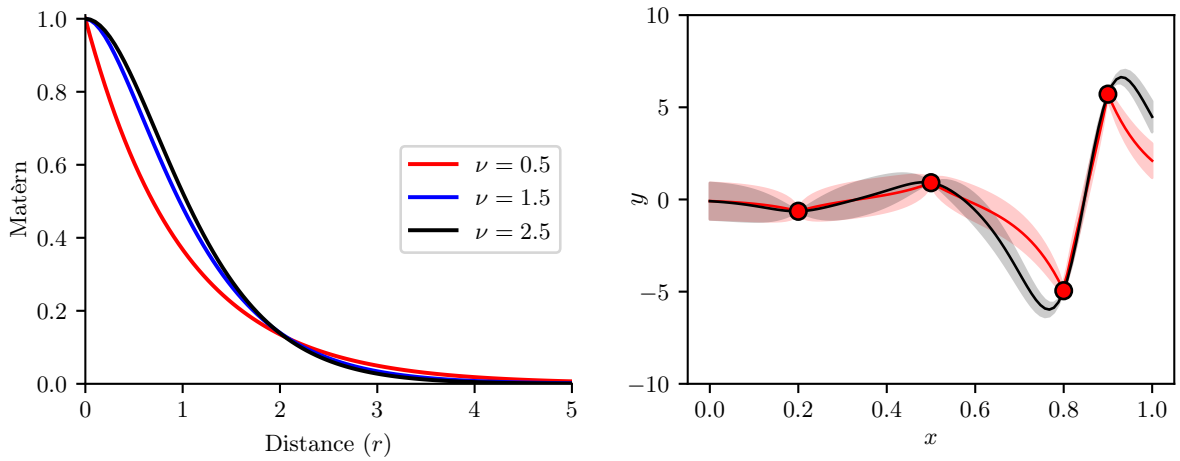
Non-stationary covariance functions lead to the predicted variances that, among others, depend on the location. Covariance models can be combined (summed, multiplied, etc.) and form a new covariance function. As long as the combined covariance is positive definite, it will be valid. Kriging in geostatistics uses a variogram instead of covariance. The variogram is the variance of the difference between two points in the field. When two points are further away, logically the difference of the field values at these locations can become larger, so the variogram increases with distance until it reaches a plateau where variance is maximum and field values vary independently of each other. This concept is similar to covariance which measures how field values at two points vary with each other, but (stationary) covariance is a decreasing function of distance.

### 2.5.3. Model selection and adaptation of hyperparameters

As mentioned before, prior to prediction, one needs to find a GPR (i.e., determine its mean and covariance) that fits the current data well enough. Usually, a certain GPR model is selected in the



**Figure 2.13** Left figure shows how the SE correlation function reduces with distance over different length scales. The right figure shows the corresponding fitted curves over the given data. For length scales  $l = 1$  and  $l = 10$ , the standard deviation becomes so small that it is not observable here.



**Figure 2.14** Left figure shows how Matérn with different  $\nu$  values reduces with increasing distance. The right figure shows the corresponding fitted curves over the given data for  $\nu = 0.5$  and  $\nu = 2.5$ .

first step; for example, universal Kriging with anisotropic SE covariance function while considering the Euclidean distance. The hyperparameters of this model are the length scales  $l_i$  of the covariance function and  $\sigma_f^2$  (see Table 2.2 and Equation (2.9)). If in addition, the distance measure was also considered as an unknown, then its corresponding parameter  $p$  in anisotropic SE (see Table 2.2) would also be added to the list of hyperparameters. Instead of selecting a certain model first, one can assume different GPR models and select the best among them, see Rasmussen and Williams (2006). However, here we are interested in finding the hyperparameters of only one selected model, which is also referred to as training a GP or adaptation of hyperparameters. There are two well-known methods to do this training, the Bayesian approach and cross-validation. In the Bayesian approach, the Bayes rule is used to update the prior distribution using the observed data (via the likelihood function) to obtain the posterior distribution. Bayes rule can be applied at several connected layers consequently. At the lowest level are model parameters, e.g.,  $\boldsymbol{\beta}$  in the predicted mean of Kriging (Equation (2.8)), where their prior probabilities are updated; then, at one

upper level, hyperparameters (e.g., length scale) that define these model parameters are updated using information from the previous layer update. If the structure of the model was also unknown, it would be at the highest level and it would have its own Bayes update. But, as mentioned above, here we assume that the model is fixed at the beginning. Therefore, what is important here is to calculate the posterior probability of hyperparameters,  $p(\mathbf{h}|\mathbf{y}, \mathbf{X})$  to calculate their most probable outcome ( $\mathbf{h}$  represents hyperparameters). But this posterior includes calculating an integral term which is difficult to evaluate; therefore, instead, a term called marginal likelihood  $p(\mathbf{y}|\mathbf{X}, \mathbf{h})$  is maximized w.r.t. the hyperparameters.  $p(\mathbf{y}|\mathbf{X}, \mathbf{h})$  is the probability of the observed data (training data) when the hyperparameters are set to a certain value. Hence, the meaning of maximizing the likelihood is to set the hyperparameters to a different value each time (within the considered range for hyperparameters), then evaluate the likelihood and see for which set of hyperparameters the highest likelihood is obtained or, in other words, find the hyperparameters that have the highest probability of generating current data. The marginal likelihood in the case of GP is a multiplication of Gaussian distributions for each observed sample, i.e., a multivariate Gaussian distribution.

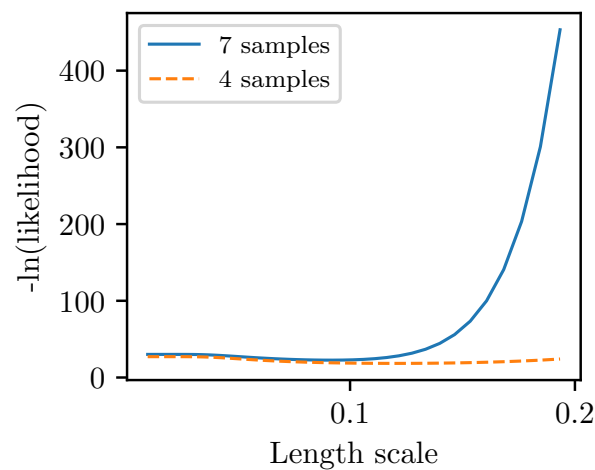
$$p(\mathbf{y}|\mathbf{X}, \mathbf{h}) = \frac{1}{(2\pi)^{\frac{n}{2}} |K + \sigma_n^2 I|^{\frac{1}{2}}} e^{\left(-\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T (K + \sigma_n^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu})\right)}. \quad (2.11)$$

Its natural logarithm is:

$$\ln p(\mathbf{y}|\mathbf{X}, \mathbf{h}) = -\frac{1}{2}(\mathbf{y}-\boldsymbol{\mu})^T (K + \sigma_n^2 I)^{-1}(\mathbf{y}-\boldsymbol{\mu}) - \frac{1}{2}\ln|K + \sigma_n^2 I| - \frac{n}{2}\ln 2\pi, \quad (2.12)$$

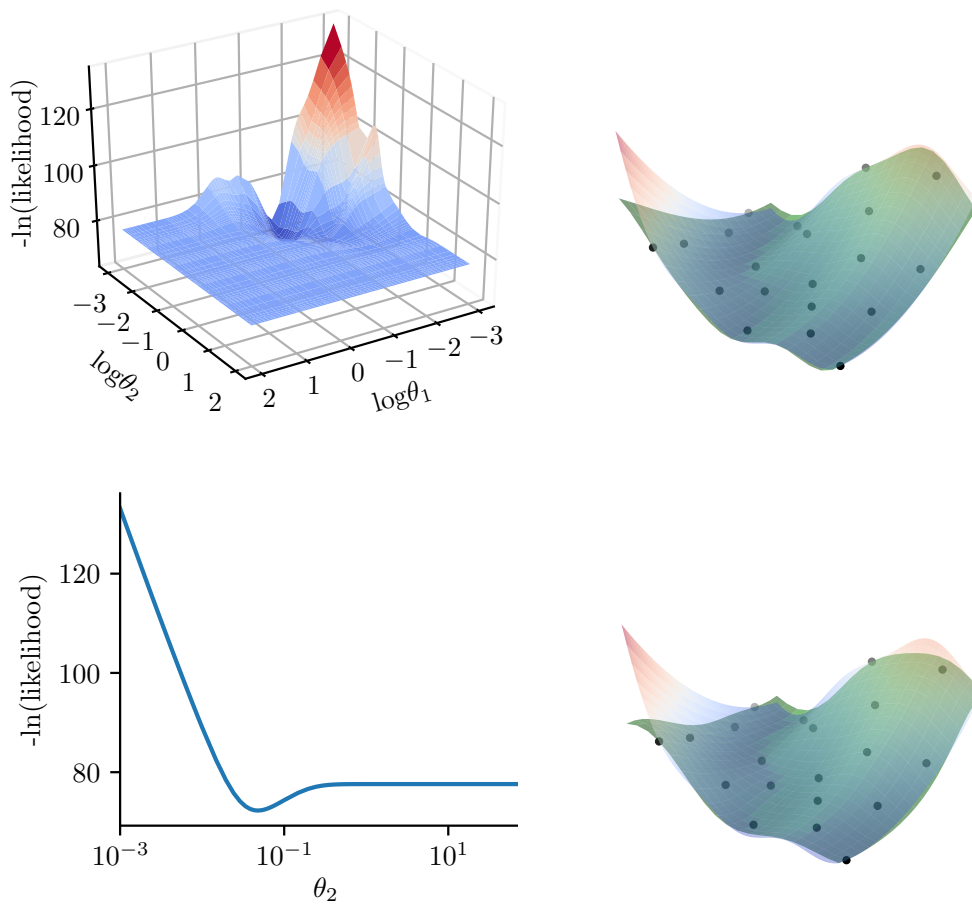
where  $n$  is the number of samples with output values of  $\mathbf{y}$ ,  $\boldsymbol{\mu}$  is the mean as discussed previously (see Equation (2.7) and the description after) and  $\sigma_n^2$  is the regularization parameter for noisy cases as discussed before. The first term  $(-\frac{1}{2}\mathbf{y}^T(\mathbf{y}-\boldsymbol{\mu})(K + \sigma_n^2 I)^{-1})$  in the equation is the only term containing the output values of the observed points and represents data-fitting capability. The second term without considering the sign  $(\frac{1}{2}\ln|K + \sigma_n^2 I|)$  is a penalty to avoid more complicated models (and consequently overfitting) and the last term is just a normalization constant. With increasing length scale, data-fit capability (first term) is decreased and prediction based on it represents more simple functions. Now, with a less complicated model, the penalty for complexity is also reduced, but in the formula, the sign of the second term is negative, so in total it will increase. In short, one term is decreasing while the other term is increasing and as the result, the likelihood optimum tends to be somewhere in between these two extremities; fitting data very well while highly risking an overfitted model on one hand and having a very simple representation of data (model tends to have fewer ups and downs) on the other hand. It should be noted that likelihood tries to achieve this balance, but this does not mean that it is always successful. The more data is available, the better likelihood can judge which set of hyperparameters ends in explaining the data well, i.e. there will be more distinct local maxima in the likelihood function (see Figure 2.15). In this thesis, we consider the negative natural logarithm (shown by  $-\ln$ ) of the likelihood. There-

fore, minimization is conducted instead of maximization. The partial derivative of the likelihood w.r.t. hyperparameters is useful for gradient-based optimizers. However, due to the nature of the likelihood, a multi-start strategy is advised to search globally. Another method to identify hyperparameters is cross-validation (CV). CV is used in general to find unknowns based on the given data; either this unknown is selecting a model from different considered models, e.g., see Garbo and German (2017), or finding the parameters of a specific model (as we are doing here). In CV, unknowns are set to certain values and then training data is divided into several groups (so-called folds), each with (almost) the same number of samples. Each time a model is fitted using samples from all but one group and the error in the group that was set aside is evaluated. This procedure is repeated until all groups have been set aside once. Then, the errors of all these cases are combined by an error measure such as root mean square error. So far, the unknowns have been set to certain values. Next, another set of values for the unknowns is considered and the procedure is repeated. In the end, the set of values that has the least CV error will be selected as the final value of the unknowns. It can be seen that CV can become computationally expensive if the number of unknowns or the number of groups increases. The most demanding one is leave-one-out cross-validation (LOOCV), which leaves only one sample out. A ten-fold cross-validation is commonly used, see e.g., Chapter 10 of Martins and Ning (2021). In general, maximizing the likelihood function is often preferred compared to CV to find hyperparameters (Martin and Simpson, 2005).



**Figure 2.15** Effect of the number of samples on the likelihood function. In general, the higher the number of samples, the more distinct the likelihood optimum. Here, the negative logarithm of likelihood is depicted.





**Figure 2.16** The left column shows the negative logarithm of the likelihood function for two cases. The top is for anisotropic SE and the bottom for isotropic SE. The left column shows the corresponding fitted curves in green and the actual underlying function on the surface with a red and blue colormap. The dots are data points. Here, ordinary Kriging was used.

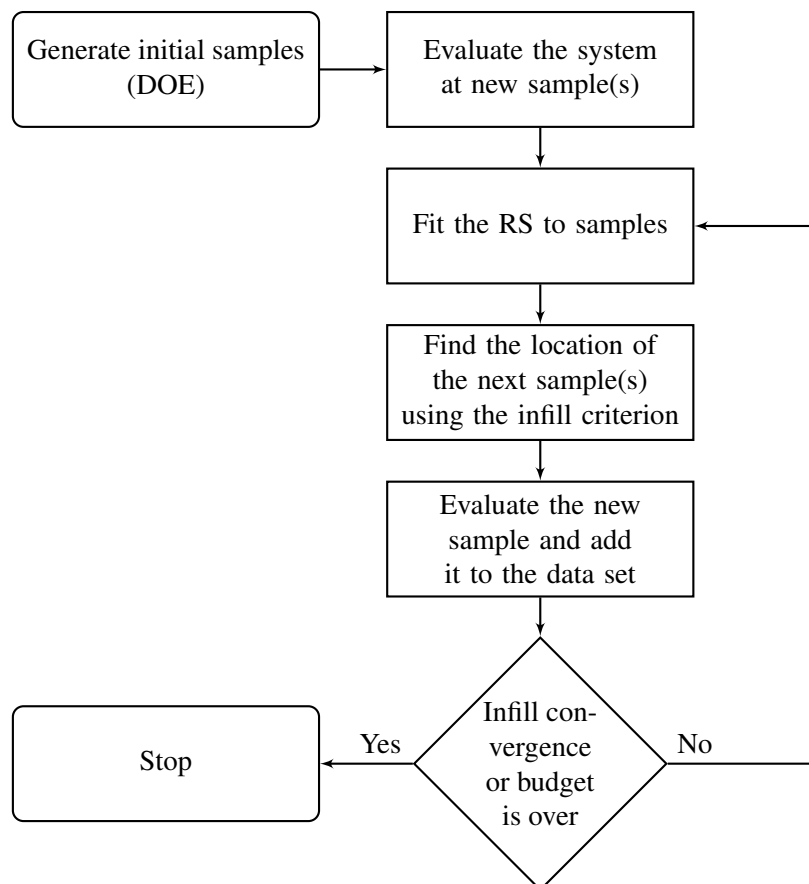
## Chapter 3

### Efficient global optimization (EGO)

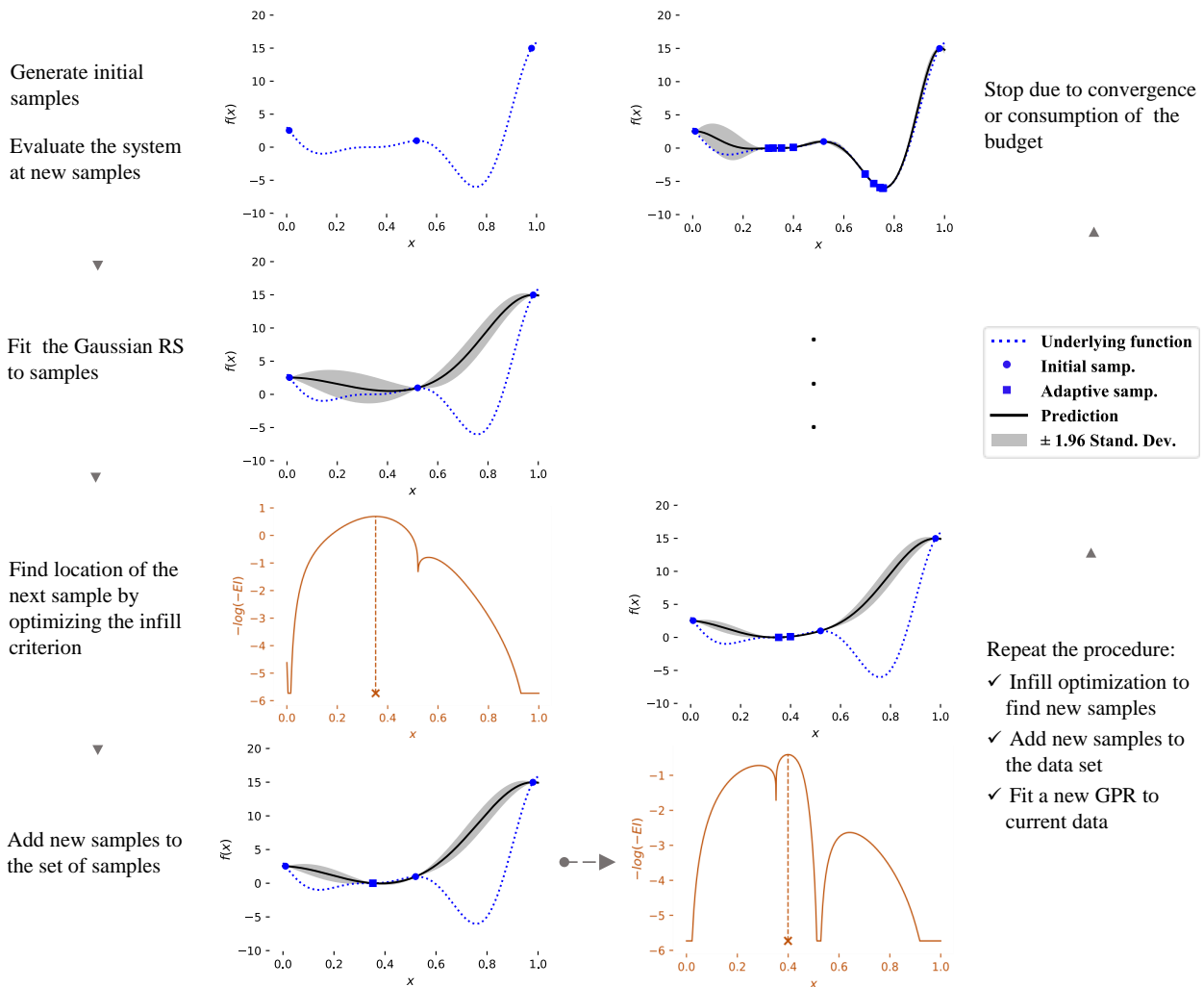
We introduce EGO in this chapter. First, the general procedure of EGO is described and then important aspects of EGO are reviewed in more detail. Starting from presenting various infill criteria followed by reviewing different methods to handle constrained problems. We also discuss how to handle noisy problems and then larger dimensions along with modifications of EGO to make it more efficient. The GPR and uniform sampling required in EGO are already introduced in the previous chapter (preliminaries).

### 3.1. Introduction

Efficient global optimization (EGO) is an adaptive response surface-based optimization (ARSBO) introduced by Schonlau (1997). The general procedure for an ARSBO is shown in Figure 3.1. In EGO, the fitted RS is the predicted mean of a GPR and this algorithm has two phases. In the first phase, samples are generated to be uniformly distributed in the design space. Then, the problem at hand is evaluated at these samples and a GPR model is fitted. This model is considered representative of the underlying response and will be adaptively improved in the second phase to make better use of the sampling budget. Therefore, in the second phase, samples are added one by one, each time with the help of the mean and standard deviation (i.e., uncertainty about the goodness of fit) of the fitted surface. The location of the next sample is decided by optimizing the so-called infill criterion (see Figure 3.3). This procedure is terminated if the infill criterion becomes less than a certain tolerance or the budget runs out due to a time constraint. Fitting a GPR model at each iteration of EGO requires determining the corresponding hyperparameters. This can be done, for example, by updating hyperparameters based on the Bayes rule, as mentioned in Section 2.5.3. However, instead, we maximize the likelihood function. Therefore, EGO can be considered as Bayesian optimization (BO) (Moćkus, 1975).



**Figure 3.1** Flowchart of some adaptive response surface-based optimization (ARSBO) methods. In EGO, the considered RS is the predicted mean of the fitted GPR.

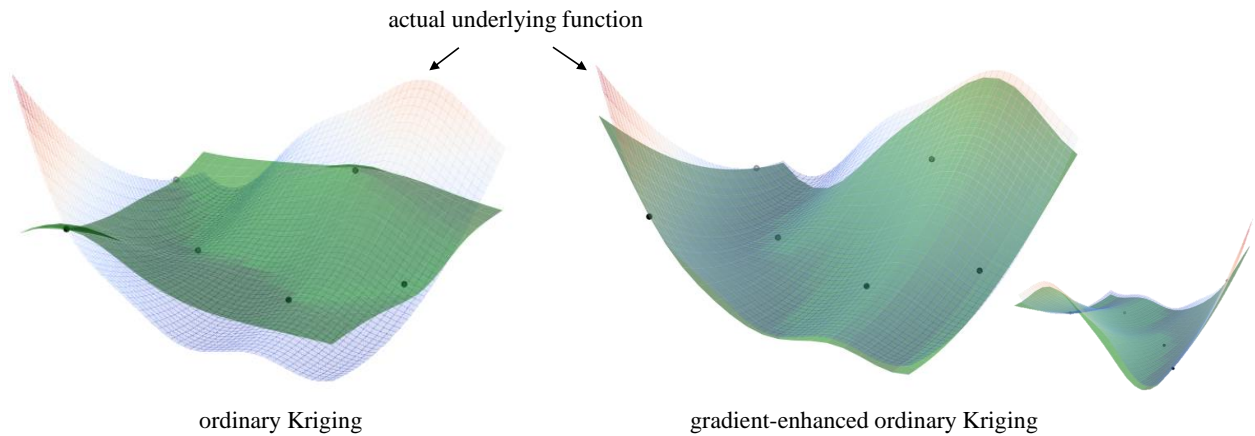


**Figure 3.2** Steps of the efficient global optimization (EGO)

EGO as a surrogate-based optimization is naturally affected by the limitation of RS methods in general and GPR in particular. In this regard, one important aspect is the loss of efficiency in higher dimensions. GPR is recommended for dimensions less than 20 (see Chapter 3 of Forrester et al. (2008)). Therefore, Bayesian optimization performs well in lower dimensions, but degrades about 15-20 dimensions (Wang et al., 2013). However, since we deal with lower dimensions in this work, EGO is a suitable candidate for expensive optimization problems such as crashworthiness. Over time, different infill criteria have been proposed, which will be covered in this chapter. The extension of EGO to constrained problems is also discussed in the following sections. There are some other extensions of EGO that will not be covered in this work, including:

- Parallel EGO, in which, in the adaptive phase, more than one sample is added in each iteration to benefit from parallel processing, see Haftka et al. (2016) and Li et al. (2016).
- Gradient-enhanced Kriging, where gradient information helps to increase efficiency, e.g., by considering not only the output values, but also the gradients at sample points when fitting a GPR. Obviously, the existence and accuracy of gradients directly affect the quality, see Laurent

et al. (2017).



**Figure 3.3** An ordinary Kriging (left) is compared to a gradient-enhanced ordinary Kriging (right) over a test function. The smaller figure on the right shows the same fit, but from another angle. With only six samples, the gradient-enhanced model achieves significantly better accuracy, but at a higher computational cost.

- **Cokriging.** When there is more than one Kriging model and they are correlated and share the same inputs, cokriging can help to improve the approximation of one of the models based on the other. This can be beneficial, e.g., when there are low- and high-fidelity models and one prefers to avoid expensive high-fidelity evaluations as much as possible. Therefore, instead, the low-fidelity model is evaluated several times to obtain a good approximation of the high-fidelity one. One example is optimization involving fluid simulations where higher and lower mesh densities are high- and low-fidelity models correspondingly, see Forrester and Keane (2009).

### 3.2. Infill criterion

In ARSBOs, the optima of the infill criteria help to find where the system should be evaluated next. Since an RS is fitted to data, there is already an approximation of the underlying function. On the one hand, regions around the current best samples can be promising areas to exploit and on the other hand, exploring the design space at regions that have not yet been evaluated can be a reasonable approach for placing the next samples. Most infill criteria try to balance exploration and exploitation. The fitted RS in case of GPR is the predicted mean and its uncertainty is the corresponding predicted standard deviation, which acts naturally as an exploratory measure. Therefore, components to form exploitation and exploration criteria are readily available. Some other RSs do not provide uncertainty estimates for their prediction. In that case, some error estimations have been suggested to play a similar role. Two rather obvious infill criteria are at the extremities of the exploration-exploitation balance:

1. Assume that the current-fitted surface is a very good representative of the actual function for which we try to find the optima; in this case, we only have to find the optimum of the surface (i.e., optimum of the predicted (mean) values of the RS).

2. We are only interested in placing samples at regions that have not yet been explored (i.e. total exploration). Here, we optimize the uncertainty error, e.g., the predicted standard deviation from GPR.

In the next sections, we will review some infill criteria, based on the seminal paper by Jones (2001). In the following,  $\hat{y}(\mathbf{x})$  is the predicted mean of GPR and  $\hat{s}(\mathbf{x})$  is the corresponding standard deviation.

### 3.2.1. Lower confidence bound (LCB)

$\hat{y}(\mathbf{x}) - k\hat{s}(\mathbf{x})$  is the lower confidence bound of the prediction, also known as the statistical lower bound, in which  $k > 0$  is a multiplier. The location of the next sample is obtained by minimizing this criterion. When  $k = 0$ , uncertainty is neglected and fitted RS is optimized directly, which is the extreme case of exploitation, as mentioned before. The higher the  $k$ , the more exploratory the behavior. However, when minimizing the LCB, any region that has a lower bound larger than that of the latest found sample will be excluded from future searches. Therefore, the LCB does not place samples densely iteratively and hence we have no guarantee of convergence (Jones, 2001).

### 3.2.2. Probability of improvement (PI)

Our aim is to find the minimum of an objective represented by a GPR, so each sample in the design space is expressed by a normal random variable  $Y(\mathbf{x})$  with mean  $\hat{y}(\mathbf{x})$  and standard deviation  $\hat{s}(\mathbf{x})$ . In optimization, we try to improve iteratively. A straightforward definition of improvement ( $I(\mathbf{x})$ ) w.r.t. a target ( $T$ ) in the case of minimization is:

$$I(\mathbf{x}) = \begin{cases} T - Y(\mathbf{x}) & Y(\mathbf{x}) < T \\ 0 & \text{otherwise,} \end{cases} \quad (3.1)$$

which is a positive value when it is lower than the target ( $Y(\mathbf{x}) < T$ ), otherwise it will be zero.  $T$  can logically be the best output value among the samples evaluated so far ( $y_{\min}$ ). Probability of improvement is then calculated via:

$$P[I(\mathbf{x})] = P(Y(\mathbf{x}) < T) = \Phi\left(\frac{T - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) = \Phi\left(\frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right), \quad (3.2)$$

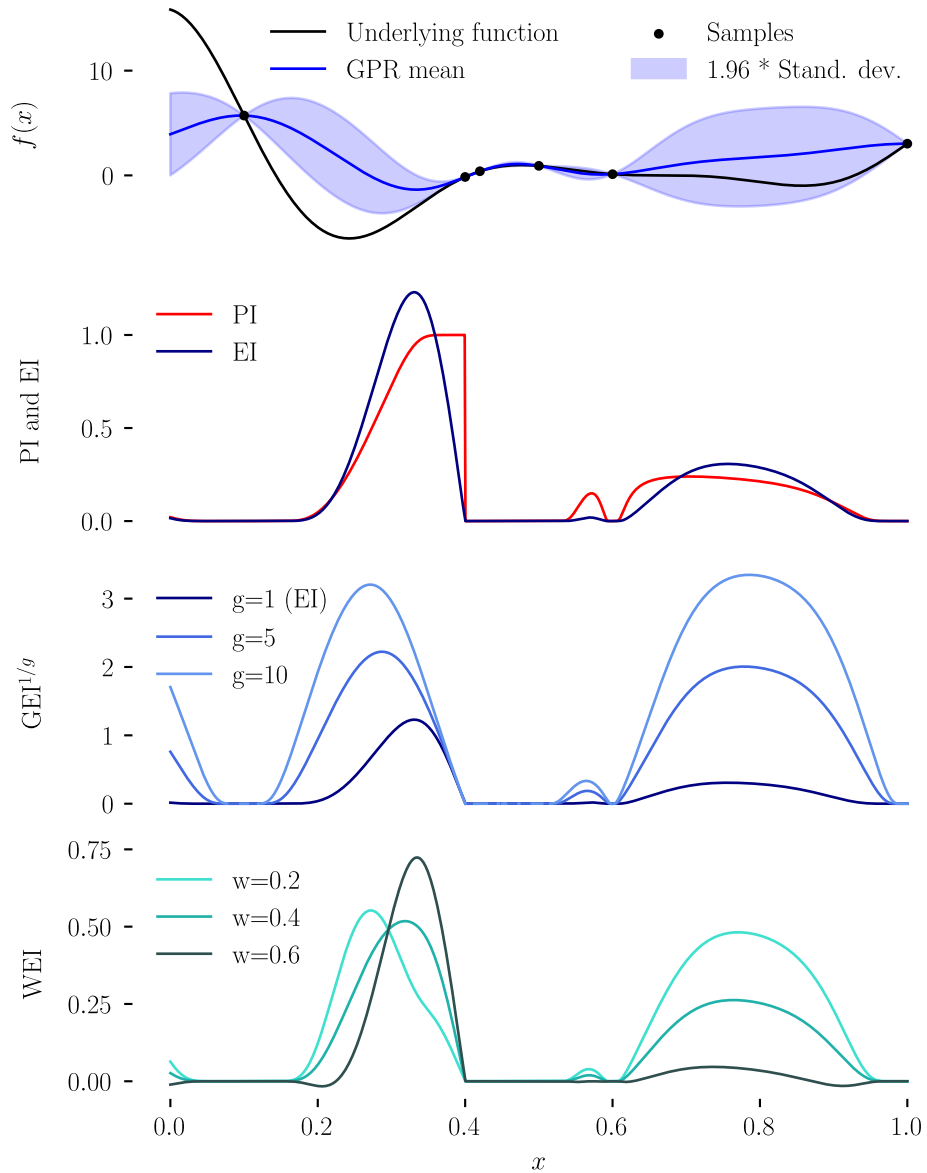
where  $\Phi$  is the cumulative distribution function (CDF) of the standard normal distribution. However, this infill criterion is sensitive to  $T$  and has a tendency to exploitation when  $T = y_{\min}$ ; on the other hand, if  $T$  is set too high, global search is performed and convergence is sacrificed for exploration (Jones, 2001).

### 3.2.3. Expected improvement family

If in the previous section the probability of improvement was maximized, here the maximum expected value of the improvement ( $E[I(\mathbf{x})]$ ) is sought:

$$E[I(\mathbf{x})] = (y_{\min} - \hat{y}(\mathbf{x}))\Phi\left(\frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right), \quad (3.3)$$

where  $\phi$  is the probability density function (PDF) of the standard normal distribution and  $\Phi$ ,  $\hat{y}(\mathbf{x})$ , and  $\hat{s}(\mathbf{x})$  are the same as above. When there is no uncertainty ( $\hat{s}(\mathbf{x}) = 0$ ), e.g., at samples with already known outputs (and no regularization), EI will be zero, since no improvement can happen. The first term in EI conducts exploitation, while the second term wants to explore. So, EI tries to make a balance between trusting what is fitted so far and on the other hand, improving the fit itself (not totally trusting it). Later, an additional integer parameter ( $g \geq 0$ ) was added to the



**Figure 3.4** Top figure shows a GPR curve fitted to current samples. Figures below depict some infill criteria for the current iteration.

improvement formula ( $I^g(\mathbf{x})$ ) in Schonlau et al. (1998), with the aim of having more flexibility in adjusting this balance. The higher the value of  $g$ , the more exploratory the new fill criterion (see Figure 3.4), which they called generalized expected improvement ( $GEI = E[I^g(\mathbf{x})]$ ). GEI returns the probability of improvement for  $g = 0$  and expected improvement for  $g = 1$ . To calculate GEI for

larger values of  $g$ , the following formula is suggested:

$$GEI = E[I^g(\mathbf{x})] = \hat{s}(\mathbf{x})^g \sum_{k=0}^g (-1)^{-k} \binom{g}{k} \left( \frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right)^{g-k} T_k, \quad (3.4)$$

where  $T_k$  is calculated recursively via:

$$\begin{aligned} T_0 &= \Phi(\mathbf{u}) \quad \text{and} \quad T_1 = -\phi(\mathbf{u}), \\ T_k &= -\mathbf{u}^{k-1} \phi(\mathbf{u}) + (k-1)T_{k-2}, \\ \mathbf{u} &= \left( \frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right). \end{aligned} \quad (3.5)$$

$E[I^g(\mathbf{x})]^{1/g}$  is compared to a tolerance to end the optimization. Schonlau explains further that GEI provides a systematic way to control global vs. local search trade-off, an improvement over a previously suggested ad hoc method, in which standard deviation was inflated by a factor to make the search more global. He also adds that a local search has a higher probability of improvement, but the amount of improvement can be small while a larger improvement can be obtained with a global search but with a smaller probability. This is perhaps like a risk-reward spectrum in an investment; gaining more rewards is riskier.

Another way of changing the local-global search trade-off is to assign a weight to each term of EI, so more emphasis can be placed on either when required. This version of EI is called weighted expected improvement (WEI) (Sobester, 2003),

$$WEI = w(y_{\min} - \hat{y}(\mathbf{x})) \Phi \left( \frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right) + (1-w) \hat{s}(\mathbf{x}) \phi \left( \frac{y_{\min} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right), \quad (3.6)$$

where  $w \in [0, 1]$ . WEI becomes more exploitative as weight ( $w$ ) increases.

### 3.3. Handling constraints

In engineering problems including crashworthiness, constraints are often an indispensable part of optimization. In order to introduce them in the solution process, there are some common ways, such as the penalty method, which adds the constraints to the objective function or approximates them in the solution process as is done in SQP or turning the problem into a multi-objective one. In EGO, in addition to some of the common methods for handling constraints, some specific ones related to the procedure of EGO and/or availability of an uncertainty measure are suggested. In this section, we briefly mention some of the suggested methods. It should be noted that the suggested classification here is one of the possible ways to do this.



### 3.3.1. Penalty methods and constrained infill criterion

A common way of transferring a constrained problem to an unconstrained one is obtained by adding the constraint violation as a penalty term to the objective function. In the context of EGO, the same procedure is performed, only the objective and constraint are the corresponding mean of the fitted GPR and it is a one-pass penalization, see, e.g., Parr et al. (2010),

$$\begin{aligned} y(\mathbf{x})_p &= \hat{y}(\mathbf{x}) + \text{Penalty}, \\ \text{Penalty} &= \mu * (\text{constraint violation})^p, \text{ mostly } p = 2. \end{aligned} \quad (3.7)$$

As is commonly known, the penalty term can make abrupt changes in the landscape and hence make optimization difficult. In addition, here we are using only the prediction of the GPR, so there is no measure for exploration. To improve, one possibility is to apply a penalty to EI directly (Parr et al., 2010),

$$E[I(\mathbf{x})]_p = (y_{\min\text{-feas}} - \hat{y}(\mathbf{x}))\Phi\left(\frac{y_{\min\text{-feas}} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) + \hat{s}(\mathbf{x})\phi\left(\frac{y_{\min\text{-feas}} - \hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right) - \text{Penalty}, \quad (3.8)$$

where  $y_{\min\text{-feas}}$  is the best feasible objective obtained so far. The minus sign for the penalty term is because EI should be maximized. Another way of handling constraint EGO is to represent constraints during the infill sampling criterion optimization, see, e.g., Sasena et al. (2001). Here, constraints are represented by the Kriging (mean) prediction, that is,

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad \text{Infill criterion}(\mathbf{x}) \\ &\text{subject to} \quad \hat{c}_i(\mathbf{x}) \leq 0, \forall i. \end{aligned} \quad (3.9)$$

Still, this is a constrained problem, so Sasena used a version of the DIRECT algorithm that can handle such cases. One problem here is that we use only the predicted value of the constraint and do not consider the uncertainty of this prediction. In other words, we are explicitly assuming that  $\hat{c}_i$  is a good representative of the actual constraint, but this is not usually the case. Therefore, the expected violation (EV) was suggested and is described in the next section. Another case that considers constraints within the optimization of the infill criterion is introduced in El Bechari et al. (2018). Here, feasibility probability (FP) of the constraint should be above a threshold,

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}} \quad -\text{EI}(\mathbf{x}) \\ &\text{subject to} \quad \text{FP}(\mathbf{x}) \geq P_{\text{tol}}, \text{ suggested } P_{\text{tol}} = 0.5. \end{aligned} \quad (3.10)$$

### 3.3.2. Expected Violation (EV)

The Expected Violation (EV) criterion is very similar to EI and was introduced to handle constraint problems (Audet et al., 2000). Basically, one calculates EI, but for the constraints, while the  $y_{\min}$  in EI is replaced by the limit of the constraint, i.e., zero, when writing the approximated constraint

(by GPR) in the standard form ( $\hat{c}_i(\mathbf{x}) \leq 0$ ). So, EV is,

$$EV(\mathbf{x}) = (-\hat{c}(\mathbf{x}))\Phi\left(\frac{-\hat{c}(\mathbf{x})}{\hat{s}_c(\mathbf{x})}\right) + \hat{s}_c(\mathbf{x})\phi\left(\frac{-\hat{c}(\mathbf{x})}{\hat{s}_c(\mathbf{x})}\right). \quad (3.11)$$

EV for each constraint is calculated and used in a suggested optimization algorithm that also attempts to balance local and global search (Audet et al., 2000). EV becomes large when the constraint violation is expected to be high or if there is a high uncertainty in the prediction of a constraint. Here, since we do minimization, -EI should be minimized. The expected violation was also used in Bichon et al. (2009). Here, the infill criterion is EI of the Augmented Lagrangian representation of the problem, in which EV replaces the Lagrange and the penalty terms,

$$\eta = \hat{Y} + \sum \lambda_i EV_i + \sum \mu_i EV_i^2. \quad (3.12)$$

In this formula, only  $\hat{Y}$  is random and hence the mean and variance of this term are

$$\begin{aligned} \text{mean}_\eta &= \hat{y} + \sum \lambda_i EV_i + \sum \mu_i EV_i^2; \\ \text{variance}_\eta &= \hat{s}_y^2, \end{aligned} \quad (3.13)$$

which can be used to calculate the intended criterion  $EI_\eta$ . The roles of  $\mu$  and  $\lambda$  based on Nocedal and Wright (2006) are described next. In the penalty method, a large penalty multiplier ( $\mu$ ) is required to make the solution of the unconstrained problem close to the (feasible) solution of the original constrained problem, but at the same time, this can create jumps in the unconstrained problem (the cliff issue), as mentioned before. In addition, near the optimum, large  $\mu$  can make the Hessian of this unconstrained problem ill-conditioned and hence cause problems to find the solution via algorithms such as conjugate gradient or Quasi-Newton methods. Augmented Lagrangian reduces this ill-conditioning by adding the Lagrangian term into the penalty formulation.

### 3.3.3. Expected Improvement with Feasibility Probability

This section is based on Schonlau et al. (1998) which introduces handling constraints to generalized expected improvement. Consider the nonlinear optimization problem:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) \\ \text{subject to} \quad & c_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m, \\ & h_j(\mathbf{x}) = 0, \quad j = 1, \dots, n, \end{aligned} \quad (3.14)$$

with objective function  $f(\mathbf{x})$ , inequality constraints  $c_i(\mathbf{x})$  and equality constraints  $h(\mathbf{x})$ . In Schonlau et al. (1998) only inequality constraints in the form of  $c_i^{\text{lb}} < c_i(\mathbf{x}) < c_i^{\text{ub}}$  are considered. Although an equality constraint ( $h_j$ ) can be defined as two inequality constraints with very close lower and upper bounds correspondingly, we will see later that there is a better way of handling equality constraints. Suppose that we are at a certain iteration and the current fitted objective and

constraints are  $F(\mathbf{x})$  and  $C_i(\mathbf{x})$  respectively. In addition, the minimum feasible objective function obtained so far is called  $y_{\min\text{-feas}}$ . The generalized expected improvement subjected to constraint ( $I_c^g(\mathbf{x})$ ) is defined by:

$$I_c^g(\mathbf{x}) = \begin{cases} [y_{\min\text{-feas}} - F(\mathbf{x})]^g & \text{if } F(\mathbf{x}) < y_{\min\text{-feas}} \text{ and } c_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, k \\ 0 & \text{otherwise,} \end{cases} \quad (3.15)$$

The expected value of  $I_c^g(\mathbf{x})$  is:

$$E(I_c^g(\mathbf{x})) = \int_{-\infty}^0 \dots \int_{-\infty}^0 \int_{-\infty}^{y_{\min\text{-feas}}} [y_{\min\text{-feas}} - F(\mathbf{x})]^g P(F, C_1, \dots, C_k) dF dC_1 \dots dC_k, \quad (3.16)$$

where  $P(F, C_1, \dots, C_k)$  is the joint probability density function of the objective and constraints, which we assume to be a multivariate normal distribution. The predicted mean and variance are already available for each response (i.e., objective and constraints), but the correlations among responses are not available to know the covariance of this multivariate normal distribution. Hence, we assume independence among responses (i.e., no correlation) and this joint multivariate normal density function will be just the product of each response probability density function. As the result, constrained generalized expected improvement (Eq. (3.16)) can be simplified to:

$$E(I_c^g(\mathbf{x})) = E(I^g(\mathbf{x})) \prod_{i=1}^k P_i = E(I^g(\mathbf{x})) \prod_{i=1}^k P(C_i \leq 0), \quad (3.17)$$

which is basically the Expected Improvement of the objective (like unconstrained problems) multiplied by the probability that each constraint is satisfied  $P(C_i \leq 0)$ ; or, in other words, feasibility probability (FP). Since for each constraint, a GPR model is fitted, the predicted mean and standard deviation are available at each point and consequently the corresponding FP can be calculated as:

$$P(C_i \leq 0) = \Phi\left(\frac{0 - \hat{y}_i(\mathbf{x})}{\hat{s}_i(\mathbf{x})}\right) = 1 - \Phi\left(\frac{\hat{y}_i(\mathbf{x})}{\hat{s}_i(\mathbf{x})}\right), \quad (3.18)$$

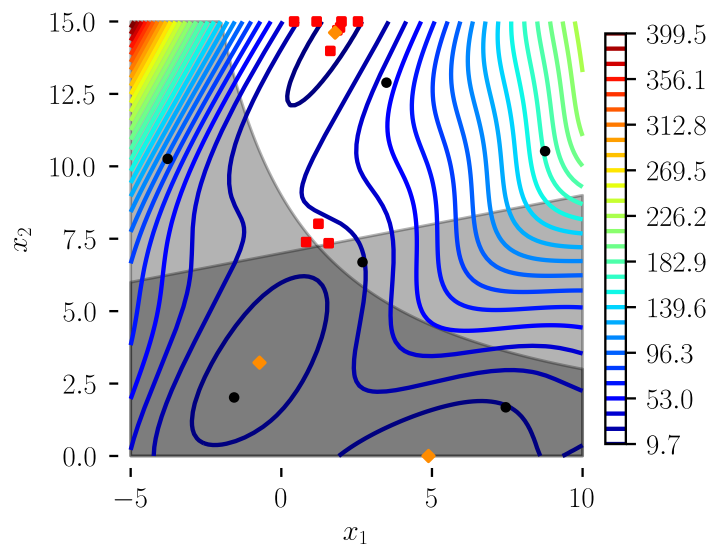
where  $\Phi$  is as before the CDF of the standard normal distribution. We will refer to this method as EI-FP. It should be noted that feasibility probability (FP) is also called "probability of feasibility (PF)" in the literature.

Durantin et al. compared EV, EI-FP and Sasena's suggestion (i.e., Equation (3.9)) and found that EI-FP was the only one that did not fail in the considered problem to reach the optimum, but Sasena's leads to better sampling points close to the boundaries of the constraints. The authors claim that the results are consistent with two other references (Durantin et al., 2016). Parr et al. compared the common penalty method (3.7) and the penalty applied to EI (3.8) with EI-FP and concluded that for complex problems, EI-FP is better because it represents a smooth transition between feasible and infeasible regions (no cliff problem due to penalty term) and has both exploration and exploitation elements (Parr et al., 2010). However, the authors also mention that if one

of the constraints has a feasibility probability close to zero, this can be problematic and good solutions might be discarded. Indeed, EI-FP also suffers from deficiencies, so some attempts have been made to improve them or introduce another way of handling constraints. Figure 3.5 shows a modified Branin-Hoo function that is subject to two constraints. EI-FP is used to find the infill points depicted by red squares. In this figure, EI-FP seems to be efficient; however, there are issues that can reduce its performance to a large degree. In EI-FP, feasibility probability of each constraint is multiplied by the EI of the objective. Probability is a number between  $[0,1]$  and when multiplying several of them, the results become (very) small and we may need to take a logarithm to distinguish them in optimization. But what is important, is the fact that this probability is calculated by CDF of a normal distribution that saturates quickly to either zero or one (Figure 3.6). So, when there are a couple of constraints, first, the multi-modality of CEI can increase and/or some actually good regions can remain unsampled, because one or more constraints are not approximated well there and block visiting these regions even after many evaluations. A modification to probability feasibility and hence CEI is suggested in Bagheri et al. (2017) as follows:

$$E(I_c(\mathbf{x})) = E(I(\mathbf{x})) * F(\mathbf{x}) = E(I(\mathbf{x})) \prod_{i=1}^k \min \left( 2\Phi \left( \frac{-\hat{y}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right), 1 \right). \quad (3.19)$$

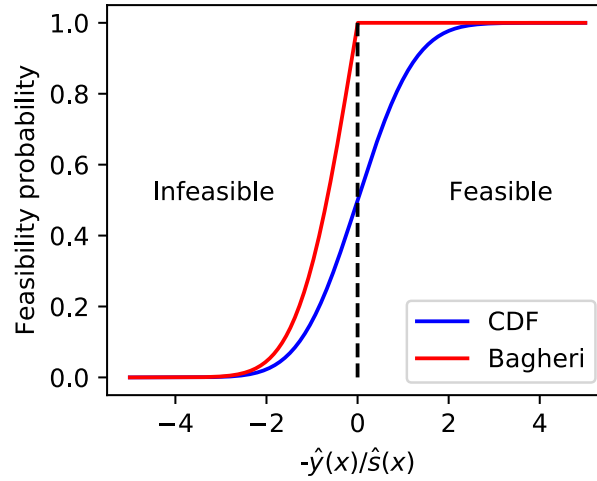
The authors suggest first maximizing the modified feasibility to find at least one feasible solution and then maximizing the modified CEI. A similar strategy to find the first feasible solution (before considering the effect of the objective function) is also suggested in Habib et al. (2016).



**Figure 3.5** Finding the feasible optimum by EI-FP. Contours represent the objective function which is a modified Branin-Hoo with three local minima. Regions where two constraints are violated are highlighted in gray. Six initial points are shown in black circles and adaptive points in red squares. Orange rhombuses mark the three local minima of which the lower right is the global minimum in the absence of constraints.

### 3.3.4. Conversion into a multi-objective problem

One way to deal with constraints is to turn the constrained problem into a multi-objective one. Since EI-FP in the previous section can suffer from dominance of a constraint, there have been



**Figure 3.6** CDF of the standard normal distribution and corresponding modification of it suggested in Bagheri et al. (2017).

suggestions to separate EI and constraints via multi-objective optimization, for example, in Parr et al. (2012), where EI of the objective function is one objective and probability of feasibility of the constraint is another objective. If there is more than one constraint and especially when there are more than three, it is better to multiply all probability of feasibility terms and use them as one objective (see the first two terms in Equation (3.20), since a multi-objective optimization with more than four objectives is inherently difficult to solve (Parr et al., 2012). The authors consider only one update point and hence from all the points on the Pareto set, the one that maximizes Schonlau's formula (Equation (3.17)) will be selected. A three-objective optimization problem has also been proposed by Durantin et al. (2016) which, as the third objective, considers the prediction variance of the constraints. One reason according to the authors, is that if constraints are not properly modeled, there is nothing to guide the infill criterion to regions with high uncertainty in Kriging prediction. The authors also mention that "In the case of active constraints, reducing the prediction variance also leads to a global minimum that is closer to the actual constraint boundaries" (Durantin et al., 2016, p. 918). Here, it is suggested that this variance as the third objective be the sum of the constraints' standard deviation  $\sum_{i=1}^k \hat{s}_{ci}$  or the sum of the second part of the expected violation  $(\sum_{i=1}^k \hat{s}_{ci} \phi(-\hat{c}_i/\hat{s}_{ci}))$ .

$$\text{minimize}_{\mathbf{x}} \left\{ -EI(\mathbf{x}), -\prod_{i=1}^k FP_i(\mathbf{x}), \sum_{i=1}^k \hat{s}_{ci}(\mathbf{x}) \right\}, \quad (3.20)$$

or

$$\text{minimize}_{\mathbf{x}} \left\{ -EI(\mathbf{x}), -\prod_{i=1}^k FP_i(\mathbf{x}), \sum_{i=1}^k \hat{s}_{ci}(\mathbf{x}) \phi\left(\frac{-\hat{c}_i(\mathbf{x})}{\hat{s}_{ci}(\mathbf{x})}\right) \right\}. \quad (3.21)$$

### 3.3.5. Following part of the (feasible) design space

From previous sections, we realized that the feasible region can be represented in various ways, e.g., by multiplying the feasibility probability of constraints which was better than some other methods, but had its own problems. One issue besides those discussed above is that such multiplication tends to search inside the feasible region better than at the boundary and hence it will find the optimum within the feasible region faster. This is simply due to the feasibility probability formula (CDF of a normal distribution) which has a higher value when there is more evidence indicating that one is deep inside the feasible region, while at the boundary the probability reduces to 0.5. Accordingly, it is reported by Audet et al. (2000) and Picheny (2014) that when the optimum is on or very close to one of the constraint thresholds, the EI-FP may face difficulties in rapidly converging to these frontier regions. The authors also mention that Sasena's method (3.9) has the same problem, but in this case, the reason is lack of an exploration measure (uncertainty) in representing the constraints. However, it is not uncommon to have an optimum on the boundary of some constraints (Cho et al., 2015); hence, some methods have been offered to deal with such cases more properly. In reliability and robustness analyses, regions close to the state limit of the constraint (i.e., taking as a boundary) are also important. This indicates that we may need to approximate a function accurately enough around a certain level-set (i.e.,  $[T - \varepsilon, T + \varepsilon]$ ). With this in mind, it has been proposed to build a Kriging surrogate that is accurate in the vicinity of the target boundary in Picheny et al. (2010). In order to have a trade-off between concentrating on the boundary regions and reduction of the global uncertainty (indicated by the Kriging prediction error), they suggested an infill criterion. This measure includes the square of the Kriging prediction error, which is weighted by the probability of being inside the desired interval  $[T - \varepsilon, T + \varepsilon]$ . This probability is calculated based on the feasibility probability and thresholds are now lower and upper bounds of the interval:

$$\Phi\left(\frac{T + \varepsilon - \hat{g}_i(\mathbf{x})}{\hat{s}_i(\mathbf{x})}\right) - \Phi\left(\frac{T - \varepsilon - \hat{g}_i(\mathbf{x})}{\hat{s}_i(\mathbf{x})}\right). \quad (3.22)$$

This idea is similar to the one in Ranjan et al. (2008), where following a contour within an  $\varepsilon$ -wide band is intended and the authors have redefined EI in a way to balance sampling both in the vicinity of the boundaries and outside the desired region where uncertainty is high. In Basudhar et al. (2012), the boundary of the feasible domain is approximated by a Support Vector Machine (SVM) model and the probability of feasibility is calculated using a new probabilistic SVM model, which is a modification of the sigmoid function. In their proposed methods, there are two stages; the first one is dedicated to the global search of the constrained optimum using EI of the objective and feasibility probability of constraints; while the second stage is focused on the local refinement of the SVM approximating the boundary of the feasible space adaptively. In Chen et al. (2014), the authors suggest adding points efficiently around an obtained design point, especially when it is close to the constraint boundaries to be able to fit a better Kriging locally. Their measure involves the probability of feasibility, which is weighted by some distance measure to current samples

and then summed over all constraints. This measure takes the approximation of the output into consideration and hence has an advantage over just adding samples based on a certain distance. It is used when there are enough samples locally. Later in the context of the reliability analysis, it was suggested that only the critical part of the limit state (i.e., the constraint boundary) needs to be fitted accurately. Therefore, two (importance) coefficients were introduced to find such critical sections of boundaries; one based on the value of the objective function and the other based on the joint probability density value of the design variables Chen et al. (2015). In Cho et al. (2015), a criterion is suggested that tries to put samples inside feasible regions in early iterations; as the approximation of this region becomes more accurate (the standard deviation of the prediction decreases), emphasis is shifted to samples around boundaries. To find the global optimum around boundaries with the least amount of sampling, the following infill criterion was introduced in Boukouvala and Ierapetritou (2014),

$$E_{\text{feas}}[I(\mathbf{x})] = \hat{s}(\mathbf{x})\phi\left(\frac{0 - \hat{c}(\mathbf{x})}{\hat{s}(\mathbf{x})}\right). \quad (3.23)$$

It is basically the second part of the EI where  $y_{\min}$  is set to zero. In other words, if  $c$  is the value of a constraint (and  $\hat{c}$  its approximation), then zero is the limit (state) value (for the standardized constraint  $c(\mathbf{x}) \leq 0$ ). By maximizing  $E_{\text{feas}}$ , on the one hand, points that maximize  $\phi(\cdot)$  will be good candidates, i.e., those that are at the boundary of the prediction. On the other hand, we are looking for maximum uncertainty (regions with a low number of samples). So, in general, we try to sample around the boundaries while trying to avoid putting samples too close to each other (role of  $\hat{s}(\mathbf{x})$ ). This is an effective criterion and as the authors claim, with a few numbers of samples, a strong improvement in the prediction of the feasibility region can be obtained. Authors use  $E_{\text{feas}}$  to concentrate on promising regions, where they also use trust-region and clustering to increase the convergence rate (compared to just globally optimizing).

It should be noted that the literature for searching close to the boundary of the feasible region for evolutionary algorithms is reviewed by Coello Coello (Coello Coello, 2002). Here, techniques such as an adaptive penalty or other operators that allow crossing the boundary via relaxation or strengthening of the constraints are employed. However, as Coello Coello mentions, these operators can be highly dependent on the chosen parameter and may have difficulties with disjoint regions.

### 3.3.6. Other constraint handling methods

A new two-phase method for solving constrained problems is proposed in Li et al. (2017b). In the first phase, a feasible solution should be found by maximizing the feasibility probability of the maximum constraint violation considering a distance penalty. This maximum violation among all constraints is approximated by a Kriging model. In the second phase, the aim is to find a better feasible solution by minimizing the lower bound infill criterion (i.e.,  $\hat{y}_{\text{objective}} - k * \hat{s}_{\text{objective}}$ ) subject to two constraints; the first is that feasibility probability should be higher than a certain value and the second is a distance penalty. The value of 'k' in the infill criterion is also defined by a routine. As another way of handling constraints, Picheny et al. calculate the expected value of the

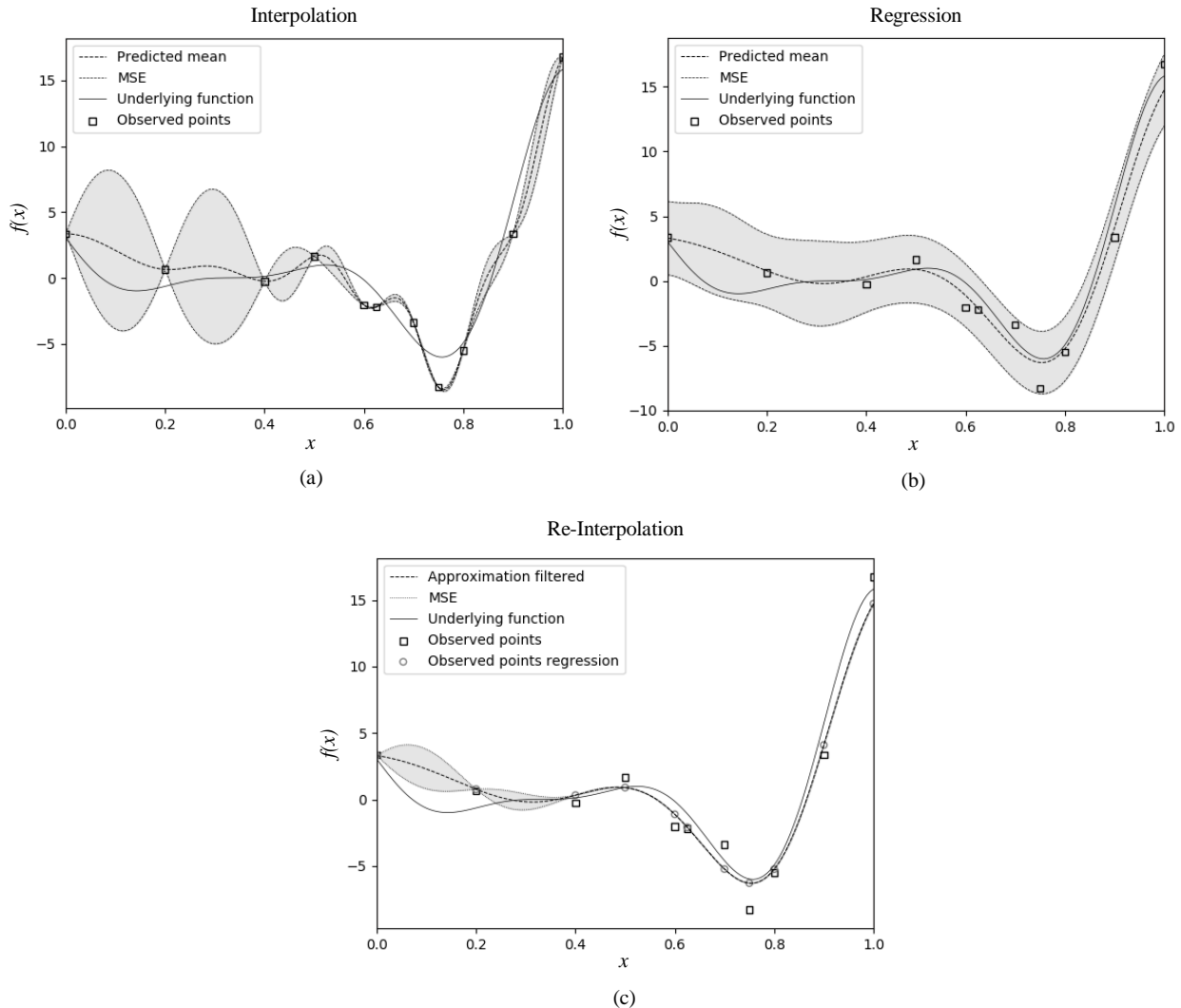
feasibility probability (i.e., the integral of the probability of improvement (CDF) over the design space). This is a measure of uncertainty about the location of the minimizer ( $\mathbf{x}_*$ ) of the desired function. When it is large, it shows that many designs are likely to be better than the current one and when it is low, very little can be obtained by conducting the optimization. This measure is calculated using numerical methods and tries to automatically balance the exploration of feasible regions and sampling in promising regions (Picheny, 2014). As the authors mention, there are limitations for problem dimensionality, regularity of the function and sample size. In the case of multiple constraints, a proposed way to represent the feasibility is by considering the maximum of all constraints in a procedure, see Chunna et al. (2020). However, this method includes fuzzy clustering, which requires a lot of samples and hence we do not review it further here.

### 3.4. Handling noisy simulations

Crash simulations are well-known for the presence of noise and consequently unreliable gradients for optimization. Fitting a GPR to a noisy problem can lead to an overfitting case. The fitted model tries to follow the noise and would have unnecessary ups and downs, which can mislead the infill criterion and consequently sacrifice the EGO efficiency. One may suggest using regularization to smooth out the fitted curve as previously shown in Figure 2.3. The larger the regularization, the more regressive (in contrast to interpolative) and smooth the fitted surface is. This regularized surface is suitable if one desires to have a model representing the simulation, but it can cause problems if it is used in EGO. When regularization is added, the predicted variance at the evaluated points increases. If the regularization is too high, the variance starts to become similar at neighboring points and therefore EGO can get stuck unnecessarily in such regions for a while. One remedy for this problem is re-interpolation, described in Forrester et al. (2006) and summarized in the following. First, a regularized GPR surface is fitted to the current data to obtain a smooth approximation of the noisy output. Next, we need to define the final GP surface that solves the non-zero variance problem at the evaluated samples. This surface has the same hyperparameters as the current surface (i.e., no fitting is required). Now, we can just use this surface in infill optimization without problems. Crash scenarios that we consider in this work are noisy, but not to a large degree, as some data sets are, for example, housing prices in terms of the building and neighborhood properties. In addition, based on our experience, EGO with properly fitted GPR puts samples not extremely close to each other and hence avoids creating a highly noisy situation, especially in dimensions more than one. Therefore, the original EGO with suitable regularization such as  $1e-6$  and  $e-7$  is good enough for the crash cases of interest within this thesis. Hefele investigated the effect of regularization on EGO performance over some crash simulations under the supervision of the author of this thesis, see Hefele (2017) for more information. In this regard, when we say that EGO is interpolative, it is meant in a relative sense. Due to the numerical precision, no case is 100% interpolative, in the sense that the predicted output will eventually be different from the actual output at locations where the problem has already been evaluated. This difference is also not necessarily much smaller than the regularization. Based on the author's experience, it is better



to keep the regularization at least to  $10^{-10}$  by default even for interpolative cases, to prevent to a large degree some numerical issues such as a non-invertible covariance matrix. It should be noted that there are other ways to deal with noisy cases with variants of EI, see Picheny et al. (2013) for more details.



**Figure 3.7** Figure (a) shows the result of an interpolative GPR curve in case of a noisy function. Here noise is added to a test function labeled as the "underlying function" (b) When regularization is applied excessively, the standard deviation of prediction, here labeled as "MSE", becomes less distinct at each point compared to its neighbors, causing problems for EGO. (c) Re-interpolation solves the problems in (a) and (b). There will be a smooth curve representing the noisy data and the predicted standard deviation has its desirable form (Hefele, 2017).

### 3.5. Handling EGO more efficiently - larger space treatments

The number of dimensions and bound constraints (i.e., range) of each dimension have a large impact on the performance of nonlinear optimization. Hence, one tries to reduce both to just the necessary values. Accordingly, one way of gaining efficiency in EGO is through the proper treatment of the hyperparameter  $\theta$ . This can be helpful not only for the optimization of the likelihood function, but also for counteracting overfitting, as we discuss later in Chapter 5. First, we have a look at the literature to see how the range for hyperparameters is set. In this regard, most of

the reviewed publications either do not mention the selected range for  $\theta$  or consider some common (large) intervals, e.g., (0.005,5000) in Marrel et al. (2008). Note that we have modified this range to match the GPR definition here, whereas originally the authors used Kriging. Picheny et al. conducted some pre-experiments to set a limited range for  $\theta$  before actual fitting (Picheny et al., 2013). A better approach is given by MacDonald et al. (2015), in which the authors try to define the range by setting limits on the correlation model ( $R$ ) in Kriging, i.e., they consider  $0.00067 \leq R_{ij} \leq 0.9999$  and assume the same average distance and smoothness in each coordinate; therefore, the hyperparameter for each dimension ( $\theta_i$ ) will be within (again, converting for the GPR covariance model that we use here):

$$\sqrt{0.5 * 10^{-\log_{10} 500 + \log_{10} d}} \leq \theta_i \leq \sqrt{0.5 * 10^{2 + \log_{10} d}}, \quad d : \text{dimension}, \quad (3.24)$$

for example, in a 10-dimensional problem, the range is  $0.1 \leq \theta_i \leq 22.36$ . We will see later in Chapter 5 that these ranges can be limiting for some problems.

Another way of gaining efficiency is by updating  $\theta$ , only when necessary and not in each iteration. However, this requires a metric and in Gano et al. (2006) such a measure is developed based on a trust-region concept, which evaluates how well the fitted Kriging approximates the true model. A more cost-effective EGO can also be obtained by a more efficient calculation of the likelihood function and its derivatives, for example, by an adjoint method, which is proposed in Toal et al. (2009). Hence, it can be used in various optimization algorithms (derivative-based or -free). Based on the application, availability of derivatives or affordability of a larger number of evaluations, various improvements of EGO have been proposed, see, e.g., Lam et al. (2018). However, in this work, we focus on the original EGO and with a quite limited budget and neglect unreliable derivatives of the underlying function.

Obtaining a more efficient result in higher dimensions commonly involves the divide-and-conquer strategy to reduce the space dimension or size. For example, trust-region methods, clustering, PCA, or other approaches are proposed that use the fact that some dimensions have limited effect, see Long et al. (2015). If all dimensions are effective more or less, an additive Bayesian approach is suggested in Kandasamy et al. (2015). This method represents the objective function as the sum of lower-dimensional covariance functions with mutually disjoint dimensions. Then, BO can be done separately in these sub-dimensions. The authors claim, this additive BO works not only for functions that are additive but also for some non-additive cases. It seems this would be more helpful for problems with more separable inputs. In cases where only some dimensions are contributing to the output, but we do not know which ones, Wang et al. (2013) proposed considering an embedded lower-dimension space; e.g.,  $x_1 = x_2$  instead of the entire two-dimensional space. Then, optimization can be done efficiently in this subspace obtained by a random transformation matrix. The assumption that only some dimensions are contributing was relaxed in Qian et al. (2016), considering that all dimensions are effective, but some have small bounded effects and consequently, a sequential random embedding can be used. If several thousands of samples are affordable, a pro-

posed method is using PCA (every 50 iterations) to select a subspace of important dimensions and conducting BO in this subspace while the least important coordinates are set to their best current values (Ulmasov et al., 2016). In Long et al. (2015) the objective and all constraints are combined into one so-called merit function and instead of fitting an RS to each output, one RS is fitted to this merit function and its optimum is obtained during the search for the important part of the design space. Authors in Li et al. (2017a) suggested dropping out randomly some dimensions at each iteration in a high-dimensional BO, which was motivated by the dropout algorithm in training an ANN. The values of the left-out variables are set as random, the best function value so far or their mixture. Wang and Simpson used an RS (e.g., Kriging) to generate a high number of points in the design space and screened variables by clustering. They reduced the space even more by selecting the promising cluster and then employing Kriging progressively to find the optimum (Wang and Simpson, 2004).

## Chapter 4

### Enhancements of EGO - Part I: Dimension adaptation and reduction, shape modification, sample translation and hybridization

This is the first result chapter in which we introduce our proposed modifications and discuss their performance at least compared to those obtained by the standard EGO. The introduction begins by providing background information required for the questions that are raised at the end of this section and addressed in the rest of this chapter. Section two describes the proposed methods in detail, which are compared over various problems in the third section. This chapter ends with a discussion and some further suggestions.

## 4.1. Introduction

Standard EGO has a tendency to visit locations that have not been explored, mainly due to the lack of samples in these regions. However, not all of these locations are actually promising ones. An important question is how we can prevent unnecessary explorations. Another issue that affects the performance of EGO is the number of hyperparameters. With each added dimension, the curse of dimensionality becomes more influential and hence, undesirable results may follow. If we do not want to reduce the dimension, the second-best option is to reduce the size of the space. So what if we change the size of the space based on the design variables' contributions to the output? In addition, how about making this change gradually based on the iteration? Considering the different contributions to the output by design variables (DVs), can we use this information to focus more on the promising part of the space?

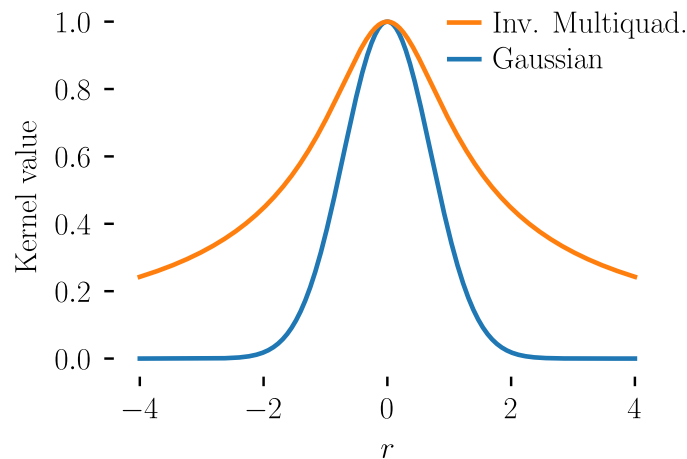
Sensitivity analysis is performed to identify the contribution of inputs to the output. Consequently, only important dimensions would be considered. However, in practice, (some) less important design variables may still be kept, see, e.g., Antinori (2017). This happens since although some inputs may not be relevant for all outputs and/or disciplines, they are relevant for some. Another reason is to simplify the whole procedure. In the end, what matters is not the reason behind keeping dimensions, but its consequence, i.e., the effects of the curse of dimensionality. So, here, the most fruitful procedure is not the most practical one. Perhaps, we need to compromise. In this regard, we may wonder if we can keep DVs, but treat them flexibly, sometimes as important and sometimes not, while using a suitable strategy for weakly connected inputs to make optimization (here EGO) more efficient.

Based on our deduction from the NFL theory in this thesis, if an algorithm is decent over various types of problem, then it must have been enriched by (rather) complementary strategies (besides the proper detection mechanism to use which strategy when). In the context of EGO, such complementary behavior can be realized via, e.g. incorporating different kernels. As reviewed in Section 2.5.2, any kernel that leads to a positive definite covariance matrix is an acceptable kernel for GPR. An interesting set of kernels is offered by the RBFs. For example, a locally refined RBF surface based on the multiquadratic kernel (MQ) was used in a sequential optimization of a strip bending process (Havinga et al., 2013). However, based on the author's experience, the MQ kernel leads to a non-positive definite covariance matrix and hence it is not appropriate for GPR directly. Nevertheless, inverse multiquadratic (IMQ) does not show such a problem. IMQ is defined by:

$$k(x_i, x_j) = \frac{1}{\sqrt{1 + (\theta|x_i - x_j|)^2}}. \quad (4.1)$$

' $\theta$ ' determines the region of influence of the kernel, which, like other hyperparameters, is defined by cross-validation or comparable methods. Given the same ' $\theta$ ', IMQ decays slower than the Gaussian kernel. This means that one can assume that IMQ carries information further than the Gaussian does (see Figure 4.1). So, it is expected that GPR based on the IMQ also carries information further than GPR based on the Gaussian kernel. This difference can have significant

consequences, including behavior close to boundaries, which is particularly of interest for EGO. Therefore, another topic to be investigated in this chapter is whether and for which cases the IMQ kernel taken from the RBF family can make significant improvements in the context of EGO in comparison to the common Gaussian kernel. In addition, what if one tries to use the advantage of both kernels and has a hybrid optimizer? How does this hybrid method perform compared to EGO?



**Figure 4.1** IMQ and Gaussian kernel values vs. distance from the kernel location for  $\theta = 1$ .

Based on what has been discussed so far in this chapter, we propose some modifications and methods to improve the performance of the EGO algorithm, including:

- Modification of the range of the GPR hyperparameters based on how influential the corresponding design variables (DVs) are on the output (minimum, medium or maximum).
- Optimizing the EI function only on the influential DVs and defining the value of the rest of DVs by using a part of the MVMO algorithm.
- Using the IMQ kernel to construct the covariance matrix in GPR (IMQ\_GPR) and check its performance compared to the common EGO using a Gaussian kernel.
- Modification of sample locations close to the boundaries using the common GPR mean and IMQ\_GPR mean.
- Developing a hybrid meta-model-based optimizer, using both common GPR and IMQ\_GPR.

These modifications are applied alone or together and their performances are compared with that of the common EGO algorithm. The author of this thesis used the GPR code from the scikit-learn (Pedregosa et al., 2011) package and modified it for the proposed methods and increased its speed. The code for IMQ was already available in a code developed by the author earlier. The integration of both codes made it possible to use the IMQ instead of the Gaussian kernel in GPR (IMQ\_GPR) and then EGO. We refer to this EGO in the images simply as IMQ. The developments proposed in this chapter have already been published largely in Hefele (2020) supervised by the author of this dissertation.

## 4.2. Suggested modifications and methods

So far in this chapter, some questions have been raised in search of enhancing EGO. In this section, the corresponding proposed modifications are described in detail.

### 4.2.1. Dimension adaptation and shape modification

Here it is proposed to divide the design variables into three classes with minimum, medium and maximum effects on the output. This can be advantageous when we want to treat the dimensions differently, which is here called dimension adaption option. Furthermore, we propose to treat each dimension individually via the hyperparameter  $\theta$  of the kernels in GPR. When anisotropic kernels are used, each dimension has its own hyperparameter. For a weakly contributing design variable, the corresponding  $\theta$  should lead to a surface in which the output does not change much along that direction; in other words,  $\theta$  should be large enough (see Figure 4.2). Therefore, we can modify the lower bound of  $\theta$  to obtain (hopefully) better results based on the contribution level of the corresponding dimension.

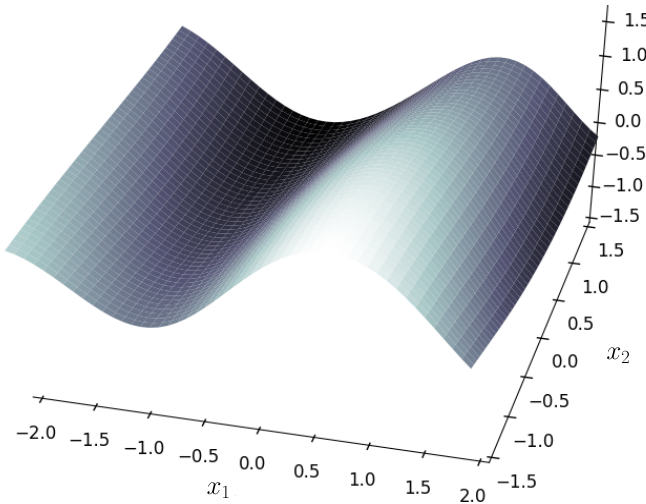
For EGO, we commonly have a default range  $[lb_{\min}, ub_{\min}]$  for all  $\theta$ s. Based on what was discussed above, the intention is to select higher  $\theta$  values for less important design variables and lower values for more influential design variables. Here,  $\theta$  is also called shape parameter and its range based on the considered classification of the influential design variables will be different. Each class will be assigned a range that is called shape adaptation (SA). In addition, ranges can also be changed linearly during optimization. Starting at the category-specific bound and ending at the default value. Hence, e.g., a lower bound at an iteration is calculated via:

$$lb_{\text{iter}} = lb_0 + (n_{\text{iter}} - n_0) \frac{lb_1 - lb_0}{n_1 - n_0}, \quad (4.2)$$

where  $n_{\text{iter}}$  is the current iteration,  $n_0$  and  $n_1$  are the initial and last iterations, respectively,  $lb_0$  is the initial lower bound (here  $lb_{\min}$ ,  $lb_{\max}$  or  $lb_{\text{med}}$ , see Table 4.1) and  $lb_1$  the final value (here  $lb_{\text{default}}$ ). This linear change is called shape interpolation (SI), which, of course, is applicable if shape adaptation is used first. The purpose of SI is to gradually improve the quality of the fitted model as more samples are added (using the information history to make the algorithm more efficient).

### 4.2.2. Dimension reduction with the help of the MVMO algorithm

As mentioned earlier, some design variables may have a small impact on a certain output, but they may still be kept in the solution. Hence, the cost of finding the values of these less important dimensions should be reduced. The initial idea was to perturb an already promising value by getting inspiration from the PS algorithm. Another idea was to be inspired by a successful algorithm for high-dimensions. In this regard, Hefele suggested using MVMO (Erlich et al., 2014) as it previously drew the attention of the author of this work to improve high-dimensional EGO. MVMO is a single parent-offspring algorithm that uses an archive of n-best samples (usually 2-5). The best sample from the archive is selected as a parent and also a mapping function is developed from the



**Figure 4.2** A realization of a GPR in two dimensions. Output (vertical axis) changes considerably less along the  $x_2$  direction than the  $x_1$  direction. Therefore,  $\theta_2$  should be higher than  $\theta_1$

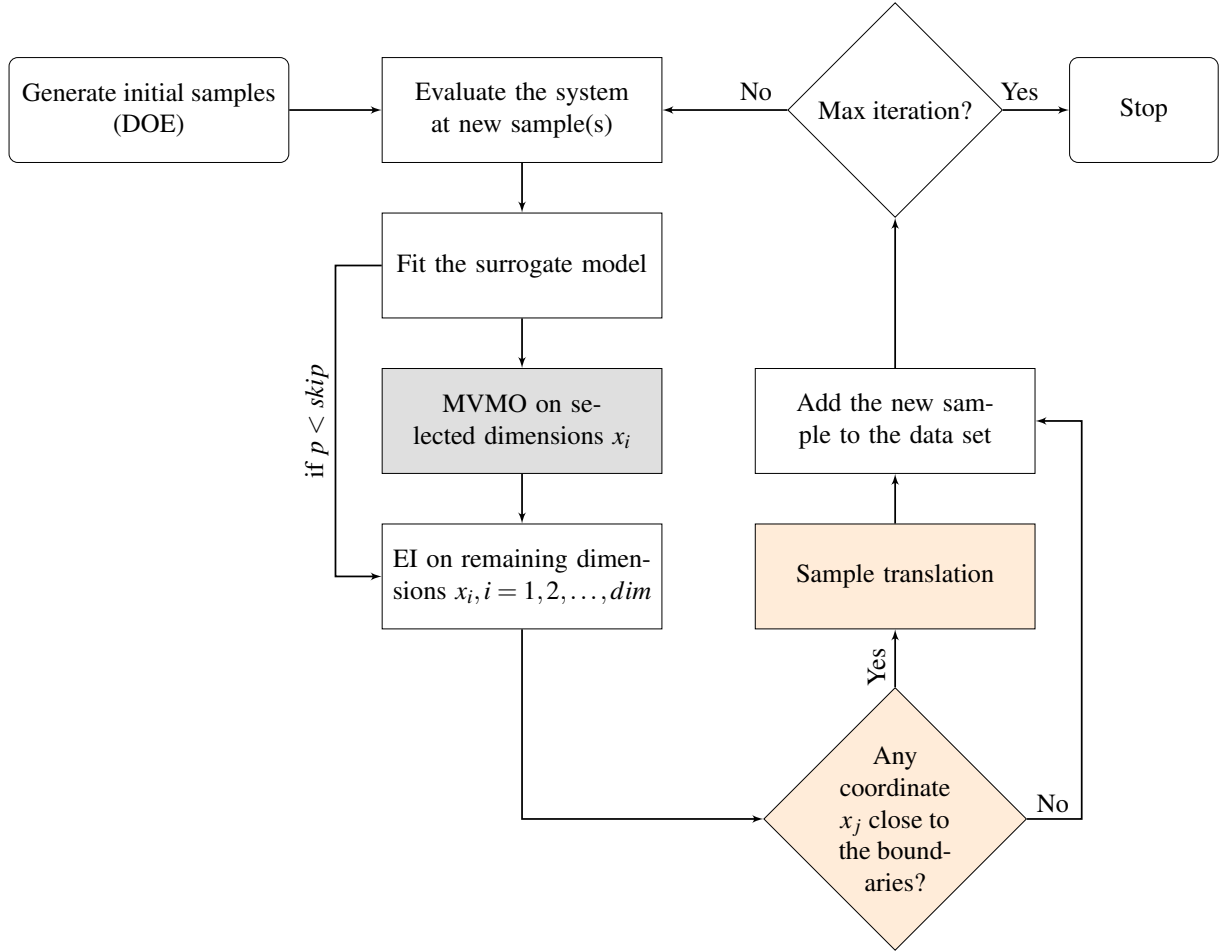
**Table 4.1** Classification of design variables based on their influence on the output. For each class and ID, a certain range for the shape parameter ( $\theta$ ) is considered.

Design variables classification		
Influence on output	ID	bounds
minimum	0	$[lb_{\min}, ub_{\min}]$
medium	1	$[lb_{\text{med}}, ub_{\text{med}}]$
maximum	2	$[lb_{\max}, ub_{\max}]$

archive which is used for mutation of the parent. The mapping function takes the mean and scaled natural logarithm of the variance of each dimension as input. Only some of the parent dimensions are mutated by transferring a random number using the mentioned map. The rest of the dimensions of the offspring are set using the values from the parent through the crossover operation, (Erlich et al., 2014). MVMO is suitable for our intention as in the offspring creation phase, it separates the dimensions and perturbs around the best solution (i.e., mutation) as we intended to do so. In EGO there is an initial phase, so we already have some samples and the MVMO initialization procedure for its mapping function is not required. This separation of DVs helps us to reduce the dimensionality of the infill criterion in EGO. Basically, we optimize the infill criterion (EI) for only important dimensions and the rest is varied around the best samples so far using the MVMO sampling strategy. This option is used on less important DVs. In order to avoid getting stuck in a subdomain, every several steps, the infill criterion (EI) is optimized on all dimensions (i.e., the common way) and MVMO is skipped. Skipping occurs when  $p < skip$ , where  $p$  is a random number in  $[0,1]$  generated in each iteration and  $skip$  is also a (fixed) parameter in  $[0,1]$  set by the user



in advance. The higher the skip parameter, the more frequent the skipping of MVMO.



**Figure 4.3** EGO algorithm with partial MVMO-sampling and sample translation. The gray box shows MVMO and the light orange boxes indicate sample translation in the proposed modified EGO.

#### 4.2.3. Sample translation

In extrapolation or when the test points are away from the points evaluated so far, the stochastic part of the GPR declines and eventually GPR converges to its trend value, which is a constant here. This possible decline in the predicted mean value plus the higher variance at these test points (because they are away from others) make them good candidates for future evaluations in the eyes of the infill criteria. Therefore, boundary regions have the potential to be considerably explored by EGO. It is proposed here to complement this by IMQ, which carries information further than Gaussian. So, if the predicted mean using IMQ is higher than the one using the Gaussian kernel (common GPR) and a coordinate is closer than 2% to the boundaries, then we relocate the coordinate based on the uniform distribution within 10% of the boundaries:

$$x_{\text{final}} = \begin{cases} \mathcal{U}(0.0, 0.1) & \text{if } x_i < 0.02 \text{ and } \hat{y}_{\text{GPR\_IMQ}} > \hat{y}_{\text{GPR}} \\ \mathcal{U}(0.9, 1.0) & \text{if } x_i > 0.98 \text{ and } \hat{y}_{\text{GPR\_IMQ}} > \hat{y}_{\text{GPR}} \\ \text{No change} & \text{else,} \end{cases} \quad (4.3)$$

where  $\mathcal{U}$  is a uniform distribution. This relocation is made at the point obtained by optimizing EI and before adding it to the data set, see Figure 4.3. We call this modification "sample translation" (ST). The intention behind this relocation is to push samples back (mostly) away from boundaries when we expect this to be appropriate. In the cases relevant to this thesis, this can enhance the efficiency of the EGO. Note that we are looking for fast improvements in optimization. If the intention is to have a better convergence, then perhaps this option may be switched off in the last iterations or its effect may be reduced gradually. In addition, it is recommended first to check if the modified candidate is away from the (so far) best samples to tune this option accordingly.

#### 4.2.4. A hybrid meta-model optimizer

In this section, we propose an optimizer based on a hybrid of two RSMs. One is the common GPR with Gaussian kernel and the other one is GPR where the covariance matrix is built from the IMQ kernel. We refer to them as GPR and IMQ\_GPR respectively. Hybrid or meta-model ensembles were previously reviewed in Section 2.2. One advantage of the proposed ensemble is that both methods (GPR and IMQ\_GPR) have uncertainty predictions in their DNA, which can help balance exploration and exploitation automatically by methods like EI in the EGO algorithm. The details of the method are explained below.

A kernel that acts as a local weighting is placed at the location of each adaptive sample. Based on the prediction, if the GPR error (previous iteration prediction compared to the actual value from the current iteration) is less than the IMQ\_GPR error, the kernel is multiplied by +1, otherwise by -1, see Equation (4.4). To have global coverage, the weights are summed up and if at a location the value is above zero, then GPR is used in EGO, otherwise, IMQ\_GPR will replace it, see Equation (4.5). When there is no difference, one of the models is chosen randomly. The function that helps us opt for one of the meta-models is defined as  $S(\mathbf{x})$ :

$$S(\mathbf{x}) = \sum_{i=1}^n w_i \exp\left(-\frac{1}{2\theta^2}(\mathbf{x}_{\text{adp}} - \mathbf{x})^T(\mathbf{x}_{\text{adp}} - \mathbf{x})\right), \text{ with } \theta = 0.2, \quad (4.4)$$

$$w_i = \begin{cases} +1 & \text{if } \text{error}_{\text{GPR}} < \text{error}_{\text{IMQ\_GPR}} \\ -1 & \text{if } \text{error}_{\text{GPR}} > \text{error}_{\text{IMQ\_GPR}}, \end{cases}$$

where  $\mathbf{x}_{\text{adp}}$  represents adaptive samples and  $\mathbf{x}$  prediction points.  $n$  is the number of adaptive samples. So, the hybrid surrogate model, which will be called HSM is obtained from:

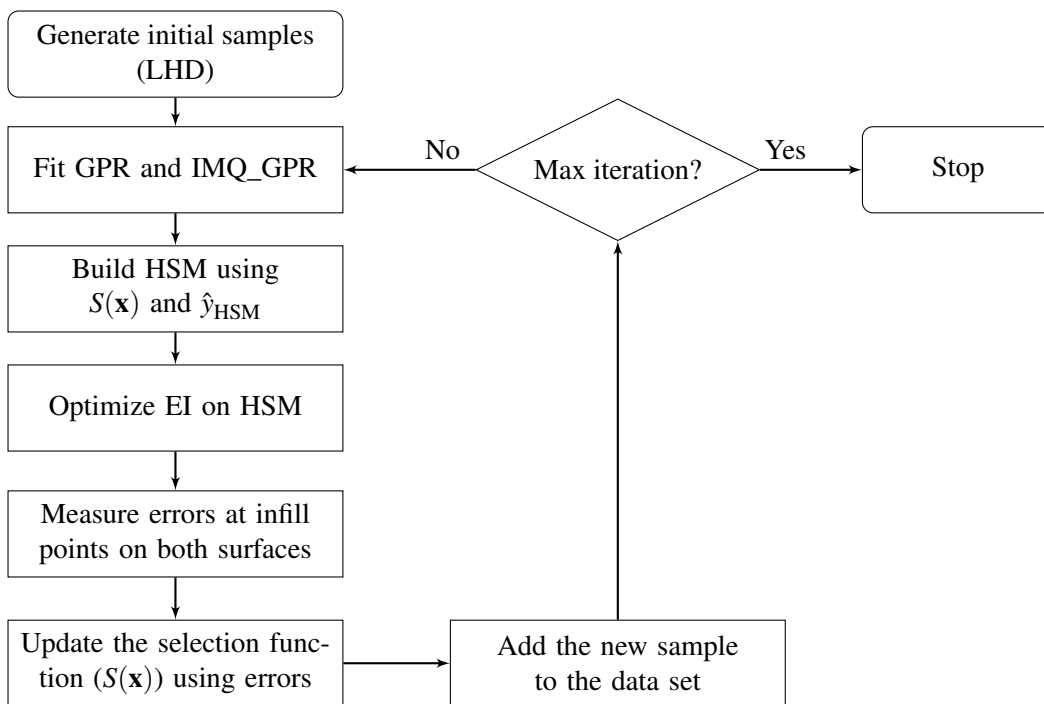
$$\hat{y}_{\text{HSM}} = \begin{cases} \hat{y}_{\text{IMQ\_GPR}} & \text{if } S(\mathbf{x}) < 0 \\ \hat{y}_{\text{GPR}} & \text{if } S(\mathbf{x}) > 0 \\ \text{randomly select } \hat{y}_{\text{IMQ\_GPR}} \text{ or } \hat{y}_{\text{GPR}} & \text{else.} \end{cases} \quad (4.5)$$

In each iteration of the HSM, both GPR and IMQ\_GPR are fitted. EI is calculated based on Equation (4.5). After optimizing EI, the error at the new suggested point is measured for both metamodels and a new kernel with the weight defined by Equation (4.4) is added to the selection

function for the next iteration, see the flowchart in Figure 4.4. Note that this proposed weighting is developed based on the following principles:

- Weights are defined based on the predicted error w.r.t. the actual function values.
- Weights are defined locally and should gradually become the same away from the samples.
- Weights are updated based on history.

The author of this work prepared the code for common EGO and EGO with IMQ\_GPR and defined the mentioned principles. Hefele suggested the procedure for weighting and combination in his Bachelor's thesis (Hefele, 2020). The weighting is based on our previous work (Komeilizadeh et al., 2018).



**Figure 4.4** The proposed hybrid surrogate model (HSM) procedure.

### 4.3. Problems and results

In this section, the modified methods mentioned above are compared with the common EGO. There are two mathematical test functions and two mechanical problems. Besides the plots in Figures 4.22 and 4.23, we have already reported the rest in Hefele (2020). Some of the plots have undergone minor changes in this work to have better visual fidelity.

### 4.3.1. Prerequisites and settings

In this chapter, we consider optimizing a single objective function  $f(\mathbf{x})$ , while constraints are handled by the penalty method.

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_p(\mathbf{x}) = f(\mathbf{x}) + \sum_{i=1}^N p_i \max(c_i(\mathbf{x}), 0) \\ \text{subject to} \quad & x_i^{\text{lb}} \leq x_i \leq x_i^{\text{ub}}, \quad i = 1, \dots, d, \\ & c_i(\mathbf{x}) \leq 0, \quad p_i > 0. \end{aligned} \quad (4.6)$$

The inequality constraints are considered in the standard form in Equation (4.6), that is:

$$\text{replace } c_i(\mathbf{x}) \leq a, \text{ with } \frac{c_i(\mathbf{x})}{a} - 1 \leq 0, \text{ or replace } c_i(\mathbf{x}) \geq b, \text{ with } 1 - \frac{c_i(\mathbf{x})}{b} \leq 0. \quad (4.7)$$

Each optimization is repeated 50 times with different initial LHD samples (but the same for the methods that are being compared). Since expensive crash simulations are the main application, we are interested in behavior under a low number of evaluations. Therefore, for EGO, the number of initial samples is set to  $n_{\text{init}} = 5 \times \text{dimension}$  and the number of adaptive samples is twice that of the initial samples.

We mentioned earlier that design variables are categorized by their influence. For test functions and the truss problem, the Sobol sensitivity analysis is performed using SALib (Herman and Usher, 2017) based on Saltelli sampling. It is suggested to use  $500 \times \text{dimension} \leq N \leq 1000 \times \text{dimension}$  samples to conduct this sensitivity and since we are considering here lower-dimensional problems, we can afford it. For more information on the sensitivity analysis, see Antinori (2017). We also need to be able to categorize the dimensions into three influential categories for some of the suggested methods, as mentioned in Table 4.1. Therefore, the result of the sensitivity analysis (total order terms from Sobol) is categorized into three groups with the help of the (one-dimensional) k-mean clustering from the scikit-learn package (Pedregosa et al., 2011). In the following, the discussed methods are applied to two test functions and two mechanical problems. The selection of cases has been based on discussing the advantages and limitations of the proposed methods.

**Table 4.2** Classification of design variables for the considered problems. See Table 4.3 for additional settings.

problem	classes	problem	classes
Hartman-6	['2', '1', '0', '2', '1', '2']	Truss	['0', '2', '1', '0', '0']
Himmelblau	['1', '0', '2', '0', '1']	Side sill impact	['0', '1', '2', '1', '0']

**Table 4.3** Common settings used for all problems.

parameter	value / type	explanation
problem dimension	$d$	number of design variables (DVs)
$n_{\text{init}}$	$5 \times d$	number of initial samples
$n_{\text{adaptive}}$	$10 \times d$	number of adaptive samples
default_bounds	[0.01, 100]	default shape bounds
min_kernel_bounds	[1, 1000]	bounds for minimum influential
medium_kernel_bounds	[0.01, 100]	bounds for medium influential
max_kernel_bounds	[0.01, 0.1]	bounds for maximum influential
influence_mapping	{'min_influence': '0', 'medium_influence': '1', 'max_influence': '2'}	labeling of the DVs' classes
MVMO applied to	{'0'}	only minimum influential DVs
SA/SI applied to	'0':[1,1000], '2': [0.01, 0.1]	min and max influential
ST applied to	{'2', '0', '1'}	all DVs
skipping MVMO	0.2	skip parameter in Figure 4.3
general optimizer	DE from SciPy	for EGO and the penalty method.

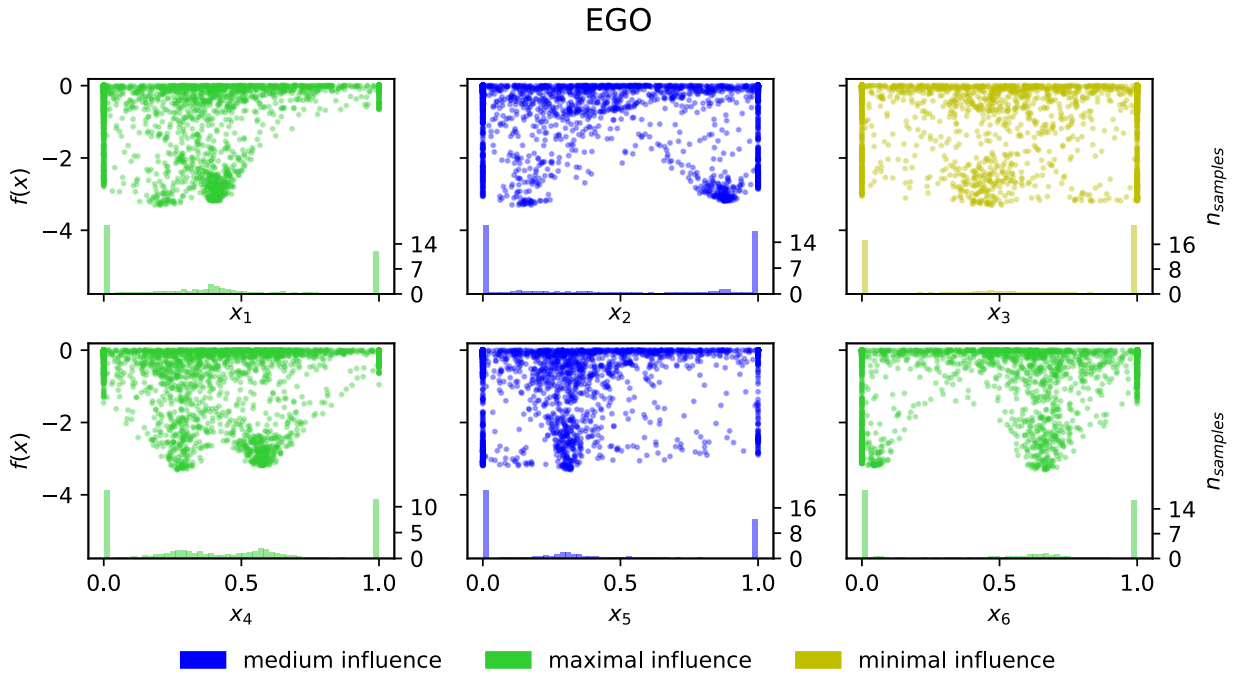
### 4.3.2. Hartman-6

This is a multi-modal six-dimensional mathematical test function (Surjanovic and Bingham, 2013).

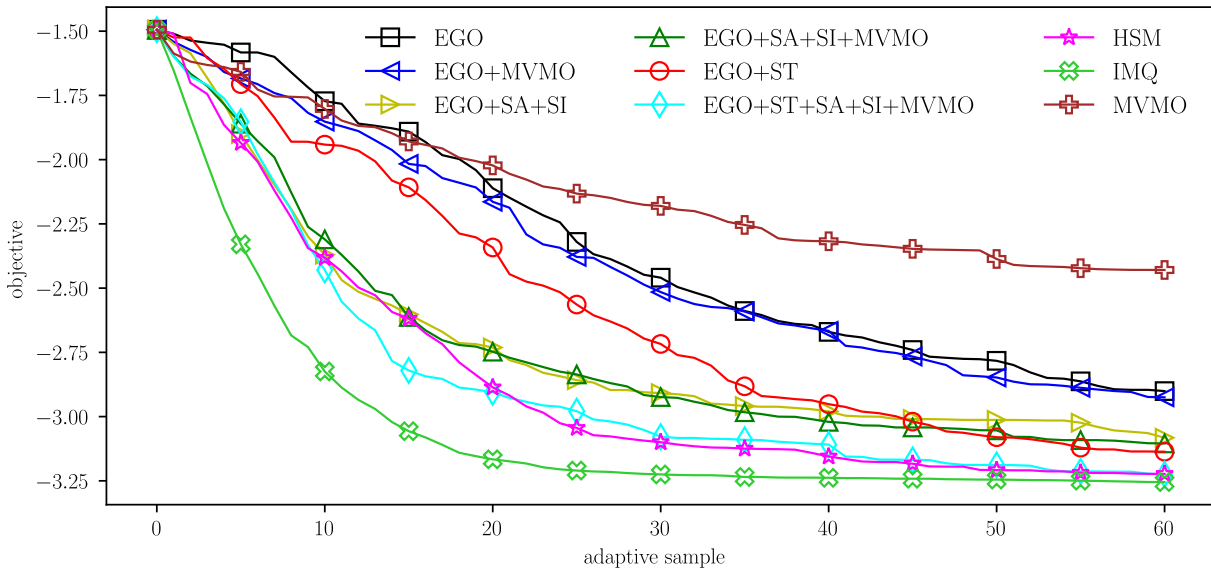
$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f(\mathbf{x}) = - \sum_{i=1}^4 \alpha_i \exp \left( - \sum_{j=1}^6 A_{ij} (x_j - P_{ij})^2 \right) \\ \text{where} \quad & \alpha = (1.0, 1.2, 3.0, 3.2)^T, \quad 0 \leq x_{1..6} \leq 1, \\ & A = \begin{pmatrix} 10 & 3 & 17 & 3.5 & 1.7 & 8 \\ 0.05 & 10 & 17 & 0.1 & 8 & 14 \\ 3 & 3.5 & 1.7 & 10 & 17 & 8 \\ 17 & 8 & 0.05 & 10 & 0.1 & 14 \end{pmatrix}, \\ & P = \begin{pmatrix} 1312 & 1696 & 5569 & 124 & 8283 & 5886 \\ 2329 & 4135 & 8307 & 3736 & 1004 & 9991 \\ 2348 & 1451 & 3522 & 2883 & 3047 & 6650 \\ 4047 & 8828 & 8732 & 5743 & 1091 & 381 \end{pmatrix} \times 10^{-4}. \end{aligned} \quad (4.8)$$

The global optimum -3.32 is at (0.20169, 0.15001, 0.47687, 0.27533, 0.31165, 0.65730) (Surjanovic and Bingham, 2013). First, let us look at the plot of all adaptive EGO samples from all

50 optimization repetitions in Figure 4.5. The concentration of samples at more than one location in each dimension hints at possible multi-modality. Many samples also end up close to the boundaries, which is not unusual for EGO.

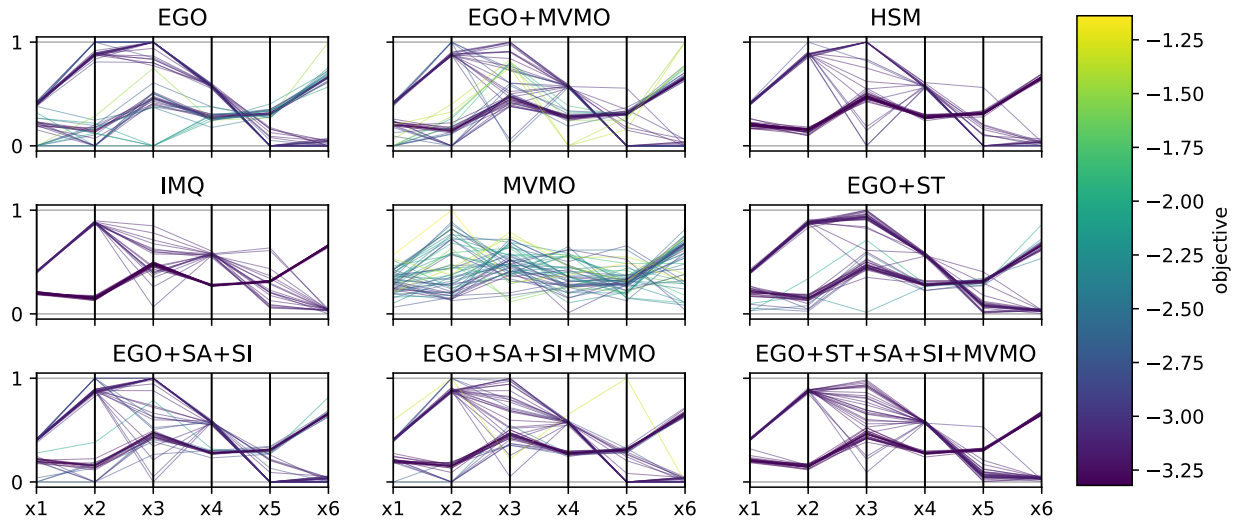


**Figure 4.5** All adaptive samples of all 50 independent optimizations are plotted vs. each dimension for the Hartman-6 problem. The distribution of each design variable is shown in the lower part.



**Figure 4.6** Mean values of the adaptive phase of all 50 repetitions of different methods are compared for the Hartman-6 test function. IMQ here means EGO with the IMQ kernel.

Figure 4.6 compares the performance of different methods. EGO with IMQ is the best among all, with the fastest improvement dominating here the traditional EGO. MVMO performs the worst, which is expected for problems with a lower number of evaluations and dimensions, according



**Figure 4.7** The parallel coordinate plot of all 50 repetitions of the proposed methods for the Hartman-6 problem. The darker the color the more desirable the value (we are conducting minimization).

to the author's experience with derivative-free generation-based optimizers. The suggested hybrid method also performs well, due to the good performance of the IMQ. Furthermore, it can be observed in Figure 4.6 that modifying the hyperparameter range and making a gradual change (SA and SI) also helps to obtain better results than for the common EGO. This is also true for the method that uses the translation of samples close to the boundary (ST). The proposed partial integration of MVMO into EGO does not lead to an improved result in this problem. Perhaps since here, only one dimension is considered to be minimally influential among 6 inputs.

One way of depicting problems with more than two inputs is to use a parallel coordinate plot with colors representing the objective values to better identify promising locations in space. Figure 4.7 shows this plot for the adaptive phase of all the considered methods. It can be seen that IMQ compared to EGO visits the boundary less often and heavily concentrates samples around certain values in some dimensions. EGO with sample translation (ST) also performs similarly to IMQ, with more concentrated samples on certain values, while MVMO is searching around more than other methods indicated by more bright colors.

### 4.3.3. Himmelblau

This function is a multi-modal constrained problem in five dimensions (Coello Coello, 2000).

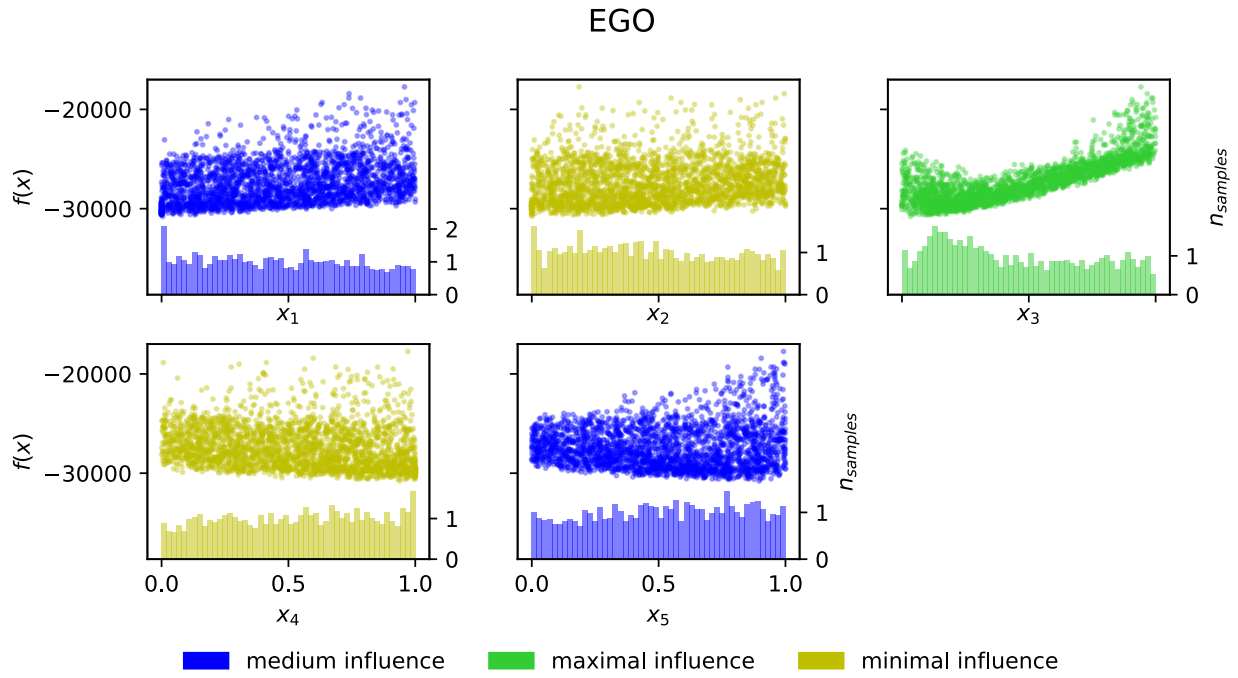
$$\begin{aligned}
& \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) = 5.3578547x_3^2 + 0.8356801x_1x_5 + 37.29329x_1 - 40792.141 \\
& \text{subject to} && c1 = 85.334407 + 0.0056858x_2x_5 + 0.00026x_1x_4 - 0.0022053x_3x_5, \\
& && c2 = 80.51249 + 0.0071317x_2x_5 + 0.0029955x_1x_2 + 0.0021813x_3^2, \\
& && c3 = 9.300961 + 0.0047026x_3x_5 + 0.0012547x_1x_3 + 0.0019085x_3x_4, \\
& && 0 \leq c_1(\mathbf{x}) \leq 92, \quad 90 \leq c_2(\mathbf{x}) \leq 110, \quad 20 \leq c_3(\mathbf{x}) \leq 25, \\
& && 78 \leq x_1 \leq 102, \quad 33 \leq x_2 \leq 45, \quad 27 \leq x_{3,4,5} \leq 45.
\end{aligned} \tag{4.9}$$

Here, the constraints are standardized as described above and we use a penalty approach (penalty factor = 50,000) to create an unconstrained objective. Our main intention is to see how the proposed methods perform on this function and not on the feasibility or constraint handling. The final formula is then given by:

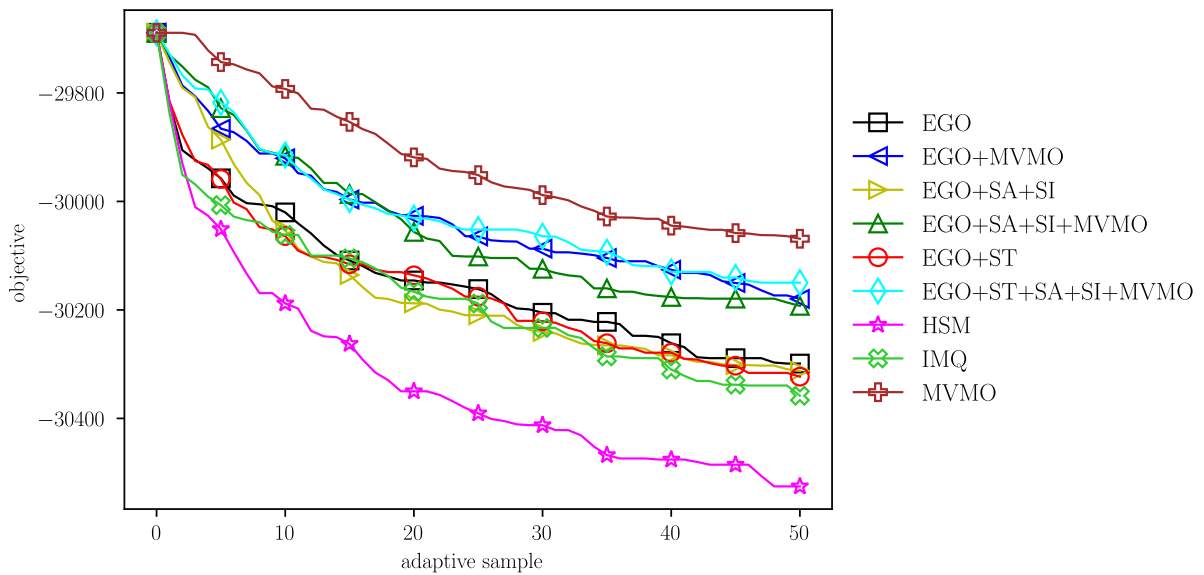
$$f_p(\mathbf{x}) = f(\mathbf{x}) + p \sum_{i=1}^3 (\max(g_{i_{lb}}(\mathbf{x}), 0) + \max(g_{i_{ub}}(\mathbf{x}), 0)), \quad p = 50,000. \tag{4.10}$$

IMQ performs slightly better than EGO, but interestingly, the hybrid method (HSM) is considerably better than all other methods. Shape translation does not improve EGO here. However, a closer look at the HSM in Figures 4.10 and 4.12 shows the tendency of having lower values close to even more than one boundary, while the ST method tries to keep the samples away from the boundaries when it finds visiting such regions is non-productive. Therefore, at least ST has not made EGO worse in this case where the desired values are closer to the boundaries. It seems that the combination of SA and SI to reduce the range of the hyperparameters here does not make an improvement over EGO. A possible reason is the change of the function characteristic at different locations due to the penalty method (there are six constraints, given the both lower and upper bounds), which can lead to the situation that a part of the space being represented by a hyperparameter considerably in a different manner compared to the other parts. The effect of partial MVMO on EGO is adverse here, either when it is the only change (i.e., EGO+MVMO) or when it is accompanied by other proposed modifications. As Figures 4.11 and 4.12 show, MVMO explores excessively and hence it is among the worst, as one may expect for such a low number of evaluations.





**Figure 4.8** All adaptive samples of all 50 independent optimizations are plotted vs. each dimension for the Himmelblau problem. The distribution of each design variable is shown in the lower part.



**Figure 4.9** Mean values of the adaptive phase of all 50 repetitions of different methods for the Himmelblau problem.

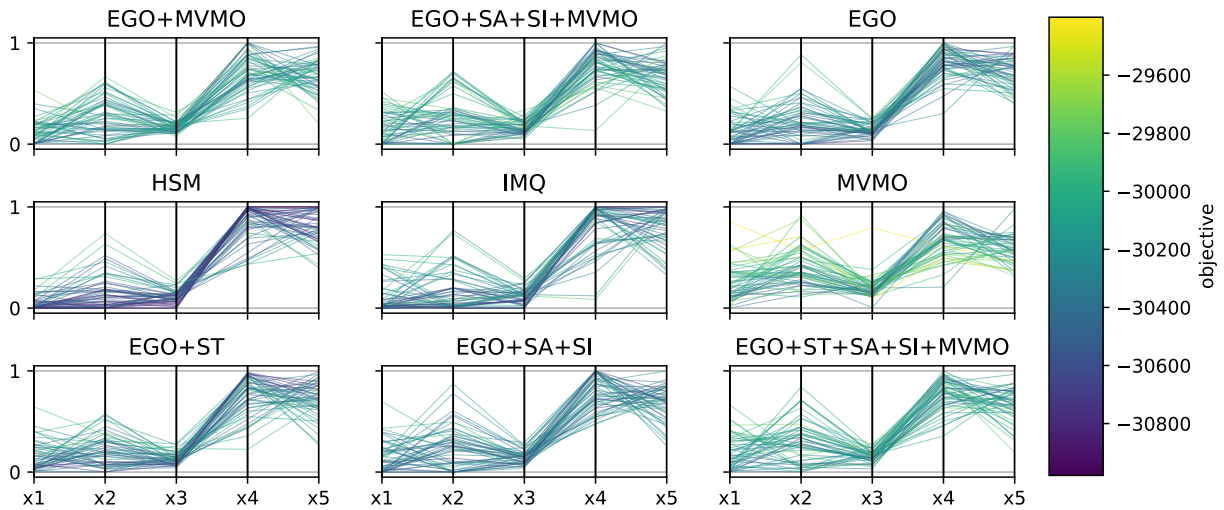


Figure 4.10 The parallel coordinate plot of all 50 repetitions of the proposed methods for the Himmelblau problem.

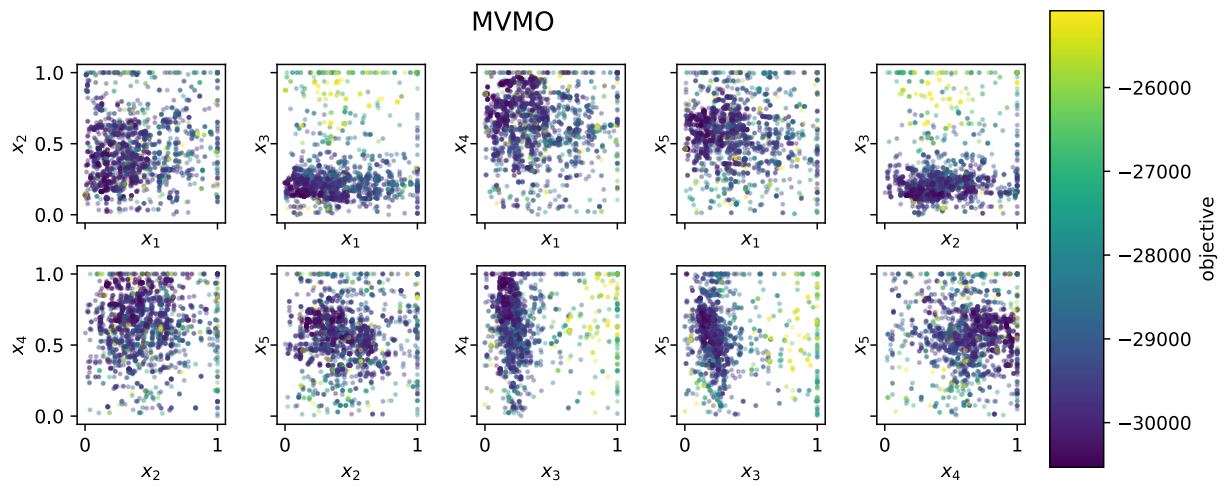


Figure 4.11 The two-dimensional projections of the design variables for MVMO.

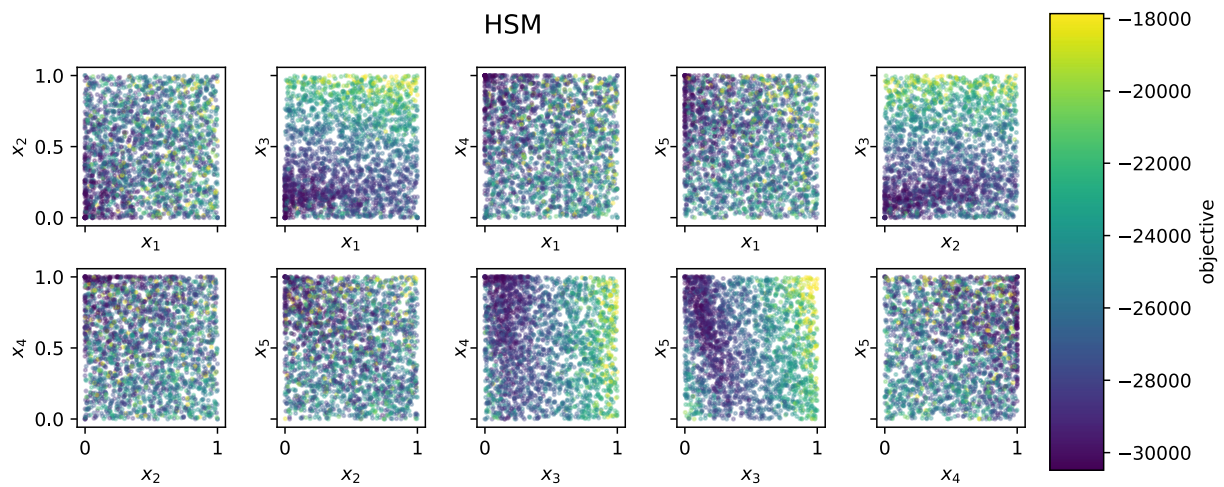
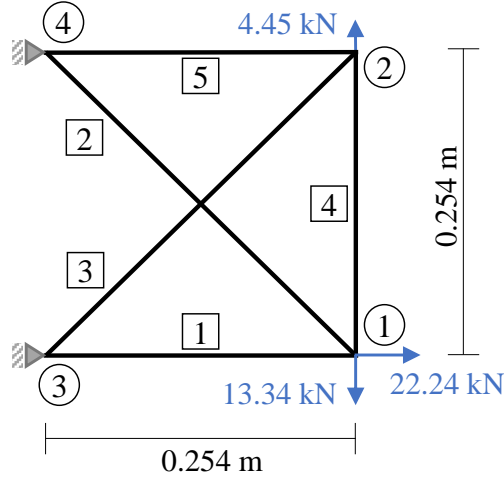


Figure 4.12 The two-dimensional projections of the design variables for HSM.

#### 4.3.4. Five-bar truss

In this section, we consider a model with a mechanical background that is also very fast to evaluate. Therefore, we decided to use a truss structure. The five-bar truss problem shown in Figure 4.13 is from Pospěšilová and Lepš (2012). The FEM Python code used to solve this problem is taken from Aranda and Bellido (2016) and modified by the author of this thesis.



**Figure 4.13** The considered linear elastic truss problem.

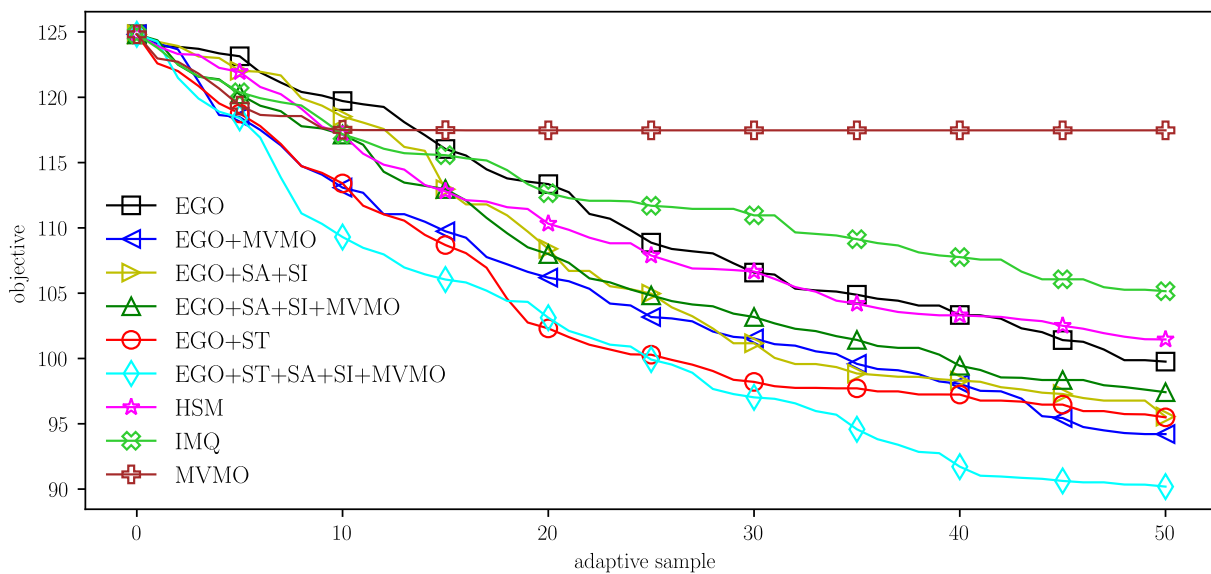
The problem is linear elastic and the objective is to minimize the entire mass, while there are constraints on the maximum displacement and stress. Design variables are the cross-sectional areas of each bar. The problem is defined as:

$$\begin{aligned}
 & \underset{\mathbf{A}}{\text{minimize}} && f(\mathbf{A}) = \rho \sum_{i=1}^5 A_i L_i \\
 & \text{subject to} && 6.4516 \leq A_i \leq 64.516 \text{ mm}^2, \quad i = 1, \dots, 5, \\
 & && \max |\sigma_i| \leq 0.414 \text{ GPa}, \\
 & && \max |w_i| \leq 1.524 \text{ mm}.
 \end{aligned} \tag{4.11}$$

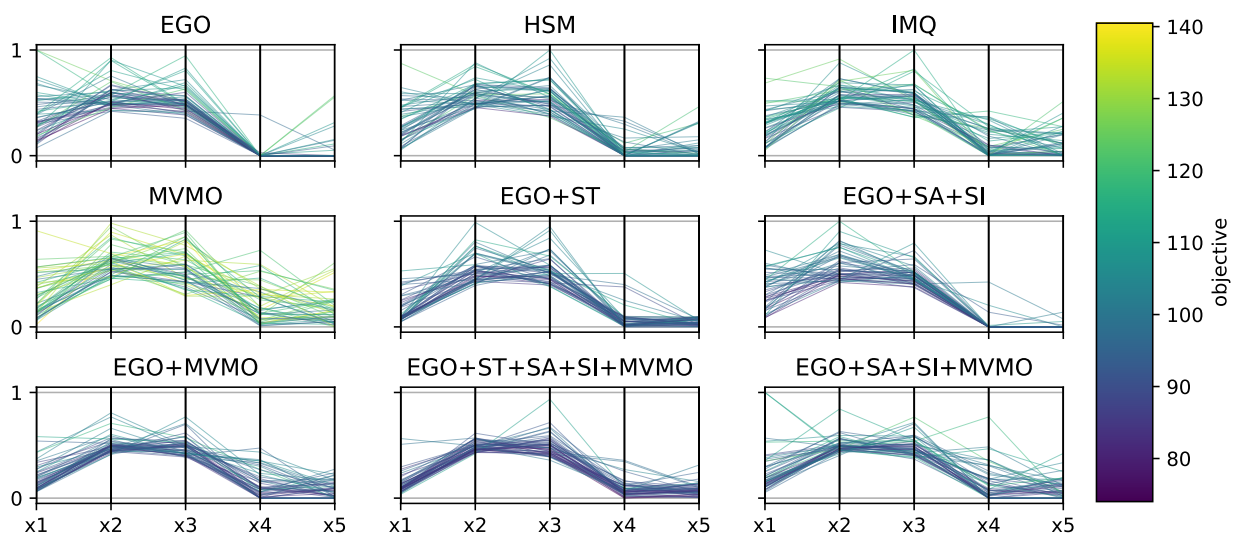
The considered material is aluminum with mass density  $\rho = 2.768 \times 10^{-6} \text{ kg/mm}^3$  and Young's modulus  $E = 70 \text{ GPa}$ . The unconstrained form by the penalty method is:

$$\begin{aligned}
 & \underset{\mathbf{A}}{\text{minimize}} && f_p(\mathbf{A}) = \rho \sum_{i=1}^5 A_i L_i + p_1 \max(c_1(\mathbf{A}), 0) + p_2 \max(c_2(\mathbf{A}), 0) \\
 & \text{subject to} && 6.4516 \leq A_i \leq 64.516 \text{ mm}^2, \quad i = 1, \dots, 5, \\
 & && c_1(\mathbf{A}) = \frac{|\sigma_i|}{0.414} - 1, \\
 & && c_2(\mathbf{A}) = \frac{|w_i|}{1.524} - 1, \\
 & && p_1 = p_2 = 4.536.
 \end{aligned} \tag{4.12}$$

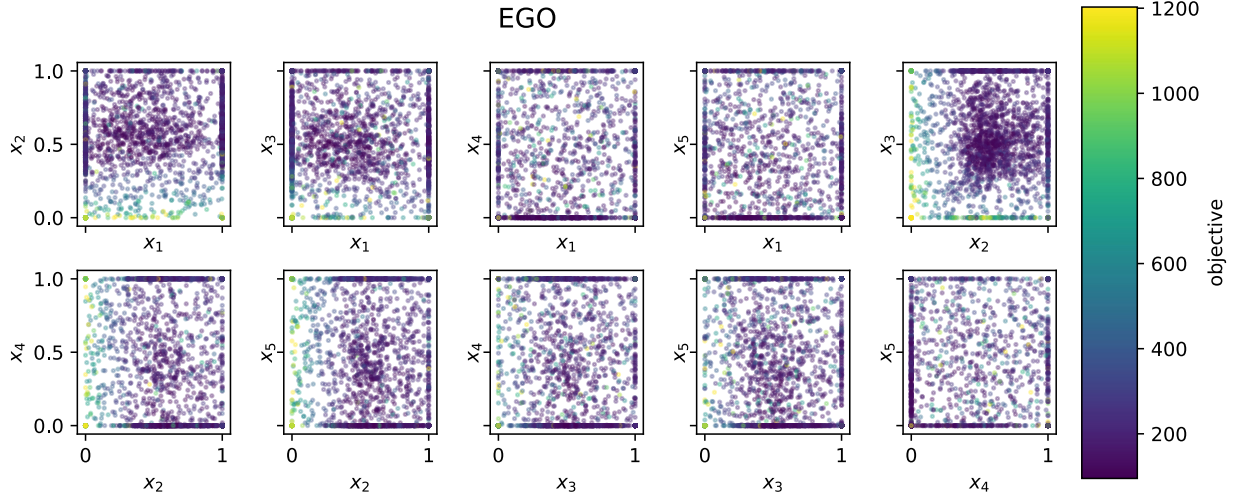
Figure 4.14 shows that the hybrid method and EGO perform similarly, while IMQ performs worse, a possible suggestion that a more local kernel is better. The influence of ST on EGO is also positive and lower output values occur for some design variables close to the boundary, as Figures 4.16 and 4.17 show. Figure 4.17 also shows that a significant number of samples during the adaptive phase ends close to the boundaries and ST redistributes them. At the same time, lower function values are happening around these boundaries, which can be helpful for EGO + MVMO, because, EGO can concentrate on more important design variables and less important ones are already in good regions. Here, MVMO has a good start, but then it is not able to find better samples and keeps searching around (for this low number of evaluations).



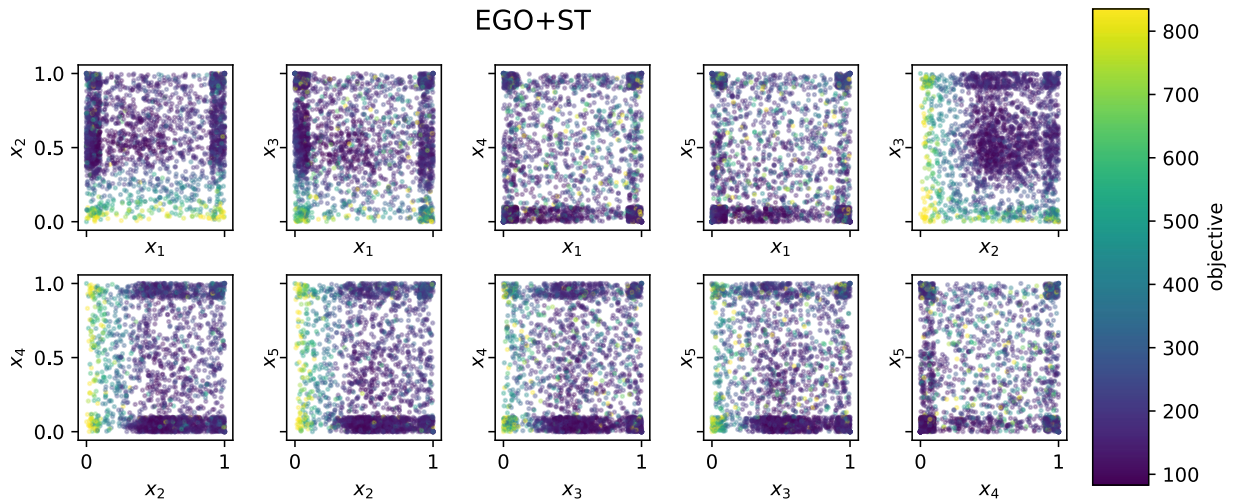
**Figure 4.14** Mean values of the adaptive phase of all 50 repetitions of different methods for the truss problem.



**Figure 4.15** The parallel coordinate plot of all 50 repetitions of the proposed methods for the truss problem. The darker the color the more desirable the value (we are conducting minimization).



**Figure 4.16** The two-dimensional projections of the design variables for the original EGO.



**Figure 4.17** The two-dimensional projection of the design variables for EGO+ST.

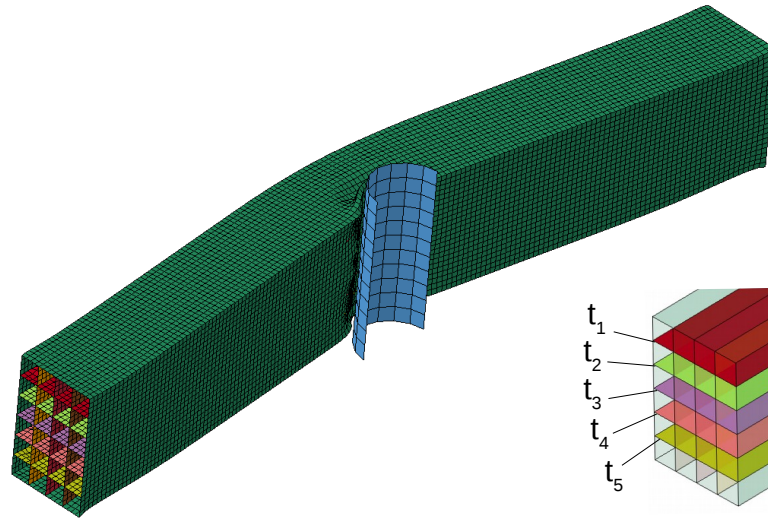
#### 4.3.5. Side sill impact

In this example, we consider a crash model representing the side sill of a car under a side pole impact. The intention is to absorb the energy and avoid intrusions greater than 50 mm. Both ends of the model are fixed and the design variables are the thicknesses of the horizontal ribs (Figure 4.18). The optimization is defined again by the penalty method via:

$$\begin{aligned}
 & \underset{\mathbf{t}}{\text{minimize}} && f_p(\mathbf{t}) = \text{mass} + p * \max(c(\mathbf{t}), 0), \quad \text{where } p = 3.75 \\
 & \text{subject to} && 0.5 \text{ mm} \leq t_i \leq 3.0 \text{ mm}, \quad i = 1, \dots, 5, \\
 & && c(\mathbf{t}) = \frac{u_{\max}}{u_{\text{allowable}}} - 1 \leq 0, \quad u_{\text{allowable}} = 50 \text{ mm}.
 \end{aligned} \tag{4.13}$$

LS-DYNA (R9.2) is used to solve the problem. Figure 4.21 shows that good results are distributed around the lower and upper bounds of some design variables; especially for  $x_3$  the desirable re-

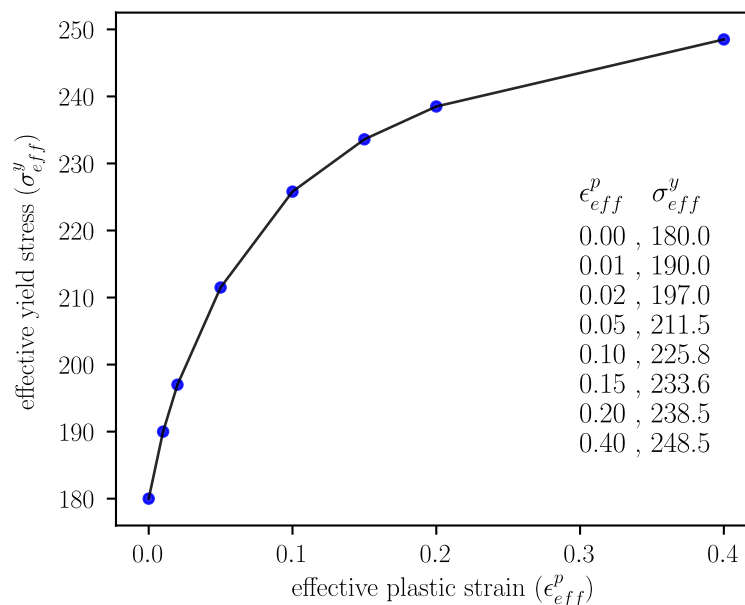




**Figure 4.18** This structure represents a side sill impacted by a cylindrical rigid body as a pole. Both ends are fixed and the design variables are the thicknesses of the horizontal ribs shown here.

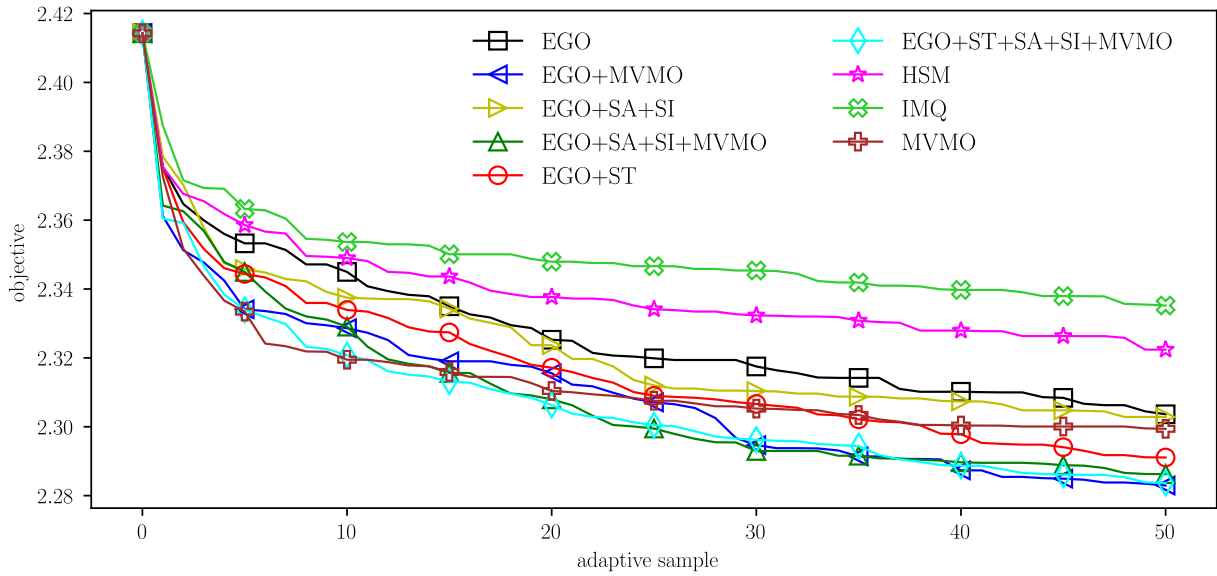
**Table 4.4** Settings of the side sill problem. The material model is \*MAT\_24 from LS-DYNA. Other required data for the plasticity part is provided in Figure 4.19.

impactor velocity	36 km/h	Young's modulus $E$	70 GPa
impactor mass	86 kg	initial yield stress $\sigma_y$	180 MPa
Poisson ratio $\nu$	0.33	density $\rho$	2700 kg/m <sup>3</sup>

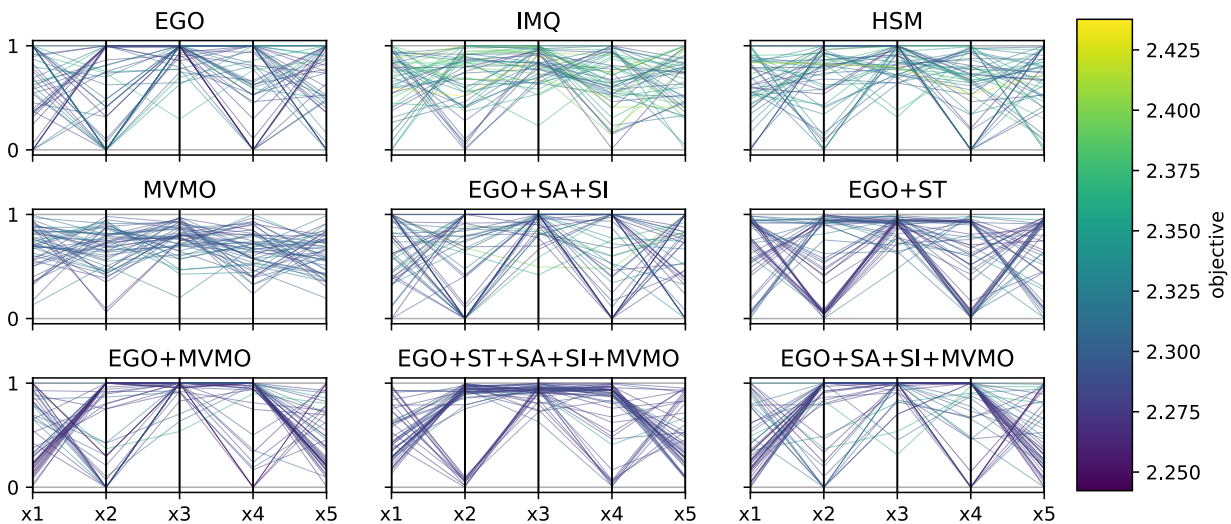


**Figure 4.19** Piecewise linear plasticity model of aluminum. Values at each point are given in the table on right.

gion is around the upper bound. Results in Figure 4.20 show that this time IMQ is the worst, while standard EGO has average performance and the hybrid of them lies in between. This is a

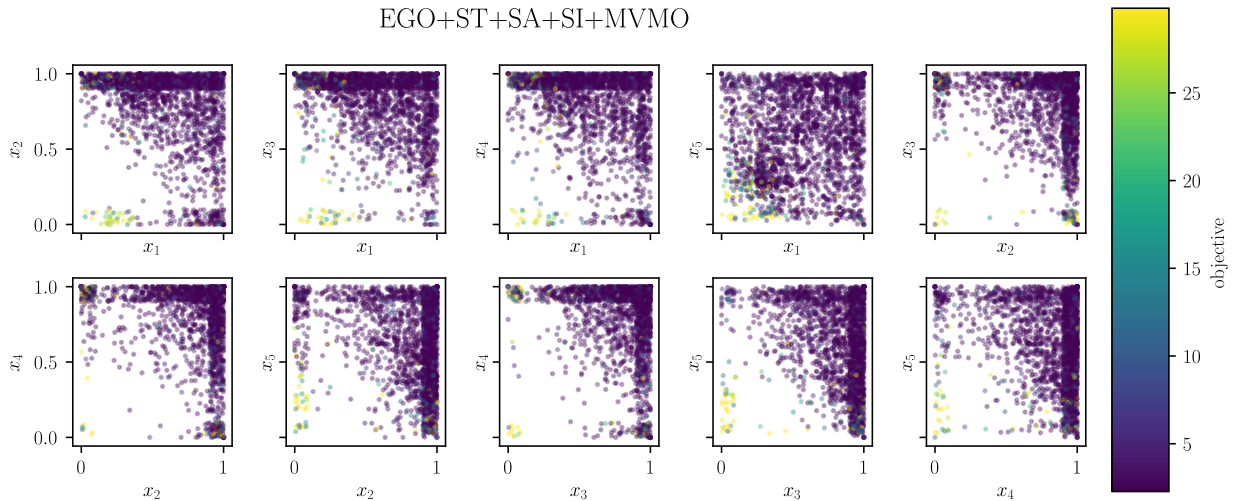


**Figure 4.20** Mean values of the adaptive phase of all 50 repetitions of different methods for the side sill problem.

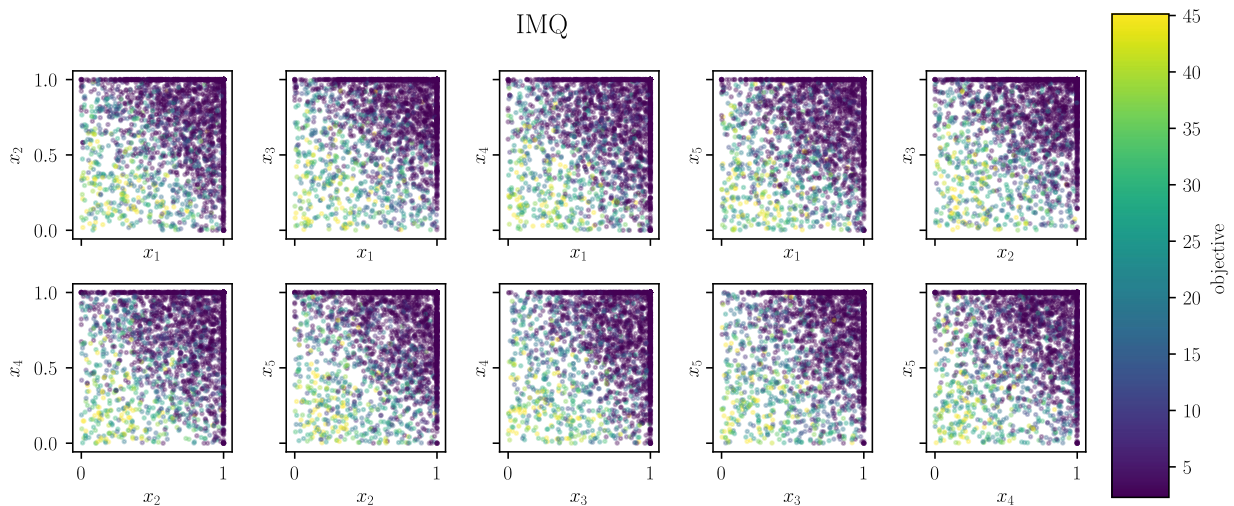


**Figure 4.21** The parallel coordinate plot of all 50 repetitions of the proposed methods for the side sill impact problem. The darker the color the more desirable the value (we are conducting minimization).

potential sign that a more localized kernel is helpful in this problem, as Figure 4.21 also indicates. Figures 4.22 and 4.23 show the worst and one of the best methods in this problem correspondingly. EGO+MVMO is among the best because it has a good local value that the method can exploit. ST also improves EGO; here, the desirable values are close to the boundaries. This case has similarities to the truss problem. Those methods that are better than EGO are also better here. The considerable difference is that MVMO is no longer the worst performer. It has a faster improvement compared to EGO and keeps that even up to 40 adaptive evaluations; after this, the difference starts to get smaller. Besides the fact that a good basin has already been found, this problem is lower-dimensional than it seems due to close to symmetric behavior and hence MVMO also becomes a decent contestant.



**Figure 4.22** The two-dimensional projections of the design variables in the side sill problem for IMQ.



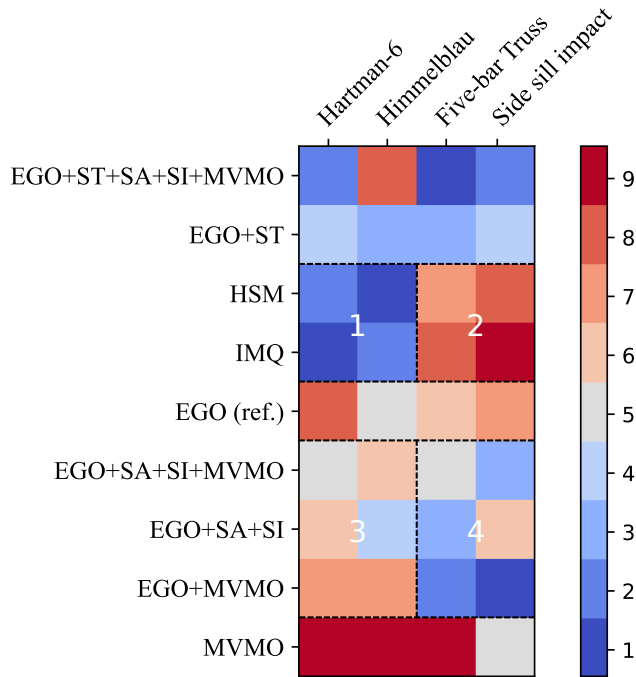
**Figure 4.23** The two-dimensional projections of the design variables in the side sill problem.

#### 4.3.6. Discussion

In this chapter, multiple proposed methods for improving EGO were described and compared with the common EGO algorithm. Figure 4.24 summarizes the results and shows the (color-coded) ranks for each of the four considered problems. The black dashed lines separate the proposed methods from the established ones (EGO and MVMO) and classify them into four groups numbered in white. This figure shows the rankings flip when comparing groups 1 with 2 and 3 with 4, which indicates that the two problem sets (Hartman-6 and Himmelblau vs. the last two) have quite non-overlapping characteristics. In addition, the developed methods HSM and IMQ also show rather complementary behavior to those methods in groups 3 and 4, since HSM and IMQ are better in the first two problems (Hartman-6 and Himmelblau) and those methods in the last two. Previously, the literature review showed the importance of adding knowledge about the problem to the optimization algorithm and the presence of complementary strategies. Knowledge can also be interpreted as knowing the properties of an algorithm and trying to compensate for its short-



comings. In this regard, we considered the IMQ kernel as a replacement for the Gaussian kernel and in addition, a hybrid model of both kernels was also developed (HSM). One of the ideas for improvement was to avoid unnecessary visits to regions close to the boundaries. To realize this, we proposed the ST method (EGO + ST), which interestingly has a better rank than EGO in all four problems. In ST, we redistribute samples around the boundary (mostly making them more distant) if the prediction from IMQ is higher than that of the Gaussian kernel. Here, we used the fact that IMQ carries information over a wider range, while the Gaussian kernel dampens it faster. Therefore, a higher IMQ is more often associated with higher output values than lower ones and since we are minimizing, samples with these higher values are not important for improving the results themselves. However, their location is still important for a proper representation of the underlying function, which is the goal of ST and is achieved by spreading samples more widely than just concentrating on the mentioned boundaries. Contrary to other methods, EGO+ST is interestingly better than EGO in all four problems, perhaps due to the nature of its modification, which neither uses a specific kernel (or amplifies its characteristics) nor limits the hyperparameter range, but instead tries to improve the fit without directly changing good solutions. However, we cannot generalize this good performance to higher dimensions (although samples at the boundaries could be beneficial), at least since the reliability of predictions is reduced due to the sparsity of samples, while ST depends on rather good relative predictions by the IMQ and the Gaussian kernel. Besides, here we have a limited number of problems and results can change for new ones, yet the simple modification in ST shows its effectiveness and potential. EGO+ST+SA+SI+MVMO is the only other tested model that includes the ST modification. This method can get better results than EGO+ST as it has other means to better handle some cases, but it also performs worse in the case of Himmelblau. The EGO with IMQ performed either better or worse than the original, as expected. However, the performance of the hybrid method was mostly between one of the two methods and only once it was the best. This may indicate that neither kernel can adequately represent various problems. Note that, hybrid or ensemble methods are more common in machine learning and using them makes sense in that field, while this is not necessarily true for our application area as explained next. The ensemble methods in machine learning deal mostly with noisy cases that for each set of inputs, more than one possible response exists (i.e., (highly) noisy, no interpolation possible), e.g., the price of homes based on the neighborhood, number of rooms and so on. In these cases, several methods are combined to explain the variability in the output. Based on trial and error, a combination of methods is found that works better than others and this ensemble will then be used for this specific case (we neglect the update of the model here). Nevertheless, our approach can also be seen under the umbrella of supervised learning. However, in our engineering applications, the problem may not be that noisy and usually, an interpolative method (with regularization) can be a decent representative of the underlying function. So, there is less variability than for the standard machine learning cases and secondly, we are usually interested in using the model also for other cases successfully, not just for the cases that are tuned for. But in practice, as one saw here and also as a reasonable conclusion of the NFL theory, for many cases,



**Figure 4.24** Colors in each column represent the ranking of the nine compared methods for each of the four considered problems. In the middle, "ref." means the (common) EGO is the reference method for comparison.

one of the methods used in the hybrid is better than the other ones, while the hybrid model ends up in between. In other words, we are mostly not being the worst performer among the considered methods in hybridization, but our intention was to be (among) the best. This issue cannot be solved unless the demands of the problem are recognized and appropriate actions are taken accordingly. In other words, one needs to include steps to realize the type of the underlying function in the first place, for example, by using some samples to test which of our initial complementary methods holds better over several iterations. After that, we may focus more on the promising method. This could be a better approach than simply selecting some methods for hybridization and then hoping for success in unknown scenarios based on some tuned parameters.

Another proposed modification was changing the ranges for the hyperparameters based on the contribution of the corresponding DVs to the output (considered for the minimum and maximum influential DVs). Results showed, in cases of Himmelblau and side sill impact, no improvement was obtained compared to the original EGO. Since perhaps the locality is important here, and moreover, corresponding objective functions include penalty terms, which potentially can intensify such locality. However, in the two other considered examples, changing the range of the hyperparameters improved the results of EGO. Another suggestion was separating the less influential design variables from the EI optimization and setting them by the MVMO mutation procedure. For three out of four cases, there were improvements and notably in combination with other modifications we obtained the best performer for both of the truss and side sill impact problems. Although being inspired by ideas from treating EGO (Bayesian optimization) in higher dimensions, test cases are only 5- and 6-dimensional here. This is mainly due to the fact that developing new methods and realizing results given a sufficiently large set of investigations is quite time-consuming. Even

current settings limited our ideas to be implemented and all possible combinations to be tested. The results indicate the proposed methods have a good potential to make EGO more efficient in higher dimensions, especially for the cases that were raised in the motivation part and are revisited here next; in practical cases which involve several outputs like turbine or car body optimization, many design variables are kept even after the sensitivity analysis, while for each output some variables are not contributing. That is why in our proposed methods we tried to improve EGO not by eliminating dimensions but with a treatment, based on the influences of DVs and/or characteristics and behavior of EGO itself. Hence, all proposed modifications are based on utilizing information from the problem at hand at a general level and avoiding fine-tuning as reasonably as possible.

## Chapter 5

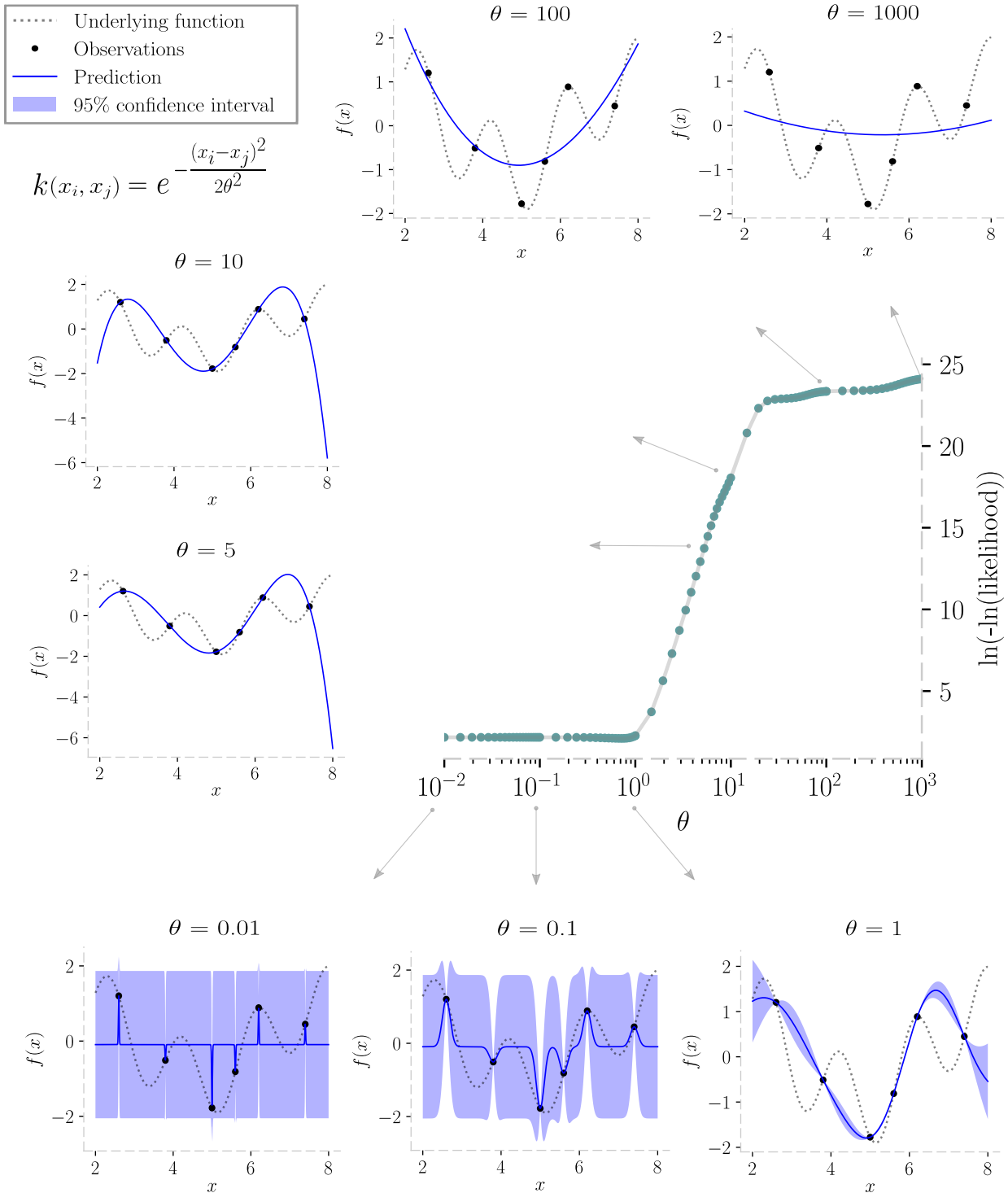
### Enhancements of EGO - Part II: GPR fitting, EI

We begin this chapter by observing a pitfall of a GPR model, i.e., overfitting and then offer a procedure to improve the performance, mainly involving the modification of the bounds of the GPR hyperparameter(s), most importantly the lower bound  $\theta_{lb}$ . This first part ends by comparing the results on some mathematical test functions and the corresponding discussion later. In the next part, we propose a new formula that tries to make better use of the EI potential. We check the results on some mathematical test functions and then on mechanical design problems with constraints. The final example is as always a crashworthiness problem. Here, the shape of a crash box is optimized considering the two proposed modifications in this chapter, followed by the corresponding discussion at the end.

## 5.1. GPR overfitting

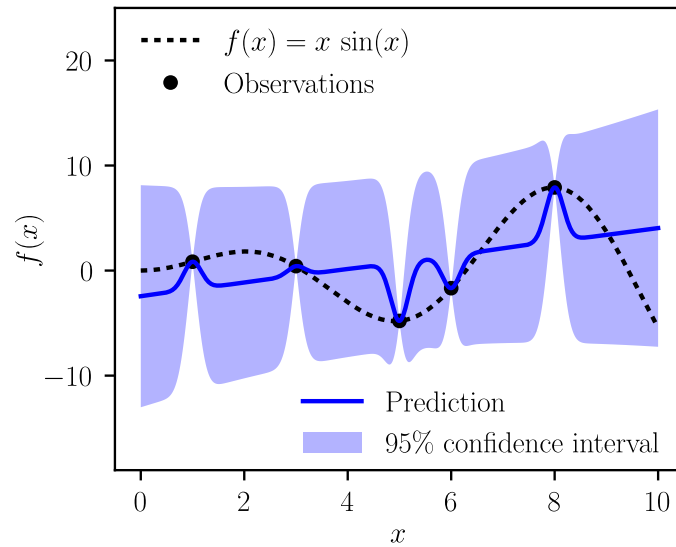
EGO is based on GPR and therefore is affected by its quality. A GPR surface is fitted to the already evaluated samples and used for prediction at new locations. A proper fitting tries to balance representing the current data accurately and yet preventing overfitting, or in other words, being able to generalize well and having a reasonable prediction for unseen cases. This was previously discussed in Section 2.2 under balancing the bias and variance dilemma. The likelihood function tries to achieve this balance by encouraging accuracy and at the same time penalizing for more complex models (e.g., having more parameters). However, this does not mean that the optimum of this function always leads to actually a good balance and even if it does, the optimizer may not be able to find it. Next, we depict likelihood behavior and discuss some relevant pitfalls that can negatively affect GPR and consequently EGO; then, we propose a method to counteract overfitting accordingly in case of using isotropic kernels.

Figure 5.1 shows how the likelihood function changes for different values of the hyperparameter  $\theta$  for a 1D function evaluated at six samples. The blue line shows the fitted curve (i.e., the mean of GPR) and the dashed black line shows the underlying function. For low values of  $\theta$  (0.01 and 0.1), there is a clear overfitting, i.e., the fitted curve goes through the available points but then suddenly follows a (predefined) trend independent of the problem. Consequently, the prediction error at new locations can be high, even if these points are close to the current samples. The trend part of the GPR is constant here. If it was linear, then overfitting would result in a linear prediction not far from the samples. This can be seen in Figure 5.2. Even if the fitting is regular and proper (e.g., for  $\theta = 1$  in Figure 5.1), still the stochastic part vanishes away from the samples and only the trend remains. These examples also show how GPR and Kriging extrapolate and indicate what to expect in higher dimensions for unexplored regions. Now, let us return to the likelihood in Figure 5.1. By increasing the hyperparameter  $\theta$ , the fitted curve becomes smoother and the variance decreases as the curvature changes less. Gradually, the behavior goes from interpolation to regression, as shown in the figure. Figure 5.3 shows the likelihood of a 2D example. Here, the anisotropic squared exponential is chosen for the covariance model, so there is one  $\theta$  for each dimension. In addition to the overfitting that was discussed above and can be seen in the plot number 4 in this figure, there is another overfitting; when one of the  $\theta$ s is small and the other one is large. This is shown in the plot number 3. Since GPR has a constant trend, the prediction is flat at locations without samples and along the direction having this large  $\theta$  (relative to the problem). In other words, GPR assumes that this dimension does not affect the output. On the other hand, the dimension with a small  $\theta$  is overfitted, like what we saw above for the 1D case. Such a fitting also affects EI and makes it vary similarly as shown in the nearby grayscale figure. These mentioned plots are an example of why a more complex model is not always good. There is more flexibility in the model and hence more chance that the fitted surface has an inappropriate shape. More samples can improve the situation; since there is more information to favor some  $\theta$ s with more certainty over others. Here, similar to the 1D case, by increasing the values of  $\theta$ , the fitted surface becomes more regressive. It



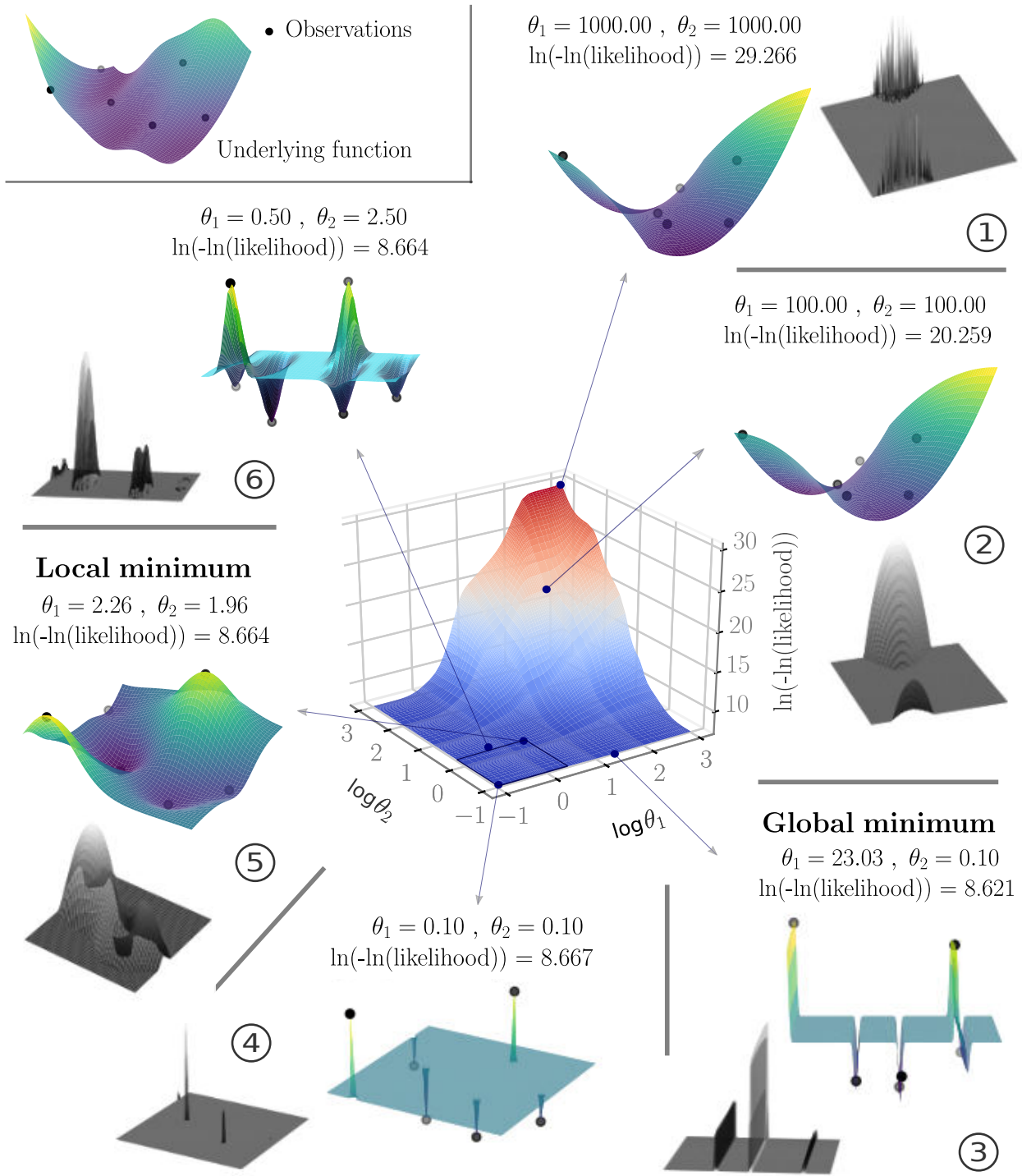
**Figure 5.1**  $\ln(-\ln(\text{Likelihood}))$  is plotted for current data set including 6 samples. The fitted curve for different values of  $\theta$  is plotted in each case (solid blue lines) along with the predicted variance (shaded area) and the actual underlying function (black dotted lines). The variance gets smaller by increasing  $\theta$  and hence it may not be observable from the figures.

should be noted that by increasing the  $\theta$ , the covariance matrix eventually becomes non-invertible as columns and rows get closer to each other. Adding regularization can improve the situation; however, an excessive amount of it leads to a regressive fitted surface and very small prediction



**Figure 5.2** overfitting of a Kriging with a linear trend.

variance, making EGO ineffective in practice. Therefore, the regularization should remain small. But then, a high condition number is not remedied. The spike-like EI shown in plot 1, is due to this high condition number, which is especially reflected in the variance of prediction. Increasing the resolution of the plot here just leads to more spikes. The considered regularization is  $1e - 10$ . Another interesting aspect is the global optimum of the likelihood. Please note that we want to find the maximum of the likelihood, but since we consistently do minimization in this thesis, the sign of the likelihood is changed to negative, as mentioned before. To have a visually better plot, an extra natural logarithm ( $\ln$ ) is also taken here. Interestingly, the global optimum, which is shown in plot 3, is the overfitted case discussed above. Plot 5 corresponds to a local optimum (at least) for the region inscribed by the rectangle shown on the likelihood plot. This local minimum has a more reasonable fitting than the global one. Therefore, even when the global optimum of the likelihood is found, overfitting can still happen, simply due to the fact that the quality of the likelihood itself is not high enough. There are more similar overfitting examples that are not shown here, even for a larger number of samples. For sure, if one increases the number of samples, the possibility of having a better fit increases, but this does not guarantee that the optimum of the likelihood function would not lead to overfitting and in case it does not, the optimizer is able to find it. Remember that we are dealing with a small budget and these arguments can still be valid for a significantly larger number of samples. We want to avoid these overfitted cases as much as possible. Here, we only focus on the isotropic kernel, where overfitting similar to the one shown in plot 4 of Figure 5.3 occurs. As mentioned above and shown earlier in Figure 5.3, the anisotropic kernel can lead to additional types of overfitting. However, we do not consider anisotropic kernels in this section, since one dimension can contribute much less than the others and hence how one should treat these cases requires more study. In addition, since the hyperparameter space is larger in case of anisotropic kernels, overfitting potentially can happen more often and hence we prefer to start with the isotropic kernel, which is much better posed against these issues. In this first part of this



**Figure 5.3**  $\ln(-\ln(\text{Likelihood}))$  is depicted for the six available samples in 2D, originating from the BarninHoo test function shown in top left. Considered covariance model is anisotropic squared exponential with regularization of  $1e - 10$ . The fitted surfaces (i.e., mean of the GPR) for the selected  $\theta$ s are shown in green and yellow colormap while the corresponding Expected Improvement is shown in gray-scale beside each case.

chapter, if the kernel is not mentioned, it is an isotropic squared exponential.

Concerning finding a way to improve the overfitting problem, let us revisit one of the main messages of the NFLT; it is necessary to introduce information from the problem at hand into the



considered optimizer, to make it more successful than others over new cases. Although the problem we are dealing with can be a black box, we use a surrogate of it, namely GPR, which on the contrary is not a black box and has an analytical formula. This means that we have the possibility to access such information tailoring the optimizer. The analytical formula is repeated below.

$$\text{Mean}(\mathbf{x}_*) = \boldsymbol{\mu}(\mathbf{x}_*) + K(\mathbf{x}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1}(\mathbf{y} - \boldsymbol{\mu}(\mathbf{X})), \quad (5.1)$$

$$\text{Variance}(\mathbf{x}_*) = K(\mathbf{x}_*, \mathbf{x}_*) - K(\mathbf{x}_*, \mathbf{X})[K(\mathbf{X}, \mathbf{X}) + \sigma_n^2 I]^{-1}K(\mathbf{X}, \mathbf{x}_*). \quad (5.2)$$

$$\text{The covariance model for GPR in this thesis : } \exp\left(-\frac{r^2}{2\theta^2}\right). \quad (5.3)$$

Obviously, the inverse of the covariance matrix ( $K^{-1}$ ) plays an important role in the overfitting issue. When such an overfitting occurs (e.g., plot 4 of Figure 5.3), information from the evaluated samples does not reach other locations and only the trend remains for prediction. The key here is the distance between samples that defines how fast the covariance and hence, the transformation of information diminishes for a certain  $\theta$ . We try to identify the sample that is the worst in terms of receiving information from others. Before going further, we neglect regularization and consider a covariance model with a unit multiplier (i.e., only the exponential term as in Equation (5.3)). Regularization will be considered later in the final modification and the multiplier will not affect the proposed method. When for a certain location in the design space, all non-diagonal elements in a row of  $K^{-1}$  are zero, it implies that this location does not receive any information from the rest of the already evaluated points. However, this is a very severe case and overfitting can still occur when the off-diagonal terms are not zero, but very small ( $\ll 1$ ). As they get larger, the situation gets better, so a threshold-like criterion might help, which requires identifying the most isolated off-diagonal terms. Alternatively, we can reduce overfitting by setting a limit on the worst diagonal term in  $K^{-1}$ . We will describe how this worst is obtained. But first, we show that each diagonal element in  $K^{-1}$  is equal to or greater than its counterparts in  $K$ . The proof is as follows. The covariance matrix is symmetric positive definite, so its singular value decomposition will be (Bogacki, 2019),

$$K = Q^T D Q, \quad (5.4)$$

where,  $Q$  is a matrix of orthonormal eigenvectors and  $D$  a diagonal matrix containing the eigenvalues. Considering another symmetric positive definite  $B$  exists such that (Bogacki, 2019),

$$B^2 = B^T B = K. \quad (5.5)$$

$B$  has the same eigenvectors and its eigenvalues are the square root of those of  $K$ . If  $Q_i$  represents the  $i^{\text{th}}$  column of  $Q$ , the diagonal terms of  $K$  and its inverse can be written as,

$$\begin{aligned} K_{ii} &= Q_i^T K Q_i = Q_i^T B^T B Q_i = (B Q_i)^T B Q_i = \|B Q_i\|^2, \\ K_{ii}^{-1} &= Q_i^T K^{-1} Q_i = Q_i^T (B^T B)^{-1} Q_i = (B^{-T} Q_i)^T B^{-T} Q_i = \|B^{-T} Q_i\|^2, \end{aligned} \quad (5.6)$$

using the square root of the Cauchy-Schwarz inequality, while  $\langle \cdot, \cdot \rangle$  is the inner product and  $I$  the identity matrix,

$$\|BQ_i\| \|B^{-T}Q_i\| \geq \langle BQ_i, B^{-T}Q_i \rangle = Q_i^T B^T B^{-T} Q_i = Q_i^T I^T Q_i = Q_i^T Q_i = 1. \quad (5.7)$$

Combining equations (5.6) and (5.7),

$$K_{ii}^{-1} \geq \frac{1}{K_{ii}}, \quad (5.8)$$

Since the diagonal elements of  $K$  are one, inequality (5.8) means that all diagonal elements of  $K^{-1}$  should be at least 1. The inequality above does not explain how each diagonal term in  $K^{-1}$  can be calculated from the elements in  $K$ . To find that, we rewrite the covariance matrix  $K$  in block form based on Zhang (2005). Suppose that  $K$  is formed from evaluations at  $n$  locations  $X_{1,\dots,n}$ . We separate one ( $X_n$ ) from the rest ( $X_{1,\dots,n-1}$ ) and rewrite  $K$  in block form accordingly,

$$K = \begin{bmatrix} A & B \\ B^T & C \end{bmatrix}, \quad (5.9)$$

where,  $C_{1*1} = 1$  is the covariance of ( $X_n$ ) with itself, and hence,  $B$  is the covariance of  $X_n$  with the rest and  $A$  is the covariance matrix neglecting  $X_n$ . The inverse of  $K$  would be,

$$K^{-1} = \begin{bmatrix} A^{-1} + A^{-1}B(C - B^T A^{-1}B)^{-1}B^T A^{-1} & -A^{-1}B(C - B^T A^{-1}B)^{-1} \\ -(C - B^T A^{-1}B)^{-1}B^T A^{-1} & (C - B^T A^{-1}B)^{-1} \end{bmatrix}, \quad (5.10)$$

with  $C = 1$ ,

$$K_{nn}^{-1} = \frac{1}{C - B^T A^{-1}B} = \frac{1}{1 - B^T A^{-1}B}. \quad (5.11)$$

Now, if we look at the last equation,  $K_{nn}^{-1}$  is the counterpart of the diagonal term we separated from others. Its denominator shows the maximum variance, which is one here and is reduced by a second-order term. In other words, the maximum uncertainty is reduced by information from the rest of the samples (contained in  $A^{-1}$ ). The minimum value of  $K_{nn}^{-1}$  would be one if no information is reached and would be greater than one if there is any influence from other samples. Therefore, when we are looking for the most isolated sample, the lower bound of the corresponding diagonal term of  $K^{-1}$  will be one and the rest of the diagonal elements will be at least the same or larger. If we neglect  $B$  or its transpose in the second-order term  $B^T A^{-1}B$ , the remaining part (e.g.,  $A^{-1}B$ ) will be of the first order. As an interesting side note, this term also appears in the prediction of mean in GPR and in addition,  $A^{-1}B$  also gives the coefficients in a multiple regression that describes  $X_n$  based on the rest of the samples  $X_{1,\dots,n-1}$  observed so far. In other words, if we represent the multiple regression as,

$$\hat{X}_n = \hat{\beta}_i X_{1,\dots,n-1}, \quad i = 1, \dots, n-1, \quad (5.12)$$

then, the best least squares approximation for unknown coefficients  $\beta_s$  can be obtained from (Raveh, 1985),

$$\hat{\beta} = A^{-1}B. \quad (5.13)$$

Here, again,  $A^{-1}$  contains information from available solutions, and  $B$  defines how much of this information reaches the desired point  $X_n$ . This section paved the way for the suggestion to reduce the overfitting problem, which is explained next.

### 5.1.1. A new proposed procedure to set the hyperparameter bounds

Based on the earlier review in this chapter, we realized that overfitting is a function of the hyperparameter values  $\theta$  when it is low; on the other hand, at high values of  $\theta$ , GPR becomes more and more regressive. We try to specially limit the lower bound of  $\theta$  for an isotropic kernel. If this new limit on the lower bound is intended to reduce the number of overfitting occurrences, the new limit for the upper bound is set with the intention of reducing the size of space when optimizing the likelihood function. This makes it easier for the optimizer to find the appropriate values for  $\theta$ . Details of the proposed method to reduce overfitting are given in Table 5.1. Here, we represent a concise version of it. We suggest modifying the lower bound of  $\theta$  ( $\theta_{lb}$ ), only if the minimum diagonal element in the inverse covariance matrix ( $\min(K_{jj}^{-1})$ ) is less than a certain threshold (e.g.,  $1 + 1e - 8$ ). If this is the case, then  $\theta_{lb}$  is varied until ( $\min(K_{jj}^{-1})$ ) is within a range that is considered acceptable. The limits of this range are defined by two multipliers, where one is 10 times larger than the other. The user can vary the multipliers to change the degree of modification. However, here we have not defined the multipliers as a function of the regularization ( $\alpha$ ) applied to the covariance matrix, since we usually set  $\alpha$  to a certain value such as  $1e - 7$  or  $1e - 6$ . It should be noted that  $\alpha$  is added to the diagonal elements of  $K$  and thus directly affects ( $\min(K_{jj}^{-1})$ ). Based on some initial results (see Appendix A), we make one extra small (yet important) change that  $\min(K_{jj}^{-1})$  should always be within an acceptable range, not only when it becomes less than a threshold. Next, we set the upper bound based on considering a threshold for the normalized difference of the actual outputs and their corresponding approximations (by GPR's mean) at evaluated points.

**Table 5.1** The proposed procedure to define the upper and lower bounds for the hyperparameter  $\theta$ .

- 
- Defining an initial lower and upper bound of  $\theta$ :
    - 1 First, an initial range is set for  $\theta$  and then, the lower and upper bounds will be modified.
    - 2 if inputs are defined by
    - 3  $X = [x_{ij}]$ ,  $i = 1 \dots n$ : sample number and  $j = 1 \dots d$ : coordinate
    - 4  $\theta_{avg} = \left[ \frac{\prod_{j=1}^d (\max(x_{:j}) - \min(x_{:j}))}{n} \right]^{1/d}$ ,  $n$ : number of samples and  $d$ : dimension
    - 5 initial bounds will be,
    - 6  $\theta_{lb} = 0.1 * \theta_{avg}$  and  $\theta_{ub} = 10 * \theta_{avg}$
  - Defining the final value of the lower bound:

7 Find the lowest value on the diagonal of the inverse covariance matrix  $K^{-1}$ ,  
8  $K_{jj\_min}^{-1} = \min(K_{jj}^{-1})$   
9 consider the following settings,  
10  $\min\_Kinv = K_{jj\_min}^{-1}$   
11  $\min\_Kinv\_init = 1.0$   
12 upper multiplier = 1.01 and lower multiplier = 0.1 \* upper multiplier  
13 if  $K_{jj\_min}^{-1} \leq 1 + 1e - 8$ :  
14  $\min\_Kinv\_init = \min\_Kinv$   
15  $\min\_Kinv\_lb = \text{lower multiplier} * \min\_Kinv\_init$   
16  $\min\_Kinv\_ub = \text{upper multiplier} * \min\_Kinv\_init$   
17 Now we set an acceptable range:  
18 if  $\min\_Kinv\_lb \leq \min\_Kinv \leq \min\_Kinv\_ub$ :  
19 the current  $\theta_{lb}$  will be the final one. stop  
20 if  $\min\_Kinv < \min\_Kinv\_lb$ :  
21 increase  $\theta_{lb}$  by 20%  
22 recalculate  $\min\_Kinv = K_{jj\_min}^{-1}$   
23 if  $\min\_Kinv > \min\_Kinv\_ub$ :  
24 decrease  $\theta_{lb}$  by 10%  
25 recalculate  $\min\_Kinv = K_{jj\_min}^{-1}$   
26 while  $\min\_Kinv < \min\_Kinv\_lb$  or  $\min\_Kinv > \min\_Kinv\_ub$ :  
27 Repeat the last two "if" conditions (lines 20-25)

- Defining the final value of the upper bound

28 At already evaluated samples, there will be a difference between the actual output values  
29 and what GPR predicts.  $\theta_{ub}$  is set to keep this deviation below a threshold, but first we  
30 need to normalize data.  
31 if the GPR (predicted) standard deviation at  $y_i < 0.0001$ :  
32  $\varepsilon = 1e - 15$   
33 else  
34  $\varepsilon = 0$   
35  $y_{normalized} = \frac{y - \text{mean}(y)}{\text{standard deviation}(y) + \varepsilon}$   
36 calculate the average deviation as:  
37 
$$\text{deviation\_avg} = \frac{\sum_{j=1}^n |y_{normalized\_actual_j} - y_{normalized\_prediction_j}|}{n}$$
  
38 while  $\text{deviation\_avg} < 0.005$ :  
39 increase  $\theta_{ub}$  by 10%  
40 recalculate the  $\text{deviation\_avg}$   
41 the current  $\theta_{ub}$  will be the final one  
End.

---

Before describing the test setup, we make some comments about the direct use of  $K^{-1}$ . First, as mentioned before, we use our modified version of GPR from scikit-learn, which is an efficient code based on Rasmussen and Williams (2006). Cholesky factorization is used for two reasons. First, to check whether the covariance matrix  $K$  is invertible and second, to help with the efficient and more robust calculation of terms involving  $K^{-1}$ . However, here we calculate the  $K^{-1}$  directly (still by using Cholesky as described below), since we need it for our proposed method. One may question the efficiency of calculating only  $K^{-1}$  instead of obtaining the (total) result of  $K^{-1}$  multiplied by another vector, as we are dealing with these multiplications and not isolated  $K^{-1}$  terms in the predicted mean and variance of a GPR. The reason is, our approach here can be actually more efficient, in contrast to the common case. This is because we call  $K^{-1}$  many times and not just once during infill optimization. So now, by having  $K^{-1}$  directly, we simply multiply this matrix into vectors each time and less work is required than solving a linear equation (e.g.,  $K^{-1}k = \text{Unknown result of this multiplication}$ ) involving the Cholesky factorization. For a small number of samples and dimensions, the common method (i.e., calculating the entire multiplication) may be more efficient; however, the total time would be already so low that it does not matter in practice. Note that we still use Cholesky factorization to find  $K^{-1}$  directly. We are just adding extra calculations for the steps afterward. Cholesky factorization of  $K$  is  $K = LL^T$ , where  $L$  is a lower triangular matrix. Accordingly,

$$\begin{aligned} (K = LL^T)^{-1}, \\ K^{-1} = (L^T)^{-1}L^{-1} = (L^{-1})^T L^{-1}. \end{aligned} \tag{5.14}$$

Therefore,  $K^{-1}$  can be calculated from  $L^{-1}$ , which itself is obtained by solving  $LL^{-1} = I$ , with  $I$  being the identity matrix. We use the "solve\_triangular()" function from SciPy to solve for  $L^{-1}$ . For the cases where  $\theta_b$  is modified,  $\theta$  is already at such low values that  $K$  would be very well-conditioned. For high values of  $\theta$ , if  $K^{-1}$  does not exist, the Cholesky factorization will return an error in the code and consequently the negative of the likelihood function is set to a high value. This is commonly done to avoid numerical problems when optimizing the likelihood function.

## 5.2. Problems and results

### 5.2.1. Prerequisites and settings

The performance of the suggested method to reduce overfitting will be assessed on several test functions in dimensions [2, 4, 6, 10]. The required settings are given in Tables 5.2 and 5.3. These test functions are selected to have different characteristics and many of them are also used in Picheny et al. (2013), but for noisy cases. Here, only Hartman4 and Hartman6 are for (small) noisy cases and the rest are taken in their common version without the extra noise. Rosenbrock, Alpine2 and Sphere are functions that are tested in all considered dimensions. A short description of these functions and their properties can be found in Surjanovic and Bingham (2013). Samples for the first phase of EGO are generated via Latin Hypercube with 30 repetitions. The reported

results are the average of these repetitions, unless otherwise shown or mentioned.

**Table 5.2** Common settings used for all problems.

Parameter	Value / Type	Explanation
covariance model	$\exp(-\frac{r^2}{2\theta^2})$	isotropic squared exponential
$\theta$ default bounds	[0.01, 100]	given by the user initially
$\alpha$	$1e-6$	regularization for the covariance matrix
weight in WEI	0.4	WEI: weighted expected improvement
general optimizer	DE from SciPy	where the optimizer is not mentioned

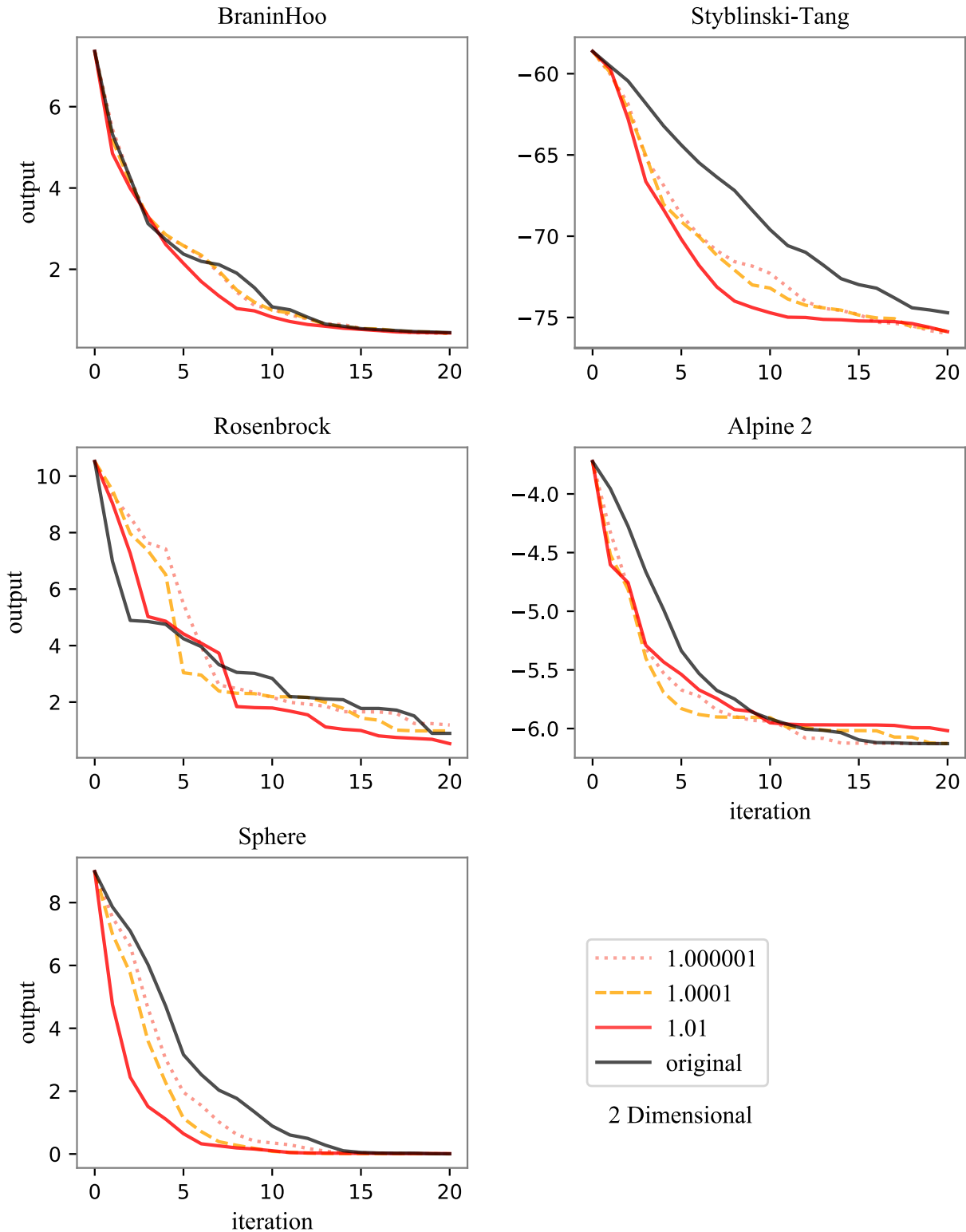
**Table 5.3** Employed Test functions.  $d$  means dimension, Rosenb. means Rosenbrock, "g07 objective" is the objective in the constrained test function g07 from Runarsson and Yao (2000). See Jamil and Yang (2013) for more information.

$d$	Test functions	Total budget	First phase budget
2	Branin, Styblinski-Tang, Rosenb., Alpine2, Sphere	30	10
4	Hartman4, Colville_4d, Rosenb., Alpine2, Sphere	72	24
6	Biggs Exp5, Hartman 6, Rosenb., Alpine2, Sphere	90	36
10	Paviani, g07 objective, Rosenb., Alpine2, Sphere	210	70

### 5.2.2. Results

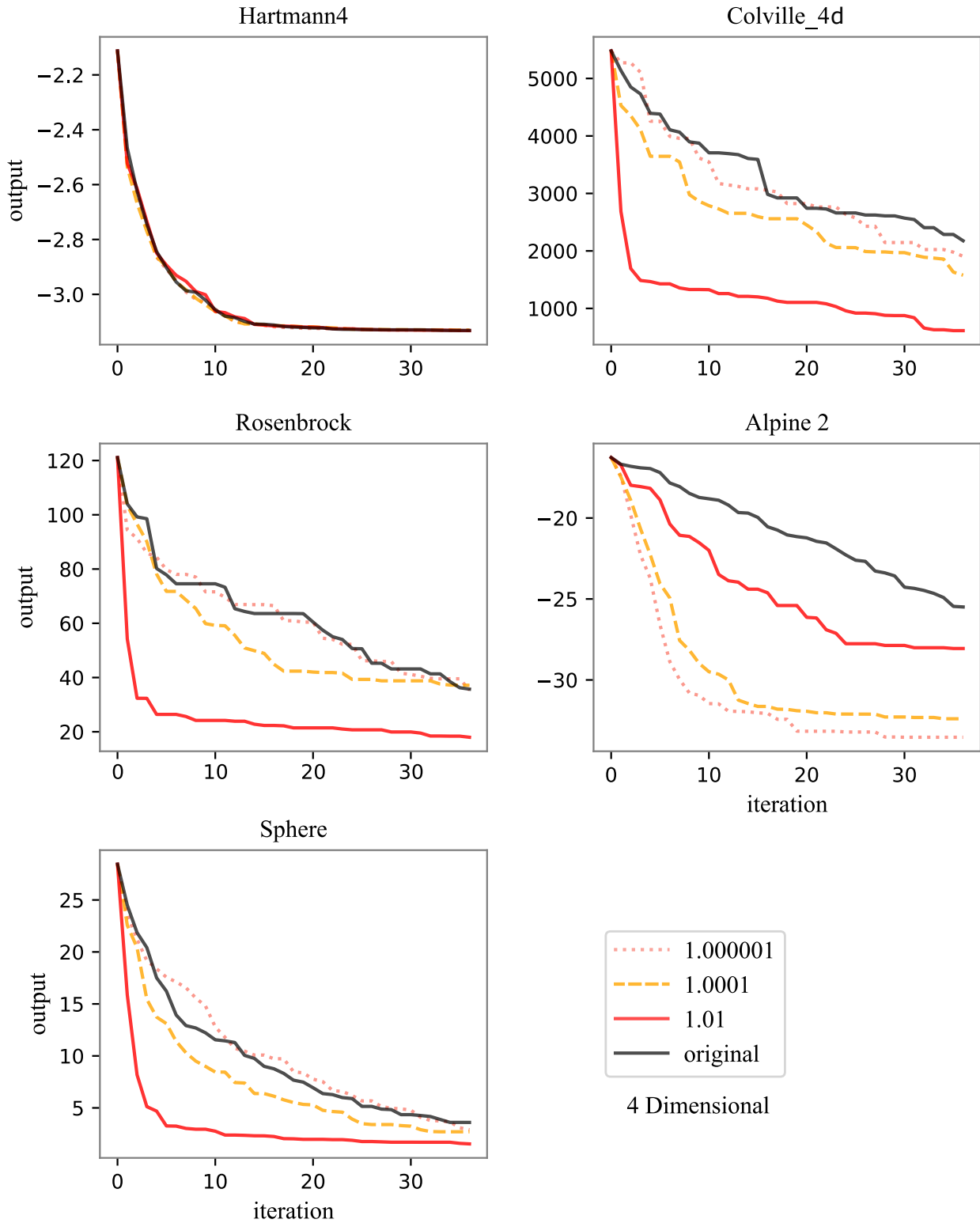
Three cases, i.e., three sets of the acceptable range for the minimum diagonal element of  $K^{-1}$  are considered. The  $\theta_{lb}$  is correspondingly changed compared to its predefined value in the common EGO. These considered ranges are [1.000001, 1.00001], [1.0001, 1.001] and [1.01, 1.1]. Any value in these ranges can be considered as an info-multiplier to '1', which is the case that the most isolated point becomes independent of other observed samples (see Equations (5.8 - 5.11)). Hence, the larger the info-multiplier the larger the threshold for accepting overfit, i.e., more information reaches from other samples to the most isolated one among them. Before discussing the results, note that the initial suggestion to modify the hyperparameter bounds was slightly different and its results (in Appendix A) were either much better or worse than EGO. Nevertheless, we wanted to have a more generally successful method, i.e., the new modified results should improve decently where they were initially considerably worse. However, this comes at the cost of becoming slightly worse where they were much better (but they are still better nonetheless). To achieve this

"more generally successful" method, a single, yet decisive change is applied. Here, the cases that were left unchanged previously (since they were above the threshold  $1+1e-8$ ) must now also be confined within a certain range. The final procedure is given in Table 5.1 and the results of the three test cases for setting the  $\theta$  bounds are given in Figures 5.4 - 5.7 for dimensions [2, 4, 6, 10], respectively. The results show how the proposed modification enhances performance for various functions, especially for the case with the largest acceptable range among the three ([1.01, 1.1]) and then for the second largest one. The values in these ranges are minuscule compared to the values that diagonal elements of  $K^{-1}$  can have and yet, they are so influential. These low values define how EGO can change locally and they become even more important for functions such as Alpine2 and especially Schwefel (10D). Here, we obtain an improvement in performance over Alpine2 and close the performance gap in Schwefel (10D) compared to the initial modification; fulfilling our goal of having a more generally successful algorithm. In the discussion section below, we go into more details of the locality. In general, the performance of the proposed modified cases is quite good and in some cases beyond the author's expectations, especially for this low number of function evaluations. What is even more interesting is the fact that such steep changes have been obtained not by dimension reduction methods, but just by modifying the hyperparameter bounds and most importantly, the lower one. This hints at the true potential of GPR and by extension, EGO that can be better utilized in the future.

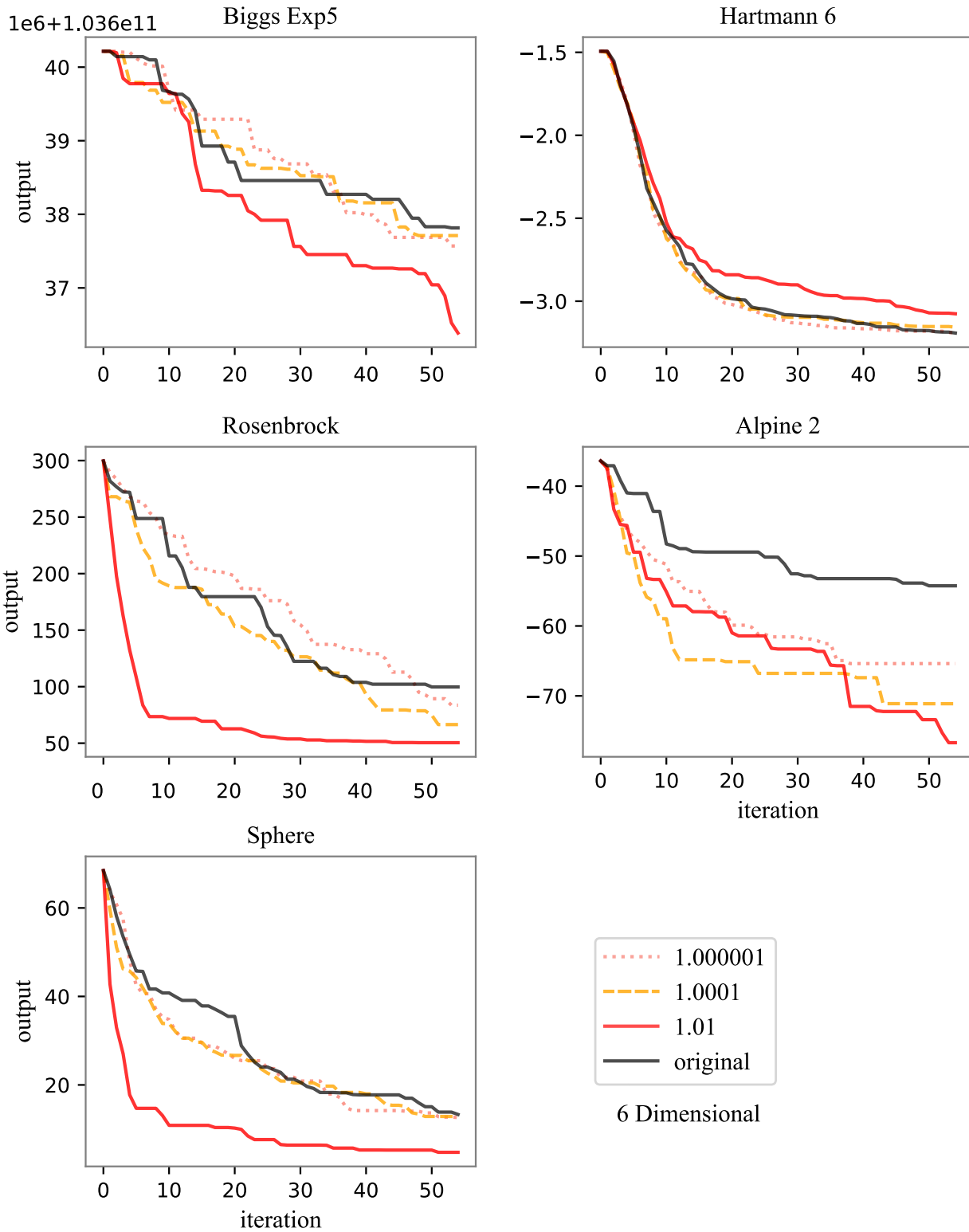


**Figure 5.4** Considered two-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. Numbers in the legend are the value of the lower multipliers in Table 5.1, the corresponding upper multipliers are 10 times their lower counterparts. So, e.g., the legend 1.01 represents the acceptable range of  $[1.01, 1.1]$  for the smallest diagonal element of the  $K^{-1}$ . The lower the value, the less the possible changes compared to the original EGO. Each plot is the average of the 30 repetitions for the iterative phase of the EGO.

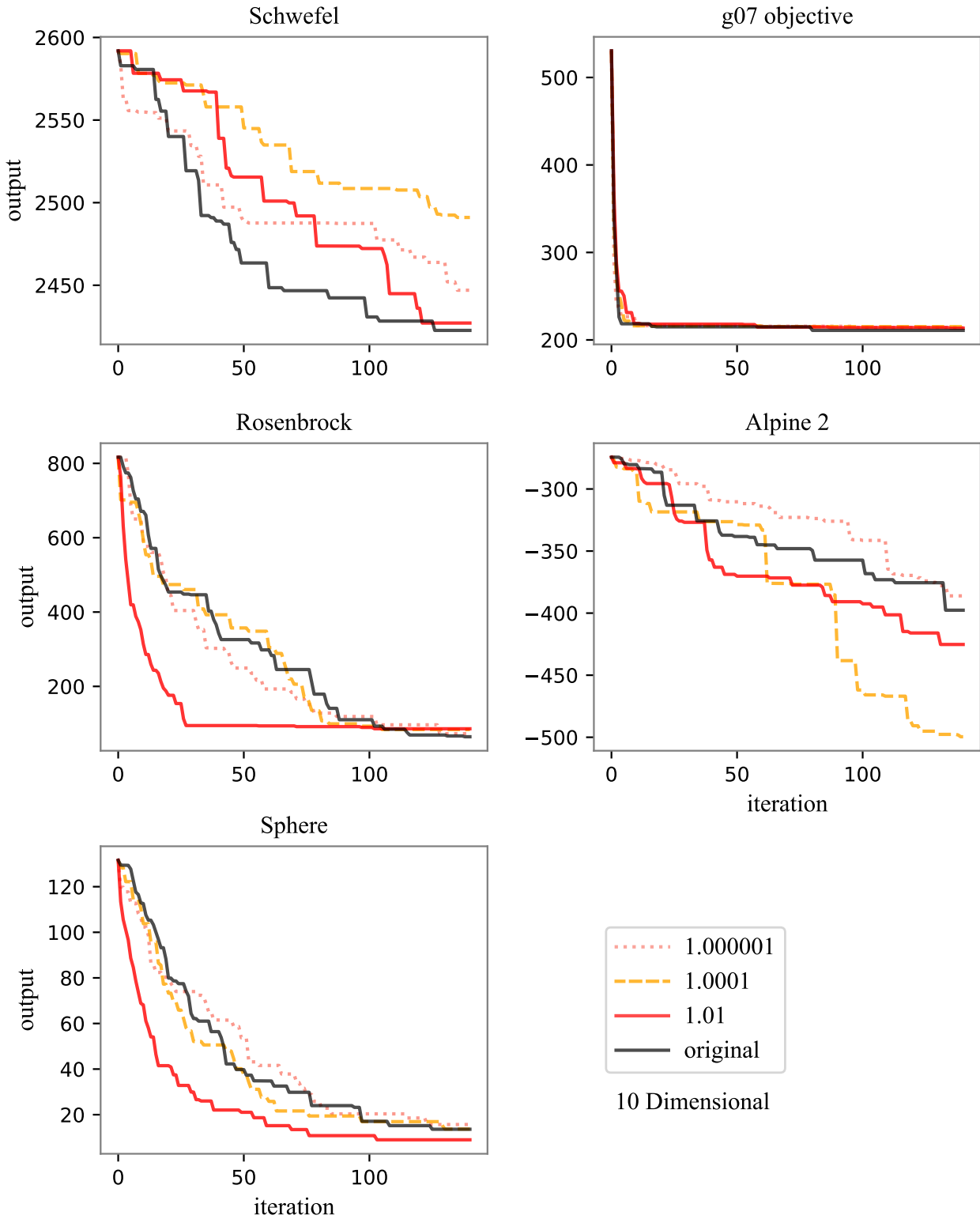




**Figure 5.5** Considered four-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4.



**Figure 5.6** Considered six-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4.



**Figure 5.7** Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4.

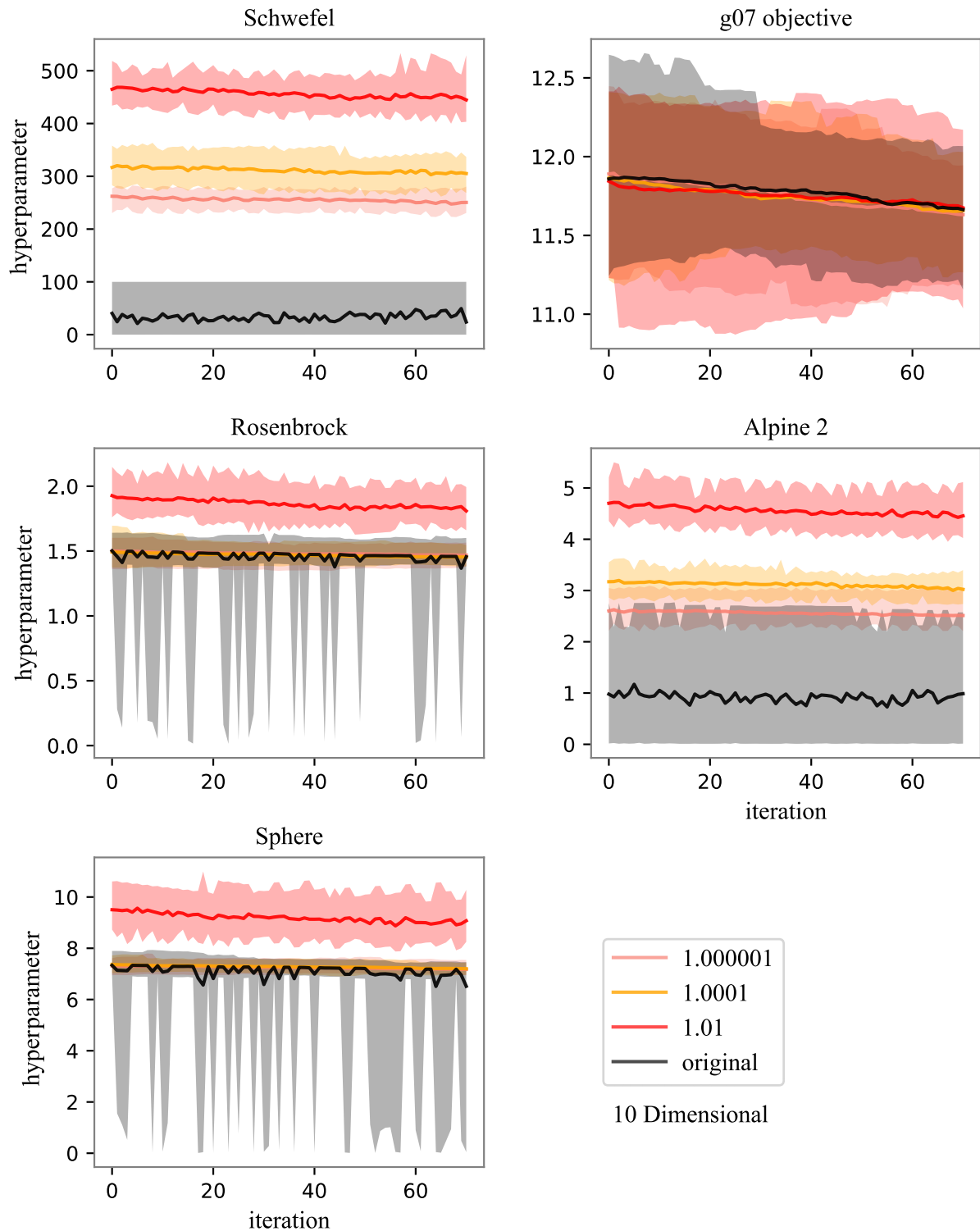
### 5.2.3. Discussion

So far, in this chapter, we have proposed a method for setting the hyperparameter ( $\theta$ ) bounds, most importantly the lower one ( $\theta_{lb}$ ). The results considering the final modification were shown above and they are more balanced than the initial modification, as they were intended to be. We treat the problems as a black box here, and hence no information is available in this regard. However, we use information from the GPR formula to develop our proposed method. By introducing the new range for the smallest diagonal element of  $K^{-1}$ , we modify the initial value of  $\theta_{lb}$ . Not too low to have more possibility of overfitting, but perhaps small enough to allow capturing the local features of the underlying function, even when the locality is so strong that the underlying landscape is actually more like an overfitting case, e.g., see the Schwefel in Figure 5.9. Contrary to the initial modification, the final one considers such strong localities and this is reflected in large improvements in the results of Alpine2 and Schwefel. Each hyperparameter ( $\theta_i$ ) can be considered as a certain global scaling applied to a corresponding dimension. However, here, the basin sizes of Alpine2 and especially Schwefel are changing. Therefore, to capture this locality, GPR needs more samples compared to the case where the basin sizes are close to each other (so a global scaling for the most of the adaptive iterations may be enough). One suggestion to further improve in these cases is to track the history of  $\theta_{lb}$  and evaluate how GPR was correct at previous infill points, as a guide to achieve better results in the remaining iterations. Figure 5.8 shows the history of  $\theta$  for 30 repetitions. Highlights show the region between the maximum and minimum values of  $\theta$  among all repetitions, while the solid lines show the average. Interestingly, Schwefel has a distinct behavior and has a large range of  $\theta$ , which may be explainable considering its formula. It is basically some ups and downs that are mounted over  $x_i$ ,

$$f_{\text{Schwefel}} = 418.9829d - \sum_{i=1}^d x_i \sin(\sqrt{|x_i|}). \quad (5.15)$$

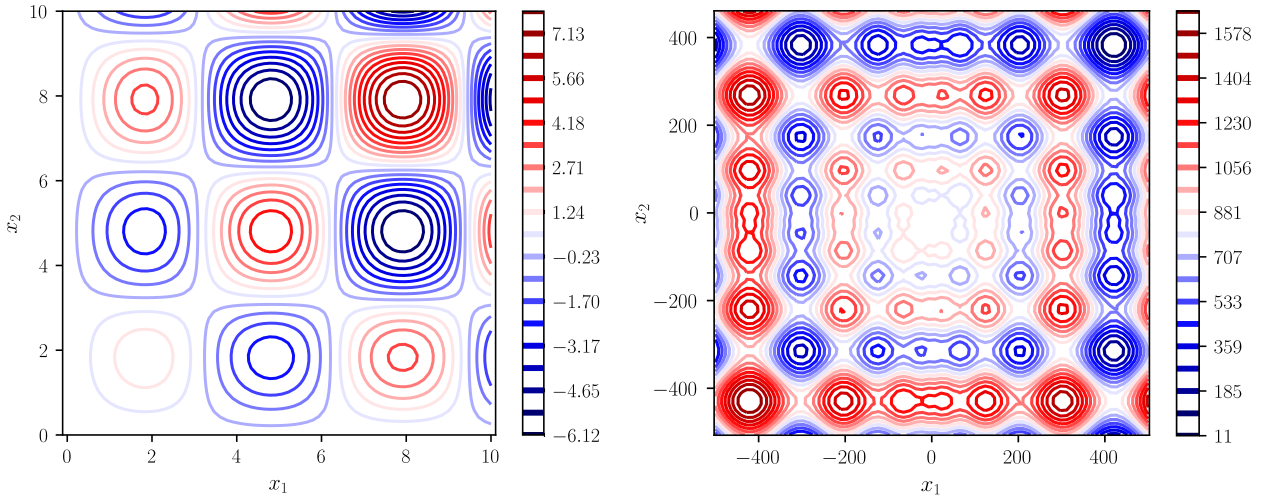
So, for such a low number of samples that we considered in 10D, EGO assumes in some iterations that current data comes from a rather smooth simple surface and hence opts for a large  $\theta$  on one hand, and on the other hand, it mostly assumes that the underlying function is changing locally, so now it favors a small  $\theta$ . Furthermore, Figure 5.8 shows that in case of Schwefel and to a lesser extent Alpine2, all three modifications have distinct regions that do not overlap with each other considerably, contrary to that of the other test functions. This may indicate the high locality and sensitivity of the hyperparameter to relatively small changes of  $\theta_{lb}$  at these low values. In other words, there is the possibility of identifying the locality if such a sensitivity is utilized. The Schwefel result for the original EGO (the gray highlight) indicates that at least some of the repetitions end up at the upper bound of 100 that was initially set by the user, which means that

the algorithm wanted to go for an even higher value of  $\theta$ , but the settings did not allow it.

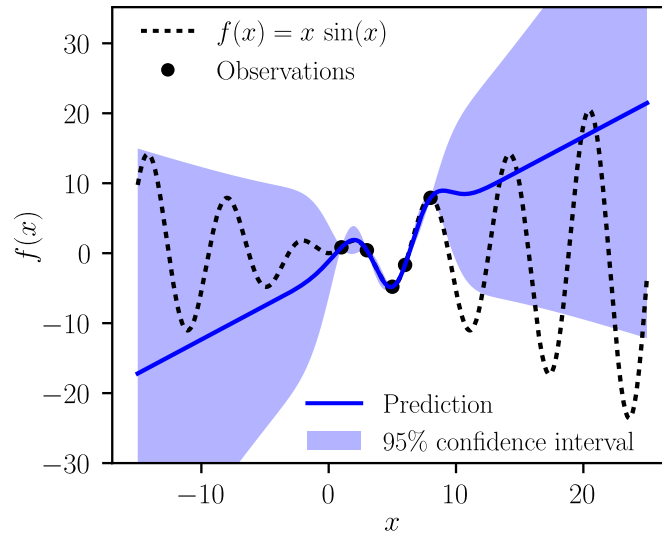


**Figure 5.8** The history of the hyperparameter  $\theta$  in each ten-dimensional test function to evaluate the effect of the final suggested modification for the  $\theta$  bound. The highlights show the region between maximum and minimum  $\theta$  values among all thirty repetitions. Solid lines are the average of thirty cases. The rest of the description is the same as the one in Figure 5.4.

Settings in the proposed method to reduce the overfitting are the result of just a few trial-and-errors



**Figure 5.9** Alpine2 (left) and Schwefel (right) test functions.



**Figure 5.10** Extrapolation of a regularly fitted (i.e., not overfitted) Kriging with a linear trend. This plot shows how extrapolation in far distances can be more and more misleading while at the same time the uncertainty also grows, both of which make EGO visit and remain in regions that it should have avoided initially.

over a couple of functions (other than those tested here). Certainly, more efficient ways of changing  $\theta$  can be devised. However, this was not the main aim of this chapter.

Interestingly, the results of overfitting reduction show that even a large extra improvement for EGO is possible by a more proper setting of the hyperparameters. This is despite the fact that EGO is already an efficient algorithm for low dimensions. This improvement may also indicate that even for an EGO with a constant trend, inappropriate fitting can occur and it is not just a rare event. Inappropriate here refers to either overfitting or less severe cases where the fitted GPR assumes the underlying function is considerably more local than it actually is. Consequently, the higher order trends could end up being even worse, especially in our case where the budget is small. This was discussed in explaining the bias-variance dilemma in Section 2.2. Another difference between a constant trend term and a higher-order one is how they extrapolate. At locations away from current

samples, the Gaussian term vanishes and only the trend term of the fitted Kriging remains. Hence, for a linear trend, extrapolation at such locations leads to a linear change in the approximation. This makes EGO search for boundaries with lower values of the linear trend (in a minimization). Hence, there is a chance of finding a good optimum faster but also getting lost for many iterations (even the entire budget), as the experience of the author shows. See Figure 5.10 as an example. Here, a constant trend is a safer bet, given our limited budget. Another difference between trends in Kriging is their evaluation time. A linear trend Kriging becomes more and more expensive by increasing the number of samples and dimensions relative to a constant trend Kriging. This can make EGO a couple of times more expensive and hence loses its efficiency quite faster.

### 5.3. Proposed modification for (W)EI

Functions like (weighted) Expected Improvement ((W)EI) can be challenging for any optimizer, due to the presence of flat regions surrounding promising ones and / or multiple shallow local optima, see Figure 5.3. In this section, our aim is to make it easier for the optimizer to find the promising regions of EI, while EI refers to both itself and WEI. This is done preferably without changing the balance of exploitation-exploration at least for the promising regions and the other parts are also modified to guide the optimizer to a region with better EI values. As far as the author knows, there are not many EI modifications in the literature, especially with our intention here. For example, in Sasena et al. (2002) prediction of GPR is added to EI as a way of modifying EI, which has a strong tendency for exploitation, according to the author's experience. The proposed modification is based on the following principles:

- Consider three levels for EI. In the first level, EI has the highest values and hence remains unchanged, as it is already indicating the most promising regions for improvement.
- Change EI values within the other two levels corresponding to less promising regions, while these levels have a certain range and they are set in a way to guide the optimizer to the level with (potentially) better EI values.
- Modifications are made by replacing the EI terms with new ones that involve normalized mean, standard deviation and some limit values. There is a gap in values at the border of neighboring levels, but this is not difficult for a derivative-free optimizer to navigate over. More details about the proposed method are given in Table 5.4.
- In total, modifications should favor neither exploration nor exploitation to a large degree.

Fortunately, EI has an analytical formula and we know its components, so we can replace them with desirable ones. This piece of information is available regardless of the problem; i.e., we have one of the necessary requirements for making the modified EI more successful (in general) over a variety of problems. There are two sets of tests to check the performance of the proposed modification. One is the same as in the previous section and can be found in Tables 5.2 and 5.3. The only difference is using an anisotropic covariance model here. The second set of tests uses mechanical design problems where constraints play an important role.

**Table 5.4** Proposed modification of EI.

---

- Definitions and initial settings:

- 1  $\Phi$ : CDF of the standard normal distribution
- 2  $\phi$ : PDF of the standard normal distribution
- 3  $y_i$ : outputs up to now
- 4  $\hat{y}(x)$ : predicted mean
- 5  $\hat{s}(x)$ : prediction standard deviation
- 6 target: minimum  $y_i$  in case of unconstrained problems or  
minimum feasible  $y_i$  or  $y_i$  with the least constraint violation
- 7  $y_{\text{mean}} = \text{mean } y_i$
- 8  $y_{\text{max}} = \max y_i$
- 9  $\hat{s}_{\text{max}} = \max \hat{s}(x) = 1$  (normalized)
- 10 original EI =  $(\text{target} - \hat{y}(x))\Phi\left(\frac{\text{target} - \hat{y}(x)}{\hat{s}(x)}\right) + \hat{s}(x)\phi\left(\frac{\text{target} - \hat{y}(x)}{\hat{s}(x)}\right)$
- 11 first relative prediction limit =  $-1$
- 12 stand. dev. limit =  $0.9\hat{s}_{\text{max}} = 0.9$
- 14 second relative prediction limit =  $-0.2$
- 13 transition value =  $1$
- 15 main quantity =  $\frac{\text{target} - \hat{y}(x)}{y_{\text{mean}} - \text{target}}$

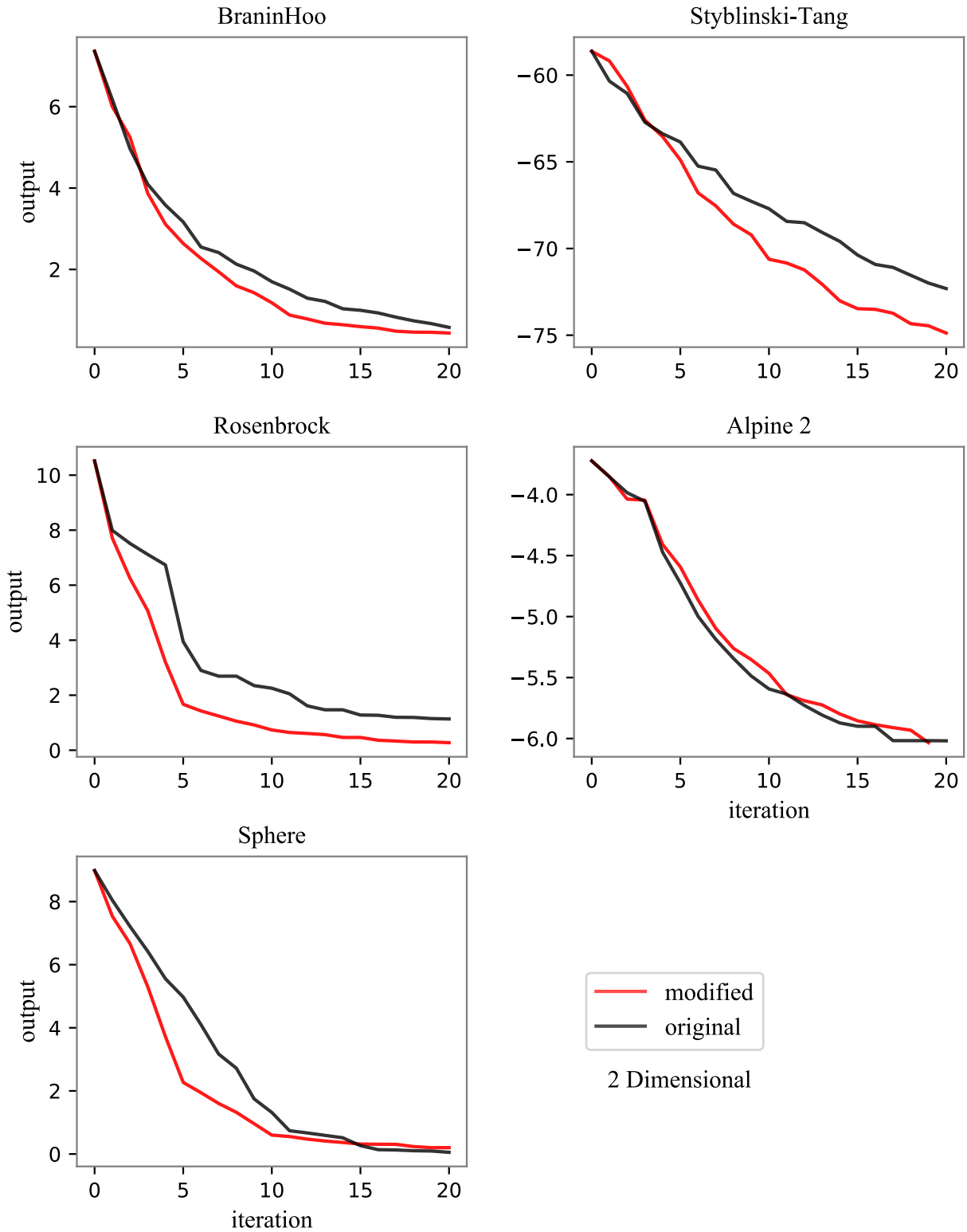
- Proposed modification:

- 16 if target =  $y_{\text{max}}$ :
  - 17     modified EI = original EI
  - 18     print the warning that "target =  $y_{\text{max}}$  ", switch to the original EI
  - 19 else
  - 20     if main quantity < first relative prediction limit:
  - 21         if  $\hat{s}(x) >$  stand. dev. limit:
  - 22             modified EI\_term1 =  $- \left(1 - \frac{\hat{s}(x)}{\hat{s}_{\text{max}}}\right) + \text{first relative prediction limit}$
  - 23             modified EI = modified EI\_term1 \*  $\frac{\hat{y}(x) - \text{target}}{y_{\text{max}} - \text{target}}$
  - 24         else
  - 25             modified EI = modified EI\_term1
  - 26         else if first relative prediction limit < main quantity < second relative prediction limit:
  - 27             modified EI = original EI + main quantity +  $\frac{\hat{s}(x)}{\hat{s}_{\text{max}}}$
  - 28         else
  - 29             modified EI = original EI + transition value
  - 30      $\log_{10}(\text{modified EI} + 10)$
-

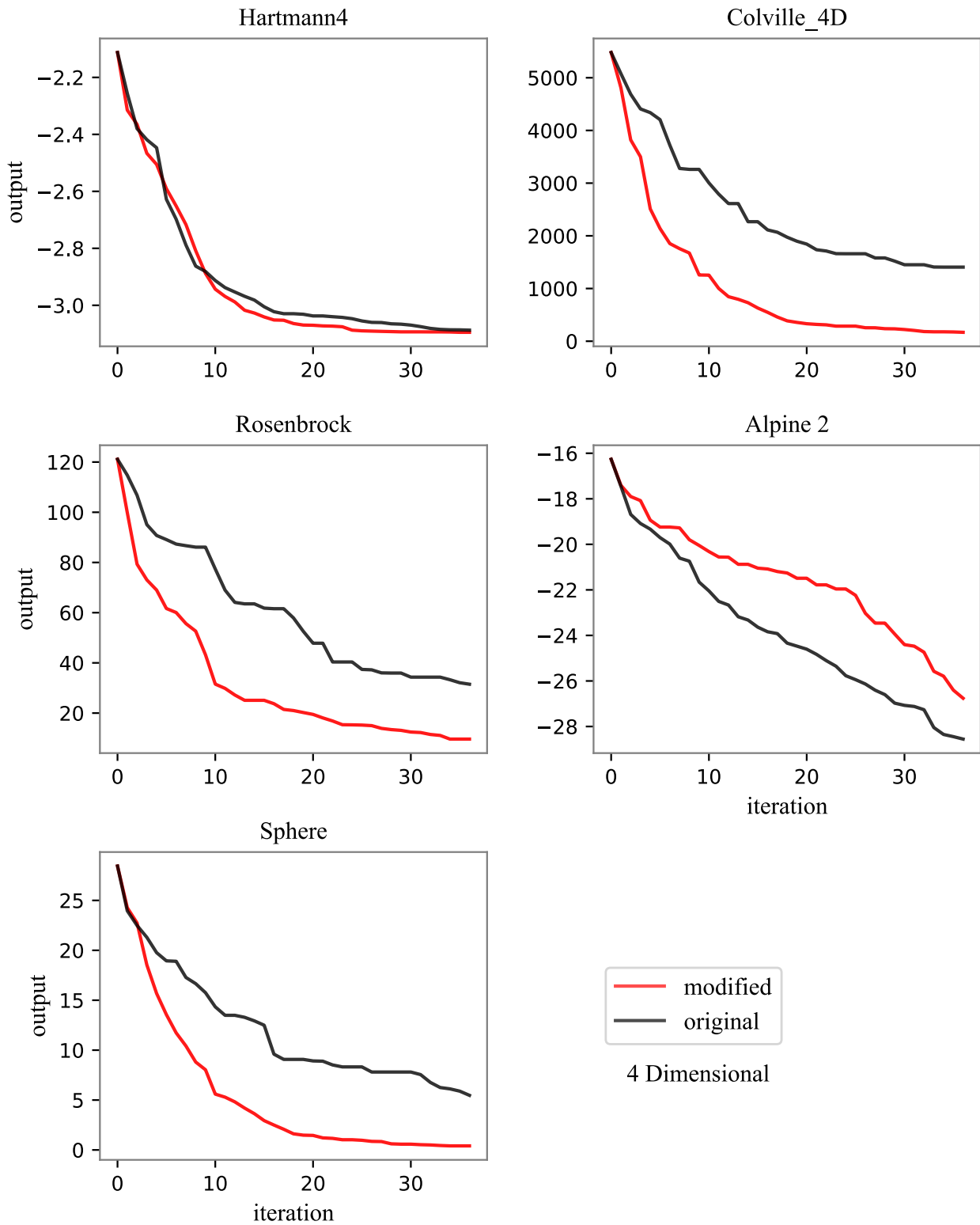


### 5.3.1. Test functions as unconstrained problems

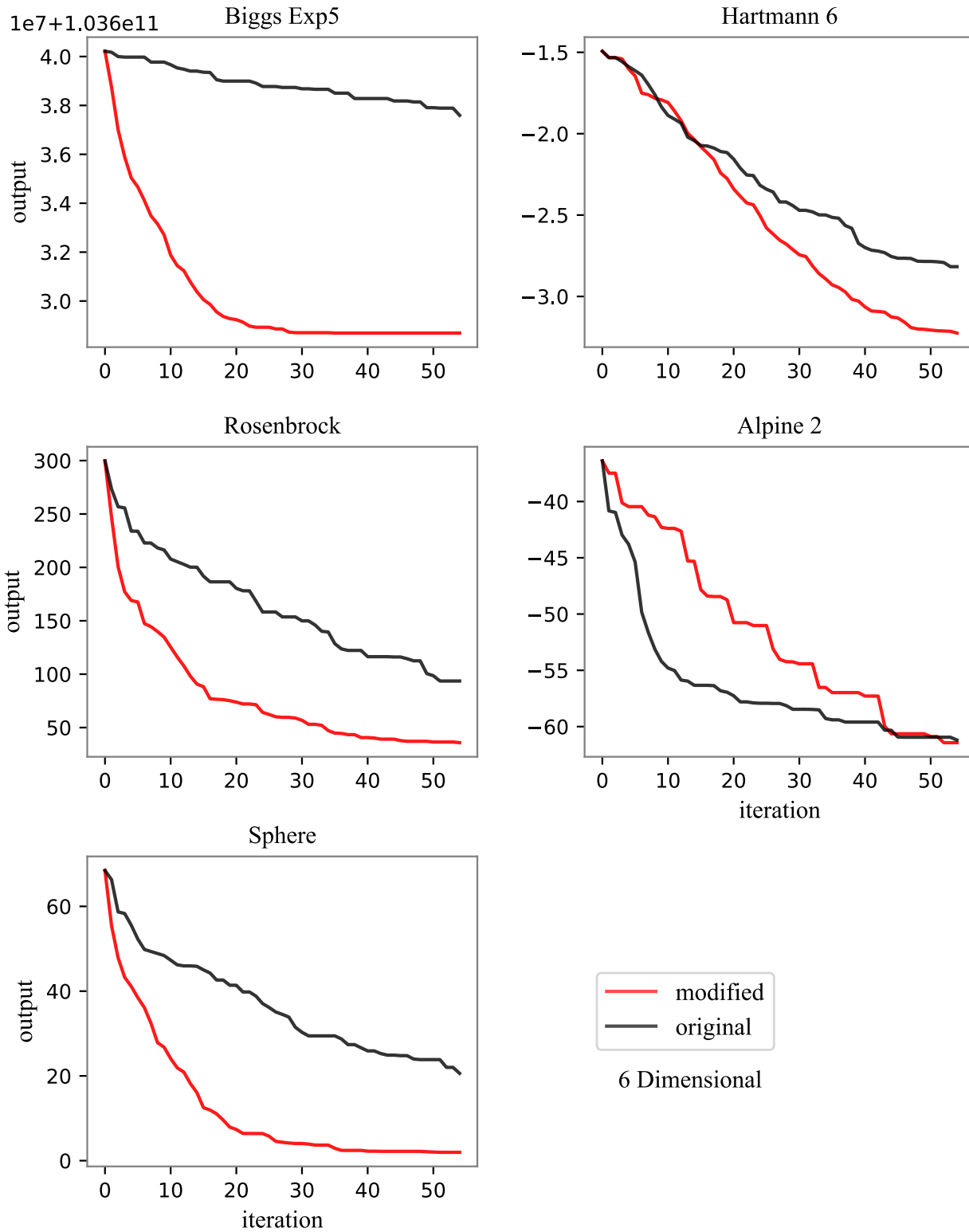
Figures 5.11 - 5.14 show the results of the original EI and the modified EI. The improvement in the considered test functions is significant in many cases. Only for Alpine2 the performance is not as good as that of the original EGO in 4D. In 6D it catches up at the end and in 10D it becomes even much better than the original EI's performance. For Schwefel in 10D, we also obtained an improvement in the modified version. It seems the proposed modifications have no strong tendency to either exploitation or exploration; fulfilling our aim to achieve such a balance. So, we improved EI at least for problems similar to the functions tested here and this is another indication of the positive effect of information inclusion into the solution procedure. Additional improvement is possible not only by coming up with better modifications, but also by incorporating even more information. Here, we do not have the maximum access to the (entire) solution procedure; since we are dealing with a black box case. However, at least we are using EI's analytical formula to order the importance of regions to EI and replace these regions with a proper simplified models. This also means that we depend on how well EI interprets the underlying function, so any deficiency can negatively affect the results, independent of the model itself. Beside reducing overfitting for EI, the principles of the proposed modification may be applicable to other infill criteria, in order to get the most out of them. In this work, we try to improve the EGO for a small amount of budget and with the application of crashworthiness.



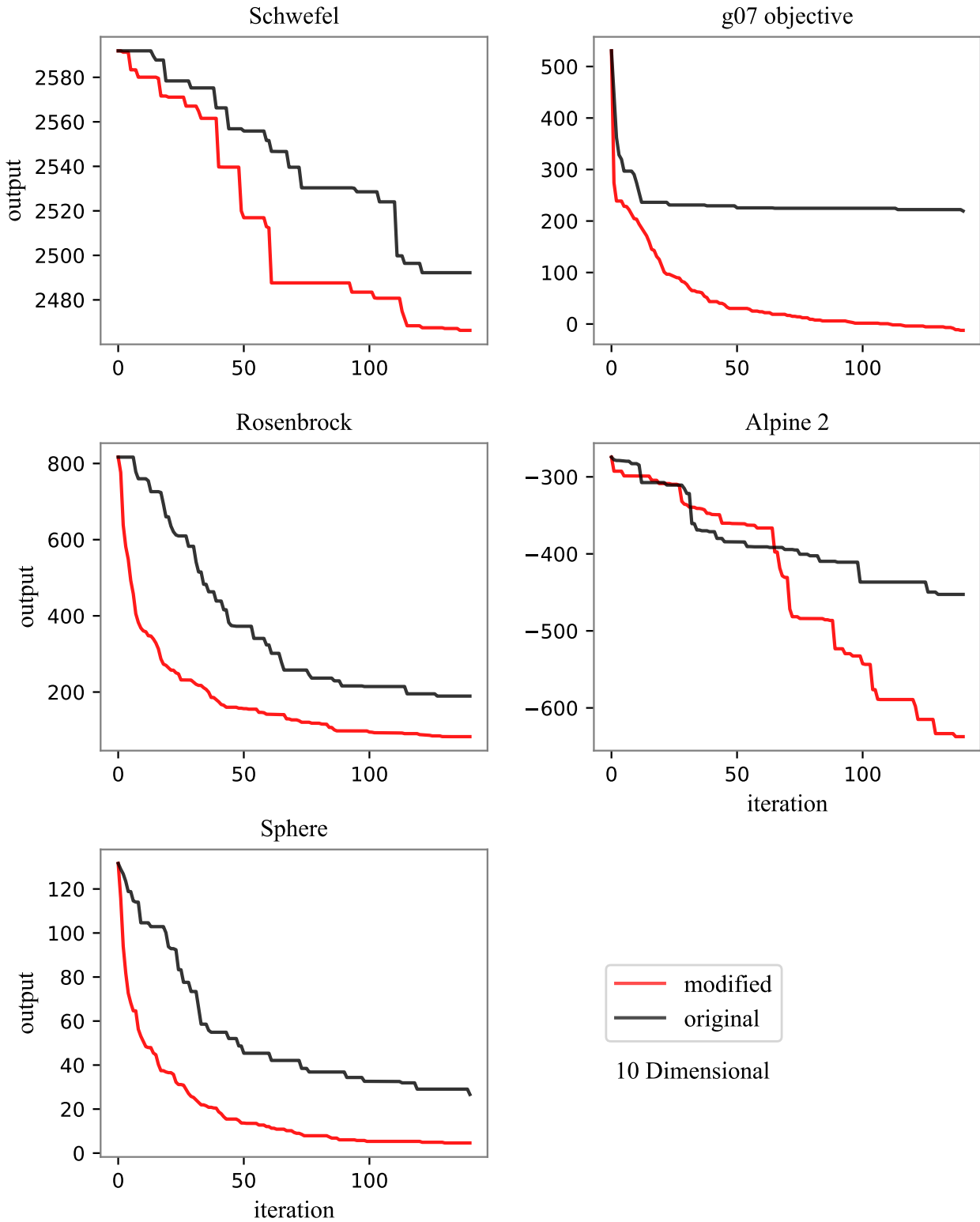
**Figure 5.11** Considered two-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO.



**Figure 5.12** Considered four-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO.



**Figure 5.13** Considered six-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO.



**Figure 5.14** Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO.

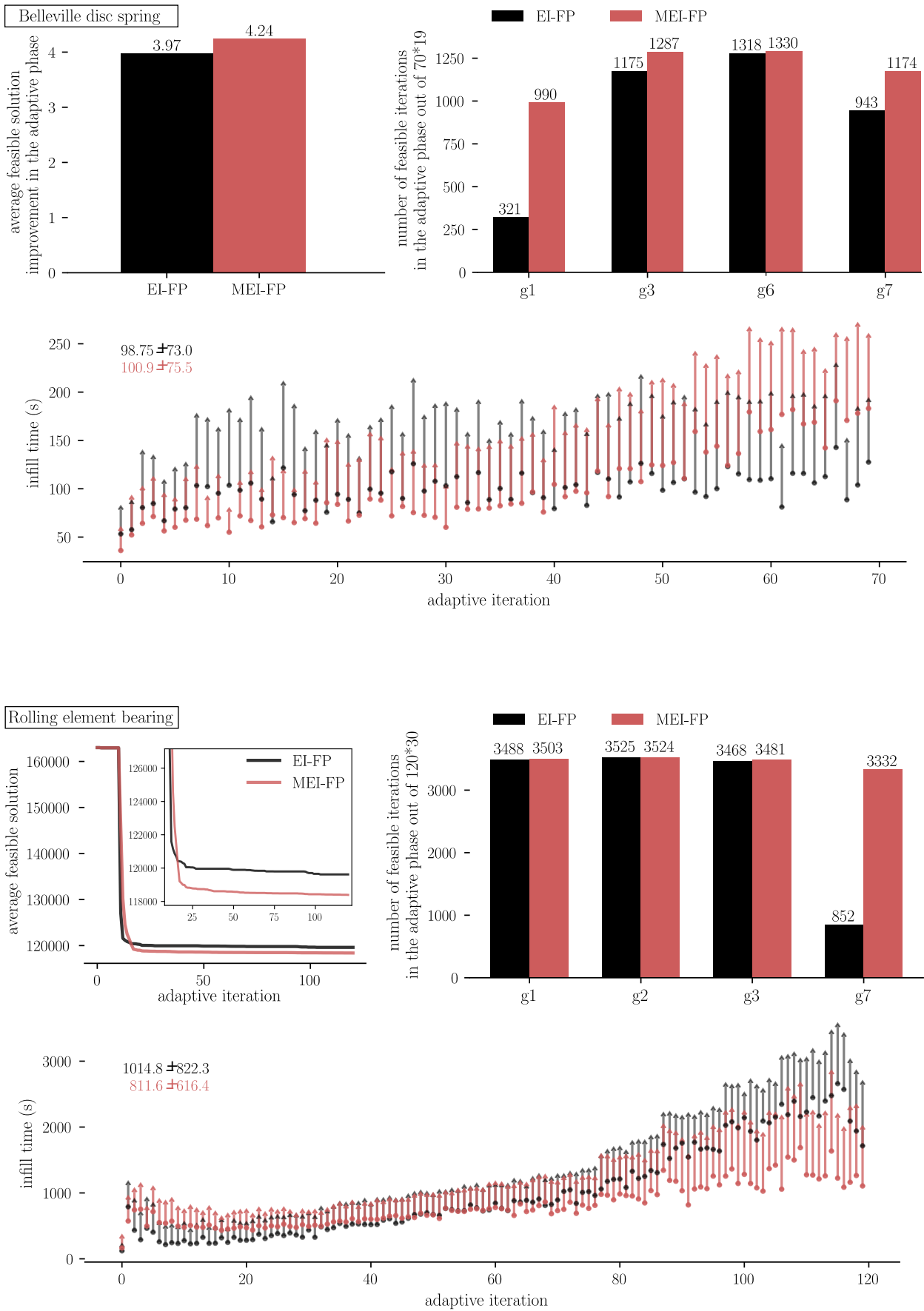
### 5.3.2. Mechanical design problems

In the previous section, the proposed modified EI was compared to the original one over several unconstrained test functions. In this section, we see how it handles constrained problems. However, now, the infill criterion is not just EI and instead EI-FP is selected. As mentioned in Section 3.3.3, EI-FP is composed of the EI of the objective function, multiplied by the feasibility probability of each constraint. Here, we are interested in how the proposed modified EI affects the performance of EI-FP. Hence, the new criterion for constrained cases is called MEI-FP to emphasize the modification of EI without changing the feasibility probability of constraints. The next chapter deals with constrained EGO by focusing on constraints.

Four mechanical design problems are considered, namely, a rolling element bearing, a Belleville disc spring, a speed reducer (gearbox) and finally a hydrostatic thrust bearing design (see all in Appendix B). Note that the first constraint in the thrust bearing problem is relaxed here to allow for a comparison of feasible results on a small budget. See Appendix B.4 for more information. Settings for the problems are the same as those of the previous section for test functions and results are given in Figures 5.15 and 5.16. Three plots are provided for each problem. One of them compares the number of feasible iterations for only some of the constraints (top right). We do not compare the performance of the two methods over the remaining constraints, since these results are very close and current plots have better readability without them (given limited space). The second plot shows the infill time (the time required to find the next infill point) in adaptive iterations, where the average value is shown by circles and the corresponding standard deviation by upward arrows (the bottom figure). The last of these three plots (top left) shows how the average feasible objective value changes. This plot has two forms; either the average feasible solution (of thirty repetitions) for each adaptive iteration is depicted or the obtained improvement in the entire adaptive phase is shown in the form of a bar plot (i.e., reducing the final feasible result from the initial one). The reason for considering the latter (i.e., bar plot) is that otherwise, the plotted average value may even go up significantly (while we are doing minimization) due to a severe lack of feasible solutions and averaging. However, this usually gets better close to the end of the adaptive iterations and the difference between the two methods becomes more stable, since more feasible solutions are available to stabilize the averaging. This type of bar plot is shown for the Belleville spring and thrust bearing problems.

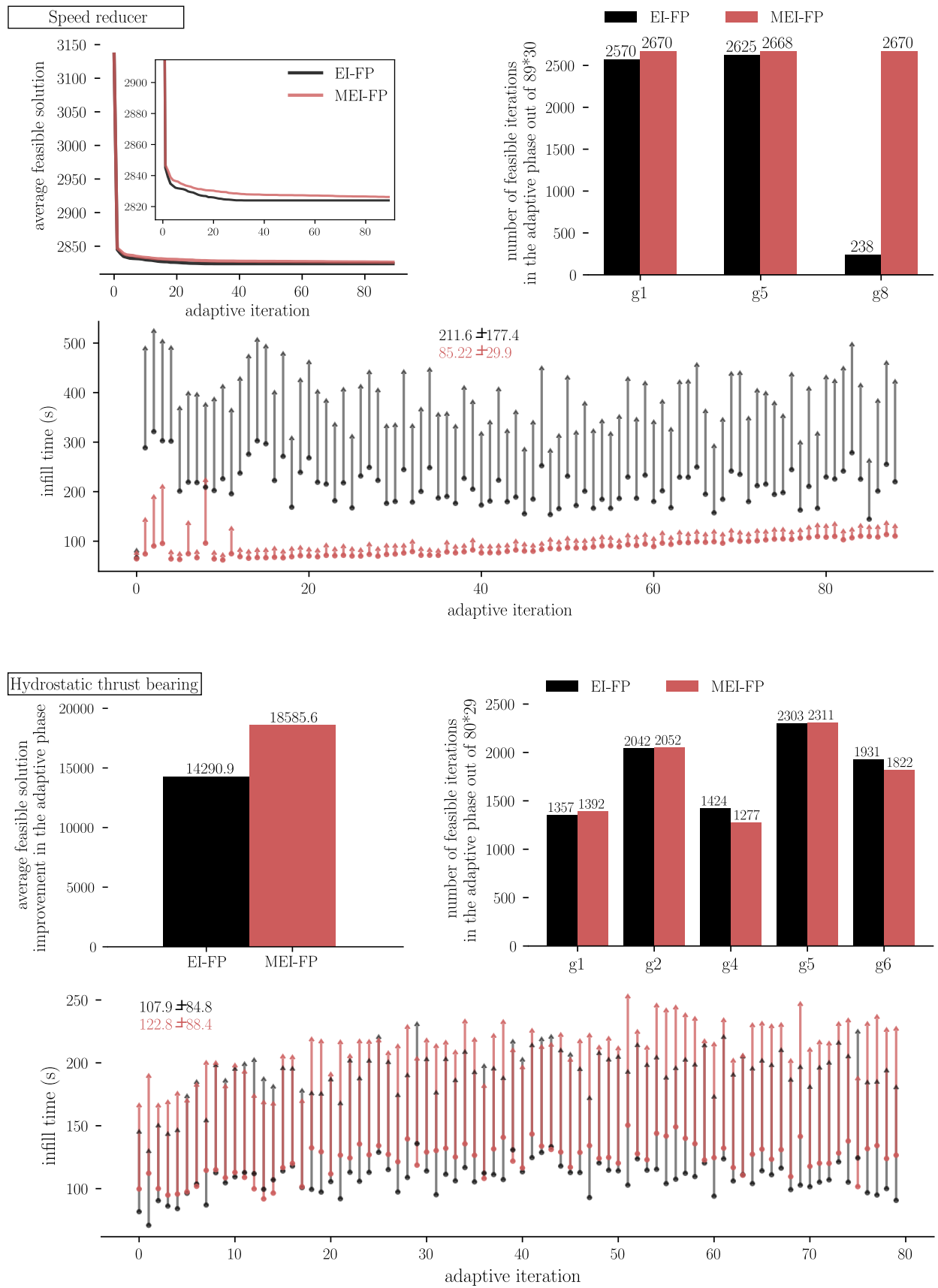
One observation from the feasibility of constraints (the top right figures) is that the proposed modification significantly increases the number of feasible constraints (in general), although the modification is just applied to the EI of the objective and nothing about the constraints has changed. Among the four considered examples, only the thrust bearing problem shows a worsening of the total number of feasible constraints. However, the feasible solution has a higher improvement. One explanation is that the proposed method (MEI-FP) obtains higher number of solutions with feasible  $g_1$ , which is important to reach better results if one studies this problem in more detail. But at the same time, some other constraints become less feasible. Hence, the total number of feasible constraints is reduced (although not a lot), but the obtained (average) feasible solution has

improved. This improvement has been achieved by spending more time in optimizing the infill criterion. However, a significantly smaller amount of infill time could be even a bad sign, as far as the author's experience shows. Because, this can indicate that the optimizer converges fast to a local optimum or ends up somewhere in the flat regions of the infill criterion, a typical landscape for EI. The additional effects of the constraints in EI-FP can have even more adverse effects. However, contrary to this behavior of the original infill criteria (EI and EI-FP), MEI-FP may lead to the same or even better solutions while also requiring significantly smaller infill time, as the results of the speed reducer and rolling bearing problems show. This is another indication that the proposed modification of EI can improve at least some (similar) constraint problems. It is not necessary to restate that there are problems for which MEI-FP performs worse than the original EI-FP, although we did not face such problems here. Before ending this section, let us comment on the results of the speed reducer problem (the three top plots of Figure 5.16) to answer some questions that may arise. As the results for both methods show, there is a very fast improvement in the initial iterations of the adaptive phase and already a very good result has been obtained, knowing that the best reported results from derivative-free methods in Zhu et al. (2019) have values around 2994. In case of the original EI-FP, the constraint  $g_8$  is feasible almost only at the beginning of the adaptive phase (not depicted here); correspondingly, the infill time of EI-FP is higher at the beginning of the adaptive phase and then reduces in the middle iterations, contrary to the more common case in which one expects to spend more on infill time as the number of samples increases accordingly. Having more feasible regions potentially makes the optimizer search further requiring more time; although, this does not necessarily mean reaching a better solution. For MEI-FP, there are large standard deviations in the very first iterations of the adaptive phase, indicating the effect of initial sampling and the difficulty of reaching distinct basins by the optimizer. Due to the (quite) low number of samples at the beginning of the adaptive phase, it is more likely that basins are not sufficiently differentiable and/or more distinct ones are not easily identifiable. Most of the improvement in feasible solutions also occurs just at these initial iterations. In the following iterations where good regions of the underlying function are found, the infill time starts to rise gradually as more samples are added. In addition to having much more feasible solutions, the modified method is also much faster than the original one in obtaining the results.



**Figure 5.15** Top three figures are the results of the Belleville spring design problem and the bottom three that of the rolling element bearing problem.





**Figure 5.16** Top three figures are the results of the speed reducer problem and the bottom three that of the thrust bearing problem.

### 5.3.3. Shape optimization of a crash box

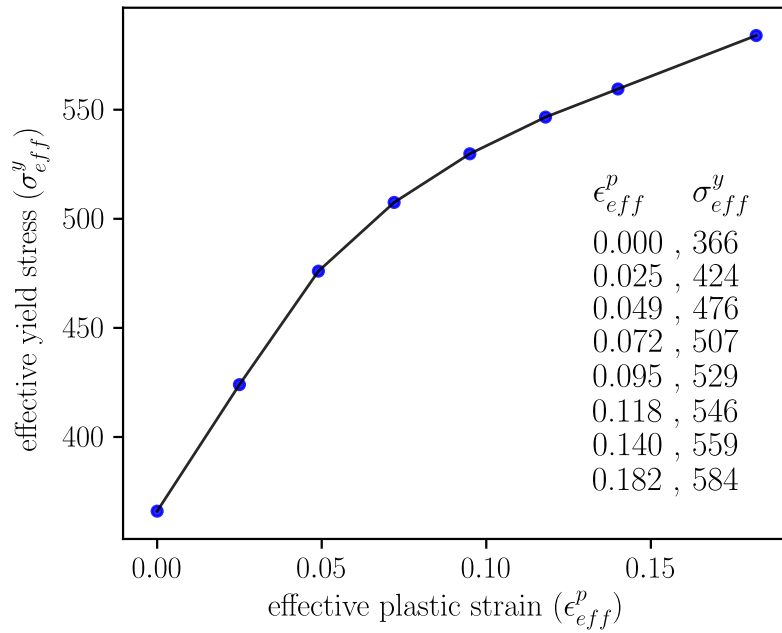
In this section, a crash box problem is considered to compare the performance of the original EGO with that of the proposed modifications of the hyperparameter bounds and EI. The considered crash box cross-section is based on the model in Hunkeler et al. (2013) and shown in Figure 5.18 with five design variables. Here, in a classical drop tower test, a moving rigid wall comes into contact with the crash box from the top with no angle and the crash box is clamped at the other end. The main settings are given in Table 5.5 and the plasticity model is defined in Figure 5.17. The aim is to minimize the mass, while the maximum acceptable intrusion ( $u_{\text{allowable}}$ ) is set at 110 mm and the maximum allowable normal force of the rigid wall ( $f_{\text{allowable}}$ ) is 120 kN. Therefore, the problem is defined as,

$$\begin{aligned}
 & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) = \text{mass} \\
 & && \mathbf{x} = (a, b, u, v, t), \\
 & \text{subject to} && c_1(\mathbf{x}) = \frac{u_{\max}}{u_{\text{allowable}}} - 1 \leq 0, \quad u_{\text{allowable}} = 110 \text{ mm}, \\
 & && c_2(\mathbf{x}) = \frac{f_{\max}}{f_{\text{allowable}}} - 1 \leq 0, \quad f_{\text{allowable}} = 120 \text{ kN}, \\
 & && 0.8 \leq t \leq 2.2, \\
 & && 60 \leq a, b \leq 100, \\
 & && -20 \leq u, v \leq 20.
 \end{aligned} \tag{5.16}$$

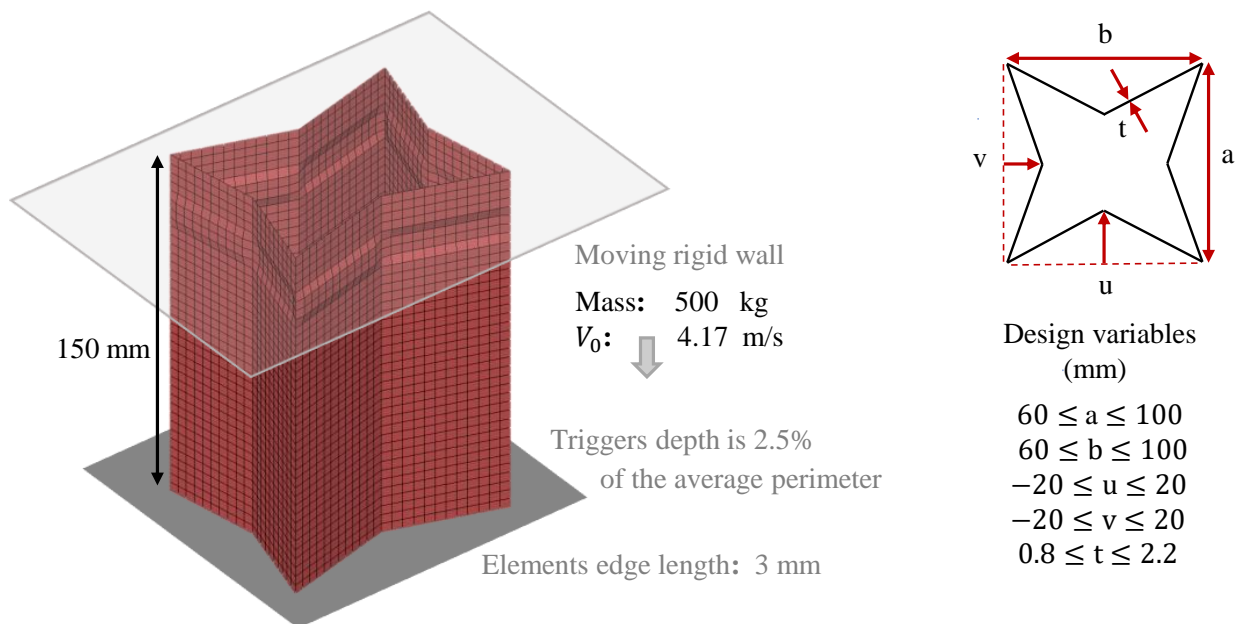
Model creation and other pre-processing steps are done in Python, based on a code developed by Pablo Lozano and modified by the author of this work. LS-DYNA (R9.2) is the solver and the entire procedure is also managed in Python. We use constrained EGO with feasibility probability (see Section 3.3.3) to solve this problem 30 times. Each repetition has a total budget of 50 samples, half of which is allocated to the initial phase and generated by LHD.

**Table 5.5** Settings of the crash box problem. The material model is \*MAT\_24 from LS-DYNA, where below parameters C and P are from the Cowper and Symonds strain rate model and other required data for the plasticity part is provided in Figure 5.17.

impactor velocity	15 km/h	Young's modulus $E$	200 GPa
impactor mass	500 kg	initial yield stress $\sigma_y$	366 MPa
Poisson ratio $\nu$	0.3	strain rate C	40 1/s
mass density $\rho$	7830 kg/m <sup>3</sup>	strain rate P	5 –



**Figure 5.17** Piecewise linear plasticity used for the material model \*MAT\_24 in LS-DYNA. Values at each point are given in the table on right.

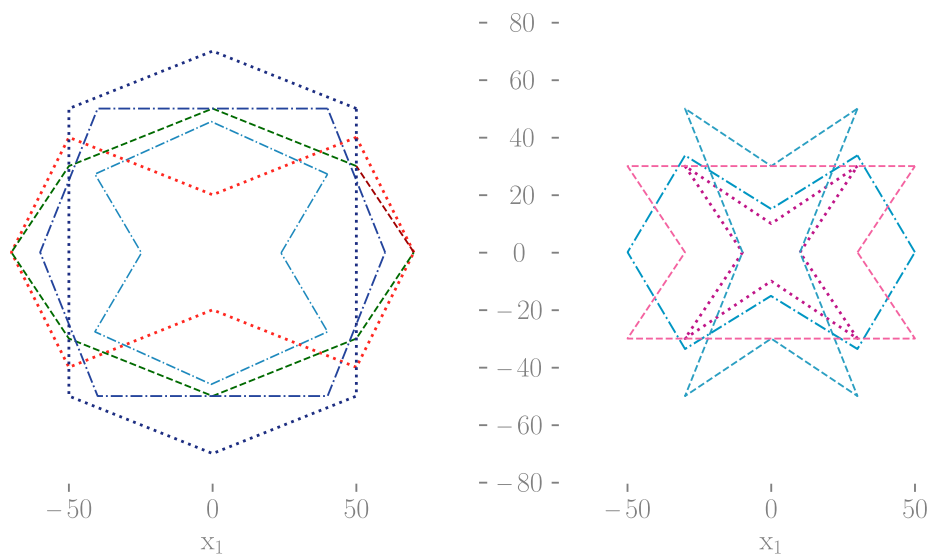


**Figure 5.18** Crash box problem shape and design variables. The transparent plane represents the moving rigid wall impacting the crash box from the top while the other end is fixed.

Three sets of runs are considered for this problem, all with the same initial phase and repetitions (of course, until the second phase of EGO). These runs are:

1. EGO where constraints are handled by EI-FP. This is the original method and will be referred to as EI-FP in the plots.

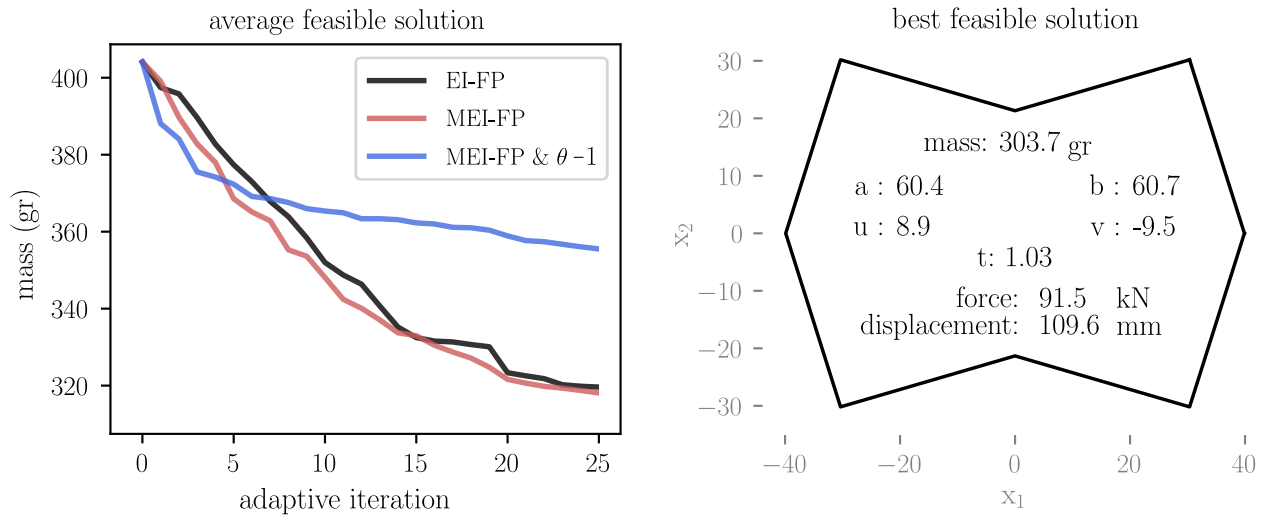
2. Constraint EGO with the proposed modified EI, i.e., MEI-FP as the infill criterion. This case will be referred to as MEI-FP in plots.
3. MEI-FP as in the previous item, plus modification of the  $\theta$  bounds. Here, the acceptable range for the minimum diagonal element in the inverse covariance matrix ( $\min(K_{jj}^{-1})$ ) is set as (1.01, 1.1). This modification is applied to both the objective and constraints. This case will be referred to as "MEI-FP &  $\theta - 1$ " in the plots.



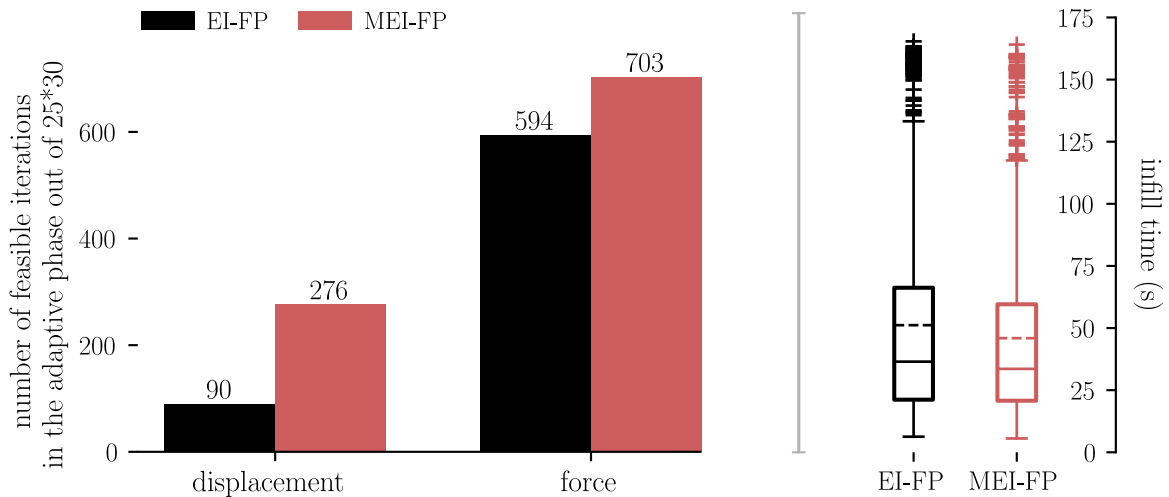
**Figure 5.19** Some possible cross-sections of the crash box problem.

#### 5.3.4. Crash box results and discussion

Figure 5.20 shows the average feasible solutions for the three considered cases. MEI-FP performs the same as the original one in terms of improving the objective function. However, when it comes to the feasibility of constraints, MEI-FP is considerably better, as Figure 5.21 shows. The number of adaptive iterations with feasible displacement has almost tripled, while the feasibility of the force constraint has also increased by 18%. More importantly, the infill time decreases on average by 7.4%, see Figure 5.21 right. The third considered case, i.e., MEI-FP &  $\theta - 1$  starts better than the others, but ends considerably worse, as Figure 5.20 shows. A possible explanation is that the modification of the hyperparameter bound is developed for an isotropic kernel, which tends to treat all input dimensions the same and it seems that this is not suitable here. The reason may be that not all inputs have the same contribution to the output and thickness has considerably more effects than other geometrical design variables. This result does not mean that the proposed modification for the isotropic kernel is always inferior when some dimensions contribute (considerably) more than others to the output. For example, contrary to what was observed here, a significant improvement was obtained for the Colville\_4d case in Section 5.2.2 of this chapter.



**Figure 5.20** Left: Average feasible solution in the adaptive phase of the constraint EGO methods. Right: Best feasible solution among 4500 ( $3 \times 30 \times 50$ ) simulations is one of the MEI-FP cases.



**Figure 5.21** Left: Number of feasible solutions for each constraint in the adaptive phase of 30 repetitions. Right: Boxplots of the infill time. The dashed lines indicate the mean values (51.2 vs. 45.9 s).

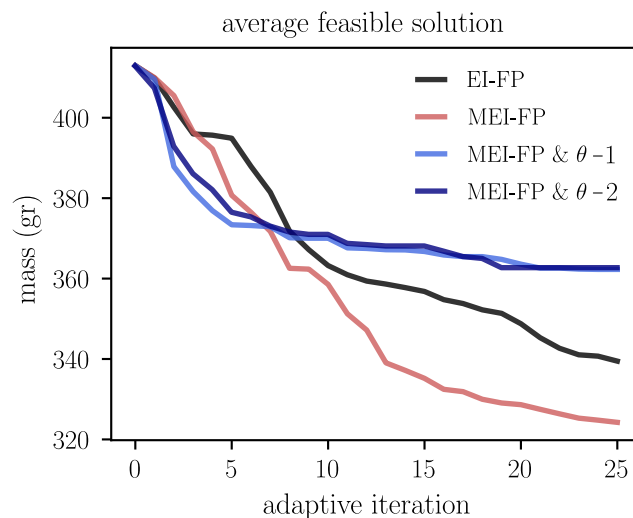
The best solution among all considered cases is one of the MEI-FP simulations and is shown in Figure 5.20. Based on the authors' experience, strong corners can be beneficial in such problems, which is translated here into four corners with almost perpendicular edges, i.e., the material is distributed in two independent directions and not being allocated more to either of them. The displacement constraint is also almost active (109.6 vs. 110 mm), which can be a good sign and at least a possible indication of being close to a (feasible) local minima. Last but not least, design variables 'a' and 'b' are also at their lower bounds, which here directly translates into lower mass (objective) values. All the above mentioned points indicate that this design is decent.

We solve this crash box problem again, but this time we remove the force constraint, since it was feasible in many adaptive iterations of both EI-FP and especially MEI-FP (703 out of 750 iterations). We also do not normalize inputs (to  $[0,1]$ ), representing situations where at least one input contributes much more than others to the output. Additionally, besides the three previous

cases, we add a new one:

- The new case "MEI-FP &  $\theta - 2$ " is the same as "MEI-FP &  $\theta - 1$ ", with the only difference in the acceptable range for reducing overfitting, which is changed from (1.01, 1.1) to (1.001, 1.01).

Figure 5.22 shows the average feasible solutions for all four considered cases in the adaptive phase. The average feasible solution at the beginning of the adaptive phase has a mass of 413 (gr) and in the end it is close to 340 (gr) for the original case (EI-FP). This means that with only 25 high-fidelity evaluations, a 17.7% improvement is obtained ( $(413-340)/413$ ), which is indeed a good performance on its own. MEI-FP is also now performing significantly better than the original method. A comparison to Figure 5.20 shows that the proposed MEI-FP method is immune to non-normalized inputs to a large degree, while the original constraint EGO loses its efficiency significantly. Of course, this statement is meant (at least) for this example, without trying to generalize it to all problems. The performance of the new case is very close to that of the third one, while both underperform compared to the other cases, implying that the modified range itself is not important here and perhaps not the main reason for their underperformance. However, compared to the previous results in the normalized situation in Figure 5.20, the third case maintains its performance significantly better than the original constraint EGO (EI-FP) does.



**Figure 5.22** Average feasible solution in the adaptive phase of the constrained EGO. " $\theta - 1$ " refers to the first modification of the  $\theta$  bounds and " $\theta - 2$ " to the second one respectively.

In this chapter, we proposed to improve EGO's fitting and infill criterion quality by identifying deficiencies and using the analytical formula of GPR as a piece of available information. The first modification was on setting a proper range for the hyperparameter  $\theta$ , especially the lower bound where the corresponding fit can be an overfitting case and at the same time be able to capture local features of the underlying function. This critical region has a considerable influence on the GPR and hence EGO performance. By setting three sets of acceptable ranges in our initially proposed

method, we observed that the one with the least change compared to EGO can have better results on functions with high local features, especially as the dimension grows. This was a motivation for the second (and final) change in which an additional modification happens (even) when the considered range is not violated. This was applied since the problem was not that much about overfitting, but mainly missing capturing local features. Consequently, we provide EGO with the opportunity to bring a better balance between avoiding overfitting and yet sufficiently representing functions with local features having a limited budget. Note that, we are not implying that the second modification is always better than the first one, as being good depends on the required functionality. So, if the aim is to find a very good performing GPR for a special case and we can also afford testing several methods, then the first modification could be a good candidate, since it can excel in what it is good at and its downside (being significantly worse in opposite cases) does not matter then. However, if we need to select an algorithm while we cannot afford to compare several methods, then the second modification is a better choice, as it has a balanced performance with decent improvements. Later, in the crash box example, we saw that the proposed modifications for reducing overfitting were performing worse than the original constraint EGO (EI-FP), since using an isotropic kernel is perhaps not suitable here. This reminds us again that if our model's assumptions do not match the underlying problem, we should not expect them to perform well and this can be thought over when setting up the solution procedure. Modifying EI was another proposed method with quite the same improvement in the objective of the crash box problem, but requiring less time and having significantly more feasible solution. The latter can be helpful for further processing of current data. E.g., using feasible data as start points for continuing optimization from different regions in parallel or having more flexibility to select several proper designs instead of just the optimum one. Here, we try to get more performance out of EI, without changing it in the regions where we expect to get the highest improvements and make it easier for the optimizer to reach these regions. This was possible by using information from EI behavior and its formula, which is one of the lessons from the NFLT. The results of the four mechanical design problems show the modified EI method at least improves one of the objective or the infill time and in addition, tends to increase the number of feasible solutions. However, here we only modified the EI part and did not deal with the constraints, which is addressed in the next chapter.

## Chapter 6

### Enhancement of EGO - Part III: constrained problems

This is the final result chapter of this work, which focuses on handling constraints with EGO. In the first part, a new criterion is proposed to enhance the exploration properties of the  $EI_{feas}$  criterion and hence, finding the segmented feasible regions faster. In the second part, we try to enhance the applicability of the established constrained EGO which employs the EI-FP criterion. This will be achieved by combining this algorithm with new proposed stages. So, in the first part, we try to gain improvement by faster sweeping the design space in case of following a certain value of a function. In the second part, this value is expanded to a narrow range, which will be identified gradually and it acts as an additional feasibility constraint for the optimization. Each part is accompanied by examples to compare the performances and a discussion section at the end.



## 6.1. Motivations for following certain function values

In optimization problems, there are several cases where a certain value or a desired range of values of the involved functions (objective or constraints) are of interest, such as:

- Reliability analysis. Here, there are random input variables and therefore, outputs can have distributions rather than a single value. In a reliable design, one would like to have objective values that mostly do not violate the constraints and hence are reliable. This means, failure should be an extreme case and one is dealing with low probabilities happening at tails of distributions. Identifying the limit value and efficiently sampling from its vicinity is desirable in this case.
- There are problems where a certain constraint is active or close to being active. For example, in problems with equality constraints. Another example that we will use later, is the solution space method, where e.g., the intrusion of a car component should reach a certain value, or even lie within a narrow range by relaxing the assumptions.
- In some cases we are interested to have several possible designs to choose from. One way to obtain multiple designs is using a multi-modal optimizer; of course, given that the problem is also multi-modal. But even then, one finds local optima, while we may want to have multiple designs that have almost a certain output value, so we can distinguish them later when more information is available to make decisions. This pursuit of the desired range is also suitable when one wants to design the next iteration of an available product. Usually, changes in performance of subsequent iterations are not too large. Hence, it makes sense to concentrate on searching around the current design performance, i.e., following certain output ranges.

As mentioned earlier in the literature review,  $EI_{\text{feas}}$  is an infill criterion that is designed to follow a certain value of a function. Based on the experience of the author of this work,  $EI_{\text{feas}}$  indeed does, what it is designed for. However, it tends to put samples too close to each other, and hence, it sweeps the design space slowly. In the next section, we propose a criterion that is less dense and leads to a better exploration. This is done with the help of a proposed pseudo error measure for  $EI_{\text{feas}}$ .

## 6.2. A new criterion to follow a certain value

In this section, we propose a new criterion for following a certain value of output with better explorative properties. Since this work focuses on optimization, this function can be the objective function or one of the constraints. As mentioned in the literature review, a criterion that helps to follow the limit value of a constraint is suggested in Boukouvala and Ierapetritou (2014),

$$EI_{\text{feas}} = \hat{s}(\mathbf{x})\phi \left( \frac{\text{limit} - \hat{c}(\mathbf{x})}{\hat{s}(\mathbf{x})} \right). \quad (6.1)$$

where  $\text{limit} = 0$  for a standard constraint ( $c(\mathbf{x}) \leq 0$ ). By maximizing  $EI_{\text{feas}}$ , samples are placed at the location where the output takes the desirable value (due to  $\phi$ ) with some distance w.r.t. previous samples (due to  $\hat{s}$ ). However, based on the author's experience, samples still may be placed too

close to each other. In order to have a criterion with better placement of samples, the variance of the  $EI_{feas}$  could help, in cases it exists. It should be noted that  $\hat{s}$  already represents the uncertainty (i.e., error estimation) of the fitted model and  $EI_{feas}$  itself is a deterministic number at each point ( $\mathbf{x}$ ) for the given hyperparameters. Here, we suggest considering a pseudo-random variable " $\mathbf{z}$ ".

$$\mathbf{z} = \frac{\text{limit} - \hat{c}(\mathbf{x})}{\hat{s}(\mathbf{x})}, \quad (6.2)$$

However, we assume  $\hat{s}(\mathbf{x})$  is a constant multiplier and " $\text{limit} - \hat{c}(\mathbf{x})$ " varies. The pseudo error measure (pseudo standard deviation (SD)) is calculated based on,

$$\begin{aligned} \text{pseudo-SD} &= \sqrt{\text{WA}(\mathbf{z}^2) - \hat{s}^2 \mathbf{z}^2 \phi_{st}(\mathbf{z})}, \quad \text{WA : weighted average,} \\ \text{weighting function} &= \hat{s}^2 \phi_{st}(\mathbf{z}), \\ \phi_{st}(\mathbf{z}) &= \frac{1}{\sqrt{2\pi}} e^{-0.5\mathbf{z}^2}. \end{aligned} \quad (6.3)$$

Weighted average (WA) is obtained via,

$$\text{WA}(\mathbf{z}^2) = \hat{s}^2 \int_{-\infty}^{\text{limit}} \mathbf{z}^2 \phi_{st}(\mathbf{z}) d\mathbf{z} = -\hat{s}^2 \int_{-\infty}^{\text{limit}} \mathbf{z}(-\mathbf{z} \phi_{st}(\mathbf{z})) d\mathbf{z} \quad (6.4)$$

which is done by integration by parts and consequently requires the calculation of  $\int_{-\infty}^{\text{limit}} e^{-0.5\mathbf{z}^2} d\mathbf{z}$ . This integral is obtained by combining two "erf" functions,

$$\begin{aligned} \left(\frac{c}{\pi}\right)^{0.5} \int_{p=-\infty}^{q=\text{limit}} e^{-ct^2} dt &= \frac{1}{2} [\text{erf}(q\sqrt{c}) - \text{erf}(p\sqrt{c})], \quad \text{here, } c = 0.5, \\ \int_{-\infty}^{\text{limit}} e^{-0.5t^2} dt &= \frac{\sqrt{2\pi}}{2} \left[ \text{erf}\left(\frac{\text{limit}}{\sqrt{2}}\right) + 1 \right], \end{aligned} \quad (6.5)$$

so finally, WA is,

$$\text{WA}(\mathbf{z}^2) = \hat{s}^2 \left[ -\frac{1}{\sqrt{2\pi}} \mathbf{z} e^{-0.5\mathbf{z}^2} \Big|_{\text{limit}} + \frac{1}{2} \left( \text{erf}\left(\frac{\text{limit}}{\sqrt{2}}\right) + 1 \right) \right]. \quad (6.6)$$

Since we standardized the constraint to the form ( $c, \hat{c} \leq 0$ ), the limit value will be zero. As mentioned before, the intention of obtaining a pseudo-SD of  $EI_{feas}$  is to employ it in a new criterion with better exploration. In this regard, as reviewed earlier (Section 3.2.1), the lower confidence bound (CB) criterion has a tendency of placing samples more away from each other than some other criteria. Hence, by using this CB concept the proposed criterion finally is,

$$\text{SLCB-}EI_{feas} = -\ln(\max(EI_{feas}, 2.3 * 10^{-308})) - k * 708 * \text{pseudo-SD}. \quad (6.7)$$

The first term includes the natural logarithm to make optimization algorithms perform (potentially) better, as working with (change of) very small numbers is now avoided. In addition, the value under logarithms should not be zero, and hence, a tiny amount ( $\simeq 2.3 * 10^{-308}$ ) is added, which is

almost the smallest positive number on a 64-bit PC in Python. Further,  $\hat{s}(\mathbf{x})$  is also normalized to be maximal one; this makes the calculation easier and leads to  $EI_{\text{feas}}$  values to be less or equal to one. Now, that the first term of SLCB- $EI_{\text{feas}}$  is set, the second term should be modified accordingly as well, so they have a balanced order, and that is why 708 exists in the second term. In more detail, since the lowest and highest values of the first term are  $\ln(2.3 * 10^{-308}) \approx -708.4$  and  $\ln(1/\sqrt{2\pi}) \approx -0.4$ , the range of the first term would be  $\approx 708$  and that is why it is multiplied to the second term. In addition, there is a multiplier  $k$  in the second term that defines how important the second term w.r.t. the first one is. We set  $k = 1$  in problems regarded here.

### 6.3. Problems and results

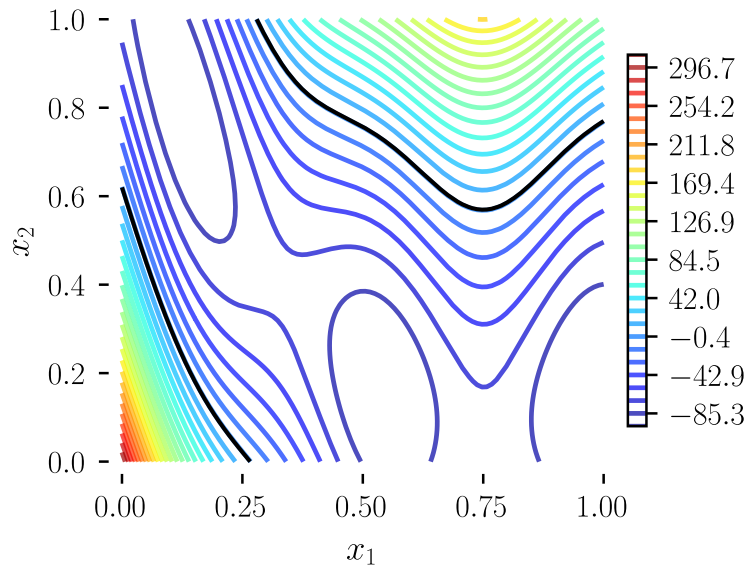
Now, we compare the proposed criterion SLCB- $EI_{\text{feas}}$  with the established one ( $EI_{\text{feas}}$ ) on two test functions, namely BraninHoo and Gomez3. The procedure is the same as for the established EGO algorithm, with the only difference being that these two criteria replace EI. The general settings are given in Table 6.1. Later, in Section 6.6.1 of this chapter, we also compare these criteria, but this time, they are part of a proposed algorithm that sweeps around an active constraint.

**Table 6.1** Common settings used for the problems in this section.

Parameter	Value / Type	Explanation
covariance model	$\exp(-\frac{r^2}{2\theta_i^2})$	anisotropic squared exponential.
$\theta_i$ default bounds	[0.01, 150]	given by the user initially.
$\alpha$	$10^{-6}$	regularization for the covariance matrix.
weights in WEI	0.4, 0.6	WEI: weighted expected improvement.
general optimizer	DE from SciPy	where an optimizer is required.
$n_{\text{init}}$	6	number of initial samples generated via Latin Hypercube.
first infill criterion	$\ln(EI_{\text{feas}})$	instead of $EI_{\text{feas}}$ .
second infill criterion	SLCB- $EI_{\text{feas}}$	the proposed criterion.
$k$	1	a multiplier in the second infill criterion.

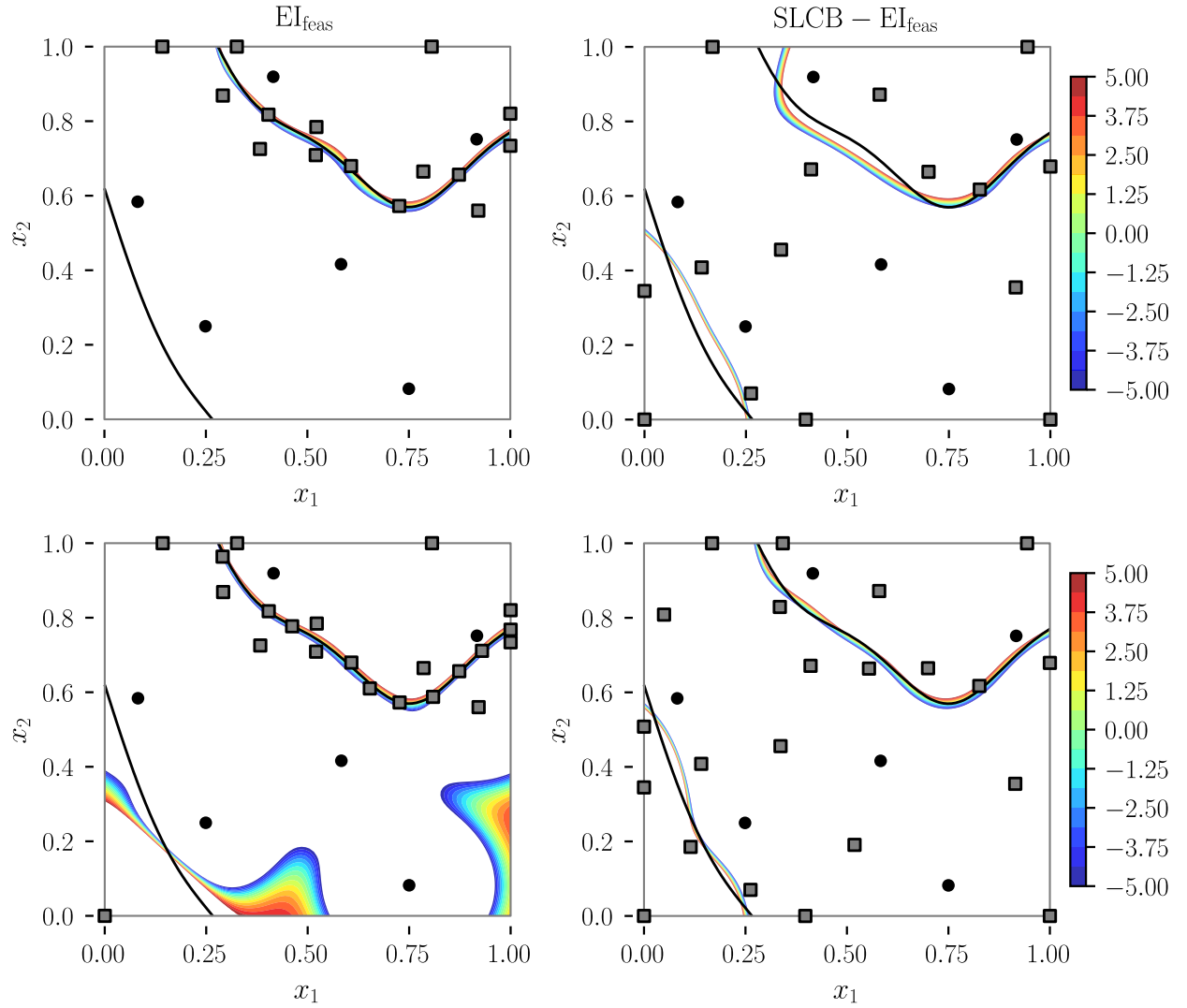
### 6.3.1. BraninHoo

The first example is BraninHoo. We are looking for locations where this function equals 75. These loci are the two open contours shown in black in Figure 6.1. In each case, the EGO optimization is done with six initial points, but the difference is due to the adaptive phase, where  $SLCB-EI_{feas}$  and  $\ln(EI_{feas})$  are optimized to find the next point.



**Figure 6.1** BraninHoo test function contours shown in color in a normalized space  $([0,1])$ . The two black contours show where this function equals 75.

Figure 6.2 shows the results after the 21<sup>st</sup> sample on top and after the 28<sup>th</sup> sample on the bottom, accordingly. The colored contours show the predicted range of  $[-5,5]$  around the target (75) at each iteration, i.e. they show prediction in the range  $[70, 80]$ . One observation is the tendency of  $EI_{feas}$  to put samples close to each other (left figures), to such extent that it takes 28 evaluations to put the first samples away from the current loci (in the upper part of the lower left figure). Accordingly, for the first time the colored contours appear in the lower part, a possible indication of the presence of new locus at these lower values of  $x_2$ . However, the current prediction of the target (75, i.e., middle of the contour) does not coincide with the actual target (the lower black line), due to the lack of samples in this region. On the other hand, the suggested criterion  $SLCB-EI_{feas}$  has a rather good grasp of both loci (lower right figure) after the 28<sup>th</sup> sample. Here, samples are not densely put along with each other. However, this criterion does not follow the loci as exactly as  $EI_{feas}$  does, instead, the proposed criterion starts looking around faster and is able to find the second loci well enough while the fitted curve based on  $EI_{feas}$  just starts discovering the lower region. This example shows that the proposed criterion can be beneficial when there are several regions. However, one could argue that although this criterion finds the second locus sufficiently well, it is only because of earlier exploration and not necessarily because it is efficient. Hence, in the next example we try to see if the proposed criterion is efficient enough compared to  $EI_{feas}$  in finding three separate, but closed curves, one of which is indeed small to capture.



**Figure 6.2** The two black curves show the locations of interest where the BranimHoo function equals to 75. Black circles are the initial samples of EGO and the gray squares are the adaptive phase samples, correspondingly. The colored regions show the prediction by the GP in the range  $[-5, 5]$  of the desired target (75), i.e., predicted BranimHoo in  $[70, 80]$ . The top row is the prediction around the target after the 21<sup>st</sup> sample and the bottom row after the 28<sup>th</sup> sample. The right plots show the proposed criterion predictions and the left that of the  $EI_{feas}$  criterion.

### 6.3.2. Gomez3

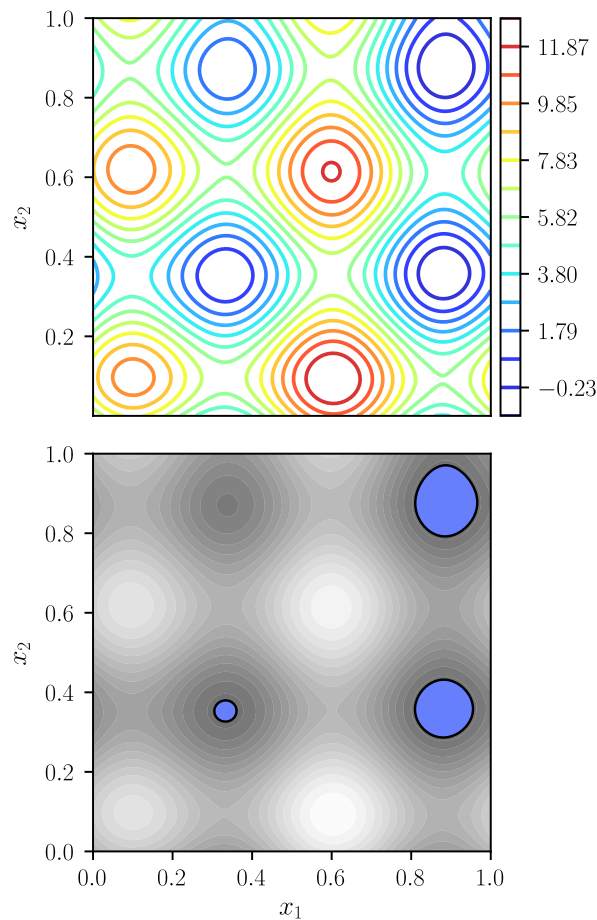
The second considered problem is based on the Gomez3 function defined by Parr et al. (2012):

$$\text{Gomez3} = \left(4 - 2.1x_1^2 + \frac{1}{3}x_1^4\right)x_1^2 + x_1x_2 + (4x_2^2 - 4)x_2^2 + 3\sin[6(1 - x_1)] + 3\sin[6(1 - x_2)]. \quad (6.8)$$

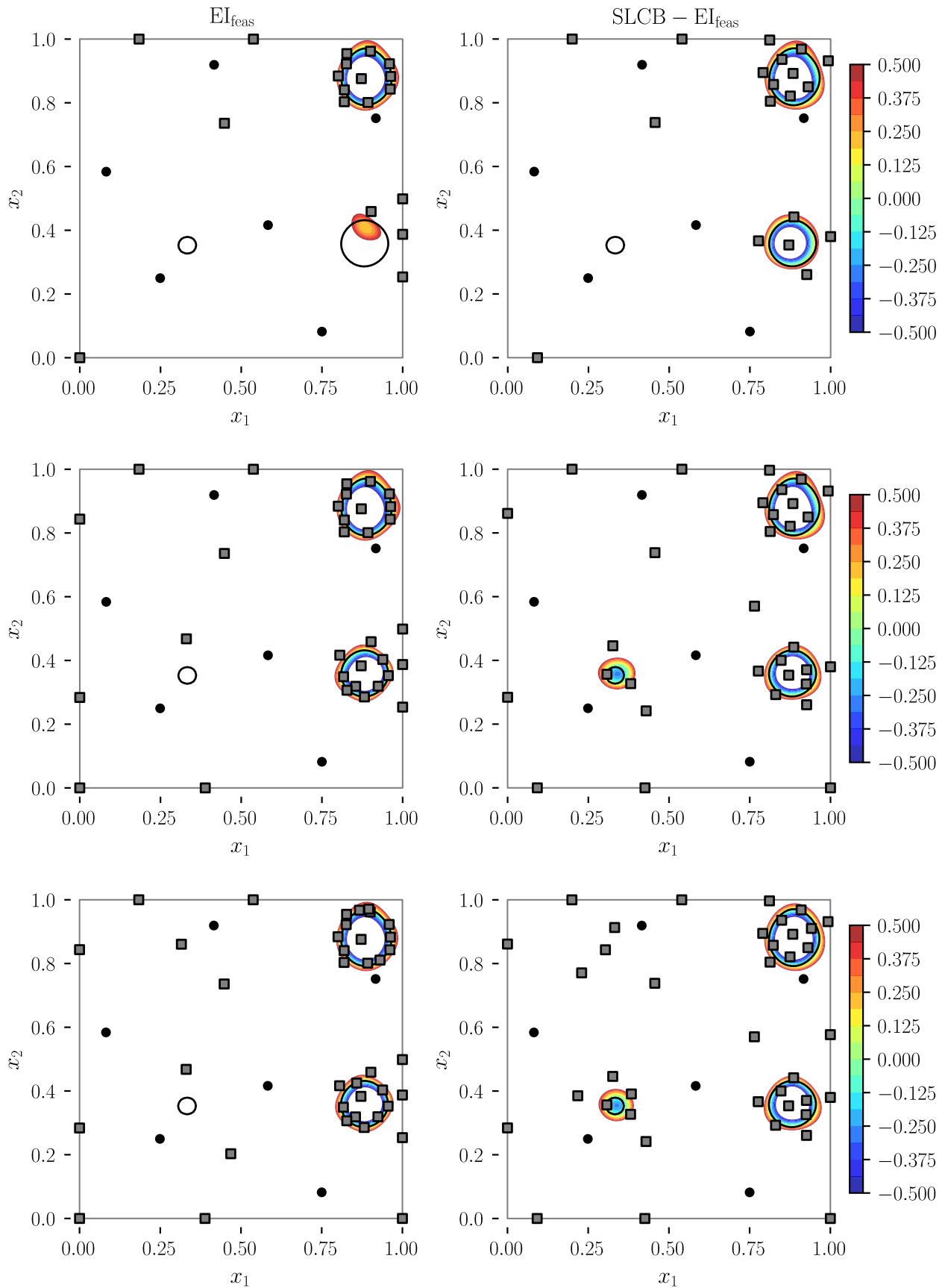
We are interested to put samples efficiently where this function equals 6. In other words, we search for the zero level contours of the "6 - Gomez3" function shown in Figure 6.3 top. These loci are three closed black curves encircling the blue regions shown in the lower figure.

Figure 6.4 shows the results for the proposed criterion on the right and those of the  $EI_{feas}$  on the left. Both criteria detect and follow the top locus, while the proposed criterion finds the other two loci before  $EI_{feas}$ . As this figure shows, by the 38<sup>th</sup> sample (the middle plots) the proposed criterion

has already detected all loci well enough, while  $EI_{\text{feas}}$  has not yet identified the smallest circle even by the 45<sup>th</sup> sample. This circle is especially of interest, which indicates that SLCB- $EI_{\text{feas}}$  is able to capture and present well enough such a small locus, as this was the purpose of this test. However, this does not guarantee that this criterion is always faster in finding loci, especially in multi-modal cases where the landscape changes fast and initial samples can also affect the results. But, we expect to get better results on average via the proposed criterion, which can be checked, e.g., in the welded beam design problem in Section 6.6.1.



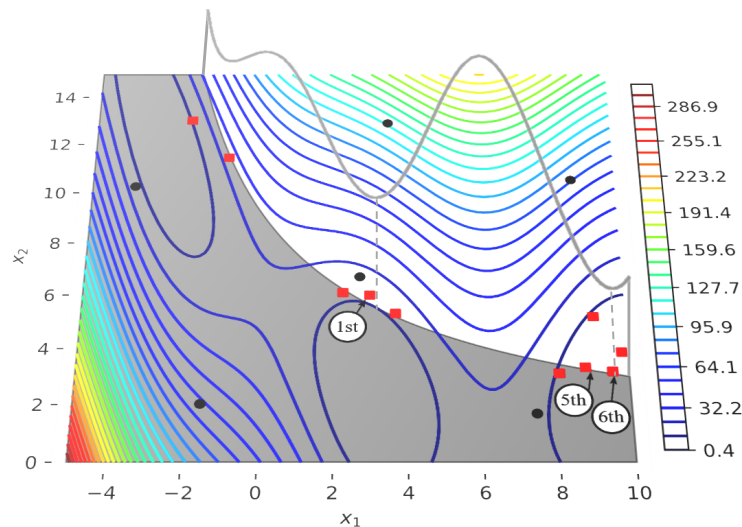
**Figure 6.3** The top figure shows the function "6 - Gomez3" and the lower figure shows where this function is above zero in grayscale and where it is below zero in blue. The closed black contours are loci where Gomez3 equals 6. This example is based on Durantin et al. (2016), where it was used as a constraint (with blue regions showing the feasible part) while, here, we try to follow the boundary of the feasible region, i.e., the limit value.



**Figure 6.4** Finding the locations where Gomez3 function equals 6 (i.e., closed black curves). Descriptions are the same as those in Figure 6.2, except that the first row shows prediction after the 25<sup>th</sup> sample, the second row after the 38<sup>th</sup> and the last row after the 45<sup>th</sup> sample.

## 6.4. Constraint handling via EI – FP

In the literature review, we mentioned different ways of handling constraints in EGO. One of the methods was optimizing "expected improvement with feasibility probability (EI – FP)", in which the EI of the objective function is multiplied by the feasibility probability of each constraint. Here, we are looking for a new algorithm, which tries to optimize around a desirable value of a constraint. But first, we will see in an example how the established EGO algorithm performs, which also reveals one of the reasons behind the proposed algorithm.

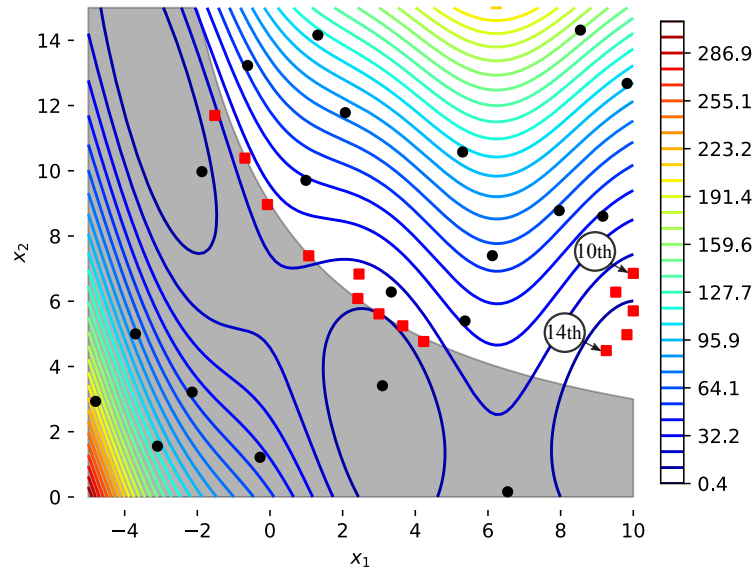


**Figure 6.5** Contour plot of the BraninHoo function subject to a constraint. The infeasible region is highlighted in gray. Black circles show the initial samples and red squares the adaptive samples, correspondingly. The BraninHoo values along the boundary of the constraint (i.e., where it is active) are shown in the gray curve on top.

Figure 6.5 shows the BraninHoo function under a constraint. The infeasible region is highlighted in gray. The global optimum is where the constraint is active and very close to the 6<sup>th</sup> adaptive sample. Here, it is clear that EI – FP is indeed highly efficient. With only six initial samples (shown in black circles) and with the 6<sup>th</sup> adaptive sample, the algorithm is able to get close to the global optimum and also finds the basin of a local optimum fast (around the 1<sup>st</sup> adaptive sample). Now, let's increase the number of initial samples from six to twenty, as shown in Figure 6.6. But then, counter-intuitively, after even the 14<sup>th</sup> adaptive sample (34 samples in total) we are still further away from the global optimum compared to the previous case after the 5<sup>th</sup> adaptive sample (11 samples in total). The difference in performance is quite considerable. But, we have not changed anything and EGO follows its procedure well enough. The 20 initial samples are also a typical LH sampling. As Figure 6.6 shows, EGO looks first around the boundary of the constraint in the middle and upper regions, where there is also a local minimum. It is only at the 10<sup>th</sup> adaptive sample that EGO starts searching the lower region where the global optimum exists, simply because the initial samples around this region are at locations where BraninHoo value is high and hence, they cover up the region behind them, which actually has lower function values (including the optimum). So, EGO searches initially the other regions that appear to it to



be more promising. When EGO finds this desirable lower region at the 10<sup>th</sup> adaptive sample, it places the next samples one by one closer to the optimum while modifying its understanding of the underlying function. This indicates a logical and efficient procedure, which is expected from the EGO. So, the large difference in performance is not because of the severe deficiency in the EGO procedure itself, but more related to the quality of the given information to EGO.



**Figure 6.6** Description is the same as for Figure 6.5.

If available information is biased and misleading about the underlying functions, then EGO like any other method has no chance of searching the design space as efficiently as possible. Yet, EGO can perform still better than some other optimizers since it updates its assumption in each iteration about the underlying function, has powerful kernels (equivalent to basis functions) and a suitable way of balancing exploration and exploitation. However, as this two-dimensional example indicates, still it can take a lot of samples (relative to the dimension) to realize there are better regions. In higher dimensions, this can simply make EGO search inferior regions for a lot of iterations. One may suggest overcoming this problem by making EGO do more exploration. But this would be only proper when we have some ideas about how the function looks and then tune for it accordingly, while in practice, we do not know the underlying function almost always and excessive exploration can actually decrease the efficiency of EGO. This problem is one of the main scenarios that EI – FP can become inefficient. As reviewed earlier, it is mentioned in the literature, that EI – FP is a good criterion, while has also downsides. We also reviewed some suggested remedies. However, in fact, some of the challenges are fundamental and may be difficult to be avoided every time. Here, we try to follow a certain range along with the desired value of a constraint as a different way of tackling a constraint problem, when possible. The example in this section is a possible scenario when such pursuit can be helpful, while some practical cases were also listed at the beginning of this chapter.

## 6.5. A new constraint handling algorithm - following a desired range

In the first section of this chapter, we introduced a new criterion to follow a certain value of an output, while here, we extend it to follow a certain range in a constrained optimization problem. It is known that narrowing the search to the promising regions of the design space helps improve the quality of optimization. This strategy is used for example in multi-fidelity optimizations or tackling high-dimensional problems, see Shan and Wang (2010).

Now, we would like to follow a certain range of a constraint, mainly, to gain the option of having various similar designs, but also getting suitable solutions in cases where the constraint is active or close to being active. We also want to take into account the lesson from the previous section about optimizing in presence of several constraints and possible challenges that it brings, such as biased approximation of the outputs that can lead to searching sub-optimal regions and segmented feasible regions. We need to overcome or reduce these adverse effects. If the feasible region is expected to be segmented, we consider employing  $EI_{feas}$  in our procedure and using  $SLCB-EI_{feas}$  in cases that identifying these segmentations becomes problematic. To reduce the challenges that several constraints cancel each other out or combine into a biased approximation, we can run the established constrained EGO only for a few iterations and then focus on only one of the constraints and try to optimize in the vicinity of its active value using  $EI - FP$ . We call this constraint CTF (constraint to be followed). This step continues until we find some predefined number of samples that are in the desired range of the CTF. In the next step, we start to consider the objective and other constraints as well. Until the end of the budget, we switch between just following the CTF in the desired range and finding the optimum within the current prediction of this range. The more detailed steps are given in Table 6.2 and Figure 6.7. This new algorithm is called Range follower (RF).

**Table 6.2** Range follower (RF) algorithm

---

Consider the following optimization problem,

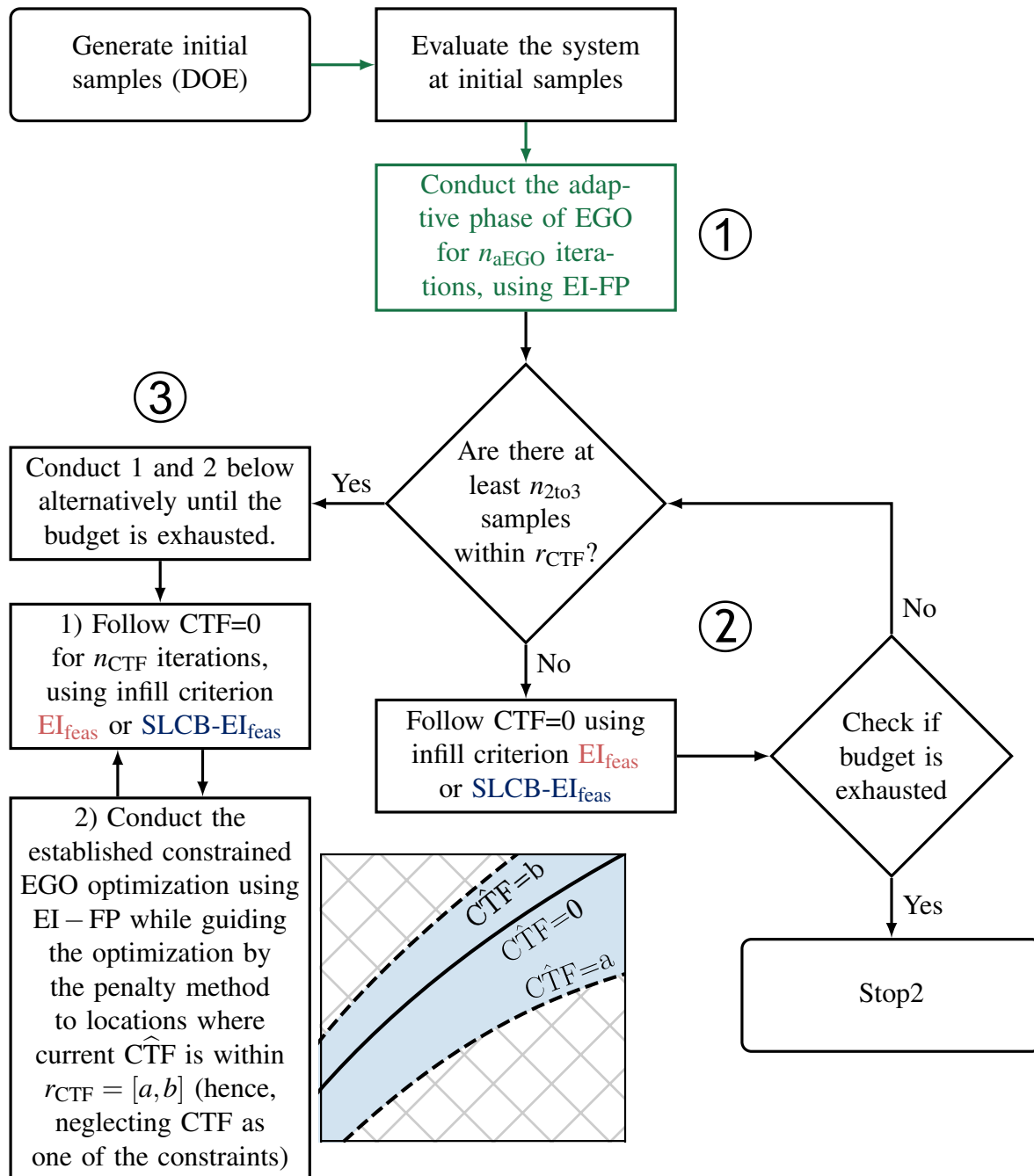
$$\begin{aligned} & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) \\ & \text{subject to} && c_i(\mathbf{x}) \leq 0, \quad i = 1, \dots, m. \end{aligned}$$

Our intention is to focus on one of the constraints and place samples in a certain range of this constraint. The constraint that we are following is called CTF and the procedure has three main parts, which are also numbered and explained in the flow chart in Figure 6.7. The steps are given below.

- 1 Select the constraint to be followed (CTF).
- 2 Define the desired range  $r_{\text{CTF}} = [a, b]$  that should be followed. Note that the constraints are standardized ( $c_i \leq 0$ ).
- 3 Define  $n_{\text{aEGO}}$ , i.e., the number of samples for the first part.
- 4 Define  $n_{2\text{to}3}$ , i.e., the number of samples that should be within  $r_{\text{CTF}} = [a, b]$  to stop following the initial CTF (second part) and start the switching phase (third part).
- 5 Define  $n_{\text{CTF}}$ , i.e., the number of times that CTF should be followed in the switching phase.
- 6 Define  $n_{\text{const}}$ , i.e., the number of times that constrained optimization should be done in the switching phase.
- 7 Define the initial number of samples ( $n_{\text{init}}$ ) and the adaptive phase budget ( $n_{\text{adp}}$ ). Total number of samples will be  $n_{\text{init}} + n_{\text{adp}}$ .
- 8 Define the distance constraints  $d_{\text{CTF}}$  and  $d_{\text{const}}$ . These constraints help to avoid putting samples closer than the defined values to each other. The former applies when we try to follow the CTF in its desired range and the latter when constrained EGO optimization is conducted in the switching phase.
- 9 Set the settings required for EGO components (e.g., type of kernels) in each part.
- 10 Evaluate the system at initial samples.
- 11 Conduct the constrained EGO with EI-FP for  $n_{\text{aEGO}}$  (first part).
- 12 Obtain the number of samples within  $r_{\text{CTF}} = [a, b]$ . If this number is equal to or higher than  $n_{2\text{to}3}$  then start the switching phase directly (item 15 below). Otherwise, first, follow the CTF (item 13 below).
- 13 Following the CTF: As long as the number of samples within  $r_{\text{CTF}} = [a, b]$  is less than  $n_{2\text{to}3}$ , follow the CTF via EGO with the infill criterion  $\text{EI}_{\text{feas}}$  or  $\text{SLCB-EI}_{\text{feas}}$ .
- 14 If the number of samples within  $r_{\text{CTF}} = [a, b]$  is still less than  $n_{2\text{to}3}$  but the budget is over, then stop, otherwise, go to the switching phase next.
- 15 Switching phase: first follow the CTF but only for  $n_{\text{CTF}}$  iterations. Then, switch to conducting the constraint optimization for  $n_{\text{const}}$ . Again, switch back to following the CTF and repeat the procedure. The switching phase continues until the budget is over.

End.

---



**Figure 6.7** Flowchart of the proposed RF and RF\_pseudo algorithms. The difference between these two algorithms is only in the infill criterion when following CTF. The former employs  $EI_{feas}$  and the latter  $SLCB-EI_{feas}$ . The figure in the lower part shows the schematic for the  $r_{CTF} = [a, b]$  range where the prediction of the value that we are following, i.e.,  $\hat{CTF} = 0$  locates within this range. The hat sign indicates the approximation of the underlying function, which is the mean of the fitted GPR. The three numbers in circles indicate three main parts of this algorithm.

## 6.6. Problems and results

In the previous section, the RF algorithm introduced, while following one of the problem constraints in a desired range stands at its heart. If this following is done via optimizing the  $EI_{feas}$ , the method is called by default RF and if instead, the new criterion  $SLCB-EI_{feas}$  is used, it will be called RF\_pseudo, see Figure 6.7. RF will be compared to the established method of constrained EGO with EI-FP as the infill criterion, or simply, the established constrained EGO algorithm. We also refer to it as EI-FP. Contrary to RF, RF\_pseudo is not compared always, since the proposed pseudo criterion was for following segmented regions faster and not particularly for finding the optimum. However, we will report it in the result tables and discuss it, especially in the welded beam design problem which has small feasible regions. Besides this welded beam example, we consider two other mechanical problems, namely, an I-beam and a pressure vessel design. These examples will be compared to show how the proposed RF algorithm is able to follow the desired range and at the same time how it is able to find the optimum even better as EI-FP in some cases. They also help to demonstrate, why the current structure is considered for the proposed algorithm, i.e., first starting with the established EGO at the beginning of the adaptive phase and then introducing the two new parts. We use these problems in their established form in the literature, which in two cases are based on non-SI units. The last example like always relates to crash, but this time, through the solution space method, in which two frontal components of a car are optimized.

### 6.6.1. A welded beam design problem

The objective of the considered welded beam design problem is to find the minimum fabricating cost, while there are some constraints on the weld geometry, deflection ( $\delta$ ), bending stress ( $\sigma$ ) and shear stress ( $\tau$ ). The applied load ( $P$ ) should not be higher than the critical buckling load ( $P_c$ ). There are four design variables  $[x_1, x_2, x_3, x_4] = [h, l, t, b]$  as depicted in Figure 6.8. The objective and constraints are (Li et al., 2017b):

$$\underset{\mathbf{x}}{\text{minimize}} \quad f(\mathbf{x}) = 1.10471x_1^2x_2 + 0.04811x_3x_4(x_2 + 14)$$

$$\text{subject to} \quad g_1(\mathbf{x}) = \tau(\mathbf{x})/\tau_{\max} - 1 \leq 0,$$

$$g_2(\mathbf{x}) = \sigma(\mathbf{x})/\sigma_{\max} \leq 0, \quad g_3(\mathbf{x}) = x_1 - x_4 \leq 0,$$

$$g_4(\mathbf{x}) = 0.020942x_1^2 + 0.009825x_3x_4(14 + x_2) - 1 \leq 0,$$

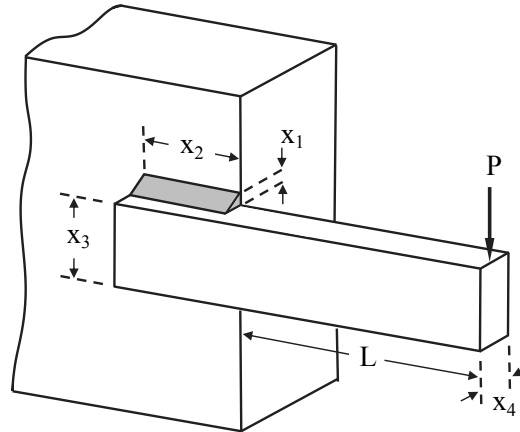
$$g_5(\mathbf{x}) = (\delta(\mathbf{x}) - \delta_{\max})/\delta_{\max} \leq 0, \quad g_6(\mathbf{x}) = (P - P_c)/P \leq 0,$$

$$\text{where} \quad \tau(\mathbf{x}) = \sqrt{(\tau')^2 + 2\tau'\tau''\frac{x_2}{2R} + (\tau'')^2}, \quad \tau' = \frac{P}{\sqrt{2}x_1x_2}, \quad \tau'' = \frac{MR}{J},$$

$$P_c(\mathbf{x}) = \frac{4.013E\sqrt{x_3^2x_4^6/36}}{L^2} \left(1 - \frac{x_3}{2L}\sqrt{\frac{E}{4G}}\right), \quad J = 2 \left\{ \sqrt{2}x_1x_2 \left[ \frac{x_2^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2 \right] \right\},$$

$$M = P \left(L + \frac{x_2}{2}\right), \quad R = \sqrt{\frac{x_2^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}, \quad \sigma(\mathbf{x}) = \frac{6PL}{x_4x_3^2}, \quad \delta(\mathbf{x}) = \frac{4PL^3}{Ex_3^3x_4},$$

$$\begin{aligned}
P &= 6000 \text{ lb}, \quad L = 14 \text{ in}, \quad E = 30 \times 10^6 \text{ psi}, \quad G = 12 \times 10^6 \text{ psi}, \\
\delta_{\max} &= 0.25 \text{ in}, \quad \sigma_{\max} = 30000 \text{ psi}, \quad \tau_{\max} = 13600 \text{ psi}, \\
0.125 &\leq x_1 \leq 10, \quad 0.1 \leq x_{2,3,4} \leq 10, \quad x_i \text{ unit : in.}
\end{aligned} \tag{6.9}$$



**Figure 6.8** Depiction of the considered welded beam design problem based on Cagnina et al. (2008).

This problem has also been solved by considering the upper bound of  $x_1$  and  $x_4$  to be 2 instead of 10, see e.g., Parnianifard et al. (2020) and Dimopoulos (2007). The mentioned difference in the upper bounds (2 vs. 10) has a considerable effect on the size of the design space and the feasible part of it. The variant of this problem with the larger upper bound has almost 27 times ( $9.875/1.9 * 9.875/1.9$ ) larger design space, while its feasible region is only 0.001 of the space size (i.e., 0.1%) compared to the 2.69% of the other definition (with the upper bound of 2). These percentages are obtained by evaluating the problems at 100 millions random samples and calculating the ratio of feasible samples out of 100 millions. Each of the percentage calculations is also done several times to make sure the differences among the results happen at decimal numbers that do not affect the accuracy of the results reported here (i.e., 0.1% and 2.69%). Considering how small the feasible region is, this problem seems to be a challenging one and the literature review also shows that many non-RS-based optimizers may need thousands of evaluations, as summarized in Table 6.4 for the first two references. This is even given the fact that, both of these references (at least in their own evaluations of the problem if not in those articles that they have compared to) have chosen the version of this problem with a much smaller design space ( $1/27 \approx 0.037$ ).

**Table 6.3** Settings of the RF method for the welded beam design problem.

Parameter	Value / Type	Explanation
CTF	g6 in Eq. (6.9)	the constraint to be followed in the desired range $r_{CTF}$ .
$r_{CTF}$	[-1.5,0]	the desired range of the selected constraint to be followed.
$n_{init}$	30	initial phase budget using Latin Hypercube.

$n_{\text{adp}}$	70	adaptive phase budget.
$n_{\text{aEGO}}$	40	number of samples to optimize EI-FP at the beginning of the adaptive phase.
$n_{2\text{to}3}$	2	number of samples that should be within $r_{\text{CTF}}$ to start the switching phase, i.e., part 3 of the algorithm. Otherwise, CTF will be followed until the end (part 2), see Figure 6.7.
$n_{\text{CTF}}$	2	number of iterations to follow the CTF after $n_{2\text{to}3}$ is reached, then alternatively switch to optimizing EI-FP within $r_{\text{CTF}}$ for $n_{\text{const}}$ iterations, see Figure 6.7.
$n_{\text{const}}$	3	number of iterations to optimize EI-FP within $r_{\text{CTF}}$ after $n_{2\text{to}3}$ is reached, then switch to following the CTF for $n_{\text{CTF}}$ iterations alternatively, see Figure 6.7.
$d_{\text{CTF}}$	0.2	distance constraint among samples when following the CTF, obtained from $0.1 * \sqrt{\text{dimension}} = 0.2$
$d_{\text{const}}$	0.01	distance constraint among samples when optimizing the constraint problem via EI-FP.
others		see Table 6.1.

We solve this problem by the established constrained EGO and also by our proposed algorithm for both variants of this problem. In the case of the larger design space, the proposed algorithm is employed twice. Once by neglecting its first part; the established constrained EGO (the green part in Figure 6.7) and hence, considering only the new parts. In the second case, we solve this problem by considering the entire proposed algorithm; the results in Table 6.4 are only for this latter case. The constraint that we follow in our proposed method is the critical buckling load (i.e., CTF is  $g_6$  in Equation (6.9)). When we neglect the first part and keep the new parts of the proposed methods, there are repetitions without a single feasible solution in 70 adaptive samples, as Figure 6.9 shows. However, RF\_pseudo that employs the proposed SLCB-EI<sub>feas</sub> criterion performs much better than RF, which uses the established EI<sub>feas</sub> criterion. This is another example which shows that the SLCB-EI<sub>feas</sub> is more successful in finding small feasible regions. Here, the established EGO performs much better than the others, opposite of what we observe later in the I-beam design problem. Even when one neglects all repetitions that are entirely infeasible, either in the case of RF and/or RF\_pseudo (in total 22 repetitions), the established EGO still performs much better with the average solution of 1.82 against 4.08 for RF and 4.51 for RF\_pseudo. This is one of the examples that shows the established constraint EGO (EI-FP) is able to find the desired promising regions faster at the beginning, and hence why it is the first part of our proposed RF algorithm. This is important for the success of the second and third parts of RF, which need to concentrate on these promising regions.

**Table 6.4** The welded beam design problem comparison. NFE stands for the number of function evaluations. The first three references (12 algorithms, 19 algorithms and SCGO) have reported the version of this problem with the upper bound of two. The best known global optimum found by the author in the literature is 1.72485 (see Wang et al. (2018)), which is valid for the both versions of the problem.

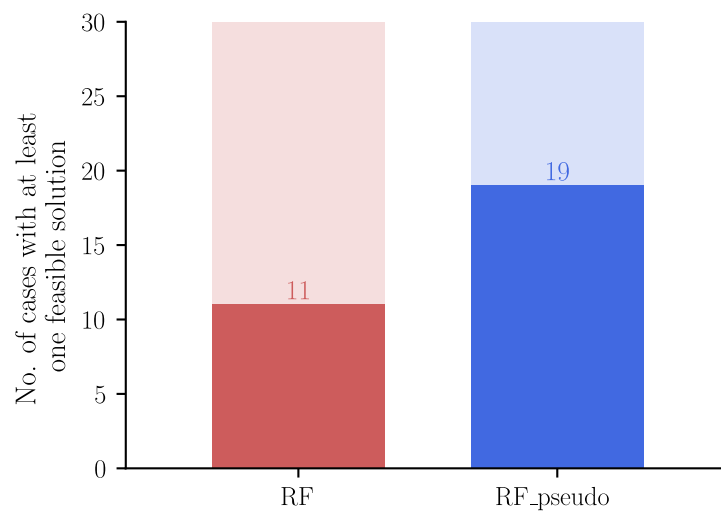
Algorithm	Best	Mean	NFE	Explanation
12 algorithms	1.72485	1.72485	20000	based on the Table 15 from (Wang et al., 2018), for the most efficient algorithm called DELC.
19 algorithms	1.72490	1.72763	10000	based on the Table 17 from Yalcin and Pekcan (2018) for the N2F algorithm, which has the best mean value in 10000 FE.
SCGO	1.756 - 1.859	-	628-695	results of different settings, (Parnianifard et al., 2020).
KCGO	2.32	-	115	(Li et al., 2017b).
SEGOKPLS	2.22	2.35	100	(Bouhleb et al., 2018), 30 repetitions.
EI-FP	1.73	7.7 $\rightarrow$ 1.91*	100	established constrained EGO. 30 repetitions.
RF	1.74	7.7 $\rightarrow$ 2.35*	100	proposed algorithm that employs EI <sub>feas</sub> . 30 repetitions.
RF_pseudo	1.76	7.7 $\rightarrow$ 2.27*	100	proposed algorithm that employs SLCB-EI <sub>feas</sub> . 30 repetitions.

\* The average value at the beginning of the adaptive phase is 7.7 and the number after the arrow is the final average value. Averaging is only based on the repetitions with at least one feasible solution in their initial phase.

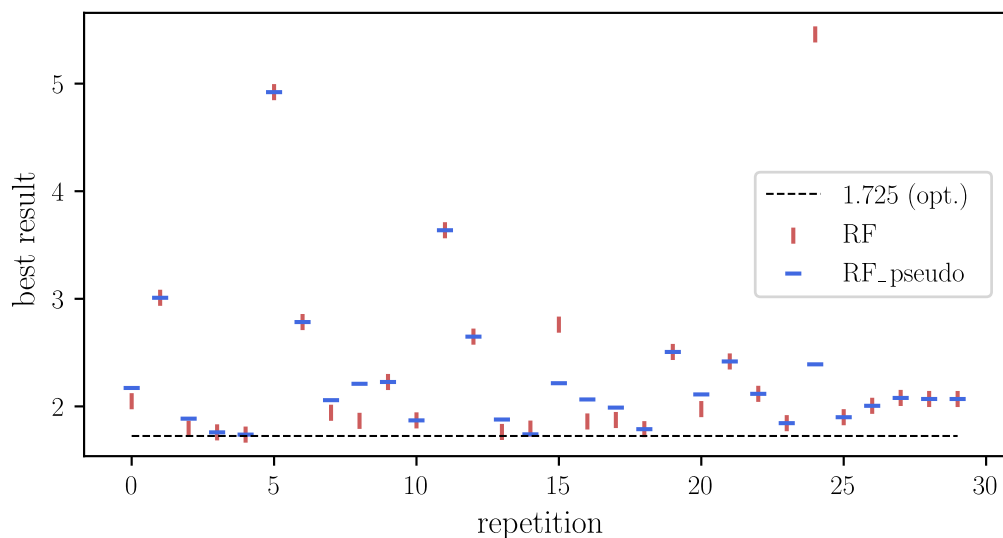
In the second run of solutions, the entire proposed algorithm is considered. Therefore, now, at the beginning of the adaptive phase, 40 iterations of the established constrained EGO are conducted initially and the rest of the budget (30 iterations) is consumed by the new parts of the algorithm. In comparison to the first run solutions (where the first part was neglected), the results are now considerably better for RF and RF\_pseudo. First, all of 30 repetitions have reached a feasible solution and second, as Table 6.4 shows, the mean of the best feasible solution has improved to 2.35 (from 4.08) and to 2.27 (from 4.51) for RF and RF\_pseudo respectively. Figure 6.10 shows individual results of the 30 repetitions for RF and RF\_pseudo. At the beginning of this example, we mentioned that there is a version of this problem that sets the upper bound to 2 instead of 10 for  $x_1$  and  $x_4$  with the percentage of feasible region at 2.69% compared to the current 0.1%. If we consider this case and run the proposed method without changing the other settings, then the results improve even further. Now, the mean of the best feasible solution (among 30 repetitions) of RF and RF\_pseudo is 1.819 and 1.822, respectively, while the established constrained EGO mean solution



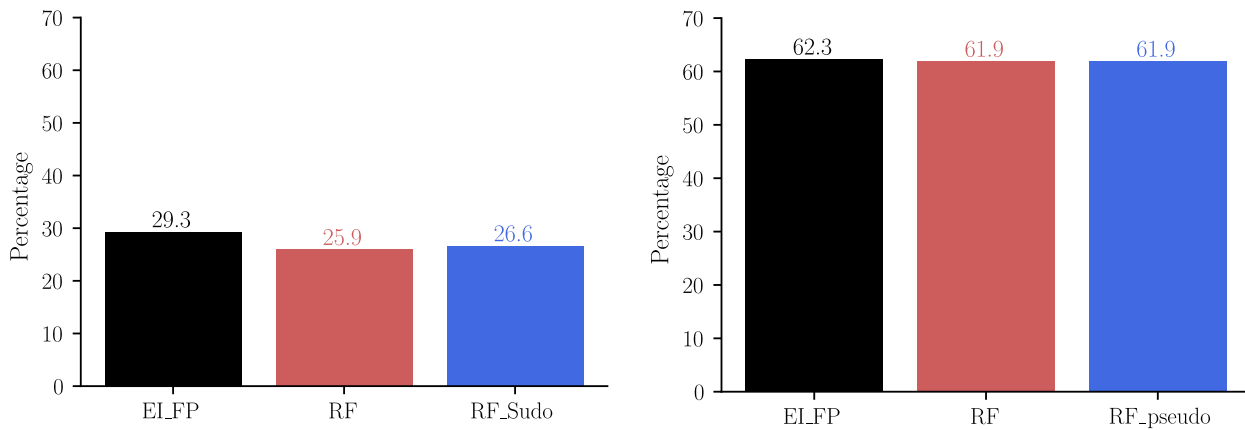
stands now at the impressive value of 1.753 (optimum is almost 1.725). However, the main aim of the proposed method was not to find the optimum, but more importantly, to have a possibility of several comparable designs within a certain desired range. In this regard, Figure 6.11 shows the solutions for both larger and smaller design space. It can be seen that the percentage of adaptive iterations in which there is at least one feasible solution within  $[1.724, 2.22]$  increases significantly for the smaller design space and the difference between the established constrained EGO and the proposed algorithms (that employs the established EGO at the beginning of its adaptive phase) becomes much smaller. This is an example that once suitable regions are found, the new steps of the proposed algorithms (parts 2 and 3 in Figure 6.7) do their intended job.



**Figure 6.9** Number of cases out of 30 repetitions in which at least one feasible solution is obtained. Each case includes 30 initial samples and 70 adaptive ones. The first part (i.e., the established constrained EGO) in both algorithms is neglected here.



**Figure 6.10** Best feasible results in all 30 repetitions of the welded beam design problem with settings given in Table 6.3 (results are for the version of the problem with the larger design space).



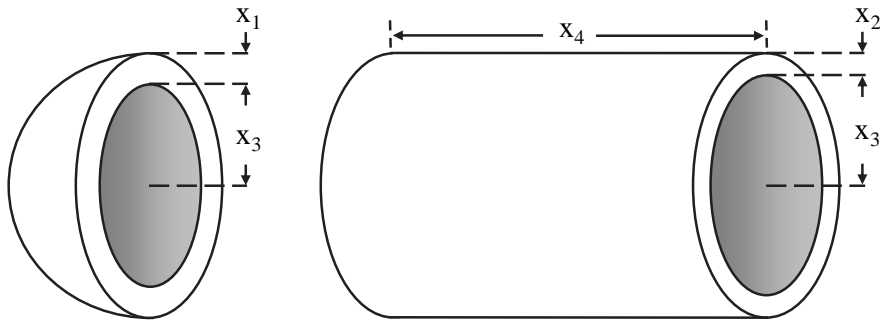
**Figure 6.11** Percentage of adaptive iterations in all 30 repetitions in which there is at least one feasible solution within  $[1.724, 2.22]$ . In total, there are 2100 adaptive iterations, considering all 30 repetitions ( $30 * 70 = 2100$ ). 1.72485 is the best known optimum to the author of this work, as mentioned earlier. 2.22 is the best result among the response surface methods from the literature that were reviewed here, see Table 6.4. The left figure is for the version of this problem where the design space is larger (27 times) and the right figure is for the smaller case.

All RS-based algorithms in Table 6.4 except the SCGO, consider the more difficult version of this problem. The best competitor among them is SEGOKPLS algorithm, which is a modification of EGO, considering partial least square and employing principal component analysis for dimension reduction. The constraints are handled when optimizing the infill criterion. Dimension reduction is an effective method that was also used in the first chapter of this work, and good performance by this algorithm with just 100 evaluations is a confirmation of its efficiency. The mean and best result in 30 repetitions are close to each other (2.35 and 2.22 respectively); while our proposed methods (RF and RF\_pseudo) have better best results, and at least the same mean results compared to SEGOKPLS. This perhaps indicates, in cases such as this problem where feasible space is sparse, removal of information by dimension reduction may prevent exploring some promising regions and hence, some good results including the optimum remain hidden even after 30 repetitions. However, this explanation requires more investigations in the future. A glance over Table 6.4 shows how highly effective the established constrained EGO in this problem is. Not only its best result is better than those of the other RS-based methods, but also its mean value is considerably lower than for the others. In comparison to the first three references in the table, which solve the easy version of this problem, established EGO has a mean value of 1.753. Although this value is more than reported for the first two references (1.72485 and 1.72763), the difference is small while the established EGO has only 100 evaluations vs. 10,000 and 20,000 evaluations of the other two. This is consistent with the experience that RS-based methods do not provide pinpoint accuracy, but in return, the number of function evaluations to reach close to the optimum is smaller. Note that, we limited the budget to 100 evaluations to compare our method with SEGOKPLS (one of the best as mentioned earlier). This is given the fact that our proposed method not only looks for the optimum, but also spends part of its budget for finding several good designs.

### 6.6.2. A pressure vessel design problem

A cylindrical pressure vessel containing compressed air of 3000 psi has hemispherical caps. The design is based on the ASME boiler and pressure vessel code with the minimum volume of 750 ft<sup>3</sup>. The objective ( $f(\mathbf{x})$ ) is to minimize the total cost (\$) including the cost of material, forming and welding. There are four design variables as shown in Figure 6.12 and with the unit of inch: The thickness of the cap ( $x_1$ ), thickness of the cylindrical part ( $x_2$ ), inner radius ( $x_3$ ) and length of the cylindrical part ( $x_4$ ), see Sandgren (1990) for more details. The objective and constraints are (Coello Coello and Mezura Montes, 2002):

$$\begin{aligned}
 &\underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) = 0.6224x_1x_3x_4 + 1.7781x_2x_3^2 + 3.1661x_1^2x_4 + 19.84x_1^2x_3 \\
 &\text{subject to} && g_1(\mathbf{x}) = -x_1 + 0.0193x_3 \leq 0, \quad g_2(\mathbf{x}) = -x_2 + 0.00954x_3 \leq 0, \\
 &&& g_3(\mathbf{x}) = \frac{-\pi x_3^2 x_4^2 - \frac{4}{3}\pi x_3^3}{750 \times 1728} + 1 \leq 0, \quad g_4(\mathbf{x}) = x_4 - 240 \leq 0, \\
 &&& 0.0625 \leq x_{1,2} \leq 6.1875, \quad 10 \leq x_{3,4} \leq 240.
 \end{aligned} \tag{6.10}$$



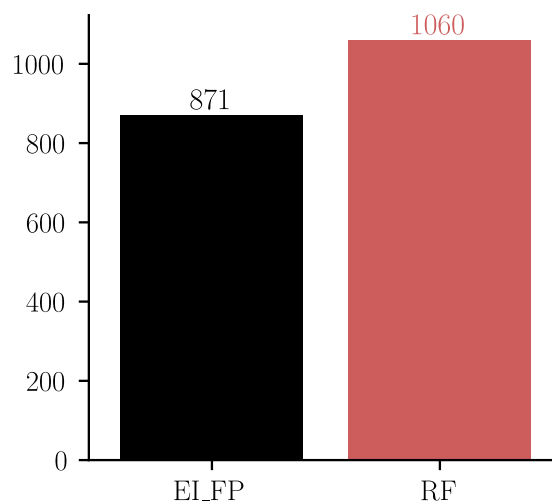
**Figure 6.12** Depiction of the considered pressure vessel design problem based on Cagnina et al. (2008).

**Table 6.5** Settings of the RF method for the pressure vessel design problem.

Parameter	Value / Type	Explanation
CTF	$g_1$ in Eq. (6.10)	the constraint to be followed in the desired range $r_{CTF}$ .
$r_{CTF}$	[-0.05,0]	desired range of the selected constraint to be followed.
$n_{init}$	20	initial phase budget using Latin Hypercube.
$n_{adp}$	80	adaptive phase budget.
$n_{aEGO}$	20	see Table 6.3.
$n_{2to3}$	4	see Table 6.3.
$n_{CTF}$	1	see Table 6.3.
$n_{const}$	3	see Table 6.3.
others		see Table 6.3.

In the literature, the upper bound of  $x_4$  has also been considered as 200 (in) instead of 240, see e.g., Cagnina et al. (2008). A comparison of some optimizers' performance for both versions of this

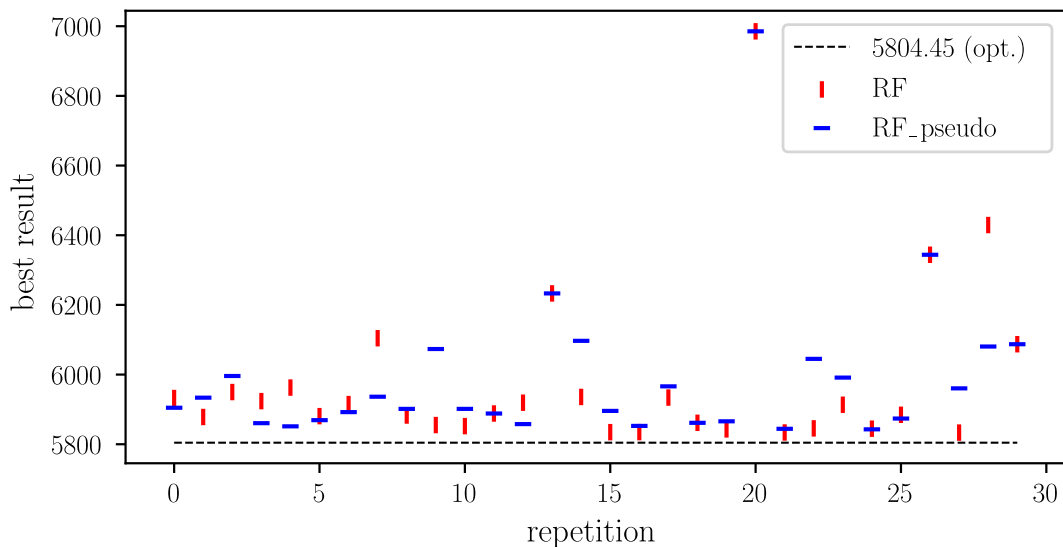
problem is given in Woldemichael and Woldeyohannes (2016), although the number of function evaluations is not reported. In this work, we consider the upper bound of  $x_4$  to be 240 (in) as mentioned in Equation (6.10). Another point is that this problem is solved in the literature based on considering all design variables to be continuous or the first two to be discrete. In the case of the latter, the optimum will be 6059.7 \$ according to Gandomi et al. (2013), while for the continuous case the best known feasible solution that the author of this work has found is 5804.45 \$, see Bouhleb et al. (2018). Table 6.6 compares the results in this work and others from the literature. A short glance shows that this problem is not as easy as it may initially seem for many optimizers. In our settings, the first part of the RF contains 20 adaptive samples and then the new parts (following a value or optimizing in a range around this value) are conducted for a maximum of 60 iterations, if there are already enough samples in the desired range. This helps the new parts to do their job better compared to the case that the entire adaptive budget is spent only on these new parts. Table 6.6 shows that the proposed RF algorithm is comparable to the established constrained EGO using EI-FP infill criterion, while it is slightly better on average and in return EGO using EI-FP is slightly better in the best result among all 30 repetitions. However, as Figure 6.13 indicates, the RF algorithm has more iterations with at least one feasible sample with very good objective values so far. In this example, the global optimum occurs when  $g_1$  is active, so like for the previous problem, the proposed method is successful in both, optimizing and finding several results in the desired region. In addition, the feasibility comparison over all 30 repetitions in Figure 6.15 shows that in RF, feasible results happen across all repetitions more or less similarly, but the performance of the EI-FP is much less uniform in that sense; in a few repetitions, it tends to stay in feasible regions for many iterations while much less in other repetitions. Yet, this figure shows, EI-FP can find good feasible solutions in early iterations, as is the case in the previous example. Note that RF\_pseudo has also quite similar results to that of RF, but it is not shown here.

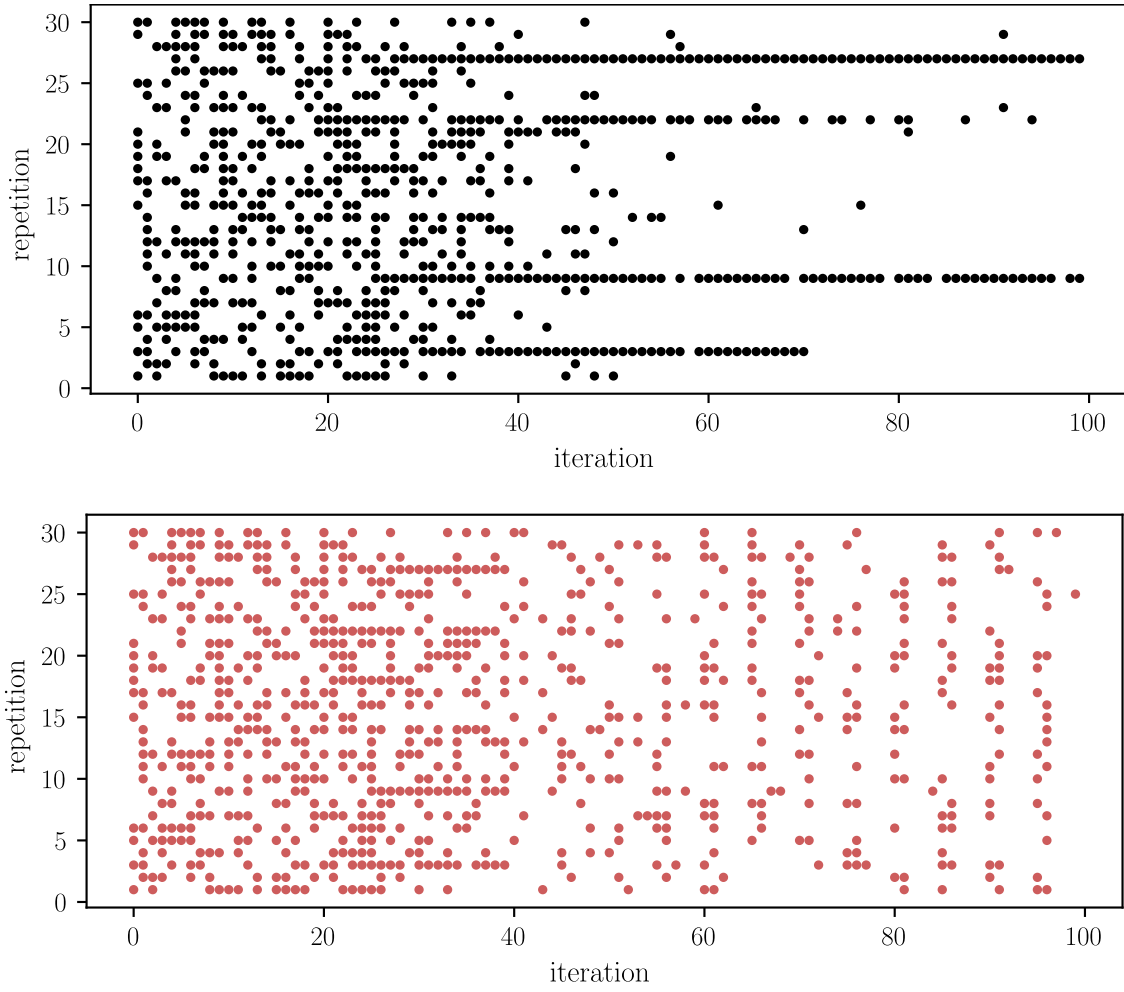


**Figure 6.13** Number of adaptive iterations in all 30 repetitions in which there is so far at least one feasible solution within [5804, 6000]. There are 2400 adaptive iterations in total considering all 30 repetitions ( $30 * 80 = 2400$ ). 5804.45 is the best known optimum to the author of this work, as mentioned earlier.

**Table 6.6** Comparison of different algorithms for the pressure vessel design problem.

Algorithm	Best	Mean	NFE	Explanation
FSA	5868.6	6164.6	108883	FSA is a simulated annealing developed for constrained problems. 30 repetitions, (Hedar and Fukushima, 2006).
PSOstr	6226.8	6341.6	3736	based on Table 13 of Dimopoulos (2007). PSOstr is a hybrid PSO algorithm. 100 repetitions.
Over 40	6059.1-8129.1	6059.7-9032.5	-	Based on Table 5 of Gandomi et al. (2013). Number of function evaluations from tens to hundreds of thousands at least in some references, e.g., (He and Wang, 2007).
SEGOKPLS	5848.7	5960.5	100	(Bouhleb et al., 2018), 30 repetitions.
EI-FP	5810.7	6025.8	100	established constrained EGO. 30 repetitions.
RF	5833.4	5980.7	100	proposed algorithm that employs $EI_{feas}$ . 30 repetitions.
RF_pseudo	5842.8	5989.8	100	proposed algorithm that employs $SLCB-EI_{feas}$ . 30 repetitions.

**Figure 6.14** Best feasible results in all 30 repetitions. 5804.45 is considered as the best known optimum in this work.

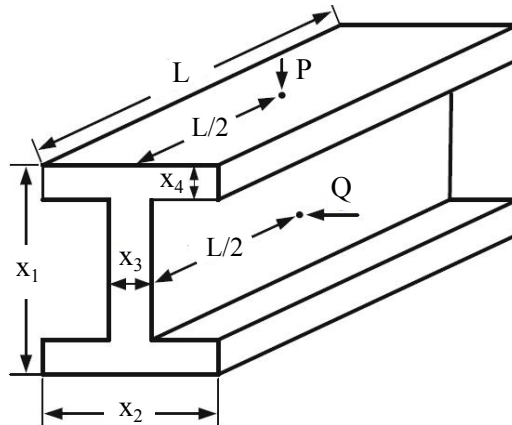


**Figure 6.15** Depiction of feasible cases in each iteration and repetition by a circle. A blank space means at least one of the constraints was violated. The top figure represents EI-FP and bottom RF, accordingly.

### 6.6.3. An I-beam design problem

We want to minimize the vertical deflection of an I-beam shown in Figure 6.16, while the cross-sectional area should be less than  $300 \text{ cm}^2$  ( $g_1$ ) and stress less than  $6 \text{ kN/cm}^2$  ( $g_2$ ) (Ye et al., 2018).

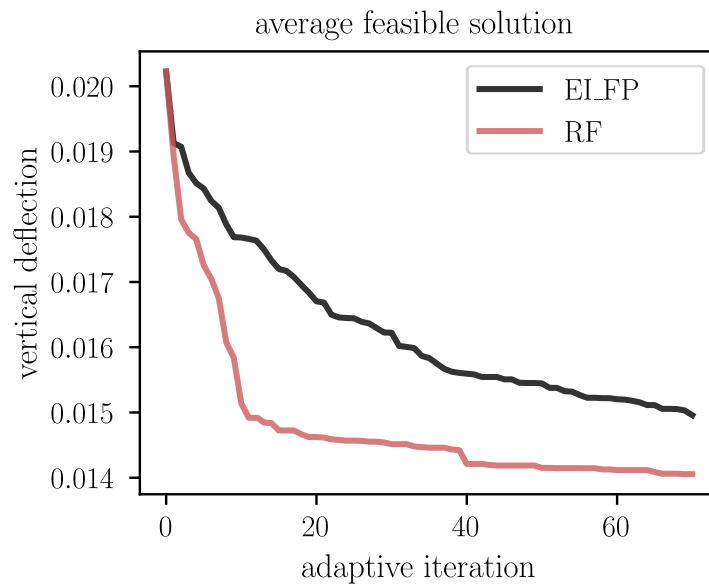
$$\begin{aligned}
 &\text{minimize}_{\mathbf{x}} \quad f(\mathbf{x}) = \frac{5000}{\frac{1}{12}x_3(x_1 - 2x_4)^3 + \frac{1}{6}x_2x_4^3 + 2x_2x_4\left(\frac{x_1 - x_4}{2}\right)^2} \\
 &\text{subject to} \quad g_1(\mathbf{x}) = \frac{2x_2x_4 + x_3(x_1 - 2x_4)}{300} - 1 \leq 0, \\
 &\quad \quad \quad g_2(\mathbf{x}) = \frac{180000x_1/6}{x_3(x_1 - 2x_4)^3 + 2x_2x_4(4x_4^2 + 3x_1(x_1 - 2x_4))} + \frac{15000x_2/6}{(x_1 - 2x_4)x_3^3 + 2x_4x_2^3} - 1 \leq 0, \\
 &\quad \quad \quad 10 \leq x_1 \leq 80, \quad 10 \leq x_2 \leq 50, \quad 0.9 \leq x_{3,4} \leq 5, \quad x_i \text{ unit : cm}, \\
 &\text{where} \quad P = 600 \text{ kN}, \quad Q = 50 \text{ kN}, \quad L = 200 \text{ cm}, \quad E = 20000 \text{ kN/cm}^2.
 \end{aligned} \tag{6.11}$$



**Figure 6.16** Schematic of the considered simply supported I-beam design problem based on Ye et al. (2018).

**Table 6.7** Settings for the I-beam design problem in the proposed algorithm RF.

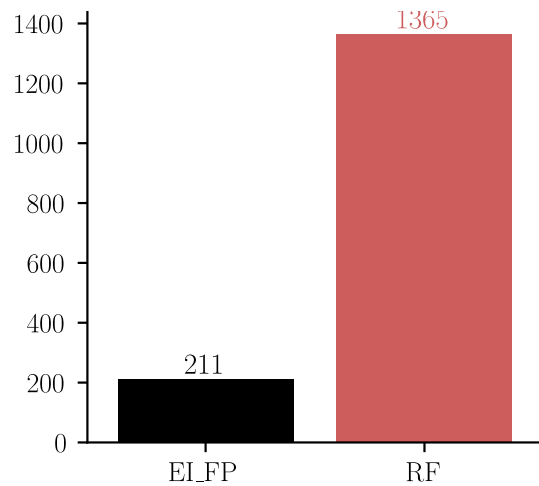
Parameter	Value / Type	Explanation
CTF	$g_1$ in Eq. (6.11)	the constraint to be followed in the desired range $r_{CTF}$ .
$r_{CTF}$	[-0.05,0]	the desired range of the selected constraint to be followed.
$n_{init}$	30	initial phase budget using Latin Hypercube.
$n_{adp}$	70	adaptive phase budget.
$n_{aEGO}$	0	see Table 6.3.
$n_{2to3}$	4	see Table 6.3.
$n_{CTF}$	1	see Table 6.3.
$n_{const}$	3	see Table 6.3.
others		see Table 6.3.



**Figure 6.17** Average feasible solution of the I-beam design problem (Equation (6.11)) with 30 repetitions.

**Table 6.8** Comparison of different algorithms for the I-beam problem.

Algorithm	Best	Mean	NFE	Explanation
FPA	0.0131	0.0270	1323	Flower Pollination algorithm, see Table 3 of Bekdaş et al. (2018). For 5000 evaluations, all repetitions find the optimum, see Yalcin and Pekcan (2020).
ESGO-HSR	0.0132	135.9	400	based on Table 8 of Ye et al. (2018). 10 repetitions.
EI-FP	0.0139	0.0149	100	established constrained EGO. 30 repetitions.
RF	0.0131	0.0140	100	proposed algorithm that employs $EI_{feas}$ . 30 repetitions.
RF_pseudo	0.0133	0.0143	100	proposed algorithm that employs SLCB- $EI_{feas}$ . 30 repetitions.



**Figure 6.18** Number of adaptive iterations in all 30 repetitions, in which there is at least one feasible solution within  $[0.01307, 0.0145]$ . There are 2400 adaptive iterations in total considering all 30 repetitions ( $30 * 80 = 2400$ ). 0.01307 is the best known optimum to the author of this work. 0.0145 is just a number close to 0.01307, and this range  $[0.01307, 0.0145]$  is selected to evaluate how well the RF method follows the desired range.

The best known global optimum that the author of this work has found in the literature is 0.01307 (Nigdeli et al., 2016), at which the first constraint (i.e., cross-section) is active. In this problem we consider only the new parts of the RF and RF\_pseudo algorithms and neglect the initial part, i.e., no established EGO iteration is done at the beginning ( $n_{aEGO} = 0$ ). Figure 6.17 depicts the average results of 30 repetitions for the considered methods. Table 6.8 shows the results compared to those of some other algorithms in the literature. For example, ESGO-HSR is one of the best algorithms in the literature, which is based on an ensemble of surrogate models in addition to a space reduction strategy (Ye et al., 2018). A simple comparison to this algorithm shows how competitive even the established EI-FP algorithm is, for just 100 evaluations, while, the proposed RF algorithm improves significantly further over EI-FP. Across all 30 repetitions, RF provides also much more iterations with at least a very good feasible solution, as shown in Figure 6.18. RF\_pseudo performs slightly worse than RF, as it naturally tends to explore more and is hence less exploitive. These new



stages considerably improve the optimization results in this problem. Although, the main goal was having a couple of good solutions in the desired region and not necessarily finding the optimum per se. One important contributing factor is that CTF is actually active at the global optimum.

#### 6.6.4. Crash simulation - Solution space optimization

In this section, we compare the proposed RF method and the established constrained EGO (using the EI-FP criterion) w.r.t. two optimization problems obtained from the solution space method (Zimmermann and von Hoessle, 2013). This method is used in the early development phase of a car as a simplified model and it is explained briefly in Appendix C. The solution space enables different development departments to change the design of a component without interfering with each other as long as the forces in the components remain within the method's defined corridors. Here, we are not interested in this method per se, but as a practical case to demonstrate the performance of the proposed algorithm. In this section, we optimize two components to fit their force within the solution space corridors. These problems are designed to be two-dimensional to show how the proposed method is different from EI-FP and at the same time to reveal how EI-FP puts samples based on its approximation of the combination of objective and constraints. The results are obtained with the help of Lukas Burmberger who did his Master's thesis under the supervision of the author of this work (Burmberger, 2020). The codes used in the previous parts of this chapter were provided to him and he set the objective and constraints based on his work in the solution space method.

Our aim is to optimize the components 4 and 7 while the developed forces should be within the corridors (bounds) over certain deformation values defined by the solution space method, see Figure 6.19. Here, each component is divided into several sections, where force corridors can vary between sections, while the displacement between the beginning and the end of each section is usually the same among all sections, unless, where another load path starts in parallel. Optimization should be conducted using the full vehicle, which is computationally expensive. Hence, instead, it is done based on the drop tower test, where the component is fixed on one side and impacted from the other side by a moving plate. Here, the focus is on the solution of the final optimization problem, which is defined by:

$$\underset{\mathbf{x}}{\text{minimize}} \quad \text{weighted mass}$$

for each section 'i' in the component, subject to:

$$g_{1,i}(\mathbf{x}) = F_{x,i}/F_i^{\text{upper limit}} - 1 \leq 0,$$

$$g_{2,i}(\mathbf{x}) = -F_{x,i}/F_i^{\text{lower limit}} + 1 \leq 0,$$

$$\mathbf{x} = [m_{\text{imp}}, t_1], \quad h(\mathbf{x}) = u_{x,\text{total}}/u_{x,\text{total}}^{\text{target}} - 1 = 0,$$

$$0 \leq m_{\text{imp}} \leq 0.0369 \text{ ton}, \quad 0.5 \leq t_1 \leq 4.5 \text{ mm},$$

$$\text{impactor initial velocity : } 15,600 \text{ mm/s, units: kN, mm, s, ton.}$$

(6.12)

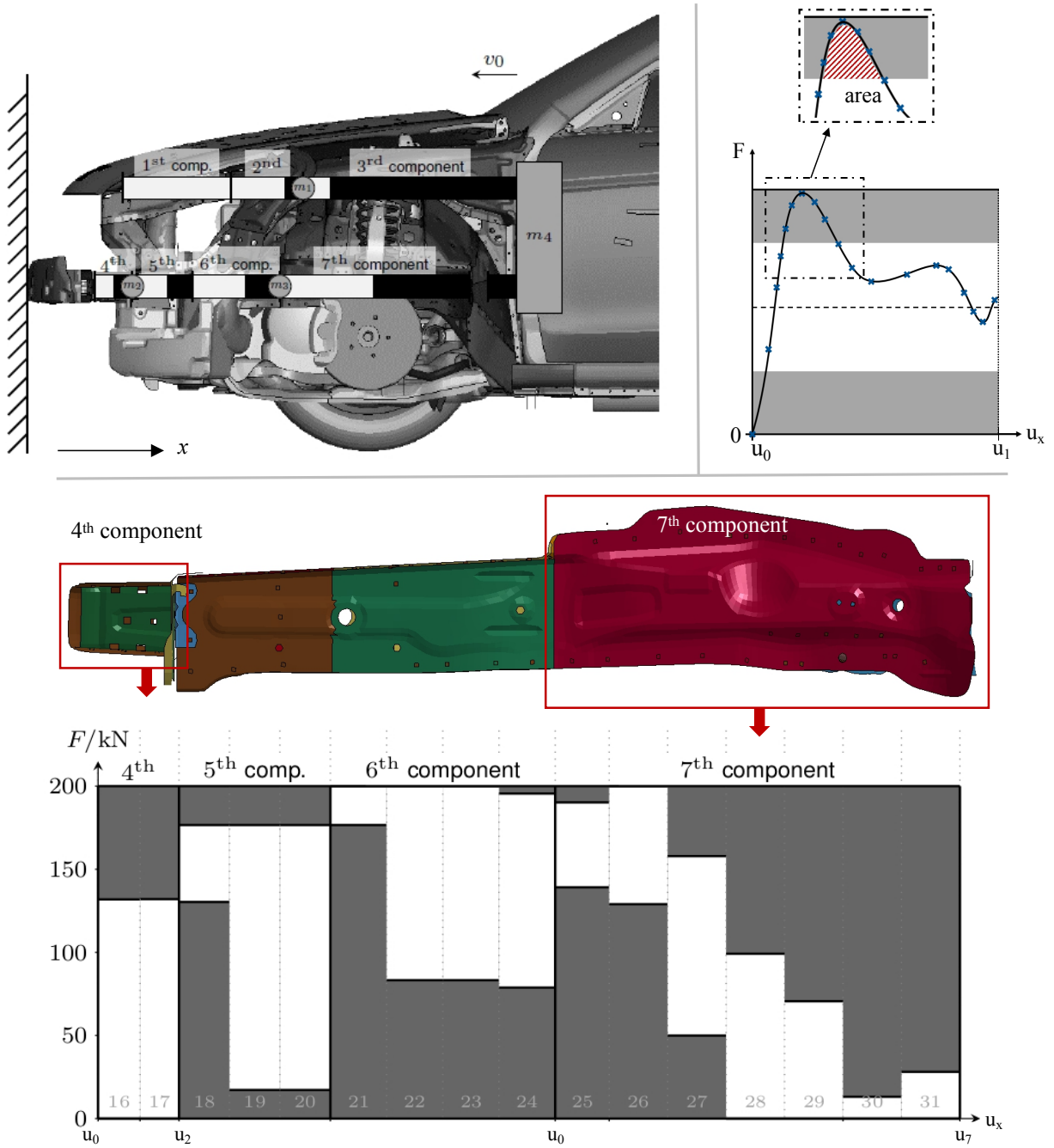
where the objective is "weighted mass", which means that the mass of the component is weighted in a way to encourage the component force to get closer to the center of the corridor suggested by the solution space.  $u_{x,\text{total}}$  is the total deformation length of the component and  $F_{x,i}$  is the obtained simulation force within the section 'i' of the component. This force should be within the corresponding corridors, i.e., constraints  $g_1$  and  $g_2$  in Equation (6.12) should be feasible. However, in practice, these two constraints do not directly calculate the force that should be within the corridor bounds, but instead represent the violation of the corridors based on how much the area under the force curve is located outside these corridor bounds, see the top right image in Figure 6.19. In addition, a small amount of violation is also allowed. Therefore, to reiterate, the definitions of  $g_1$  and  $g_2$  are symbolic representations of the force feasibility, while we ensure this feasibility through the area of the force curve and not the value of the force directly. Table 6.9 reports the upper and lower limits of force in each section and the total deformation of each component that is required for optimization in Equation (6.12). However, since optimization is done via the drop tower test instead of directly optimizing the component within the car, the force limits and total deformation are further calibrated, the details of which are beyond the aim of this chapter, see Burmberger (2020) for more information. Design variables are the impactor mass ( $m_{\text{imp}}$ ) and the thickness of the component ( $t_1$ ). The constraint on deformation ( $h(\mathbf{x})$ ) is an equality one, which originates from the solution space assumptions. To apply an equality constraint, we follow the common procedure of considering two inequality constraints with a relaxation (see, e.g., Sasena (2002), p. 119). In addition, we take the square distance to reduce the number of constraints to one, i.e., the replacement constraint for the equality constraint in Equation (6.12) is,

$$h(\mathbf{x}) = \frac{(u_x - u_x^{\text{target}})^2}{(\beta u_x^{\text{target}})^2} - 1 \leq 0, \quad \beta = 0.05, \quad (6.13)$$

where a relaxation of  $\beta = 0.05$  is considered on each side. This handling of equality constraint is used for the case that we solve the problems using the established constrained EGO. For the RF algorithm, the constraint that should be followed is naturally the equality constraint (for deformation). Settings for the RF algorithm are given in Table 6.10. First, we evaluate the results of optimizing the 4<sup>th</sup> component (i.e., crash box, see Figure 6.19).

**Table 6.9** Force corridors' bounds and total deformation for components 4 and 7 based on the solution space, see Figure 6.19 and Equation (6.12). It should be noted that here, forces are half of what is depicted in Figure 6.19, since components 4 and 7 of only one side of the car are considered and not both.

Component	4		7						
	1	2	1	2	3	4	5	6	7
$F_i^{\text{lower limit}}$ (kN)	0	0	70	65	25	0	0	0	0
$F_i^{\text{upper limit}}$ (kN)	67	66	95	100	80	50	37.5	7.5	12.5
$u_{x,\text{total}}^{\text{target}}$ (mm)	46.3		202.7						



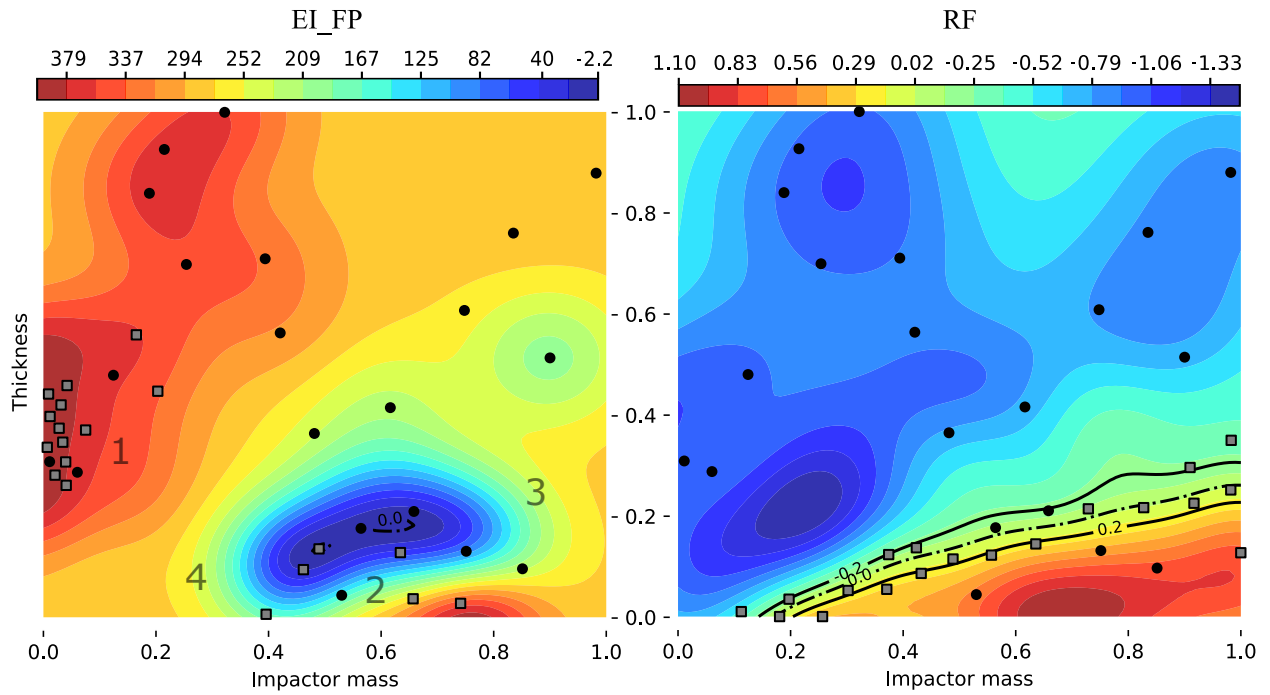
**Figure 6.19** Top left figure shows the side view of the frontal part of the Honda Accord model from (NHTSA, Last access February 2020). Numbered components show the two load paths including components 1 to 3 and 4 to 7, respectively. The middle figure shows the lower path members in a different view, especially the 4<sup>th</sup> and 7<sup>th</sup> components that are optimized here. The lowest figure shows the force corridors obtained from the solution space for this lower path (Daub, 2020). The top right figure shows how to evaluate violation of the force corridors by calculating the violated area under the force curve as suggested in Burmberger (2020).

**Table 6.10** Settings of the RF algorithm to optimize components 4 and 7 in Figure 6.19.

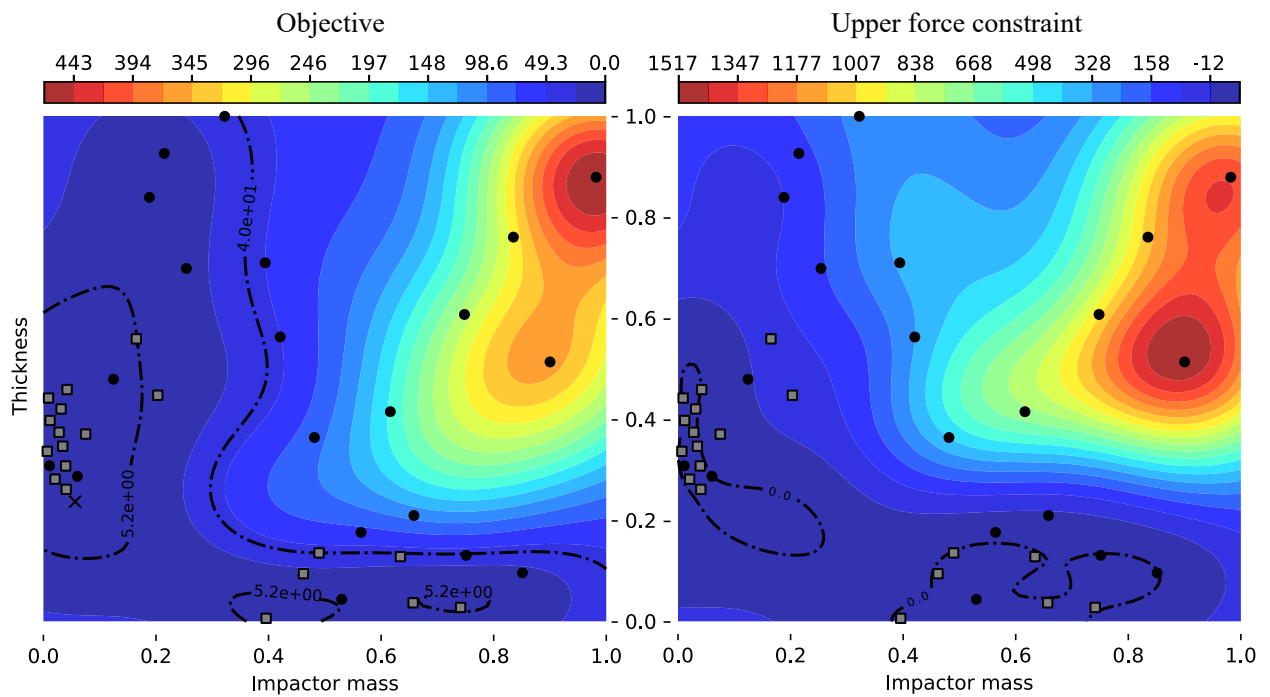
Parameter	Value / Type	Explanation
CTF	$h(\mathbf{x})$ in Eq. (6.12)	the constraint to be followed in the desired range $r_{CTF}$ .
$r_{CTF}$	[-0.2,0.2]	desired range of the selected constraint to be followed.
$n_{init}$	20	initial phase budget using Latin Hypercube.

$n_{\text{adp}}$	20	adaptive phase budget.
$n_{\text{aEGO}}$	0	see Table 6.3.
$n_{2\text{to}3}$	3	see Table 6.3.
$n_{\text{CTF}}$	1	see Table 6.3.
$n_{\text{const}}$	2	see Table 6.3.
$d_{\text{CTF}}$	$0.05 * \sqrt{2}$	see Table 6.3.
$d_{\text{const}}$	$0.01 * \sqrt{2}$	see Table 6.3.

Figure 6.20 shows the considered constraint for deformation (Equation (6.13)) at the end of the forty evaluations. The left figure is the solution of the established EI-FP algorithm and the right figure shows that of the RF, while the adaptive samples are shown in gray squares. Even at first glance, the results show different patterns. The RF method places samples within the considered bound of  $[-0.2, 0.2]$  as requested, while the established method concentrates on two regions, mainly close to the left side of the design space, where the deformation constraint is largely violated (roughly twice the limit of 44.9 mm). As mentioned in the literature review, EI-FP can remain for many iterations in part of the space where constraints are violated, especially when there are several constraints that can cancel each other's feasible regions and make the problem even more multi-modal. With a larger number of samples, the situation can improve, but there is no guarantee; as the example at the beginning of this chapter has shown, so does the current example, where the number of samples is large enough for a typical EGO in two dimensions. Here, the imbalanced representation of the underlying function and more importantly, the combined effect of constraints and the objective lead to the obtained result by the established EGO. To better explain, deformation is not the only factor, but there are other ones, such as constraints on the force and also the objective function that should be minimized. What the established method (EI-FP) considers is the combined effect. Here, the constraint representing the deformation is largely violated (region 1, Figure 6.20 right); but, on the other hand, as Figure 6.21 shows, the constraint representing the upper bound of the force has a high probability of feasibility and in addition, the objective becomes quite low in this region (1), which both are desirable in minimization. Another important point is that, at other locations, a violation of the upper bound of the force can be even worse than that of the deformation. So, by placing samples in region 1, the established method has actually opted for the case that has fewer violations in total. However, it could have potentially placed a couple of samples in regions 3 and 4 (see Figure 6.20), as other rather suitable candidate locations. But here, the imbalanced representation of the underlying function, i.e., lack of samples around these suitable regions while there are several at other locations, makes regions 3 and 4 to remain hidden and be explored later. In this problem, the lower bound of the force constraint is always feasible, i.e., the probability of feasibility is one over the entire design space in all iterations.

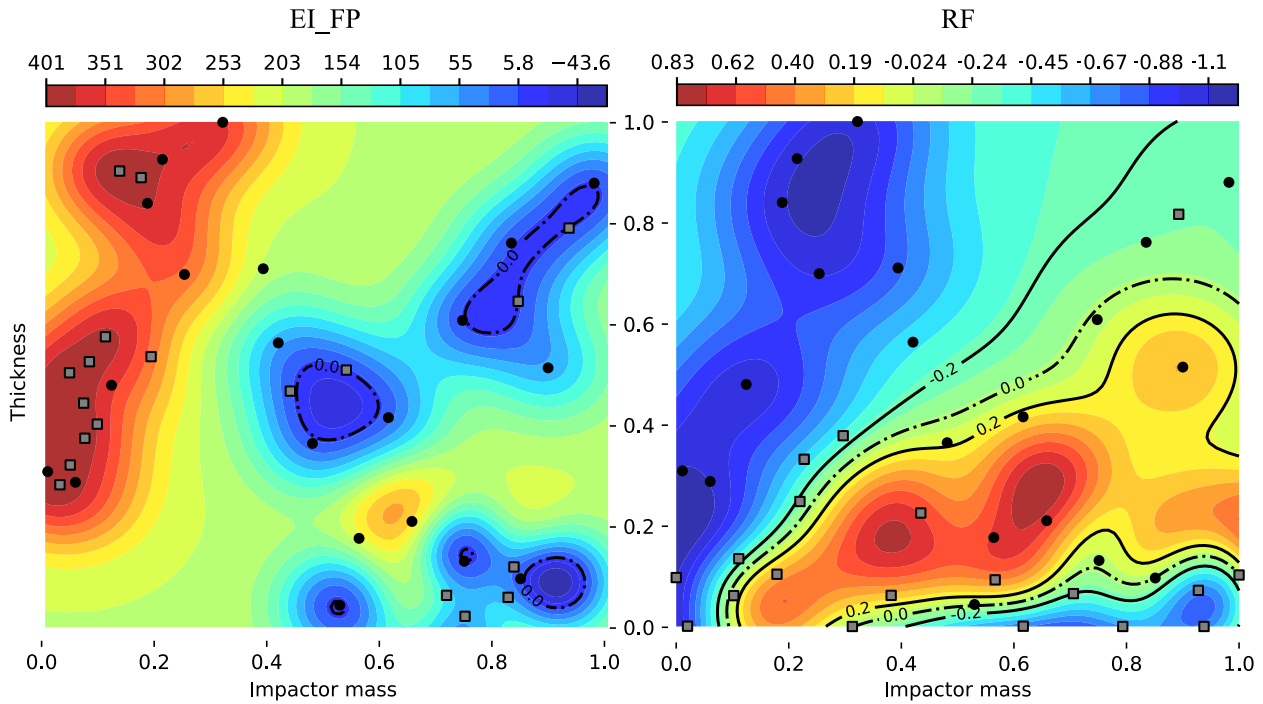


**Figure 6.20** Component 4 deformation at the end of the 40 simulations. The left figure shows the results of the established algorithm and the right one that of the RF algorithm. The initial phase samples are shown in black circles and those of the adaptive phase in the gray squares. The large numbers 1-4 in the left figure refer to their surrounding regions in the discussion.

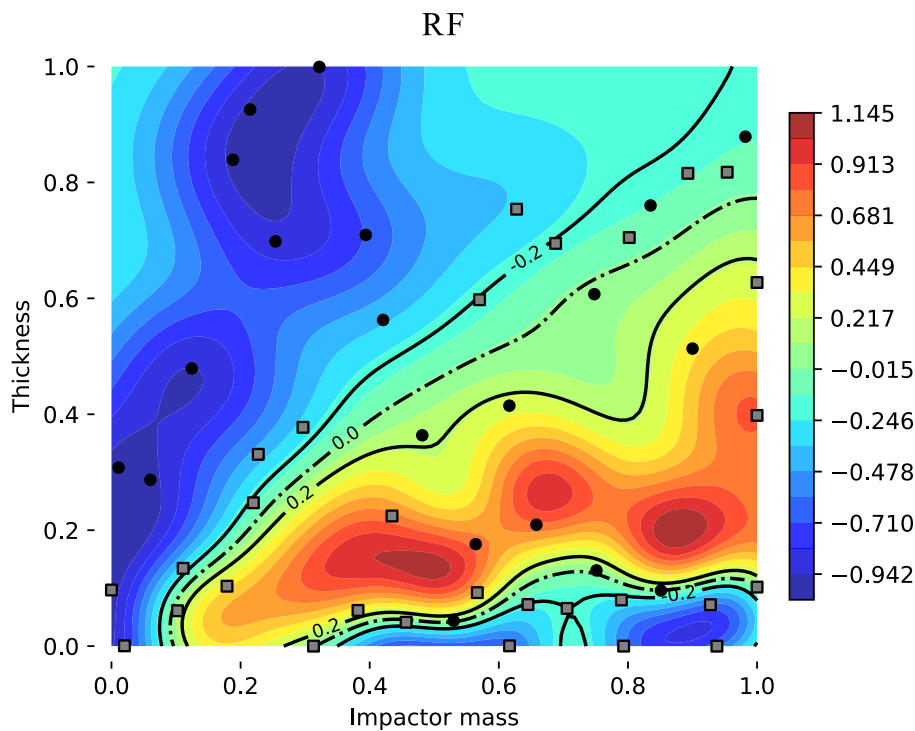


**Figure 6.21** The left figure shows the objective and the right figure the constraint representing the upper bound of the force corridor of the 4<sup>th</sup> component when optimizing with the established method.

The results for component 7 also show a similar trend. The values of the deformation constraint after 40 simulations are depicted in Figure 6.22. Here, the established method again considers



**Figure 6.22** Component 7 deformation at the end of the 40 simulations. The rest of the explanation is the same as that of Figure 6.20.



**Figure 6.23** Component 7 deformation at the end of the 50 simulations. The rest of the explanation is the same as that of Figure 6.20.

all effects together, while the suggested method focuses mainly on the deformation constraints and tries to find the minimum within the predefined bound of  $[-0.2, 0.2]$  for this constraint, which is originally an equality constraint. By running 10 extra adaptive samples, the RF method still

follows its guideline, as shown in Figure 6.23 by refining the boundary of the desired range and also placing some samples within it.

## 6.7. Discussion

In the first part of this chapter, we proposed a criterion to follow a certain value of a function faster than the  $EI_{feas}$  criterion, which tends to put samples very close to each other. This more exploratory behavior is obtained at the expense of less local accuracy. The proposed criterion  $SLCB-EI_{feas}$  identified even the small locus in the test function "6 - Gomez3" in significantly fewer iterations than  $EI_{feas}$ , showing its ability to properly explore and not just to search randomly. This was again confirmed in the welded beam design problem, where the total feasible region is a very small fraction of the entire space. However, the boundaries of these subregions that we are following can still be large themselves for a limited budget. The concept of introducing a pseudo-error measure may also be helpful for an ensemble of methods where the considered response surfaces (RSs) lack such a measure. We proposed our ensemble in Chapter 4 and as already mentioned, an error measure is required for exploration. But, if the involved surrogates in an ensemble do not provide one, it is suggested to import error measures from other RSs. As an example, in Viana and Haftka (2009) and Viana et al. (2013) the authors try to import the error estimation from one RS (e.g., Kriging) to another (e.g., SVM). Their proposed new measure relates the variance of Kriging to some error measure of SVM, for example, via cross-validation. However, this way of importing the prediction uncertainty requires a considerable amount of samples and according to the authors, the involved RSs should not be too different from each other. To elaborate further, variance in Kriging is the error estimation of the prediction and nothing else. A simple change of a hyperparameter can completely change both (mean) prediction and its variance. Therefore, if an adaptive optimization based on an SVM prediction is performed while the variance is taken from a GPR model, efficiency can (easily) be lost. This mismatched combination can remain in sub-optimal regions, as the experience of the author of this work shows. These references in the literature that import the error measure may additionally consider distance constraints, perhaps to alleviate such stagnation in sub-optimal regions as one of their functionalities. However, contrary to our proposed pseudo-error measure, a distance constraint does not guide the search to new promising regions automatically, since it is just a passive preventive measure. Indeed, the relation between a prediction and its error measure is essential for the efficiency of EGO. In a GPR, a low variance can be an indication of a smooth surface that does not need to be explored in a very detailed manner, which translates into enhanced efficiency. If this smoothness actually happens in the underlying function or not is another discussion, which largely depends on how current samples are unbiased representatives.

In the literature review, a couple of ways of handling constraints within EGO were summarized, including the EI-FP criterion. As mentioned, although this is a good criterion, there are cases where it does not perform well. E.g., by remaining in sub-optimal regions when there are several constraints, as they may create more multi-modality and/or cancel each other out (since zero

or very low numbers are multiplied to each other). One suggested way from the literature was modifying EI-FP by changing the definition of feasibility probability to further relax it. By doing so, the optimizer is given an opportunity to search more and not get lost easily, when several constraints have contradictory feasible regions. However, for less severe cases, this may lead to less efficient performance, as regions that should not actually be explored are now set artificially feasible. Here, we tried to see the known drawbacks of EI-FP from another point. Having several contradictory constraints can happen by nature and independently of the selected algorithm and so does a rather deceptive representation of the underlying function caused by sampling (and not the algorithm). By increasing the number of constraints, these drawbacks can potentially become even more problematic. Here, we tried to accept the facts that each of the constraints imposes self-limitation, and one can only afford a small number of evaluations (so, information) to counteract. Hence, as a possible solution, sometimes it is better to change the approach of how and in what order we decide to include these functions into an optimization. In some cases, it may be better to concentrate on one of the constraints, especially if an important one exists and then check the veracity of the rest altogether. In this way, we avoid wandering around due to considering demands of all constraints at the same time and just focus on the one that is interesting for us in the desired region, while checking the rest of the demands within this narrowed region. So, we concentrate the limited budget to gain more reliable information where it actually matters. An optimizer can handle various problems more decently if it is composed of several complementary criteria/procedures and has efficient means to decide to employ which criterion when. This is our practical interpretation of the NFLT. Here, we try to realize this to some extent in the proposed RF algorithm. First, as NFLT implies, to always have a good algorithm, information from the underlying problem should be incorporated into the algorithm. In other words, it should be somehow tailor-made. Of course, in the vast majority of cases, an optimization algorithm neither has access to the underlying function nor can it accommodate such information due to its inflexible structure. So, the question is what to do instead, especially in cases like crashworthiness where even obtaining information is expensive. EGO already has an approximation (i.e., information) of the underlying function. Here, a hybrid of procedures that complement each other may be helpful. Although we may not have a step that judges to use which method when, its replacement can be an alternative use of the considered procedures. That is one of the bases of the suggested RF method, which is realized in its third part, where the procedures are alternatively employed, while the first part is also in some occasions, complementary to the third one itself. The second part of RF uses a similar measure to what is suggested in the literature to improve EI-FP performance, in which, at first, at least a feasible solution is obtained before conducting the common EI-FP. So similarly, in this (second) part of RF, a certain number of samples within the desired range are found initially, before proceeding to the next stage.

The RF algorithm was tested for three mechanical design problems. We observed that the first part (established constrained EGO using EI-FP) was important in the welded beam design problem, while parts 2 and 3 were quite important in the I-Beam design problem. In the pressure vessel



problem, neither of those had a dominant influence. So basically, part 1 may overlap with parts 2 and 3 in how efficient they make the performance, but they complement each other too, as intended. The first part can help to find good regions faster than parts 2 and 3 themselves; on the other hand, parts 2 and 3 can help to reduce wasting samples by the first part in regions that should not be visited. The solution space examples also confirm that EI-FP considers the combined effect of involved functions, while RF follows the desired range and evaluates the problem demands within this range successfully, as intended. Another observation from the mechanical design problems is how the RS-based algorithms outperform other heuristic optimization methods in lower dimensions. Among the RS methods, the established EGO using EI-FP is already a highly performant algorithm and the RF methods try to improve it further when it is possible to follow desired output values.

The range that is followed in the RF algorithm should not be too wide. On the other hand, following a very narrow range could require significantly more samples, especially when the corresponding output rarely has any values within this range. This would be basically like optimizing  $EI_{feas}$  that follows a certain value and puts samples (too) close to each other. Here, we did not check the effect of changing the desired range in every problem, but we did change the range from one problem to another to show that performance over different ranges can still lead to reasonable results, especially in cases where the number of samples was limited to just 100. The only case where a larger range was considered was the design of the welded beam problem, and this range was also approximated based on the distribution of the corresponding output obtained in the initial phase runs. The number of iterations for each of the three parts of the RF was also set without trial and error; nevertheless, good performance was obtained. In the third part of RF, optimization should be done within the desired range and not outside it, which is considered infeasible. This is achieved currently by guiding the optimizer towards the desired regions via a penalty method. However, a possible improvement is to use several local optimizers instead of a global one, while the initial samples of these local optimizers are also properly distributed. These samples could be selected from the regions containing feasible solutions in the desired range. In this way, the chance of getting good solutions may increase and consequently, RF could perform even better. Another possible modification of the RF algorithm is to use a bi-objective optimization to follow certain ranges of two constraints instead of one at the same time. This can be helpful, e.g., for designs when yield strength and buckling are both important constraints or for crashworthiness problems limited by allowable force and intrusion. Here, we tried to provide a critical review of the results and make suggestions for possible future research. Some further ideas and comments remain for the concluding chapter of this work.

## Chapter 7

### Conclusion

This work provides several modifications and enhancements to EGO, which are distributed over three main Chapters 4-6. The structure of the result chapters in this thesis is different from the case where they are (rather) built upon previous ones and there is more independence here. In this concluding chapter, parts of the outlook for future work are mentioned, besides the corresponding critical review of the results, but not in a completely separate sections. In this way, we avoid repeating concepts. There are also some general remarks on conducting optimization at the end of this chapter.

## 7.1. Critical review and outlook

The aim of this work is to improve the original EGO algorithm for a limited budget with the main application to crashworthiness. We tried to achieve a general improvement by the inclusion of information from the problem demands, EGO characteristics and availability of the analytical formula in substeps of EGO. In Chapter 4, we proposed several enhancements and modifications. One case was to change the Gaussian kernels in GPR to the IMQ kernel. Then, we used it within EGO as a replacement for the Gaussian kernel, made a hybrid model out of it along with the Gaussian kernel and additionally, we used the relative prediction of these kernels to relocate samples close to the (less important) boundaries. Results improved in all problems where this relocation was the only considered enhancement for the original EGO. This shows how much boundaries affect EGO even in low dimensions and number of evaluations. We have already mentioned that although this relocation is effective, it is still a simple strategy and can be further enhanced to be adaptive. One suggestion for future work is to replace the IMQ kernel with, for example, the linear kernel from RBF. The linear kernel may be suitable to represent simpler functions without having too many ups and downs. Hence, it is quite resistant to overfitting. In this sense, it is complementary to the Gaussian kernel and even less exploitive than the IMQ kernel, according to the author's experience. Hence, this kernel may be quite good for specific cases, but even less general than IMQ. This can be further investigated, especially for cases where the underlying function does not change considerably locally. Hybridization is another way of improvement that we employed in this work. We commented on the ensemble of methods in the discussion section of Chapter 4 rather extensively. We saw that our proposed method is either the best among those in the ensemble or performs in between. This behavior is according to the norms for a decent ensemble method, with the following main message, there is no guarantee that the ensemble is better than any of the participating methods when it comes to prediction for new cases. We proposed another hybridization, this time between EGO and MVMO, in which less influential design variables are separated from the rest. These variables are then found based on perturbation around the best solution, by borrowing some steps from the MVMO algorithm, while the more influential variables are defined in the common way, i.e., by optimizing the infill criterion. The results were promising, especially for practical examples (truss and crash box) and also successful on mathematical test functions when combined with other modifications. This opens the door to further investigations, e.g., one can gather information from the history of perturbations and propose a more sophisticated approach or an exploration element can be also introduced, especially if parallel EGO is the base method. We also proposed changing the range of hyperparameters of the fitted GPR based on how influential the corresponding design variables are by assuming three levels. The results show that this simple change can be successful when the locality of the underlying functions is not high.

In Chapter 5 we set the range for hyperparameters in a more elaborate manner and with the purpose of reducing the overfitting problem of the GPR, which naturally affects EGO. Here, especially the lower bound for the hyperparameter ( $\theta_{lb}$ ) is important and hence, it is set, so the smallest diag-

onal element of the inverse covariance matrix is placed within a certain range, e.g., [1.01, 1.1]. This proposed method can be improved by reducing the number of times that  $\theta_{lb}$  needs changing to be within the considered range. For example, a one-dimensional RS can help to have a better prediction than the current way of changing  $\theta_{lb}$  based on fixed percentages. We observed that for cases where all inputs affect the output more or less the same, the isotropic kernel can be a good representative and the proposed method brings considerable improvements, more than what the author of this work expected. But, in adverse cases, performance becomes worse than that of the original EGO. A potentially promising improvement can be an extension of the proposed modification for anisotropic cases, which is more suitable for practical examples. Here, one needs to establish how influential inputs are on the output and find out whether off-diagonal elements should also be considered or not and if yes, how to normalize their effect (given their different length scale and (actual) influence). In the second part of Chapter 5 we proposed modifying the EI formula, in which it remains unchanged only when it has the highest values (above a threshold), corresponding to most promising regions to evaluate the next infill point, but it is modified elsewhere, to guide the optimizer to a better region than the current one. So, the aim was to improve EGO results by making it considerably easier for the optimizer to find the promising regions of EI and hence, employing the true potential of this successful infill criterion. These promising regions (i.e., certain values of EI) can be defined in advance and independently of the underlying function, since EI is expressed by an analytical formula. Results over unconstrained mathematical test functions and four constrained mechanical design problems were promising. For constrained problems, not only EI, but also FP of each constraint is involved and naturally affects the results. Hence, future modifications can be based on, e.g., an improvement on the FP front. One interesting observation from this chapter is the effect of the locality of the underlying function on the quality of EGO and the proposed modifications. These problems were actually the main ones that created the main challenge. When the locality is high, the most important regions may not be easily reachable from other parts of the design space and this can be a challenge for any optimizer. In some extreme cases, they may even resemble an overfitted function. In such cases, the main cornerstone of this work, i.e., relying on incorporation of information to improve the solution procedure, could actually make performance even worse. Here, a random selection can even be better at a time. First, since the underlying function is highly changing and needs a lot of explorations and second, randomness provides a chance to break the circle of ill-sampling as the fitting method may have wrong assumptions about the underlying function. The use of randomness is not a new topic in optimization, as, e.g., stochastic optimizers employ it as part of their procedure. These algorithms usually have a certain (simple) structure, while they have to navigate various landscapes as a global optimizer. The presence of randomness helps to break this constantly applied simple view, which may not be suitable for the underlying function. On the other hand, excessive use of randomness is clearly an inefficient practice.

In both sections of Chapter 5, we used the available analytical formula in EGO to first improve the overfitting cases and second to enhance the results from EI. This is an example of information

inclusion with general applicability, which fulfills the main goal of this thesis. Inclusion of information is also used in Chapter 6 to improve EGO for handling constrained problems. In the first section, we use the fact that we are interested to follow a certain value of a function and correspondingly developed a criterion for it. This new criterion complements an existing one, which is quite exploitive and has a tendency to place samples (too) close to each other. In comparison, the new criterion puts more distance between samples and it can be a better candidate in cases such as following the boundaries of segmented feasible regions, especially for a low amount of budget as one of the main assumptions of this work. On the other hand, this new criterion is less exact than the already available one in pinpointing the followed value. In the second part of Chapter 6 we extended the idea of following a certain value to a certain range and consequently optimizing for such values efficiently. This is particularly helpful in obtaining several good candidates instead of pursuing just the best solution. The proposed algorithm has three parts, which can also be complementary to each other as explained in Chapter 6. Results from the mechanical design problems and especially the crash simulation problem showed that the proposed method achieved its goal. Even if one considers how good the methods are based on the best solution, the proposed method was in some cases as good as or even better than the common EGO. But, we should not forget that the proposed method spends some of its already limited budget for finding the desired range, while the common EGO employs all of it for finding the best solution. Considering that the proposed method is modular with three parts and the common EGO (with FP criterion for handling constraints) is its first part, one can allocate the budget to prioritize a desired performance. For example, one may just want to follow the best solution by allocating all budget to the first part (common EGO), or additionally following a desired range and obtaining multiple good solutions by considering parts two and three. A more detailed explanation about future improvements of the proposed method is already given in the discussion section at the end of Chapter 6.

The various proposed modifications in this work can be applied at the same time. We did this more or less in each chapter, but we did not combine and study methods from different chapters. Here, for example, one may use a dimension reduction method from Chapter 4 and/or modification of EI from Chapter 5 to see if the proposed RF method in Chapter 6 can be further improved. So, there are various possibilities for further investigation. The proposed enhancements can also be applied to other types of EGO, such as parallel EGO or those developed for multi-fidelity. On the other hand, parallel EGO itself is a suitable option for further improvement of the proposed methods in this work. This is due to the availability of several samples per iteration instead of just one in the original EGO. We concentrated on the inclusion of information and when there is no such information, one way of realization is via trying out complementary methods and observing which one is better for the remaining iterations. Parallel EGO is inherently suitable for such scenarios and especially for higher-dimensional problems that may require many more samples.

EGO becomes continuously less efficient in higher dimensions. So, improvement of EGO for higher dimensions still requires further study. In this regard, we try to mention some important aspects. One effective way to counteract high-dimensionality is to provide a proper relation between

dimensions (i.e., inputs) given the problem at hand. This relation, in the form of an approximation, should hopefully be good enough for at least regions around the evaluated samples; otherwise, by adding each dimension, unlimited possibilities are introduced. We can consider this as information inclusion and based on what was reviewed in this work, either an RS with gradient or low-fidelity as physical surrogate can be helpful here. Due to the problem of having reliable gradients, the former is not practically realizable in many situations. However, the latter has the potential to counteract the curse of high-dimensionality. We solve problems using a numerical method like FEM. So, in the end, it is the system of equations of the method that defines how inputs are related to each other and affect the outputs. In other words, even if there are many dimensions, the system of equations provides certain relations among them, reducing the infinite possibility to a finite one. In FEM, the (tangent) stiffness is an essential quantity that contains information about the underlying problem (at least geometry and material) and hence, it could be a prime candidate for information inclusion in the optimization procedure. This could be necessary for the development of an optimizer which is expected to be highly successful over a wide range of problems (NFLT discussions). In practice, there is already an equivalent static load method (ESLM) by Park (Park, 2011) that utilizes the initial stiffness matrix in its process (although the author of this work does not know whether this development was actually inspired by the NFLT or not). Since the derivative of this initial stiffness is analytically available and can be evaluated fast (even for nonlinear problems), hence, the approximation of the actual relation between different dimensions is available well enough, at least around evaluated points. In general, ESLM requires a (very) small number of expensive high-fidelity evaluations and to the best of the author's knowledge, its scalability to higher dimensions is unparalleled. For example, a FEM model of an airplane with over 2500 design variables as shell thicknesses is studied in Kim et al. (2008). ESLM optimizes this problem in mere tens of high-fidelity evaluations, while optimizing such a high number of design variables is either beyond the reach of many optimizers or, optimistically, it is possible only after hundreds of thousands of evaluations and realistically millions of evaluations. Nevertheless, ESLM like any other method has its own limitations, see e.g., Stolpe et al. (2018). Some optimization algorithms achieve very high efficiency by considering a special purpose objective. For example, the hybrid cellular automata (HCA) algorithm. One of the improved versions of this optimizer assumes a homogeneous distribution of, e.g., the internal energy density in the considered regions and adjusts the optimization accordingly (Zeng and Duddeck, 2017). Hence, it is less generally applicable than ESLM, as its assumption may not always be valid. The intention here is not to discuss the merits or disadvantages of the ESLM or other methods, but to emphasize again that, as dimension increases, it becomes more crucial to provide proper approximated relations between inputs. Otherwise, the number of evaluations can get prohibitively large, even for a system that can be evaluated fast. Recalling that no approximation could be better than gaining information from the source material, which for FEM-based optimization is the system of equations and quantities like the (tangent) stiffness matrix.

We end this thesis by mentioning a couple of practical issues that especially early researchers need

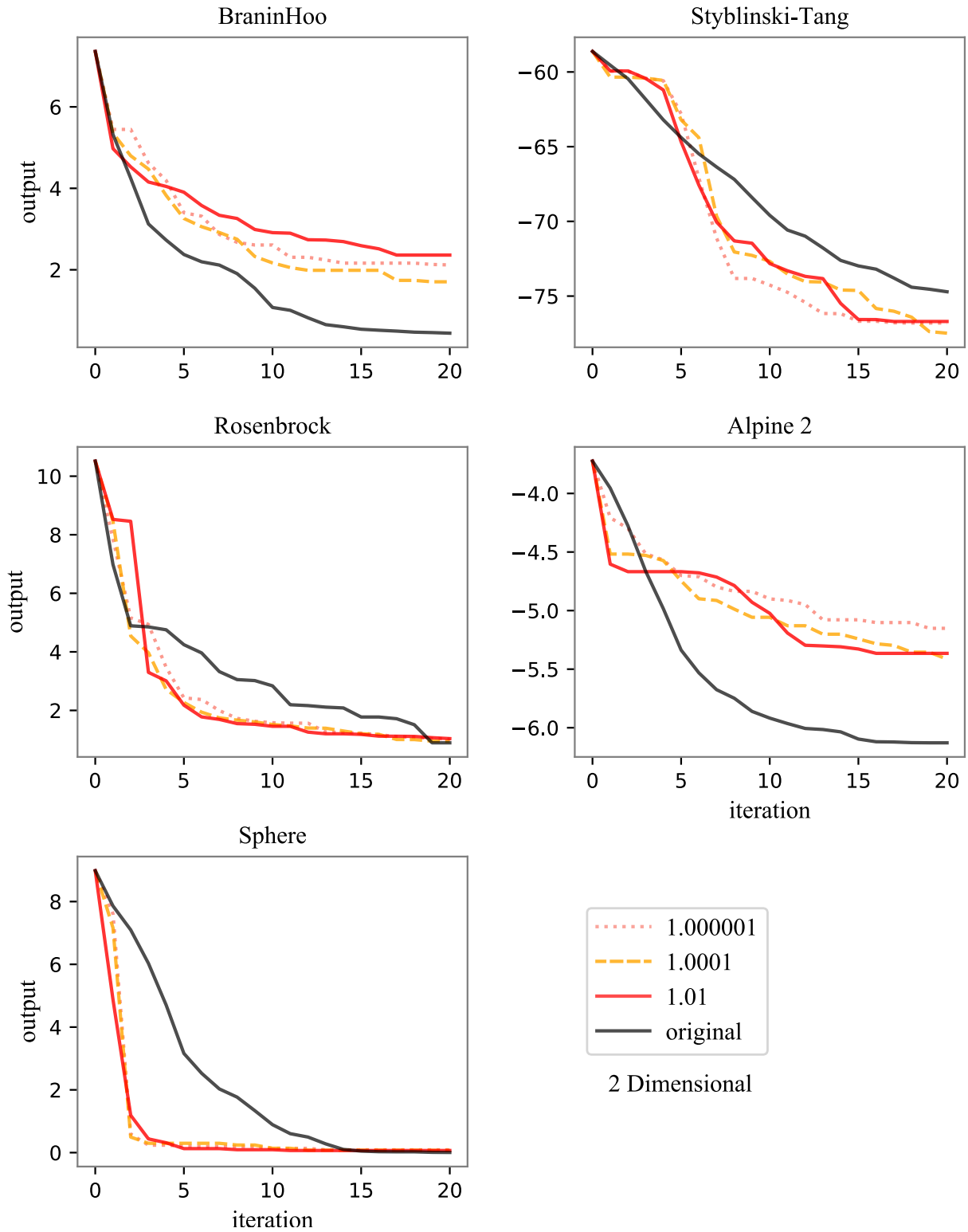
to be aware of. Working on optimization algorithms like any other discipline starts with a literature review. Accordingly, one tries to find new ideas, understand the context, strategies and get inspired by them to propose new improvements. However, such a necessary step can become less fruitful due to the lack of representing negative results in the literature. This has affected the author of this work on occasions. In search of high-performing metaheuristic algorithms, the author came across an algorithm called teaching-learning-based optimization (TLBO) by Rao et al. (2011), which had been referenced many times and according to Hussain et al. (2019), it is among the most popular swarm-based algorithms (in terms of the number of publications). However, later in comparison to some high-performing algorithms, TLBO seemed to have a relatively simple structure to cover various cases. The claims that TLBO has supreme performance compared to some other famous metaheuristic algorithms (such as DE, PSO, ...) are also rejected in Čre-pinšek et al. (2012). Here, the intention is not to question the decency of this successful algorithm, but the practice of generalization from limited results. During working on this thesis, the author gained a better understanding of the notion called "best" himself. This term can be deceptive if it is considered abstractly and out of any context and intended goal. Similarly, in pursuit of good algorithms one faces many metaheuristic algorithms that carry the word 'novel' in their title. At least 112 of such algorithms which are published (in English) within 2009-2015 have been listed in Sotoudeh-Anvari and Hafezalkotob (2018). The main question as mentioned above is, what are the new ideas and strategies and against which challenges (i.e., context). In this regard, Sörensen mentions that, after some initial metaphor-based algorithms like Simulated Annealing and Ant Colony that had some new ideas and gained the attention of researchers, there has been many metaphor-based algorithms (e.g., fireflies, intelligent water drop, ...), while their difference proved marginal at best (Sörensen, 2015). He actually does not provide concrete proof for his claim against all algorithms that he names. However, without the intention of singling out, he discusses the Harmony Search (HS) algorithm as an example. Sörensen refers to another article (Weyland, 2010) that, HS is nothing else but a special case of a  $(\mu + 1)$  evolution strategies.  $(\mu + 1)$  means in each iteration, a new solution is generated and if it is better than the worst in the current population, it would replace it (Weyland, 2010). He claims further that many of these metaphor-based algorithms offer no new ideas, but resell old ones using opaque vocabularies related to the metaphor and hence sometimes the concepts and names has to be stretched so much, that it does not resemble the process they are based on (e.g., an animal does something that no real animal in its species does). Consequently, here, the important point is even when there are new and novel ideas, it can be quite time-consuming to recognize them among mountains of vague expressions. This can be even more problematic, when vagueness becomes a common trend. So, perhaps it would be a good practice to describe the role of newly developed steps and their possible side effect as far as possible. On the other hand, there are also many helpful and informative references for better understanding the optimization strategies, such as, (Molina et al., 2018), (Hansen, 2006) and (Larson et al., 2019). In this work, the author tried to provide reasons for the proposed methods and avoid promoting only the positive results. It remains for future researchers to take better steps.

## Appendix A

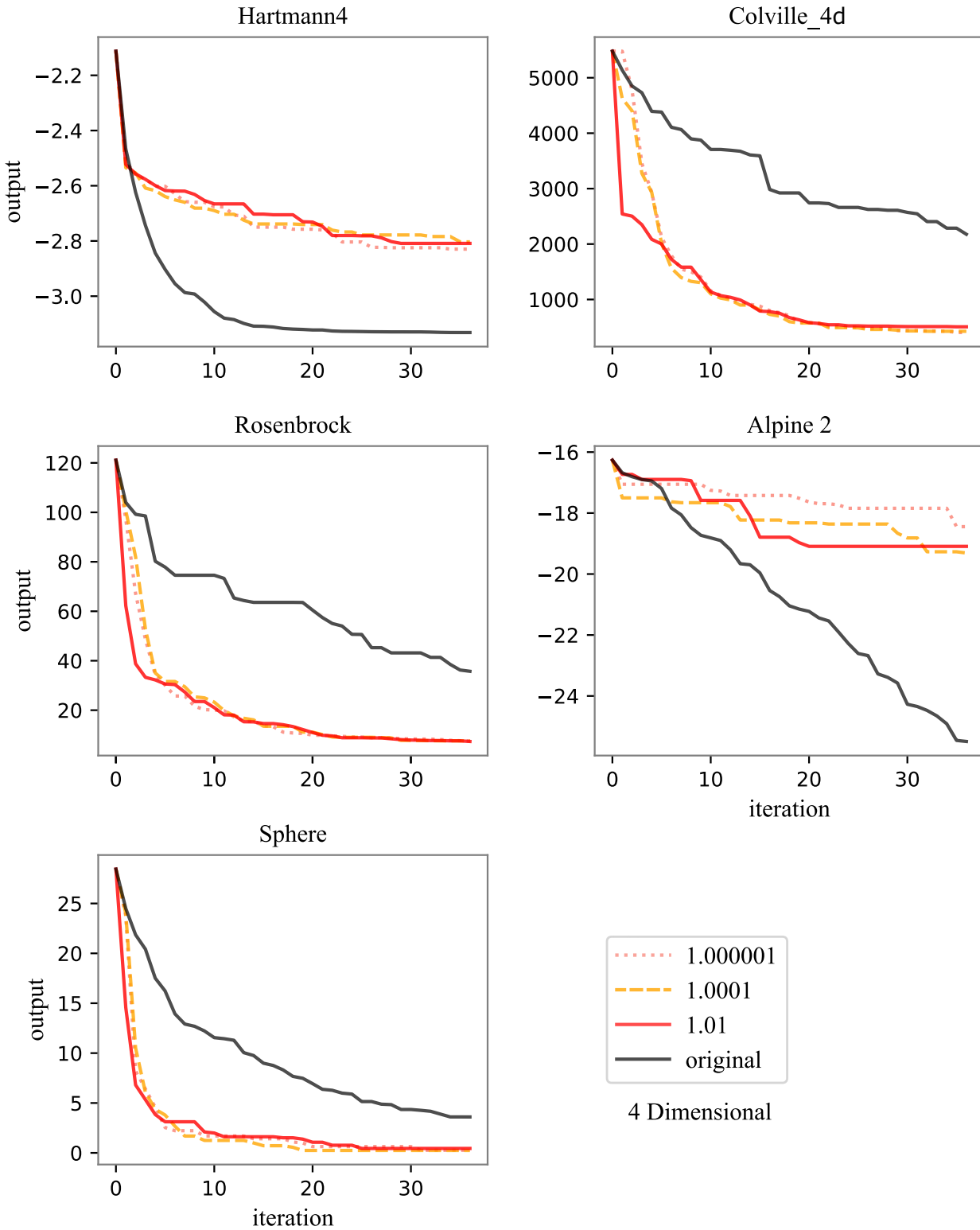
### Setting the hyperparameter bounds

Results of the first iteration of the proposed modifications for setting the hyperparameter bounds in Chapter 5.

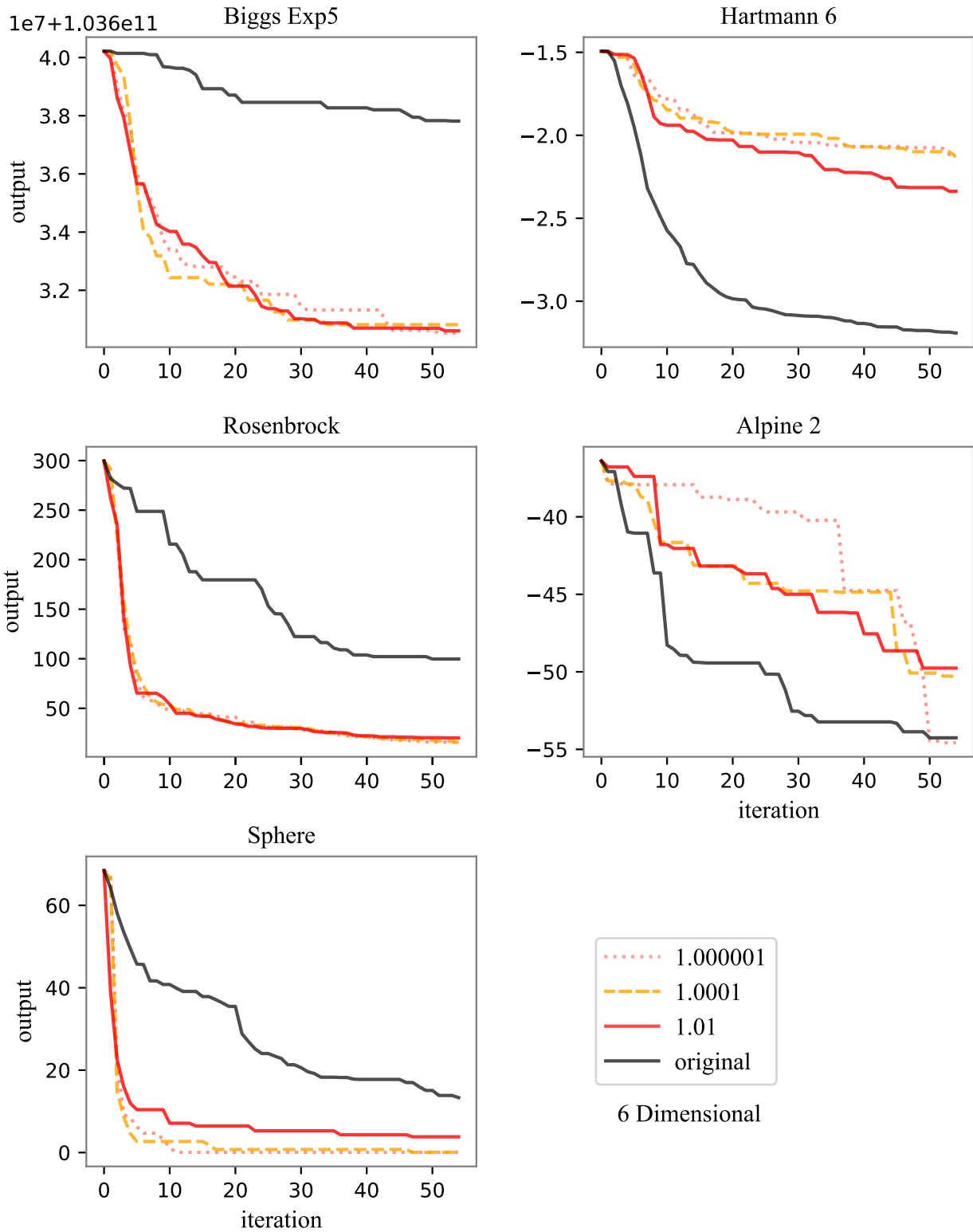




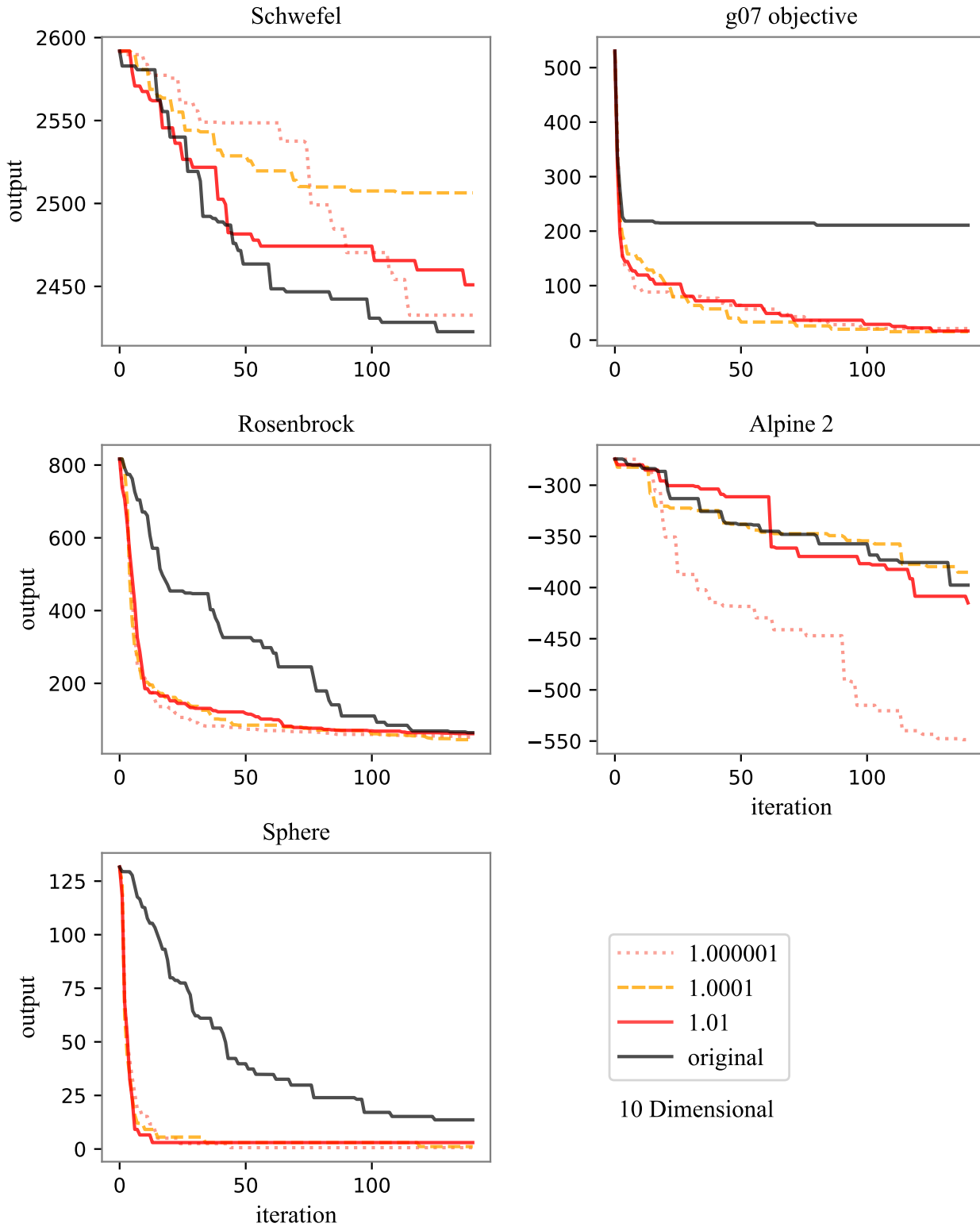
**Figure A.1** Considered two-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. Numbers in the legend are the value of the lower multipliers in Table 5.1. The lower the value, the smaller the changes compared to the original EGO. Each plot is the average of the 30 repetitions for the iterative phase of the EGO.



**Figure A.2** Considered four-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1.



**Figure A.3** Considered six-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1.



**Figure A.4** Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1.

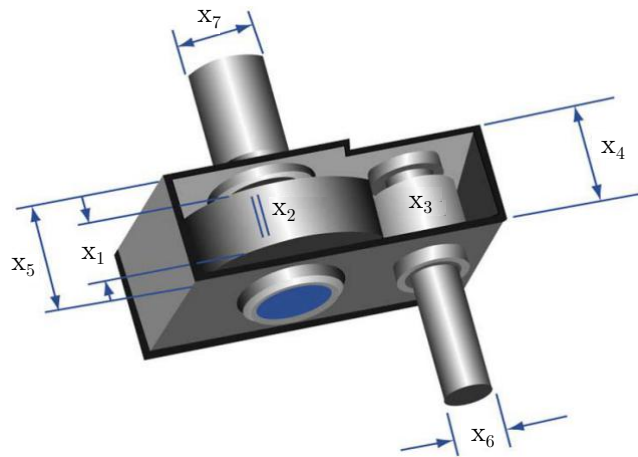
## Appendix B

### Mechanical design problems

## B.1. A speed reducer design problem

The aim is to minimize the weight of a speed reducer (in a gearbox), while there are constraints on various stresses, such as bending stress of teeth and deflection of shafts. The seven design variables are face width ( $x_1$ ), teeth module ( $x_2$ ), the number of teeth in the pinion ( $x_3$ ), length of the first and second shafts between their bearings ( $x_4$  and  $x_5$ ), the diameter of the first and second shafts ( $x_6$  and  $x_7$ ) (Zhu et al., 2019). We set  $x_3 = 17$  in our simulations, since we are only working with continuous variables.

$$\begin{aligned}
 & \underset{\mathbf{x}}{\text{minimize}} & f(\mathbf{x}) &= 0.7854x_1x_2^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) \\
 & & & + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2) \\
 & \text{subject to} & g_1(\mathbf{x}) &= \frac{27}{x_1x_2^2x_3} - 1 \leq 0, \quad g_2(\mathbf{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0, \quad g_3(\mathbf{x}) = \frac{1.93x_4^3}{x_2x_6^4x_3} - 1 \leq 0, \\
 & & g_4(\mathbf{x}) &= \frac{1.93x_5^3}{x_2x_7^4x_3} - 1 \leq 0, \quad g_5(\mathbf{x}) = \frac{[(745x_4/x_2x_3)^2 + 16.9 \times 10^6]^{0.5}}{110x_6^3} - 1 \leq 0, \\
 & & g_6(\mathbf{x}) &= \frac{[(745x_5/x_2x_3)^2 + 157.5 \times 10^6]^{0.5}}{85x_7^3} - 1 \leq 0, \\
 & & g_7(\mathbf{x}) &= \frac{x_2x_3}{40} - 1 \leq 0, \quad g_8(\mathbf{x}) = \frac{5x_2}{x_1} - 1 \leq 0, \quad g_9(\mathbf{x}) = \frac{x_1}{12x_2} - 1 \leq 0, \\
 & & g_{10}(\mathbf{x}) &= \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0, \quad g_{11}(\mathbf{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0, \\
 & & & 2.6 \leq x_1 \leq 3.6, \quad 0.7 \leq x_2 \leq 0.8, \quad 17 \leq x_3 \leq 28, \quad 7.3 \leq x_4 \leq 8.3, \\
 & & & 7.3 \leq x_5 \leq 8.3, \quad 2.9 \leq x_6 \leq 3.9, \quad 5.0 \leq x_7 \leq 5.5.
 \end{aligned} \tag{B.1}$$

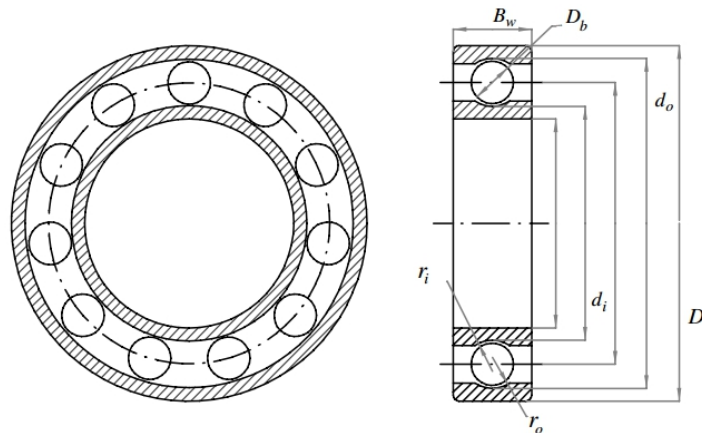


**Figure B.1** Design variables of the speed reducer problem (Mehmood et al., 2016).

## B.2. A rolling element bearing design problem

The objective is to optimize the dynamic load carrying capacity of a rolling element bearing ( $C_d$  (N)), while there are ten design variables ( $D_m, D_b, Z, f_i, f_o, K_{Dmin}, K_{Dmax}, \varepsilon, e, \zeta$ ). The first five define the internal geometry directly, and the latter five only exist in constraints. The units of  $D_m$  and  $D_b$  are "mm" and the rest are unitless. The only discrete design variable is the number of balls in the bearing ( $Z$ ) (Sadollah et al., 2013). More information can be found in (Gupta et al., 2007), where the multi-objective version of this problem was originally introduced. We solve the single objective version by setting  $Z = 11$  and based on the below settings.

$$\begin{aligned} \underset{\mathbf{x}}{\text{maximize}} \quad & C_d = f_c Z^{2/3} D_b^{1.8} \quad \text{if } D \leq 25.4 \text{ mm} \\ & 3.647 f_c Z^{2/3} D_b^{1.4} \quad \text{if } D > 25.4 \text{ mm} \\ g_1(\mathbf{x}) = & \frac{\phi_0}{2 \sin^{-1}(D_b/D_m)} - Z + 1 \geq 0, \quad g_2(\mathbf{x}) = 2D_b - K_{Dmin}(D - d) \geq 0, \\ g_3(\mathbf{x}) = & K_{Dmax}(D - d) - 2D_b \geq 0, \quad g_4(\mathbf{x}) = \zeta B_w - D_b \geq 0, \quad g_5(\mathbf{x}) = D_m - 0.5(D + d) \geq 0, \\ g_6(\mathbf{x}) = & (0.5 + e)(D + d) - D_m \geq 0, \quad g_7(\mathbf{x}) = 0.5(D - D_m - D_b) - \varepsilon D_b \geq 0, \quad g_8(\mathbf{x}) = f_i \geq 0.515, \\ f_c = & 37.91 \left[ 1 + \left\{ 1.04 \left( \frac{1 - \gamma}{1 + \gamma} \right)^{1.72} \left( \frac{f_i(2f_o - 1)}{f_o(2f_i - 1)} \right)^{0.41} \right\}^{10/3} \right]^{-0.3} \left[ \frac{\gamma^{0.3}(1 - \gamma)^{1.39}}{(1 + \gamma)^{1/3}} \right] \left[ \frac{2f_i}{2f_i - 1} \right]^{0.41}, \\ \phi_0 = & 2\pi - 2\cos^{-1} \left( \frac{\{(D-d)/2-3(T/4)\}^2 + \{D/2-T/4-D_b\}^2 - \{d/2+T/4\}^2}{2\{(D-d)/2-3(T/4)\}\{D/2-T/4-D_b\}} \right), \quad D = 160, \quad d = 90, \\ B_w = & 30, \quad r_i = r_o = 11.033, \quad \gamma = \frac{D_b}{D_m}, \quad f_i = \frac{r_i}{D_b}, \quad f_o = \frac{r_o}{D_b}, \quad T = D - d - 2D_b, \\ & 0.5(D + d) \leq D_m \leq 0.6(D + d), \quad 0.15(D - d) \leq D_b \leq 0.45(D - d), \quad 0.515 \leq f_i \text{ and } f_o \leq 0.6, \\ & 0.4 \leq K_{Dmin} \leq 0.5, \quad 0.6 \leq K_{Dmax} \leq 0.7, \quad 0.3 \leq \varepsilon \leq 0.4, \quad 0.02 \leq e \leq 0.1, \quad 0.6 \leq \zeta \leq 0.85. \end{aligned} \tag{B.2}$$



**Figure B.2** Design variables of the rolling element bearing design problem (Gupta et al., 2007).

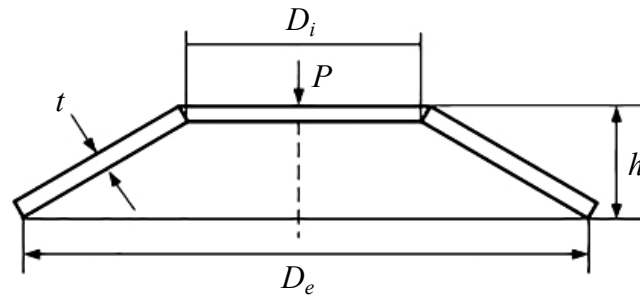
### B.3. A Belleville disc spring design problem

The objective is to minimize the weight of a Belleville disc spring. Design variables are the external ( $D_e$ ) and internal ( $D_i$ ) diameters, thickness ( $t$ ) and height ( $h$ ) of the spring as shown in Figure B.3. Constraints are applied to compressive stress ( $g_1$ ), height to deflection ( $g_2$ ), deflection ( $g_3$ ), height to maximum height ( $g_4$ ), outer diameter ( $g_5$ ), inner diameter ( $g_6$ ), and slope ( $g_7$ ), see Rao and Savsani (2012).

$$\begin{aligned}
 & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) = 0.07075\pi(D_e^2 - D_i^2)t \\
 & \text{subject to} && g_1(\mathbf{x}) = \frac{4E\delta_{\max}}{(1-\mu^2)\alpha D_e^2} \left[ \beta \left( h - \frac{\delta_{\max}}{2} \right) + \gamma t \right] / S - 1 \leq 0, \\
 & && g_2(\mathbf{x}) = 1 - \frac{4E\delta_{\max}}{(1-\mu^2)\alpha D_e^2} \left[ \left( h - \frac{\delta_{\max}}{2} \right) (h - \delta_{\max})t + t^3 \right] / P_{\max} \leq 0, \\
 & && g_3(\mathbf{x}) = 1 - \delta_l / \delta_{\max} \leq 0, \quad g_4(\mathbf{x}) = (h+t)/H - 1 \leq 0, \quad g_5(\mathbf{x}) = D_e/D_{\max} - 1 \leq 0, \\
 & && g_6(\mathbf{x}) = D_i/D_e - 1 \leq 0, \quad g_7(\mathbf{x}) = \frac{h}{D_e - D_i} / 0.3 - 1 \leq 0, \\
 & && \alpha = \frac{6}{\pi \ln K} \left( \frac{K-1}{K} \right)^2, \quad \beta = \frac{6}{\pi \ln K} \left( \frac{K-1}{\ln K} - 1 \right), \quad \gamma = \frac{6}{\pi \ln K} \left( \frac{K-1}{2} \right), \\
 & && P_{\max} = 5400 \text{ lb}, \quad \delta_{\max} = 0.2 \text{ in}, \quad S = 200 \text{ kpsi}, \quad E = 30e06 \text{ psi}, \quad \mu = 0.3, \\
 & && H = 2 \text{ in}, \quad D_{\max} = 12.01 \text{ in}, \quad K = \frac{D_e}{D_i}, \quad \delta_l = f_l(a)a, \quad a = h/t, \quad \text{see the below Table,} \\
 & && 5 \leq D_e \leq 15, \quad 5 \leq D_i \leq 15, \quad 0.01 \leq t \leq 0.6, \quad 0.05 \leq h \leq 0.5, \quad \text{units : in.}
 \end{aligned} \tag{B.3}$$

**Table B.1** Variation of  $f_l(a)$  with  $a$ .

$a$	$\leq 1.4$	1.5	1.6	1.7	1.8	1.9	2.0	2.1	2.2	2.3	2.4	2.5	2.6	2.7	$\geq 2.8$
$f_l(a)$	1	0.85	0.77	0.71	0.66	0.63	0.6	0.58	0.56	0.55	0.53	0.52	0.51	0.51	0.5



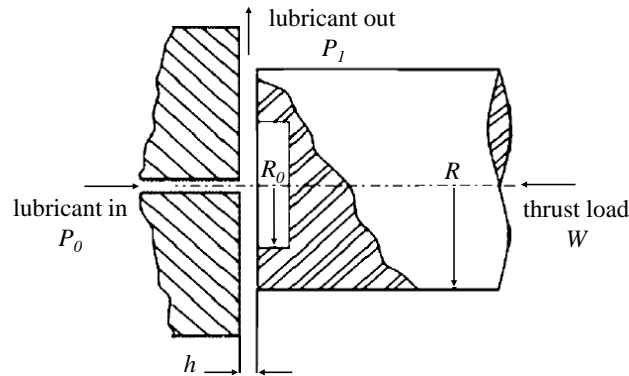
**Figure B.3** Design variables of the Belleville disc spring design problem (Zhu et al., 2019).



## B.4. A hydrostatic thrust bearing design problem

The objective is to minimize the total power loss, which is the sum of the pumping energy and the friction loss of a hydrostatic plain bearing under an axial load. The efficiency of the pump is 0.7 and design variables are flow rate ( $Q$ ), recess radius ( $R_0$ ), bearing step radius ( $R$ ) and viscosity ( $\mu$ ). There are seven nonlinear constraints; load carrying capacity ( $g_1$ ), inlet oil pressure requirements ( $g_2$ ), oil temperature rise ( $g_3$ ), oil film thickness ( $g_4$ ), and some other physical requirements ( $g_5 - g_7$ ), see He et al. (2004) and Sahin et al. (2019). The first constraint is relaxed here to have enough feasible solutions for a small number of evaluations.

$$\begin{aligned}
 & \underset{\mathbf{x}}{\text{minimize}} && f(\mathbf{x}) = \frac{QP_0}{0.7} + 9336Q\gamma C\Delta T \\
 & \text{subject to} && g_1(\mathbf{x}) = h - 3\mu Q(R^2 - R_0^2)/W_s, \quad h = h^{1/3}, \text{ if } h < -1 \text{ or } h > 1, \\
 & && g_2(\mathbf{x}) = P_0 - P_{\max}, \quad g_3(\mathbf{x}) = \Delta T - \Delta T_{\max} \leq 0, \\
 & && g_4(\mathbf{x}) = h - h_{\min} \leq 0, \quad g_5(\mathbf{x}) = R - R_0 \leq 0, \\
 & && g_6(\mathbf{x}) = \frac{\gamma}{gP_0} \frac{Q}{2\pi Rh} - 0.001 \leq 0, \quad g_7(\mathbf{x}) = \frac{W}{\pi(R^2 - R_0^2)} - 5000 \leq 0, \\
 & && W = \frac{\pi P_0}{2} \frac{R^2 - R_0^2}{\ln(R/R_0)}, \quad P_0 = \frac{6\mu Q}{\pi h^3} \ln(R/R_0), \quad E_f = 9336Q\gamma C\Delta T, \\
 & && \Delta T = 2(10^P - 560), \quad P = \frac{\log_{10}\log_{10}(8.122 \times 10^6\mu + 0.8) - C_1}{n}, \\
 & && W_s = 101000 \text{ lb}, \quad P_{\max} = 1000 \text{ psi}, \quad \Delta T_{\max} = 50 \text{ }^\circ\text{F}, \quad h_{\min} = 0.001 \text{ in}, \\
 & && h = \left(\frac{2\pi N}{60}\right)^2 \frac{2\pi\mu}{E_f} \left(\frac{R^4}{4} - \frac{R_0^4}{4}\right), \quad \gamma = 0.0307 \text{ lb/in}^3, \quad C = 0.5 \text{ Btu/lb}^\circ\text{F}, \\
 & && C_1 = 10.04 \text{ and } n = -3.55 \text{ oil properties}, \quad g = 386.4 \text{ in/s}^2, \quad N = 750 \text{ rpm}, \\
 & && 1 \leq R, R_0, Q \leq 16, \quad 1e-6 \leq \mu \leq 16e-6.
 \end{aligned} \tag{B.4}$$



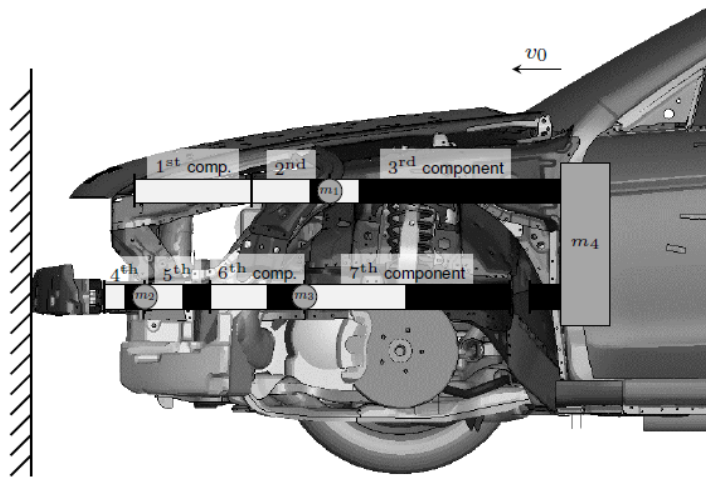
**Figure B.4** Design variables of the hydrostatic thrust bearing problem. Based on Sahin et al. (2019).

## Appendix C

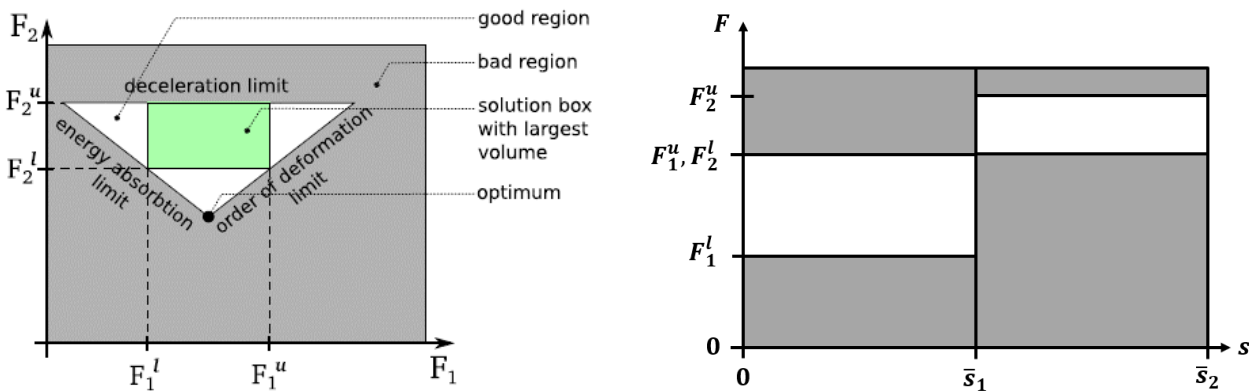
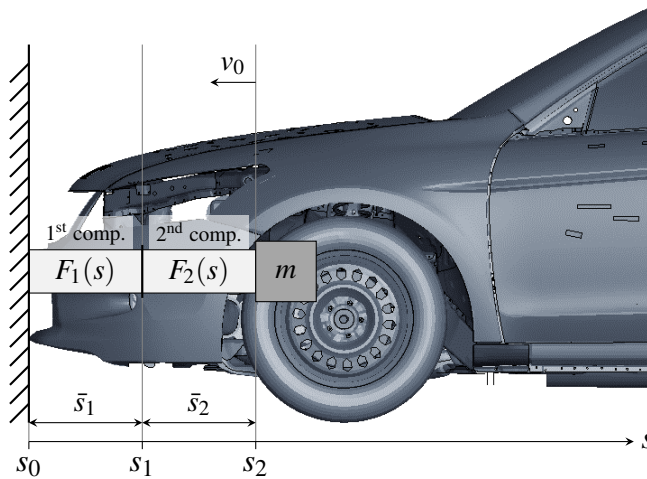
### Solution space method

## C.1. Solution space method

Here, we briefly describe the solution space method based on Lange et al. (2019). Initial ideas about the solution space were suggested by Zimmermann and von Hoessle (Zimmermann and von Hoessle, 2013). In the early stage of a car design, many aspects are not fixed; yet, in the end, the design must fulfill requirements from safety to comfort. Hence, it is preferable to work with several designs rather than a single value, such as an optimum (which is not even obtainable yet). The solution space method enables decoupled component development, which should fulfill functional constraints; so, it defines a subset of the feasible region to allow involved departments to work independently as much as possible. Therefore, a subset of the feasible space is chosen to make the decoupling of components possible, which is done via analytical models (representing simplified crash mechanics) or via Monte Carlo simulation. One tries to select the part of the feasible region that makes decoupling as large as possible to have more flexibility in design down the stream, where many decision parameters start to be fixed more and more. This selection leads to the definition of an upper and lower bound, or, in other words, a corridor for the force of each component over its deformation; see Figure C.2. Here, the solution space is developed for an idealized and simplified model of a car, e.g., an initial design for a full frontal crash can be considered to have a couple of load paths. In each load path, the components are one-dimensional and in series and load paths are in parallel. Here, the simplified model is a mass-spring one. Each component can deform up to a certain amount and the remaining undeformed part is called block length, see Figure C.1. Each component can also have a gap with the component that comes afterward. In the end, forces and displacements matter. So, one needs to transfer this geometry space representation to the one in which, a force in a component is related to a deformable part of it, while these components are in series within a load path, see Figure C.1. The mentioned force is one-dimensional and is in a sense an average representative of a quantity with the unit of force, which is developed within the component during the simulation. This force-type quantity is obtained based on some assumptions, e.g., it can be derived by considering the one-dimensional representation having the same internal energy and deformation as the actual (three-dimensional) component that it represents. Functional constraints are also introduced, e.g., acceleration or displacement should be less than a certain value, while these constraints are represented properly for this simplified one-dimensional representation. An additional constraint is introduced based on the assumption that the component which is closer to the impact deforms earlier than those that come after it in a load path. This leads to a constraint among forces in components. Now, one can consider a space spanned via these force quantities of each component, while the mentioned constraints define a feasible region of this space, see Figure C.2. As mentioned above, the largest box within the feasible region is found next. Finding this box means knowing the lower and upper bounds of the force (or force-like quantity) for each component and a solution space is formed by considering these bounds over the deformable part of the components. In other words, corridors are developed for all involved components, while they are in series for components within a load path and parallel to other paths' corridors. Now, every involved department can modify its components independently, if the responses remain within the corresponding corridors.



**Figure C.1** Frontal part of the Honda Accord model from (NHTSA, Last access February 2020). Two load paths are shown each containing some components. Black regions are parts of the component that cannot deform (block length) (Daub, 2020).



**Figure C.2** Top figure shows a simple, single load path solution space (C.1) model where block length is already removed. Below figures show the corresponding procedure to find the corridors for solution space (Daub, 2020) and (Zimmermann and von Hoessle, 2013).

## List of Figures

1.1	Four different types of cost functions. (a) and (b) are taken from Weise (2011) with modifications and (c) from (Liang et al., 2005). (d) is an Expected Improvement function from an adaptive stage of EGO. . . . .	4
2.1	Methods for solving a nonlinear optimization . . . . .	10
2.2	Polynomial of $2^{nd}$ order is underfitting and of $10^{th}$ order is overfitting. . . . .	16
2.3	The figure on the left shows an example of fitting noisy data very well, but making poor predictions (within (4,6)) while there is a regularization. By increasing the regularization to a sufficient level, predictions become much better, but noisy data is now regressed, so accuracy is lost (not a bad feature here). The left figure is the preferred one. . . . .	17
2.4	Halton in 15-dimensional space. The line-up of samples is shown in a 2D projection on dimensions 10 and 11. The correlation between these two dimensions decreases as the number of samples increases. . . . .	21
2.5	Some samples can get very close to each other compared to their distances to other samples, see the highlighted sample pairs. The right figure shows that this non-uniformity can be alleviated by skipping some initial samples (here, 10). . . . .	21
2.6	Two common Sobol patterns in which samples are concentrated in 2D projections. . .	22
2.7	Four cases of two-dimensional samples generated by uniform MC on the top, LH in the middle, and OLH on the bottom are shown on the left side. On the right, the value of the potential energy criterion for each case is depicted in bar plots correspondingly.	24
2.8	Comparison of the value of the potential energy criterion in three different cases and in DOE methods. LH designs that are optimized in OLH are different from LH designs.	24
2.9	Effects of dimensionality on the uniformity of the design using the normalized MST criterion for 1000 samples (Garud et al., 2017). Larger MST values are considered to have better space-filling. MCS: Monte Carlo sampling, SOB: Sobol, HAL: Halton, HAM: Hammersley and LHD: Latin Hypercube design. . . . .	25
2.10	Left: Change in the volume of a hypersphere with unit radius in terms of dimensionality. Right: relative change of volume between hyperspheres with radii 0.9 and 1, respectively. Based on Verleysen (2003). . . . .	26

- 2.11 10 millions samples are generated from the uniform MC within hypercubes of different dimensions (2, 3, 5 and 10). The closest distance of each sample to a boundary (of the hypercube) is calculated and its histogram's density (i.e., dividing the number of samples in each histogram's bin by 10 millions) is plotted for the considered dimensions. For each dimension, a trend line is also fitted at the top center of the bins. Here, due to the high number of samples, the bins are quite narrow and hence difficult to be seen. This figure shows that, by increasing the number of dimensions, samples get closer to the boundaries and away from the center of the design space. . . . . 27
- 2.12 Prior GP mean is shown in bold solid black line and highlighted area depicts one standard deviation around each point. Figure (a) shows prior GP with zero mean and standard deviation of one (a common choice when there is no default information about data). The posterior distribution obtained based on four samples (red dots) is shown in Figure (b). In both figures, 10 sample functions are drawn from each case. . 28
- 2.13 Left figure shows how the SE correlation function reduces with distance over different length scales. The right figure shows the corresponding fitted curves over the given data. For length scales  $l = 1$  and  $l = 10$ , the standard deviation becomes so small that it is not observable here. . . . . 32
- 2.14 Left figure shows how Matèrn with different  $\nu$  values reduces with increasing distance. The right figure shows the corresponding fitted curves over the given data for  $\nu = 0.5$  and  $\nu = 2.5$ . . . . . 32
- 2.15 Effect of the number of samples on the likelihood function. In general, the higher the number of samples, the more distinct the likelihood optimum. Here, the negative logarithm of likelihood is depicted. . . . . 34
- 2.16 The left column shows the negative logarithm of the likelihood function for two cases. The top is for anisotropic SE and the bottom for isotropic SE. The left column shows the corresponding fitted curves in green and the actual underlying function on the surface with a red and blue colormap. The dots are data points. Here, ordinary Kriging was used. . . . . 35
- 3.1 Flowchart of some adaptive response surface-based optimization (ARSBO) methods. In EGO, the considered RS is the predicted mean of the fitted GPR. . . . . 37
- 3.2 Steps of the efficient global optimization (EGO) . . . . . 38
- 3.3 An ordinary Kriging (left) is compared to a gradient-enhanced ordinary Kriging (right) over a test function. The smaller figure on the right shows the same fit, but from another angle. With only six samples, the gradient-enhanced model achieves significantly better accuracy, but at a higher computational cost. . . . . 39
- 3.4 Top figure shows a GPR curve fitted to current samples. Figures below depict some infill criteria for the current iteration. . . . . 41

3.5	Finding the feasible optimum by EI-FP. Contours represent the objective function which is a modified Branin-Hoo with three local minima. Regions where two constraints are violated are highlighted in gray. Six initial points are shown in black circles and adaptive points in red squares. Orange rhombuses mark the three local minima of which the lower right is the global minimum in the absence of constraints. . . . .	46
3.6	CDF of the standard normal distribution and corresponding modification of it suggested in Bagheri et al. (2017). . . . .	47
3.7	Figure (a) shows the result of an interpolative GPR curve in case of a noisy function. Here noise is added to a test function labeled as the "underlying function" (b) When regularization is applied excessively, the standard deviation of prediction, here labeled as "MSE", becomes less distinct at each point compared to its neighbors, causing problems for EGO. (c) Re-interpolation solves the problems in (a) and (b). There will be a smooth curve representing the noisy data and the predicted standard deviation has its desirable form (Hefele, 2017). . . . .	51
4.1	IMQ and Gaussian kernel values vs. distance from the kernel location for $\theta = 1$ . . . . .	56
4.2	A realization of a GPR in two dimensions. Output (vertical axis) changes considerably less along the $x_2$ direction than the $x_1$ direction. Therefore, $\theta_2$ should be higher than $\theta_1$ . . . . .	58
4.3	EGO algorithm with partial MVMO-sampling and sample translation. The gray box shows MVMO and the light orange boxes indicate sample translation in the proposed modified EGO. . . . .	59
4.4	The proposed hybrid surrogate model (HSM) procedure. . . . .	61
4.5	All adaptive samples of all 50 independent optimizations are plotted vs. each dimension for the Hartman-6 problem. The distribution of each design variable is shown in the lower part. . . . .	64
4.6	Mean values of the adaptive phase of all 50 repetitions of different methods are compared for the Hartman-6 test function. IMQ here means EGO with the IMQ kernel. . . . .	64
4.7	The parallel coordinate plot of all 50 repetitions of the proposed methods for the Hartman-6 problem. The darker the color the more desirable the value (we are conducting minimization). . . . .	65
4.8	All adaptive samples of all 50 independent optimizations are plotted vs. each dimension for the Himmelblau problem. The distribution of each design variable is shown in the lower part. . . . .	67
4.9	Mean values of the adaptive phase of all 50 repetitions of different methods for the Himmelblau problem. . . . .	67
4.10	The parallel coordinate plot of all 50 repetitions of the proposed methods for the Himmelblau problem. . . . .	68
4.11	The two-dimensional projections of the design variables for MVMO. . . . .	68
4.12	The two-dimensional projections of the design variables for HSM. . . . .	68

4.13	The considered linear elastic truss problem. . . . .	69
4.14	Mean values of the adaptive phase of all 50 repetitions of different methods for the truss problem. . . . .	70
4.15	The parallel coordinate plot of all 50 repetitions of the proposed methods for the truss problem. The darker the color the more desirable the value (we are conducting minimization). . . . .	70
4.16	The two-dimensional projections of the design variables for the original EGO. . . . .	71
4.17	The two-dimensional projection of the design variables for EGO+ST. . . . .	71
4.18	This structure represents a side sill impacted by a cylindrical rigid body as a pole. Both ends are fixed and the design variables are the thicknesses of the horizontal ribs shown here. . . . .	72
4.19	Piecewise linear plasticity model of aluminum. Values at each point are given in the table on right. . . . .	72
4.20	Mean values of the adaptive phase of all 50 repetitions of different methods for the side sill problem. . . . .	73
4.21	The parallel coordinate plot of all 50 repetitions of the proposed methods for the side sill impact problem. The darker the color the more desirable the value (we are conducting minimization). . . . .	73
4.22	The two-dimensional projections of the design variables in the side sill problem for IMQ. . . . .	74
4.23	The two-dimensional projections of the design variables in the side sill problem. . . . .	74
4.24	Colors in each column represent the ranking of the nine compared methods for each of the four considered problems. In the middle, "ref." means the (common) EGO is the reference method for comparison. . . . .	76
5.1	$\ln(-\ln(\text{Likelihood}))$ is plotted for current data set including 6 samples. The fitted curve for different values of $\theta$ is plotted in each case (solid blue lines) along with the predicted variance (shaded area) and the actual underlying function (black dotted lines). The variance gets smaller by increasing $\theta$ and hence it may not be observable from the figures. . . . .	80
5.2	overfitting of a Kriging with a linear trend. . . . .	81
5.3	$\ln(-\ln(\text{Likelihood}))$ is depicted for the six available samples in 2D, originating from the BarninHoo test function shown in top left. Considered covariance model is anisotropic squared exponential with regularization of $1e - 10$ . The fitted surfaces (i.e., mean of the GPR) for the selected $\theta$ s are shown in green and yellow colormap while the corresponding Expected Improvement is shown in gray-scale beside each case. . . . .	82



5.4	Considered two-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. Numbers in the legend are the value of the lower multipliers in Table 5.1, the corresponding upper multipliers are 10 times their lower counterparts. So, e.g., the legend 1.01 represents the acceptable range of [1.01, 1.1] for the smallest diagonal element of the $K^{-1}$ . The lower the value, the less the possible changes compared to the original EGO. Each plot is the average of the 30 repetitions for the iterative phase of the EGO. . . . .	90
5.5	Considered four-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4. . . . .	91
5.6	Considered six-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4. . . . .	92
5.7	Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure 5.4. . . . .	93
5.8	The history of the hyperparameter $\theta$ in each ten-dimensional test function to evaluate the effect of the final suggested modification for the $\theta$ bound. The highlights show the region between maximum and minimum $\theta$ values among all thirty repetitions. Solid lines are the average of thirty cases. The rest of the description is the same as the one in Figure 5.4. . . . .	95
5.9	Alpine2 (left) and Schwefel (right) test functions. . . . .	96
5.10	Extrapolation of a regularly fitted (i.e., not overfitted) Kriging with a linear trend. This plot shows how extrapolation in far distances can be more and more misleading while at the same time the uncertainty also grows, both of which make EGO visit and remain in regions that it should have avoided initially. . . . .	96
5.11	Considered two-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO. . . . .	100
5.12	Considered four-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO. . . . .	101
5.13	Considered six-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO. . . . .	102
5.14	Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on EI. Each plot is the average of the (30) repetitions for the iterative phase of EGO. . . . .	103
5.15	Top three figures are the results of the Belleville spring design problem and the bottom three that of the rolling element bearing problem. . . . .	106

5.16	Top three figures are the results of the speed reducer problem and the bottom three that of the thrust bearing problem. . . . .	107
5.17	Piecewise linear plasticity used for the material model *MAT_24 in LS-DYNA. Values at each point are given in the table on right. . . . .	109
5.18	Crash box problem shape and design variables. The transparent plane represents the moving rigid wall impacting the crash box from the top while the other end is fixed. .	109
5.19	Some possible cross-sections of the crash box problem. . . . .	110
5.20	Left: Average feasible solution in the adaptive phase of the constraint EGO methods. Right: Best feasible solution among 4500 (3*30*50) simulations is one of the MEI-FP cases. . . . .	111
5.21	Left: Number of feasible solutions for each constraint in the adaptive phase of 30 repetitions. Right: Boxplots of the infill time. The dashed lines indicate the mean values (51.2 vs. 45.9 s). . . . .	111
5.22	Average feasible solution in the adaptive phase of the constrained EGO. " $\theta - 1$ " refers to the first modification of the $\theta$ bounds and " $\theta - 2$ " to the second one respectively. . .	112
6.1	BraninHoo test function contours shown in color in a normalized space ([0,1]). The two black contours show where this function equals 75. . . . .	118
6.2	The two black curves show the locations of interest where the BraninHoo function equals to 75. Black circles are the initial samples of EGO and the gray squares are the adaptive phase samples, correspondingly. The colored regions show the prediction by the GP in the range [-5,5] of the desired target (75), i.e., predicted BraninHoo in [70,80]. The top row is the prediction around the target after the 21 <sup>st</sup> sample and the bottom row after the 28 <sup>th</sup> sample. The right plots show the proposed criterion predictions and the left that of the $EI_{feas}$ criterion. . . . .	119
6.3	The top figure shows the function "6 - Gomez3" and the lower figure shows where this function is above zero in grayscale and where it is below zero in blue. The closed black contours are loci where Gomez3 equals 6. This example is based on Durantin et al. (2016), where it was used as a constraint (with blue regions showing the feasible part) while, here, we try to follow the boundary of the feasible region, i.e., the limit value. . . . .	120
6.4	Finding the locations where Gomez3 function equals 6 (i.e., closed black curves). Descriptions are the same as those in Figure 6.2, except that the first row shows prediction after the 25 <sup>th</sup> sample, the second row after the 38 <sup>th</sup> and the last row after the 45 <sup>th</sup> sample. . . . .	121
6.5	Contour plot of the BraninHoo function subject to a constraint. The infeasible region is highlighted in gray. Black circles show the initial samples and red squares the adaptive samples, correspondingly. The BraninHoo values along the boundary of the constraint (i.e., where it is active) are shown in the gray curve on top. . . . .	122
6.6	Description is the same as for Figure 6.5. . . . .	123

6.7 Flowchart of the proposed RF and RF\_pseudo algorithms. The difference between these two algorithms is only in the infill criterion when following CTF. The former employs  $EI_{feas}$  and the latter  $SLCB-EI_{feas}$ . The figure in the lower part shows the schematic for the  $r_{CTF} = [a, b]$  range where the prediction of the value that we are following, i.e.,  $\hat{CTF} = 0$  locates within this range. The hat sign indicates the approximation of the underlying function, which is the mean of the fitted GPR. The three numbers in circles indicate three main parts of this algorithm. . . . . 126

6.8 Depiction of the considered welded beam design problem based on Cagnina et al. (2008). . . . . 128

6.9 Number of cases out of 30 repetitions in which at least one feasible solution is obtained. Each case includes 30 initial samples and 70 adaptive ones. The first part (i.e., the established constrained EGO) in both algorithms is neglected here. . . . . 131

6.10 Best feasible results in all 30 repetitions of the welded beam design problem with settings given in Table 6.3 (results are for the version of the problem with the larger design space). . . . . 131

6.11 Percentage of adaptive iterations in all 30 repetitions in which there is at least one feasible solution within [1.724, 2.22]. In total, there are 2100 adaptive iterations, considering all 30 repetitions ( $30 * 70 = 2100$ ). 1.72485 is the best known optimum to the author of this work, as mentioned earlier. 2.22 is the best result among the response surface methods from the literature that were reviewed here, see Table 6.4. The left figure is for the version of this problem where the design space is larger (27 times) and the right figure is for the smaller case. . . . . 132

6.12 Depiction of the considered pressure vessel design problem based on Cagnina et al. (2008). . . . . 133

6.13 Number of adaptive iterations in all 30 repetitions in which there is so far at least one feasible solution within [5804, 6000]. There are 2400 adaptive iterations in total considering all 30 repetitions ( $30 * 80 = 2400$ ). 5804.45 is the best known optimum to the author of this work, as mentioned earlier. . . . . 134

6.14 Best feasible results in all 30 repetitions. 5804.45 is considered as the best known optimum in this work. . . . . 135

6.15 Depiction of feasible cases in each iteration and repetition by a circle. A blank space means at least one of the constraints was violated. The top figure represents EI-FP and bottom RF, accordingly. . . . . 136

6.16 Schematic of the considered simply supported I-beam design problem based on Ye et al. (2018). . . . . 137

6.17 Average feasible solution of the I-beam design problem (Equation (6.11)) with 30 repetitions. . . . . 137

6.18	Number of adaptive iterations in all 30 repetitions, in which there is at least one feasible solution within [0.01307,0.0145]. There are 2400 adaptive iterations in total considering all 30 repetitions ( $30 * 80 = 2400$ ). 0.01307 is the best known optimum to the author of this work. 0.0145 is just a number close to 0.01307, and this range ([0.01307,0.0145]) is selected to evaluate how well the RF method follows the desired range. . . . .	138
6.19	Top left figure shows the side view of the frontal part of the Honda Accord model from (NHTSA, Last access February 2020). Numbered components show the two load paths including components 1 to 3 and 4 to 7, respectively. The middle figure shows the lower path members in a different view, especially the 4 <sup>th</sup> and 7 <sup>th</sup> components that are optimized here. The lowest figure shows the force corridors obtained from the solution space for this lower path (Daub, 2020). The top right figure shows how to evaluate violation of the force corridors by calculating the violated area under the force curve as suggested in Burmberger (2020). . . . .	141
6.20	Component 4 deformation at the end of the 40 simulations. The left figure shows the results of the established algorithm and the right one that of the RF algorithm. The initial phase samples are shown in black circles and those of the adaptive phase in the gray squares. The large numbers 1-4 in the left figure refer to their surrounding regions in the discussion. . . . .	143
6.21	The left figure shows the objective and the right figure the constraint representing the upper bound of the force corridor of the 4 <sup>th</sup> component when optimizing with the established method. . . . .	143
6.22	Component 7 deformation at the end of the 40 simulations. The rest of the explanation is the same as that of Figure 6.20. . . . .	144
6.23	Component 7 deformation at the end of the 50 simulations. The rest of the explanation is the same as that of Figure 6.20. . . . .	144
A.1	Considered two-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. Numbers in the legend are the value of the lower multipliers in Table 5.1. The lower the value, the smaller the changes compared to the original EGO. Each plot is the average of the 30 repetitions for the iterative phase of the EGO. . . . .	155
A.2	Considered four-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1. . . . .	156
A.3	Considered six-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1. . . . .	157
A.4	Considered ten-dimensional test functions to evaluate the effect of the proposed modifications on the hyperparameter bounds. The rest of the description is the same as the one in Figure A.1. . . . .	158
B.1	Design variables of the speed reducer problem (Mehmood et al., 2016). . . . .	160

B.2 Design variables of the rolling element bearing design problem (Gupta et al., 2007). . . 161

B.3 Design variables of the Belleville disc spring design problem (Zhu et al., 2019). . . . 162

B.4 Design variables of the hydrostatic thrust bearing problem. Based on Sahin et al. (2019). 163

C.1 Frontal part of the Honda Accord model from (NHTSA, Last access February 2020).  
Two load paths are shown each containing some components. Black regions are parts  
of the component that cannot deform (block length) (Daub, 2020). . . . . 166

C.2 Top figure shows a simple, single load path solution space (C.1) model where block  
length is already removed. Below figures show the corresponding procedure to find  
the corridors for solution space (Daub, 2020) and (Zimmermann and von Hoessle,  
2013). . . . . 166

## List of Tables

2.1	Some distance-based criteria for uniformly distributing samples in the design space. $\mathbf{x}^{(j)}$ is the $j^{th}$ sample. $N$ is dimension and $M$ is the number of samples. Here, $d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)}) = \sum_{i=1}^N  \mathbf{x}_i^{(j)} - \mathbf{x}_i^{(k)} $ and $d(\mathbf{x}^{(j)}, \mathbf{x}^{(k)})^p$ is a distance function where $p > 0$ defines the distance norm. . . . .	19
2.2	List of some common covariance functions. $d$ is dimension and $r =  \mathbf{x} - \mathbf{x}' $ is a distance measure. The column "S" defines whether the covariance functions are stationary or not. Based on Table 4.1 from Rasmussen and Williams (2006). In this work, we also refer to the hyperparameter $l$ as $\theta$ . * $T = \text{diag}\left(\frac{1}{2l_1^p}, \dots, \frac{1}{2l_D^p}\right)$ , $p > 0$ , $p = 2$ is the most common. ** $\Gamma(\nu)$ is the gamma function and $K_\nu$ is the modified Bessel function. . . . .	31
4.1	Classification of design variables based on their influence on the output. For each class and ID, a certain range for the shape parameter ( $\theta$ ) is considered. . . . .	58
4.2	Classification of design variables for the considered problems. See Table 4.3 for additional settings. . . . .	62
4.3	Common settings used for all problems. . . . .	63
4.4	Settings of the side sill problem. The material model is *MAT_24 from LS-DYNA. Other required data for the plasticity part is provided in Figure 4.19. . . . .	72
5.1	The proposed procedure to define the upper and lower bounds for the hyperparameter $\theta$ . . . . .	85
5.2	Common settings used for all problems. . . . .	88
5.3	Employed Test functions. $d$ means dimension, Rosenb. means Rosenbrock, "g07 objective" is the objective in the constrained test function g07 from Runarsson and Yao (2000). See Jamil and Yang (2013) for more information. . . . .	88
5.4	Proposed modification of EI. . . . .	98
5.5	Settings of the crash box problem. The material model is *MAT_24 from LS-DYNA, where below parameters C and P are from the Cowper and Symonds strain rate model and other required data for the plasticity part is provided in Figure 5.17. . . . .	108
6.1	Common settings used for the problems in this section. . . . .	117
6.2	Range follower (RF) algorithm . . . . .	125
6.3	Settings of the RF method for the welded beam design problem. . . . .	128

6.4 The welded beam design problem comparison. NFE stands for the number of function evaluations. The first three references (12 algorithms, 19 algorithms and SCGO) have reported the version of this problem with the upper bound of two. The best known global optimum found by the author in the literature is 1.72485 (see Wang et al. (2018)), which is valid for the both versions of the problem. . . . . 130

6.5 Settings of the RF method for the pressure vessel design problem. . . . . 133

6.6 Comparison of different algorithms for the pressure vessel design problem. . . . . 135

6.7 Settings for the I-beam design problem in the proposed algorithm RF. . . . . 137

6.8 Comparison of different algorithms for the I-beam problem. . . . . 138

6.9 Force corridors' bounds and total deformation for components 4 and 7 based on the solution space, see Figure 6.19 and Equation (6.12). It should be noted that here, forces are half of what is depicted in Figure 6.19, since components 4 and 7 of only one side of the car are considered and not both. . . . . 140

6.10 Settings of the RF algorithm to optimize components 4 and 7 in Figure 6.19. . . . . 141

B.1 Variation of  $f_l(a)$  with  $a$ . . . . . 162

## Acronyms

<b>ANN</b>	Artificial neural network
<b>ARSBO</b>	Adaptive response surface based optimization
<b>BFGS</b>	Broyden–Fletcher–Goldfarb–Shanno algorithm
<b>BIPOP-CMA-ES</b>	Bi-population covariance matrix adaptation evolution strategy
<b>BOBYQA</b>	Bound optimization by quadratic approximation
<b>CDF</b>	Cumulative density function
<b>CEI</b>	Constrained expected improvement
<b>CMA-ES</b>	Covariance matrix adaptation evolution strategy
<b>CTF</b>	The constraint to be followed
<b>CV</b>	Cross-validation
<b>D or d</b>	Dimension, 10D: 10-dimensional or 10 dimensions
<b>DE</b>	Differential evolution
<b>DOE</b>	Design of Experiments
<b>DOF</b>	Degree of freedom
<b>DV</b>	Design variable
<b>EA</b>	Evolutionary algorithm
<b>EGO</b>	Efficient global optimization
<b>EI</b>	Expected improvement
<b>EI-FP</b>	Constrained EGO using expected improvement and feasibility probability
<b>ES</b>	Evolutionary strategy
<b>ESE</b>	Enhanced stochastic evolutionary
<b>ESL(M)</b>	Equivalent static load (method)
<b>EV</b>	Expected violation
<b>FEM</b>	Finite element method
<b>FP</b>	Feasibility probability
<b>GA</b>	Genetic algorithm
<b>GP(R)</b>	Gaussian process (regression)
<b>HCA</b>	Hybrid cellular automata
<b>HS</b>	Harmony search
<b>HSM</b>	Hybrid surrogate model
<b>IMQ</b>	Inverse multiquadratic
<b>LH(D)</b>	Latin Hypercube (design)
<b>LOOCV</b>	Leave one out cross-validation
<b>MA</b>	Memetic algorithm
<b>MC</b>	Monte Carlo
<b>MEI-FP</b>	Constrained EGO using modified expected improvement and feasibility probability
<b>MOR</b>	Model order reduction
<b>MQ</b>	Multiquadratic
<b>MST</b>	Minimal spanning tree
<b>MVMO</b>	Mean-variance mapping optimization
<b>NFL(T)</b>	No free lunch (theorem)



- OLH(D)** Optimal Latin Hypercube (design)
- PCA** Principal component analysis
- PDF** Probability density function
- PE** Potential energy
- PSO** Particle swarm optimization
- QMC(M)** Quasi Monte Carlo (method)
- RBF** Radial basis function
- RF** The proposed range follower algorithm
- RS(M)** Response surface (method)
- SA** Shape adaptation
- SD** Standard deviation
- SE** Squared exponential
- SI** Shape interpolation
- SLCB-EI** The proposed method using pseudo-SD and lower confidence bound concept with EI
- SQP** Sequential quadratic programming
- ST** Sample translation
- SVM** Support vector machine
- TLBO** Teaching-learning-based optimization
- WEI** Weighted expected improvement

## Bibliography

- Acar E (2010) Various approaches for constructing an ensemble of metamodels using local measures. *Struct Multidisc Optim* 42:879–896, DOI 10.1007/s00158-010-0520-z.
- Aggarwal CC, Hinneburg A, Keim DA (2001) On the surprising behavior of distance metrics in high dimensional space. In: *Database Theory — ICDT 2001*, Springer Berlin Heidelberg, pp 420–434, DOI 10.1007/3-540-44503-X\_27.
- Ambrósio J (2005) Crash analysis and dynamical behaviour of light road and rail vehicles. *Vehicle System Dynamics* 43(6-7):385–411, DOI 10.1080/00423110500151788.
- Antinori G (2017) Uncertainty analysis and robust optimization for low pressure turbine rotors. PhD thesis, Technische Universität München, Munich, Germany.
- Aranda E, Bellido JC (2016) Introduction to truss structures optimization with python. *The Electronic Journal of Mathematics and Technology* 10(1).
- Arsenyev I (2016) Efficient surrogate-based robust design optimization method. PhD thesis, Technische Universität München, Munich, Germany.
- Audet C, Booker A, Dennis J, Frank P, Moore D (2000) A surrogate-model-based method for constrained optimization. In: *8th Symposium on Multidisciplinary Analysis and Optimization*, vol 4891, DOI 10.2514/6.2000-4891.
- Auger A, Teytaud O (2007) Continuous lunches are free plus the design of optimal optimization algorithms. *Algorithmica* 57:121–146, DOI 10.1007/s00453-008-9244-5.
- Bach C (2019) Data-driven model order reduction for nonlinear crash and impact simulations. PhD thesis, Technische Universität München, Munich, Germany.
- Bagheri S, Konen W, Allmendinger R, Branke J, Deb K, Fieldsend J, Quagliarella D, Sindhya K (2017) Constraint handling in efficient global optimization. In: *Proceedings of the Genetic and Evolutionary Computation Conference*, Association for Computing Machinery, New York, NY, USA, GECCO '17, p 673–680, DOI 10.1145/3071178.3071278.
- Basudhar A, Dribusch C, Lacaze S, Missoum S (2012) Constrained efficient global optimization with support vector machines. *Struct Multidisc Optim* 46:201–221, DOI 10.1007/s00158-011-0745-5.
- Bekdaş G, Nigdeli SM, Kayabekir AE, Toklu YC (2018) Minimization of vertical deflection of an optimum i-beam by jaya algorithm. In: *AIP Conference Proceedings*, AIP Publishing LLC, vol 1978, p 260002, DOI 10.1063/1.5043887.
- Bhosekar A, Ierapetritou M (2018) Advances in surrogate based modeling, feasibility analysis, and optimization: A review. *Computers & Chemical Engineering* 108:250–267, DOI 10.1016/j.compchemeng.2017.09.017.
- Bichon BJ, Eldred MS, Swiler LP, Mahadevan S, McFarland JM (2008) Efficient global reliability analysis for nonlinear implicit performance functions. *AIAA journal* 46(10):2459–2468, DOI 10.2514/1.34321.

- Bichon BJ, Mahadevan S, Eldred M (2009) Reliability-based design optimization using efficient global reliability analysis. In: 50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference.
- Bogacki P (2019) *Linear Algebra: Concepts and Applications*. American Mathematical Society. ISBN 9781470443849.
- Bouhrel MA, Bartoli N, Regis RG, Otsmane A, Morlier J (2018) Efficient global optimization for high-dimensional constrained problems by using the kriging models combined with the partial least squares method. *Engineering Optimization* 50(12):2038–2053, DOI 10.1080/0305215X.2017.1419344.
- Boukouvala F, Ierapetritou MG (2014) Derivative-free optimization for expensive constrained problems using a novel expected improvement objective function. *AIChE Journal* 60(7):2462–2474, DOI 10.1002/aic.14442.
- Boussaïd I, Lepagnot J, Siarry P (2013) A survey on optimization metaheuristics. *Information Sciences* 237:82 – 117, DOI 10.1016/j.ins.2013.02.041.
- Bratley P, Fox BL (1988) Algorithm 659: Implementing Sobol’s quasirandom sequence generator. *ACM Trans Math Softw* 14(1):88–100, DOI 10.1145/42288.214372.
- Brell E (2005) Simplified models of vehicle impact for injury mitigation. PhD thesis, Queensland University of Technology, Brisbane, Australia.
- Burmberger L (2020) Efficient global optimization of structural components for solution spaces in vehicle crash design. Master’s thesis, Technische Universität München, Munich, Germany.
- Cagnina LC, Esquivel S, Coello Coello CA (2008) Solving engineering optimization problems with the simple constrained particle swarm optimizer. *Informatica (Slovenia)* 32:319–326.
- Cai X, Qiu H, Gao L, Li X, Shao X (2018) A hybrid global optimization method based on multiple metamodels. *Engineering Computations* 35:71–90, DOI 10.1177/1687814018769542.
- Cartis C, Roberts L, Sheridan-Methven O (2018) Escaping local minima with derivative-free methods: a numerical investigation. arXiv DOI 10.48550/ARXIV.1812.11343.
- Cartis C, Fiala J, Marteau B, Roberts L (2019) Improving the flexibility and robustness of model-based derivative-free optimization solvers. *ACM Transactions on Mathematical Software* 45(3):1–41, DOI 10.48550/ARXIV.1804.00154.
- Carvalho M, Ambrósio J, Eberhard P (2011) Identification of validated multibody vehicle models for crash analysis using a hybrid optimization procedure. *Struct Multidisc Optim* 44(1):85–97, DOI 10.1007/s00158-010-0590-y.
- Chase N, Sidhu R, Averill R (2012) A new method for efficient global optimization of large systems using sub-models: Heeds compose demonstrated on a crash optimization problem. In: LS-DYNA user forum.
- Chen Z, Qiu H, Gao L, Li X, Li P (2014) A local adaptive sampling method for reliability-based design optimization using kriging model. *Struct Multidisc Optim* 49:401–416, DOI 10.1007/s00158-013-0988-4.

- Chen Z, Peng S, Li X, Qiu H, Xiong H, Gao L, Li P (2015) An important boundary sampling method for reliability-based design optimization using kriging model. *Struct Multidisc Optim* 52:55–70, DOI 10.1007/s00158-013-0988-4.
- Cho Sg, Jang J, Kim J, Lee M, Choi JS, Hong S, Lee TH (2015) Statistical surrogate model based sampling criterion for stochastic global optimization of problems with constraints. *Journal of Mechanical Science and Technology* 29(4):1421–1427, DOI 10.1007/s12206-015-0313-9.
- Chu W, Gao X, Sorooshian S (2011a) A new evolutionary search strategy for global optimization of high-dimensional problems. *Information Sciences* 181(22):4909 – 4927, DOI 10.1016/j.ins.2011.06.024.
- Chu W, Gao X, Sorooshian S (2011b) A solution to the crucial problem of population degeneration in high-dimensional evolutionary optimization. *IEEE Systems Journal* 5(3):362–373, DOI 10.1109/JSYST.2011.2158682.
- Chunna L, Hai F, Chunlin G (2020) Development of an efficient global optimization method based on adaptive infilling for structure optimization. *Struct Multidisc Optim* 62:3383–3412, DOI 10.1007/s00158-020-02716-y.
- Cioppa TM, Lucas TW (2007) Efficient nearly orthogonal and space-filling Latin Hypercubes. *Technometrics* 49(1):45–55, DOI 10.1198/004017006000000453.
- Coello Coello CA (2000) Use of a self-adaptive penalty approach for engineering optimization problems. *Computers in Industry* 41(2):113 – 127, DOI 10.1016/S0166-3615(99)00046-9.
- Coello Coello CA (2002) Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: a survey of the state of the art. *Computer methods in applied mechanics and engineering* 191(11-12):1245–1287, DOI 10.1016/S0045-7825(01)00323-1.
- Coello Coello CA, Mezura Montes E (2002) Constraint-handling in genetic algorithms through the use of dominance-based tournament selection. *Advanced Engineering Informatics* 16(3):193–203, DOI 10.1016/S1474-0346(02)00011-3.
- Cornette D, Thirion JL, Markiewicz E, Drazetic P, Ravalard Y (1999) Localization of collapse mechanisms for simplified vehicle crash simulation. In: Batoz JL, Chedmail P, Cognet G, Fortin C (eds) *Integrated Design and Manufacturing in Mechanical Engineering*, Springer Netherlands, pp 19–26, DOI 10.1007/978-94-015-9198-0\_3.
- Daub M (2020) Optimizing flexibility for component design in systems engineering under epistemic uncertainty. PhD thesis, Technische Universität München, Munich, Germany.
- De Rainville FM, Gagn C, Teytaud O, Laurendeau D (2012) Evolutionary optimization of low-discrepancy sequences. *ACM Trans Model Comput Simul* 22(2), DOI 10.1145/2133390.2133393.
- Dimopoulos GG (2007) Mixed-variable engineering optimization based on evolutionary and social metaphors. *Computer Methods in Applied Mechanics and Engineering* 196(4):803–817, DOI 10.1016/j.cma.2006.06.010.
- Domingos P (2012) A few useful things to know about machine learning. *Commun ACM* 55(10):78–87, DOI 10.1145/2347736.2347755.

- Duddeck F, Volz K (2012) A new topology optimization approach for crashworthiness of passenger vehicles based on physically defined equivalent static loads. In: ICrash International Crashworthiness Conference, Milano, Italy.
- Duddeck F, Wehrle E (2015) Recent advances on surrogate modeling for robustness assessment of structures with respect to crashworthiness requirements. In: 10th Europ. LS-DYNA Conf.
- Durantin C, Marzat J, Balesdent M (2016) Analysis of multi-objective kriging-based methods for constrained global optimization. *Comput Optim Appl* 63:903–926, DOI 10.1007/s10589-015-9789-6.
- Dusser C, Rasigni M, Palmari J, Rasigni G, Llebaria A, Marty F (1987) Minimal spanning tree analysis of biological structures. *Journal of Theoretical Biology* 125(3):317 – 323, DOI 10.1016/s0022-5193(87)80063-2.
- Eid R (2009) Time domain model reduction by moment matching. PhD thesis, Technische Universität München, Munich, Germany.
- El Bechari R, Brisset S, Clénet S, Mipo J (2018) Enhanced meta-model-based optimization under constraints using parallel computations. *IEEE Transactions on Magnetics* 54(3):1–4, DOI 10.1109/TMAG.2017.2761997.
- Erlich I, Rueda JL, Wildenhues S, Shewarega F (2014) Solving the IEEE-CEC 2014 expensive optimization test problems by using single-particle MVMO. In: 2014 IEEE Congress on Evolutionary Computation (CEC), pp 1084–1091, DOI 10.1109/CEC.2014.6900517.
- Fang J, Sun G, Qiu N, Kim NH, Li Q (2017) On design optimization for structural crashworthiness and its state of the art. *Struct Multidisc Optim* 55:1091 – 1119, DOI 10.1007/s00158-016-1579-y.
- Faucher V, Combescure A (2003) A time and space mortar method for coupling linear modal subdomains and non-linear subdomains in explicit structural dynamics. *Computer Methods in Applied Mechanics and Engineering* 192(5):509 – 533, DOI 10.1016/S0045-7825(02)00549-2.
- Feoktistov V (2006) *Differential Evolution: In Search of Solutions*. Springer, DOI 10.1007/978-0-387-36896-2.
- Forrester AIJ, Keane AJ (2009) Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences* 45(1):50 – 79, DOI 10.1016/j.paerosci.2008.11.001.
- Forrester AIJ, Keane AJ, Bressloff NW (2006) Design and analysis of "noisy" computer experiments. *AIAA journal* 44(10):2331–2339, DOI 10.2514/1.20068.
- Forrester AIJ, Sóbester A, Keane AJ (2007) Multi-fidelity optimization via surrogate modelling. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 463(2088):3251–3269, DOI 10.1098/rspa.2007.1900.
- Forrester AIJ, Sóbester A, Keane AJ (2008) *Engineering Design via Surrogate Modelling*. Wiley, DOI 10.1002/9780470770801.
- Gandomi AH, Yang XS, Alavi AH, Talatahari S (2013) Bat algorithm for constrained optimization tasks. *Neural Computing and Applications* 22(6):1239–1255, DOI 10.1007/s00521-012-1028-9.

- Gano SE, Renaud JE, Martin JD, Simpson TW (2006) Update strategies for kriging models used in variable fidelity optimization. *Struct Multidisc Optim* 32(4):287–298, DOI 10.1007/s00158-006-0025-y.
- Garbo A, German B (2017) Adaptive sampling with adaptive surrogate model selection for computer experiment applications. In: 18th AIAA/ISSMO multidisciplinary analysis and optimization conference, p 4430, DOI 10.2514/6.2017-4430.
- Garud SS, Karimi IA, Kraft M (2017) Design of computer experiments: A review. *Computers & Chemical Engineering* 106:71–95, DOI 10.1016/j.compchemeng.2017.05.010.
- Géron A (2017) Hands-on machine learning with scikit-learn and TensorFlow: concepts, tools, and techniques to build intelligent systems. O'Reilly Media, Inc. ISBN: 9781491962299.
- Goel T, Haftka RT, Shyy W (2007) Ensemble of surrogates. *Struct Multidisc Optim* 33:199–216, DOI 10.1007/s00158-006-0051-9.
- Gupta S, Tiwari R, Nair SB (2007) Multi-objective design optimisation of rolling bearings using genetic algorithms. *Mechanism and Machine Theory* 42(10):1418–1443, DOI 10.1016/j.mechmachtheory.2006.10.002.
- Habib A, Singh HK, Ray T (2016) A study on the effectiveness of constraint handling schemes within efficient global optimization framework. In: 2016 IEEE Symposium Series on Computational Intelligence (SSCI), IEEE, pp 1–8, DOI 10.1109/SSCI.2016.7850205.
- Haftka R, Villanueva D, Chaudhuri A (2016) Parallel surrogate-assisted global optimization with expensive functions – a survey. *Struct Multidisc Optim* 54:3–13, DOI 10.1007/s00158-016-1432-3.
- Hamza K, Shalaby M (2014) A framework for parallelized efficient global optimization with application to vehicle crashworthiness optimization. *Engineering Optimization* 46(9):1200–1221, DOI 10.1080/0305215X.2013.827672.
- Hansen N (2006) The CMA Evolution Strategy: A Comparing Review. In Lozano JA, Larrañaga P, Inza I, Bengoetxea E. (eds) *Towards a New Evolutionary Computation: Advances in the Estimation of Distribution Algorithms*, vol 192, Springer Berlin Heidelberg, pp 75–102. DOI 10.1007/3-540-32494-1.
- Hansen N (2009) Benchmarking a BI-Population CMA-ES on the BBOB-2009 function testbed. In: *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference: Late Breaking Papers*, Association for Computing Machinery, New York, NY, USA, GECCO '09, p 2389–2396, DOI 10.1145/1570256.1570333.
- Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation* 9:159–195, DOI 10.1162/106365601750190398.
- Hansen N, Auger A, Ros R, Finck S, Pošík P (2010) Comparing results of 31 algorithms from the black-box optimization benchmarking bbob-2009. In: *Proceedings of the 12th Annual Conference Companion on Genetic and Evolutionary Computation*, Association for Computing Machinery, p 1689–1696, DOI 10.1145/1830761.1830790.

- Hansen N, Ros R, Mauny N, Schoenauer M, Auger A (2011) Impacts of invariance in search: When CMA-ES and PSO face ill-conditioned and non-separable problems. *Applied Soft Computing* 11(8):5755 – 5769, DOI 10.1016/j.asoc.2011.03.001.
- Harase S (2019) Comparison of Sobol' sequences in financial applications. *Monte Carlo Methods and Applications* 25(1):61 – 74, DOI doi={ 10.1515/mcma-2019-2029}.
- Harrison KR, Engelbrecht AP, Ombuki-Berman BM (2018) Self-adaptive particle swarm optimization: a review and analysis of convergence. *Swarm Intelligence* 12(3):187 – 226, DOI 10.1007/s11721-017-0150-9.
- Havinga G, van den Boogaard A, Klaseboer G (2013) Sequential optimization of strip bending process using multiquadric radial basis function surrogate models. *Key engineering materials* 554-557:911–918, DOI 10.4028/www.scientific.net/KEM.554-557.911.
- He Q, Wang L (2007) An effective co-evolutionary particle swarm optimization for constrained engineering design problems. *Engineering Applications of Artificial Intelligence* 20(1):89–99, DOI 10.1016/j.engappai.2006.03.003.
- He S, Prempain E, Wu QH (2004) An improved particle swarm optimizer for mechanical design optimization problems. *Engineering optimization* 36(5):585–605, DOI 10.1080/03052150410001704854.
- Hedar AR, Fukushima M (2006) Derivative-free filter simulated annealing method for constrained continuous global optimization. *J Glob Optim* 35(4):521–549, DOI 10.1007/s10898-005-3693-z.
- Hefele R (2017) A multi-fidelity approach using physical and mathematical surrogates for crash optimization. Master's thesis, Technische Universität München, Munich, Germany.
- Hefele R (2020) Dimension adaptive efficient global optimization for expensive blackbox problems. Bachelor's thesis, Technische Universität München, Munich, Germany.
- Herman J, Usher W (2017) Salib: An open-source python library for sensitivity analysis. *Journal of Open Source Software* 2(9):97, DOI 10.21105/joss.00097.
- Hillis WD (1990) Co-evolving parasites improve simulated evolution as an optimization procedure. *Physica D: Nonlinear Phenomena* 42(1-3):228–234, DOI 10.1016/0167-2789(90)90076-2.
- Holmström K, Quttineh N, Edvall M (2008) An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization. *Optim Eng* 9:311–339, DOI 10.1007/s11081-008-9037-3.
- Hunkeler S, Duddeck F, Rayamajhi M, Zimmer H (2013) Shape optimisation for crashworthiness followed by a robustness analysis with respect to shape variables. *Struct Multidisc Optim* 48:367–378, DOI 10.1007/s00158-013-0903-z.
- Iuliano E (2011) Aerodynamic shape optimization with physics-based surrogate models. PhD thesis, Università degli Studi di Napoli Federico II, Naples, Italy.
- Jamil M, Yang XS (2013) A literature survey of benchmark functions for global optimisation problems. *International Journal of Mathematical Modelling and Numerical Optimisation* 4(2):150, DOI 10.1504/ijmmno.2013.055204.

- Jin R, Chen W, Sudjianto A (2005) An efficient algorithm for constructing optimal design of computer experiments. *Journal of Statistical Planning and Inference* 134(1):268–287, DOI 10.1016/j.jspi.2004.02.014.
- Joe S, Kuo FY (2003) Remark on algorithm 659: Implementing Sobol’s quasirandom sequence generator. *ACM Trans Math Softw* 29(1):49–57, DOI 10.1145/641876.641879.
- Joe S, Kuo FY (2008) Constructing Sobol sequences with better two-dimensional projections. *SIAM Journal on Scientific Computing* 30(5):2635–2654, DOI 10.1137/070709359.
- Johnson ME, Moore LM, Ylvisaker D (1990) Minimax and maximin distance designs. *Journal of Statistical Planning and Inference* 26(2):131 – 148, DOI 10.1016/0378-3758(90)90122-B.
- Jones DR (2001) A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization* 21:345–383, DOI 10.1023/A:1012771025575.
- Joyce T, Herrmann JM (2018) *A Review of No Free Lunch Theorems, and Their Implications for Metaheuristic Optimisation*, Springer International Publishing, pp 27–51. DOI 10.1007/978-3-319-67669-2\_2.
- Kamel H, Sedaghati R, Medraj M (2008) An efficient crashworthiness design optimization approach for frontal automobile structures. In: *ASME International Mechanical Engineering Congress and Exposition*, vol 48784, pp 31–35, DOI 10.1115/IMECE2008-66760.
- Kandasamy K, Schneider J, Póczos B (2015) High dimensional Bayesian optimisation and bandits via additive models. In: *Proceedings of the 32nd International Conference on Machine Learning - Volume 37*, JMLR.org, ICML’15, p 295–304, DOI 10.48550/ARXIV.1503.01673.
- Keane A, Nair P (2005) *Computational Approaches for Aerospace Design: The Pursuit of Excellence*. Wiley, DOI 10.1002/0470855487.
- Kim YI, Park GJ, Kolonay RM, Blair M, Canfield RA (2008) Nonlinear response structural optimization of a joined wing using equivalent loads. *AIAA journal* 46(11):2703–2713, DOI 10.2514/1.33428.
- Komeilizadeh K, Hefele R, Duddeck F (2018) A multi-fidelity approach using physical and mathematical surrogates for crash optimization. *PAMM* 18(1):e201800,287, DOI 10.1002/pamm.201800287.
- Koutsovasilis P (2009) *Model order reduction in structural mechanics coupling the rigid and elastic multi body dynamics*. PhD thesis, Technische Universität Dresden, Dresden, Germany.
- Kuhn M, Johnson K (2013) *Applied Predictive Modeling*. Springer, DOI 10.1007/978-1-4614-6849-3.
- Kumar U, Soman S, Jayadeva (2016) Benchmarking NLopt and state-of-the-art algorithms for continuous global optimization via IACOR. *Swarm and Evolutionary Computation* 27:116–131, DOI 10.1016/j.swevo.2015.10.005.
- Lam R, Poloczek M, Frazier P, Willcox KE (2018) Advances in Bayesian optimization with applications in aerospace engineering. In: *2018 AIAA Non-Deterministic Approaches Conference*, p 1656, DOI 10.2514/6.2018-1656.



- Lange VA, Fender J, Song L, Duddeck F (2019) Early phase modeling of frontal impacts for crashworthiness: From lumped mass–spring models to deformation space models. *Proceedings of the Institution of Mechanical Engineers, Part D: Journal of Automobile Engineering* 233(12):3000–3015, DOI 10.1177/0954407018814034.
- Larson J, Menickelly M, Wild SM (2019) Derivative-free optimization methods. *Acta Numerica* 28:287–404, DOI 10.1017/s0962492919000060.
- Laurent L, Le Riche R, Soulier B, Boucard PA (2017) An overview of gradient-enhanced meta-models with applications. *Archives of Computational Methods in Engineering* 26:61–106, DOI 10.1007/s11831-017-9226-3.
- Lee SB, Park JR, Yim HJ (2002) Numerical approximation of vehicle joint stiffness by using response surface method. *Int J Automot Technol* 3(3):117–122.
- Lee SH, Kim HY, Oh SI (2002) Cylindrical tube optimization using response surface method based on stochastic process. *Journal of Materials Processing Technology* 130:490–496, DOI 10.1016/S0924-0136(02)00794-X.
- Li C, Gupta S, Rana S, Nguyen V, Venkatesh S, Shilton A (2017a) High dimensional Bayesian optimization using dropout. In: *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pp 2096–2102, DOI 10.48550/arXiv.1802.05400.
- Li Y, Wu Y, Zhao J, Chen L (2017b) A kriging-based constrained global optimization algorithm for expensive black-box functions with infeasible initial points. *Journal of Global Optimization* 67(1-2):343–366, DOI 10.1007/s10898-016-0455-z.
- Li Z, Ruan S, Gu J, Wang X, Shen C (2016) Investigation on parallel algorithms in efficient global optimization based on multiple points infill criterion and domain decomposition. *Struct Multidiscip Optim* 54(4):747–773, DOI 10.1007/s00158-016-1441-2.
- Liang JJ, Suganthan PN, Deb K (2005) Novel composition test functions for numerical global optimization. In: *Proceedings 2005 IEEE Swarm Intelligence Symposium, 2005. SIS 2005.*, pp 68–75, DOI 10.1109/SIS.2005.1501604.
- Lichtenstern A (2013) Kriging methods in spatial statistics. Bachelor’s thesis, Department of Mathematics, Technische Universität München, Munich, Germany.
- Liu Y (2005) Development of simplified models for crashworthiness analysis. PhD thesis, University of Louisville, Louisville, USA.
- Long T, Wu D, Guo X, Wang G, Liu L (2015) Efficient adaptive response surface method using intelligent space exploration strategy. *Struct Multidisc Optim* 51:1335–1362, DOI 10.1007/s00158-014-1219-3.
- MacDonald B, Ranjan P, Chipman H (2015) Gpfit: An R package for fitting a Gaussian process model to deterministic simulator outputs. *Journal of Statistical Software* 64(12), DOI 10.18637/jss.v064.i12.
- Mahdavi S, Shiri ME, Rahnamayan S (2015) Metaheuristics in large-scale global continuous optimization. *Inf Sci* 295(C):407–428, DOI 10.1016/j.ins.2014.10.042.

- Marrel A, Iooss B, Dorpe FV, Volkova E (2008) An efficient methodology for modeling complex computer codes with Gaussian processes. *Computational Statistics & Data Analysis* 52(10):4731 – 4744, DOI 10.48550/ARXIV.0802.1099.
- Martin JD, Simpson TW (2005) Use of kriging models to approximate deterministic computer models. *AIAA journal* 43(4):853–863, DOI 10.2514/1.8650.
- Martins JRR, Ning A (2021) *Engineering Design Optimization*. Cambridge University Press, DOI 10.1017/9781108980647.
- Marzbanrad J, Pahlavani M (2011) A system identification algorithm for vehicle lumped parameter model in crash analysis. *International Journal of Modeling and Optimization* 1(2):163–168, DOI 10.7763/IJMO.2011.V1.29.
- McBride K, Sundmacher K (2019) Overview of surrogate modeling in chemical process engineering. *Chemie Ingenieur Technik* 91(3):228–239, DOI 10.1002/cite.201800091.
- Mehmood R, Qazi MH, Ata H, Zaheer R (2016) Golinski's speed reducer problem revisited using genetic algorithm. *American Scientific Research Journal for Engineering, Technology, and Sciences (ASRJETS)* 16(1):55–65.
- Močkus J (1975) On Bayesian methods for seeking the extremum. In: *Optimization techniques IFIP technical conference*, Springer, pp 400–404, DOI 10.1007/3-540-07165-2\_55.
- Molina D, Moreno-García F, Herrera F (2017) Analysis among winners of different IEEE CEC competitions on real-parameters optimization: Is there always improvement? In: *2017 IEEE Congress on Evolutionary Computation (CEC)*, pp 805–812, DOI 10.1109/CEC.2017.7969392.
- Molina D, Latorre A, Herrera F (2018) An insight into bio-inspired and evolutionary algorithms for global optimization: Review, analysis, and lessons learnt over a decade of competitions. *Cognitive Computation* 10:1–28, DOI 10.1007/s12559-018-9554-0.
- Montgomery D (2012) *Design and Analysis of Experiments*. Wiley, DOI 10.1002/ep.11743.
- Morgan R, Gallagher M (2014) Sampling techniques and distance metrics in high dimensional continuous landscape analysis: Limitations and improvements. *IEEE Transactions on Evolutionary Computation* 18(3):456–461, DOI 10.1109/TEVC.2013.2281521.
- Morris MD, Mitchell TJ (1995) Exploratory designs for computational experiments. *Journal of Statistical Planning and Inference* 43(3):381 – 402, DOI 10.1016/0378-3758(94)00035-T.
- NHTSA (Last access February 2020) Honda accord crash simulation model. URL <https://www.nhtsa.gov/document/honda-accord-lsdyna-zip>.
- Nigdeli SM, Bekdaş G, Yang XS (2016) Application of the flower pollination algorithm in structural engineering, Springer, pp 25–42. DOI 10.1007/978-3-319-26245-1\_2.
- Nocedal J, Wright SJ (2006) Penalty and augmented Lagrangian methods. *Numerical Optimization* pp 497–528, DOI 10.1007/978-3-319-58356-3\_7.
- Pan F, Zhu P, Zhang Y (2010) Metamodel-based lightweight design of b-pillar with twb structure via support vector regression. *Computers & Structures* 88(1):36 – 44, DOI 10.1016/j.compstruc.2009.07.008.

- Park GJ (2011) Technical overview of the equivalent static loads method for non-linear static response structural optimization. *Struct Multidisc Optim* 43(3):319–337, DOI 10.1007/s00158-010-0530-x.
- Parkinson AR, Balling RJ, Hedengren JD (2018) *Optimization Methods for Engineering Design*. Brigham Young University, URL <http://apmonitor.com/me575/index.php/Main/BookChapters>.
- Parnianifard A, Chanchaen R, Phanomchoeng G, Wuttisittikulij L (2020) A new approach for low-dimensional constrained engineering design optimization using design and analysis of simulation experiments. *International Journal of Computational Intelligence Systems* 13:1663–1678, DOI 10.2991/ijcis.d.201014.001.
- Parr J, Holden CME, Forrester AIJ, Keane AJ (2010) Review of efficient surrogate infill sampling criteria with constraint handling. In: 2nd International conference on engineering optimization, pp 1–10.
- Parr JM, Forrester AIJ, Keane AJ, Holden CME (2012) Enhancing infill sampling criteria for surrogate-based constrained optimization. *Journal of Computational Methods in Sciences and Engineering* 12(1-2):25–45, DOI 10.1080/0305215X.2011.637556.
- Pedregosa F, Varoquaux G, Gramfort A, Michel V, Thirion B, Grisel O, Blondel M, Prettenhofer P, Weiss R, Dubourg V, Vanderplas J, Passos A, Cournapeau D, Brucher M, Perrot M, Duchesnay E (2011) Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research* 12:2825–2830.
- Picheny V (2014) A stepwise uncertainty reduction approach to constrained global optimization. In: *Artificial Intelligence and Statistics*, pp 787–795.
- Picheny V, Ginsbourger D, Roustant O, Haftka RT, Kim NH (2010) Adaptive designs of experiments for accurate approximation of a target region. *Journal of Mechanical Design* 132(7), DOI 10.1115/1.4001873, 071008.
- Picheny V, Wagner T, Ginsbourger D (2013) A benchmark of kriging-based infill criteria for noisy optimization. *Struct Multidisc Optim* 48:607–626, DOI 10.1007/s00158-013-0919-4.
- Pospěšilová A, Lepš M (2012) Global optima for size optimization benchmarks by branch and bound principles. *Acta Polytechnica*, Vol 52 No 6/2012 DOI 10.14311/1682.
- Pošík P, Huyer W (2012) Restarted local search algorithms for continuous black box optimization. *Evol Comput* 20(4):575–607, DOI 10.1162/EVCO\_a\_00087.
- Powell MJD (2009) The bobyqa algorithm for bound constrained optimization without derivatives. Tech. Rep. NA2009/06, Department of Applied Mathematics and Theoretical Physics, University of Cambridge.
- Preuss M (2015) *Multimodal Optimization by Means of Evolutionary Algorithms*. Springer Series in Natural Computing, Springer, DOI 10.1007/978-3-319-07407-8.
- Pronzato L (2017) Minimax and maximin space-filling designs: some properties and methods for construction. *Journal de la Société Française de Statistique* 158(1):7–36.

- Qian H, Hu YQ, Yu Y (2016) Derivative-free optimization of high-dimensional non-convex functions by sequential random embeddings. In: Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI'16, p 1946–1952.
- Ranjan P, Bingham D, Michailidis G (2008) Sequential experiment design for contour estimation from complex computer codes. *Technometrics* 50(4):527–541, DOI 10.1198/0040170080000000541.
- Rao RV, Savsani VJ (2012) *Mechanical Design Optimization Using Advanced Optimization Techniques*. Springer Series in Advanced Manufacturing, Springer, DOI 10.1007/978-1-4471-2748-2.
- Rao RV, Savsani VJ, Vakharia DP (2011) Teaching–learning-based optimization: A novel method for constrained mechanical design optimization problems. *Computer-Aided Design* 43(3):303–315, DOI 10.1016/j.cad.2010.12.015.
- Rao SS (2009) *Engineering Optimization: Theory and Practice*. Wiley, DOI 10.1002/9781119454816.
- Raponi E, Bujny M, Olhofer M, Boria S, Duddeck F (2019) Hybrid strategy coupling EGO and CMA-ES for structural topology optimization in statics and crashworthiness. In: International joint conference on computational intelligence, Springer, pp 55–84, DOI 10.1007/978-3-030-70594-7\_3.
- Raponi E, Fiumarella D, Boria S, Scattina A, Belingardi G (2021) Methodology for parameter identification on a thermoplastic composite crash absorber by the sequential response surface method and efficient global optimization. *Composite Structures* 278:114,646, DOI 10.1016/j.compstruct.2021.114646.
- Rasmussen CE, Williams CKI (2006) *Gaussian Processes for Machine Learning*. MIT Press, DOI 10.7551/mitpress/3206.001.0001.
- Raveh A (1985) On the use of the inverse of the correlation matrix in multivariate data analysis. *The American Statistician* 39(1):39–42, DOI 10.2307/2683904.
- Rios LM, Sahinidis NV (2013) Derivative-free optimization: a review of algorithms and comparison of software implementations. *Journal of Global Optimization* 56(3):1247–1293, DOI 10.1007/s10898-012-9951-y.
- Rowan TH (1990) *Functional stability analysis of numerical algorithms*. PhD thesis, The University of Texas at Austin, Austin, USA.
- Runarsson TP, Yao X (2000) Stochastic ranking for constrained evolutionary optimization. *IEEE Transactions on evolutionary computation* 4(3):284–294, DOI 10.1109/4235.873238.
- Sadollah A, Bahreininejad A, Eskandar H, Hamdi M (2013) Mine blast algorithm: A new population based algorithm for solving constrained engineering optimization problems. *Applied Soft Computing* 13(5):2592–2612, DOI 10.1016/j.asoc.2012.11.026.
- Sahin I, Dörterler M, Gökce H (2019) Optimization of hydrostatic thrust bearing using enhanced grey wolf optimizer. *Mechanika/Mechanics* 25(6):480–486, DOI 10.5755/j01.mech.25.6.22512.

- Sandgren E (1990) Nonlinear Integer and Discrete Programming in Mechanical Design Optimization. *Journal of Mechanical Design* 112(2):223–229, DOI 10.1115/1.2912596.
- Sasena MJ (2002) Flexibility and efficiency enhancements for constrained global design optimization with kriging approximations. PhD thesis, University of Michigan, Ann Arbor, USA.
- Sasena MJ, Papalambros PY, Goovaerts P (2001) The use of surrogate modeling algorithms to exploit disparities in function computation time within simulation-based optimization. In: *The Fourth World Congress of Structural and Multidisciplinary Optimization*.
- Sasena MJ, Papalambros P, Goovaerts P (2002) Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization* 34(3):263–278, DOI 10.1080/03052150211751.
- Schonlau M (1997) Computer experiments and global optimization. PhD thesis, University of Waterloo, Waterloo, Canada.
- Schonlau M, Welch WJ, Jones DR (1998) Global versus local search in constrained optimization of computer models. *Lecture Notes-Monograph Series* 34:11–25, DOI 10.1214/lnms/1215456182.
- Shan S, Wang GG (2010) Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Struct Multidisc Optim* 41(2):219–241, DOI 10.1007/s00158-009-0420-2.
- Shin MK, Park KJ, Park GJ (2007) Optimization of structures with nonlinear behavior using equivalent loads. *Computer Methods in Applied Mechanics and Engineering* 196(4):1154 – 1167, DOI 10.1016/j.cma.2006.09.001.
- Singh A, Jana ND (2017) A survey on metaheuristics for solving large scale optimization problems. *International Journal of Computer Applications* 170(5):1–7, DOI 10.5120/ijca2017914839.
- Sobester A (2003) Enhancements to global design optimization techniques. PhD thesis, University of Southampton, Southampton, England.
- Sotoudeh-Anvari A, Hafezalkotob A (2018) A bibliography of metaheuristics-review from 2009 to 2015. *International Journal of Knowledge-based and Intelligent Engineering Systems* 22(1):83–95, DOI 10.3233/KES-180376.
- Sousa L, Veríssimo P, Ambrósio J (2008) Development of generic multibody road vehicle models for crashworthiness. *Multibody System Dynamics* 19(1):133–158, DOI 10.1007/s11044-007-9093-z.
- Stander N, Basudhar A, Roux W, Witowski K (2019) *LS-OPT User’s Manual, Version 6.0*. LIVERMORE SOFTWARE.
- Stolpe M, Verbart A, Rojas-Labanda S (2018) The equivalent static loads method for structural optimization does not in general generate optimal designs. *Struct Multidisc Optim* 58:139–154, DOI 10.1007/s00158-017-1884-0.
- Sun G, Wang X, Fang J, Pang T, Li Q (2021) Parallelized optimization design of bumper systems under multiple low-speed impact loads. *Thin-Walled Structures* 167:108,197, DOI 10.1016/j.tws.2021.108197.

- Surjanovic S, Bingham D (2013) Virtual library of simulation experiments: Test functions and datasets. Retrieved June 26, 2020. URL <https://www.sfu.ca/~ssurjano/hart6.html>.
- Toal DJJ, Forrester AIJ, Bressloff NW, Keane AJ, Holden C (2009) An adjoint for likelihood maximization. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences* 465(2111):3267–3287, DOI 10.1098/rspa.2009.0096.
- Ulmasov D, Baroukh C, Chachuat B, Deisenroth MP, Misener R (2016) Bayesian optimization with dimension scheduling: Application to biological systems. *26th European Symposium on Computer Aided Process Engineering* p 1051–1056, DOI 10.48550/arXiv.1511.05385.
- Verleysen M (2003) Learning high-dimensional data. *Nato Science Series Sub Series III Computer And Systems Sciences* 186:141–162.
- Viana FAC (2013) Things you wanted to know about the Latin Hypercube design and were afraid to ask. In: *10th World Congress on Structural and Multidisciplinary Optimization*.
- Viana FAC, Haftka RT (2009) Importing uncertainty estimates from one surrogate to another. In: *50th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, DOI 10.2514/6.2009-2237.
- Viana FAC, Haftka RT, Watson LT (2013) Efficient global optimization algorithm assisted by multiple surrogate techniques. *J Glob Optim* 56:669–689, DOI 10.1007/s10898-012-9892-5.
- Virtanen P, Gommers R, Oliphant TE, Haberland M, Reddy T, Cournapeau D, Burovski E, Peterson P, Weckesser W, Bright J, van der Walt SJ, Brett M, Wilson J, Millman KJ, Mayorov N, Nelson ARJ, Jones E, Kern R, Larson E, Carey CJ, Feng Y, Moore EW, VanderPlas J, Laxalde D, Perktold J, Cimrman R, Henriksen I, Quintero EA, Harris CR, Archibald AM, Ribeiro AH, Pedregosa F, van Mulbregt P, SciPy 10 Contributors (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17:261–272, DOI 10.1038/s41592-019-0686-2.
- Vořechovský M, Novák D (2009) Correlation control in small-sample Monte Carlo type simulations i: A simulated annealing approach. *Probabilistic Engineering Mechanics* 24(3):452–462, DOI 10.1016/j.probengech.2009.01.004.
- Wang GG (2003) Adaptive Response Surface Method Using Inherited Latin Hypercube Design Points. *Journal of Mechanical Design* 125(2):210–220, DOI 10.1115/1.1561044.
- Wang GG, Simpson T (2004) Fuzzy clustering based hierarchical metamodeling for design space reduction and optimization. *Engineering Optimization* 36(3):313–335, DOI 10.1080/03052150310001639911.
- Wang H, Hu Z, Sun Y, Su Q, Xia X (2018) Modified backtracking search optimization algorithm inspired by simulated annealing for constrained engineering optimization problems. *Computational intelligence and neuroscience* 2018, DOI 10.1155/2018/9167414.
- Wang JM, Fleet DJ, Hertzmann A (2007) Gaussian process dynamical models for human motion. *IEEE transactions on pattern analysis and machine intelligence* 30(2):283–298, DOI 10.1109/TPAMI.2007.1167.

- Wang Z, Zoghi M, Hutter F, Matheson D, De Freitas N (2013) Bayesian optimization in high dimensions via random embeddings. In: Proceedings of the 23rd International Joint Conference on Artificial Intelligence, AAAI Press, p 1778–1784, DOI 10.48550/arXiv.1301.1942.
- Weise T (2011) Global Optimization Algorithms - Theory and Application, 3rd edition. Self-Published, URL <http://sourceforge.net/projects/goa-taa/>.
- Woldemichael DE, Woldeyohannes AD (2016) Optimization of pressure vessel design using pyopt. ARPN Journal of Engineering and Applied Sciences 11(24):14,264–14,268.
- Wolpert D, Macready W (2006) Coevolutionary free lunches. Evolutionary Computation, IEEE Transactions on 9:721 – 735, DOI 10.1109/TEVC.2005.856205.
- Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Transactions on Evolutionary Computation 1(1):67–82, DOI 10.1109/4235.585893.
- Yalcin Y, Pekcan O (2018) Nuclear fission–nuclear fusion algorithm for global optimization: a modified big bang–big crunch algorithm. Neural Computing and Applications 32:2751–2783, DOI 10.1007/s00521-018-3907-1.
- Yalcin Y, Pekcan O (2020) Nuclear fission–nuclear fusion algorithm for global optimization: a modified big bang–big crunch algorithm. Neural Computing and Applications 32(7):2751–2783, DOI 10.1007/s00521-018-3907-1.
- Yang XS (2012) Swarm-based metaheuristic algorithms and no-free-lunch theorems. Theory and new applications of swarm intelligence 9:1–16, DOI 10.5772/30852.
- Ye KQ (1998) Orthogonal column Latin Hypercubes and their application in computer experiments. Journal of the American Statistical Association 93(444):1430–1439, DOI 10.2307/2670057.
- Ye P, Pan G, Dong Z (2018) Ensemble of surrogate based global optimization methods using hierarchical design space reduction. Struct Multidisc Optim 58:537–554, DOI 10.1007/s00158-018-1906-6.
- Zeng D, Duddeck F (2017) Improved hybrid cellular automata for crashworthiness optimization of thin-walled structures. Struct Multidisc Optim 56(1):101–115, DOI 10.1007/s00158-017-1650-3.
- Zhang F (2005) The Schur Complement and Its Applications. Springer, DOI 10.1007/b105056.
- Zhang J, Chowdhury S, Messac A (2012) An adaptive hybrid surrogate model. Struct Multidisc Optim 46(2):223–238, DOI 10.1007/s00158-012-0764-x.
- Zhu H, Hu Y, Zhu W (2019) A dynamic adaptive particle swarm optimization and genetic algorithm for different constrained engineering design optimization problems. Advances in Mechanical Engineering 11(3):1687814018824,930, DOI 10.1177/1687814018824930.
- Zimmermann M, von Hoessle JE (2013) Computing solution spaces for robust design. International Journal for Numerical Methods in Engineering 94(3):290–307, DOI 10.1002/nme.4450.