Sequential Parameter Optimization for Mixed-Discrete Problems

Thomas Bartz-Beielstein
thomas.bartz-beielstein@th-koeln.de

Lorenzo Gentile
lorenzo.gentile@th-koeln.de

Martin Zaefferer
martin.zaefferer@th-koeln.de

Contents

1 Introduction **1**

2 Problem Definition **2**

3 Challenges in Real-World Optimization **2**
3.1 Problem Features 2
3.2 High Dimensionality 3
3.2.1 Screening 4
3.2.2 Mapping 5
3.3 Uncertainty 5

4 Sequential Parameter Optimization **6**
4.1 Initial Design 7
4.1.1 Strategies for Design of Experiment 7
4.1.2 Latin Hypercube Sampling 7
4.1.3 Factorial Designs 7
4.2 Modeling 8
4.2.1 Modelling in Mixed-Integer Space 8
4.2.2 The Naive Approach 9
4.2.3 Inherently Discrete Models 9
4.2.4 Similarity-based Models 10
4.2.5 Handling Factor Variables in Kriging Model 10
4.3 Optimization Algorithms for the Metamodel 11
4.4 MIES 12

5 Case Study: Optimization of Composite Multi-layered Plate **13**
5.1 Overview 13
5.2 Optimization problem 14
5.3 Methodology 15
5.4 Results 16

1 Introduction

Many real-world optimization problems consider the optimization of ordinal integers, categorical integers, binary variables, permutations, strings, trees, or graphs structures in general. These real-world problems pose complex search spaces which require a deep understanding of the underlying solution representations. Some of them, for example integers, are more suitable to be treated by classic optimization algorithms. Others, such as trees, have to be handled by specifically developed optimization algorithms. In general, solving these kinds of problems usually necessitates a significant number of objective function evaluations. However, in many engineering problems, a single evaluation is based on either on experimental or numerical analysis. This causes significant costs with respect to time or resources. Surrogate model-based optimization (SMBO) aims to handle the complex variable structures and the limited budget simultaneously. Sequential Parameter Optimization (SPO) pursues the identification of global optima making advantage of a budget allocation process that maximizes the information gaining in promising regions. This chapter aims to show

an efficient method to face mixed-discrete optimization problems utilizing SPO. Particularly, the chapter is structured as follows: Sec. 2 will introduce the problem definition, Sec. 3 describes the challenges that are common in this problem domain, Sec. 4 contains a thorough description of SPO, and finally an application of SPO on real-world discrete-mixed optimization problem is presented in Sec. 5 .

2 Problem Definition

Optimization can be seen as the process of searching for the best candidate solution in the search space, that maximizes or minimizes an objective function. Without loss of generality, we refer to optimization as a minimization process. In this chapter, we will focus on describing problems including real-valued variables, ordinal integers, and categorical (uncountable) variables.

Let $f : \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z} \times \mathbb{D}^{n_d} \to \mathbb{R}$ denote the objective function to be optimized, $g_j : \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z} \times \mathbb{D}^{n_d} \to \mathbb{R}, 1 \leq j \leq n_g$ the inequality constraints, and $h_k : \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z} \times \mathbb{D}^{n_d} \to \mathbb{R}, 1 \leq k \leq n_h$ the equality constraints.

The problem of mixed-discrete optimization can be formalized as follows:

$$\begin{aligned} &\min_x f(x) \quad \text{where } x \in \mathbb{R}^{n_r} \times \mathbb{Z}^{n_z} \times \mathbb{D}^{n_d} \\ &\text{subject to} \begin{cases} g_j \leq 0, 1 \leq j \leq n_g \\ h_k = 0, 1 \leq k \leq n_h \\ r_i \in [r_i^{\min}, r_i^{\max}], 1 \leq i \leq n_r \\ z_i \in [z_i^{\min}, z_i^{\max}], 1 \leq i \leq n_z \\ d_i = \left\{ d_i^1, ..., d_i^{N_i} \right\}, 1 \leq i \leq n_d \end{cases} \end{aligned} \tag{1}$$

where $r_i^{\min}$ and $r_i^{\max}$ define the lower and upper bounds that the n_r real variables r_i can assume, $z_i^{\min}$ and $z_i^{\max}$ define the lower and upper bounds that the n_z integer variables z_i can assume, d_i is the set of the possible values that the i-th discrete variable can assume, and finally n_d is the number of discrete variables. The input variables will be referred to design variables.

3 Challenges in Real-World Optimization

3.1 Problem Features

Over the years, a large number of optimization methods have been proposed and new algorithms are developed every day to improve their general performance. However, it has been stated by Wolpert and Macready [57] that any algorithm's improved performance over one class of problems is offset by a performance loss over another class. Hence, the identification of problem features becomes a crucial stage in the development and selection of optimization algorithms. Among all potential problem features, the ones that mostly affect the performances in mixed-discrete optimization can be listed as follows: high-dimensionality, uncertainties, computationally expensive evaluations, complex landscapes, and black-box problems [55].

Here, "black-box" implies that no knowledge about the function is available, and any knowledge can only be derived by evaluating the function itself. It is often impossible to predict the response of the function because the physical phenomena are not fully understood or the modeling strategy leads to bias and undesired, unknown sensitivities. The integration of optimization methodologies with computational analysis and simulations is of some importance in this context. This lack of knowledge is especially problematic when function evaluations are expensive: Black-box optimization processes inherently require numerous evaluations of objective functions [29]. Therefore, although numerous non-gradient optimization methods are available for cheap black-box functions, more sophisticated methods are necessary to deal with limited evaluation budgets.

Determining whether the best solution currently known is a local or a global optimum is often difficult. This issue typically arises for multi-modal problems (if the function has multiple maxima and minima). Moreover, difficult fitness landscapes may exhibit *deceptiveness* [10]. Deceptive objective functions can trap the optimizer by a large basin of attraction, which leads the search process away from the global optimum in favour of a local one.

If an area with a better average fitness compared to other regions has been found, the optimization algorithm will consider it as promising and will focus on the exploration of this region. This assumes that such areas are likely to contain the true optimum. Hence, developing an algorithm that is able to interpret the function response correctly is a demanding task.

In many cases, this problem can be solved by choosing the correct optimization strategy and performing a preliminary algorithm tuning. For example, a population-based optimizer's ability to distinguish the global optimum from a local optimum often relies on the chosen population size. Moreover, maintaining diversity in the population helps to avoid premature convergence [55]. It is also clear that the optimality of the algorithms' parameters changes during the optimization process in case of multi-modal problems: In the beginning, algorithms should be more explorative. This leads to a fastest identification of all the promising areas and would help escaping misleading local-minima. On the contrary, at the end of the process, exploitation would gain more importance, assuming that the most promising area has already been identified. For these reasons, algorithms able to auto-tune (i.e., perform on-the-fly parameter control) all along the optimization process, such as [40, 31], can be a promising choice. More details on auto-tune can be found in Chapter 11. The combination of these features strongly increases the problem difficulty. Optimization algorithms need to be designed to solve specific problems, presenting different combinations of these features. We will focus on two essential problems in this chapter, namely the dimensionality and the uncertainty.

3.2 High Dimensionality

The dimensionality of the search space is defined by the number of design variables. Referring to the notation in Eq. (1), we define the dimensionality as $n = n_r + n_z + n_d$. It is intuitive that a large number of variables poses an demanding challenge that affects many algorithm's aspects. Dealing with this particular problem requires a great modeling capability, a huge amount of acquired data and, consequently a large budget of objective function evaluations.

Every modeling technique requires a sufficiently large data-set such that an accurate model can be trained.

Furthermore, high dimensionality leads to severe practical issues in the development of surrogate-models as well. Kriging, besides linear regression, is one of the most popular techique in SMBO, see [9]. For example, depending on the employed distance measures [1], it is widely recognized that Kriging may perform poorly for problems with more than approximately 20 variables [20].

A spectrum of countermeasures to these issues comes from different fields of engineering and data analytics. Most commonly, methods attempt to use some screening or mapping approach. The former attempts to remove insignificant variables, while the latter attempts to map the original search space to a low-dimensional subspace.

3.2.1 Screening

In the effort of reducing the problem complexity and dimensionality, screening identifies and retains important input variables and interaction terms. Screening is often implemented via sampling and the analysis of sampling results [46].

Sensitivity analysis studies how the variability of a function's output responds to changes of its inputs. It includes local and global sensitivity analyses. The local sensitivity indicates the variability of the output with respect to input variable changes at a given point, in other words it evaluates the numeric partial derivatives. The global sensitivity, contrarily, explains the global variability of the output over the entire design space, which provides an overall view of the impact of input variables on the output. One example of sensitivity analysis applied to aircraft design is given in [48].

A classic method for screening and sensitivity analysis using experimental designs is the modeling and analysis of regression models [15]. Common examples are the analysis of p-values in linear regression, mean decrease impurity in random forests [12], and the theta values of Kriging models [20].

For example: we consider a simplified variant of the optimization problem introduced in Eq. 1. The input variables are real valued and no equality and inequality constraints are imposed. In this situation we obtain the following optimization problem:

$$\min_{\vec{x} \in \mathbb{R}^2} f(\vec{x})$$

with $\vec{x} = (x_1, x_2)$ we obtain the linear regression model

$$\widehat{f}(x_1, x_2) = \beta_0 + \beta_1 x_1 + \beta_2 x_2$$

The coefficients β_1 and β_2 are interpreted as the estimated change in the objective function corresponding to one unit change in a variable, when all the other variables are held constant [37]. The p-values [37] for the coefficients indicate whether these relationships are statistically significant. Intuitively, it is possible to set a threshold value for the p-values over which the corresponding variables are considered uncorrelated to the objective function and thus, can be neglected. In the scope of linear regression, screening designs have been developed, which are applicable in the context of small function evaluation budgets [52].

A model-independent approach for variable screening that varies one-factor-at-a-time has been proposed by Morris [39]. This method, unlike the stepwise

variable selection [25], requires a number of model evaluations that are linearly-dependent by the number of design variables. This strategy aims to estimate the overall effect of the variables and the ensemble of the second-order and higher-order effects addressing two sensitivity measures per design variable [43].

Screening processes can be also directly integrated in the meta-model building. A cross-validated moving least squares approach in which one variable of the problem design represented the screening has been proposed in [52]. However, there is the risk of reducing accuracy due to the omitted dimensions.

3.2.2 Mapping

Mapping has a broad meaning including projection, non-linear mapping, parameter space transformation. A mapping procedure transforms a set of correlated variables into a smaller set of new uncorrelated ones that retain most of the original information. One popular approach that relies on linear analyses is the Principal Component Analysis (PCA). This method is especially used for problems with only continuous variables [14].

Contrarily, other methods are based on non-linear mapping and projection. Space-mapping (SM) intends to map the design space of a "coarse" and low-dimensional model to a fine, "expensive" and higher-dimensional one [5]. A good survey of related approaches is given in [46].

Another class of dimensionality reduction approaches is based on unsupervised learning. Two promising examples are *Autoencoders* and *Self-organising maps*. Autoencoders are neural networks that aim to reconstruct their own inputs. As such, an encoder network maps from the high-dimensional space to the coded space. Then, a decoder network maps back to the high dimensional space, with as little loss of information as possible. By constraining the coded space to have a smaller dimension than the input space, the autoencoder is forced to learn the most salient features of the input data [42]. Self-organising maps are a particularly interesting class of unsupervised systems that are based on competitive learning. The output neurons compete amongst themselves to be activated. Hence, only one is activated at one particular time. This competitive system forces the neurons to organise themselves. Commonly, self-organising maps target to map from the high-dimensional space to one or two-dimensional space [30].

3.3 Uncertainty

A large variety of optimization problems in scheduling, finance, transportation, and engineering design requires that decisions are made in the presence of uncertainty. Uncertainties are present in all real-world application problems, e.g., due to inaccuracies in the manufacturing process, uncertain operating conditions, or system component failures. However, different forms of uncertainties can be distinguished, a good overview can be found in [28]. In the following a description of the most common forms of uncertainty in real-world application problems, *noise* and *robustness*, will be given.

An optimization problem is considered subjected to noise if the objective function is perturbed. This can be due to several factors such as sensor measurements errors or heuristic simulations. Mathematically, noise is often assumed normally distributed with zero mean and variance [28].

In other cases, perturbation can afflict the design variables. Therefore, it is often required that the optimal solution should still work satisfactorily when the design variables change slightly, e.g., due to manufacturing tolerances [28].

In these cases, successfully performing global optimization means facing a variety of challenging issues. Resources have to be allocated to perform an uncertainty analysis in order to direct the research to stable and robust optima.

The correct balance between exploitation, exploration, and uncertainty quantification has to be addressed. Furthermore, adopting surrogate model-based optimization in noisy functions causes an additional problem. The use of derivative-based optimization techniques can lead to regions with seemingly good function caused by a misinterpretation of the noisy data. This issue appears if surrogate models confuse noise with the actual behaviour of the objective function.

Thus, finding efficient methods to deal with uncertainty appears to be a non-trivial problem. A popular approach is searching for *robust* optima. This is done by replacing the deterministic objective function f in favour of a modified $\tilde{f}$ that feeds an estimation of f back to the optimizer. This is evaluated observing the response of f a number of times with the same design solution. Popular examples of $\tilde{f}$ are the *expected value*, *expected value + k standard deviation* (the importance of the *standard deviations* in respect of the *mean* is weighted by the coefficient k) or the 95% *quantile*. The obvious drawback of this method is the considerable number of repetitions of f that are needed to make an accurate prediction of the *robust* objective function. However, researchers are proposing methods to mitigate the computational effort in estimating $\tilde{f}$ replacing classic uncertainty quantification methods, such as quasi-Monte Carlo quadrature, in favour of polynomial chaos with coefficients determined by sparse quadrature and by point collocation, radial basis function or Kriging models [33].

4 Sequential Parameter Optimization

The Sequential Parameter Optimization Toolbox (SPOT) [6] is an optimization framework which is based on surrogate model-based optimization. The aim of SMBO is to train a cheap numerical model that approximates the objective function and utilizes it to reduce the computational effort.

Initially proposed for algorithm tuning of metaheuristics, SPOT is a sophisticated tool capable of handling both continuous and mixed-discrete problems [9]. SPOT spends the available budget in a sequential manner to maximize information gain, and is particularly efficient for expensive problems. SPOT finds improved solutions in the following way (see Algorithm 1): First, the search space is sampled with an experimental design plan (see Sec. 4.1). With these samples, a first surrogate model is constructed. Then, an infill criterion is optimized on the surrogate to find new promising candidate solutions. The suggested candidates are evaluated with the real objective function, and the surrogate is updated with the observed information. In the following we will describe the fundamental steps of the SPOT methodology. A more detailed description of SPOT can be found in [7].

Algorithm 1 Sequential parameter optimization

1: $t = 0$.
2: Initialize a number of k samples $P_t = \{x_i, 1 \leq i \leq k\}$.
3: Select a suitable surrogate model $M_t(x)$.
4: Evaluate P_t on f to get observations $O_t = \{(x_i, y_i), 1 \leq i \leq k\}$.
5: **while** not Termination Criterion **do**
6: Build a model M_t with O_t
7: Optimize M_t infill criteria to get x_t^*
8: Evaluate x_t^* on f to get y_t^*
9: Update dataset $O_{t+1} = \{O_t, (x_t^*, y_t^*)\}$.
10: $t = t + 1$.
11: **end while**

4.1 Initial Design

4.1.1 Strategies for Design of Experiment

The first step of SPOT (see Algorithm 1) is the determination of the initial data set that will be used to train the first surrogate model. In order to build a moderately accurate model, the initial design should cover, if possible, the complete feasible search space. To that end, we rely on sampling methods.

Sampling methods can be classified as *deterministic* and *stochastic* methods. Examples of pure deterministic sampling are grid designs, full factorial designs, and Sobol sequences [49]. Stochastic methods try to create unbiased subsets of the original search space. They often optimize a certain criterion such as D-optimality or I-optimality [37]. This class includes basic random sampling, stratified sampling (e.g. Latin hypercube sampling) and fractional factorial designs.

4.1.2 Latin Hypercube Sampling

As a representative of stratified sampling, one of the most commonly employed methods is Latin Hypercube Sampling (LHS) [35]. LHS creates multidimensional designs. Given the number of samples n, all $n_r + n_z + n_d$ dimensions are divided into n intervals. LHS samples a point from each stratum. Different variants for choosing a point in each stratum exist. For example, median LHS uses the median value of each interval, while random LHS selects a random point within each interval.

This procedure has to be adapted to also treat categorical and discrete variables. One of the simplest solutions for ordinal variables is to assume that all variables are continuous, then using floor, ceiling or rounding operations. Dummy variables may be employed for categorical parameters. Or else, categorical parameters may be mapped to ordinal integers.

4.1.3 Factorial Designs

In the field of design of experiments, a set of statistically well profound designs have emerged, which are commonly applied to analyze and optimize industrial problems. Common designs of this field are full factorial, fractional

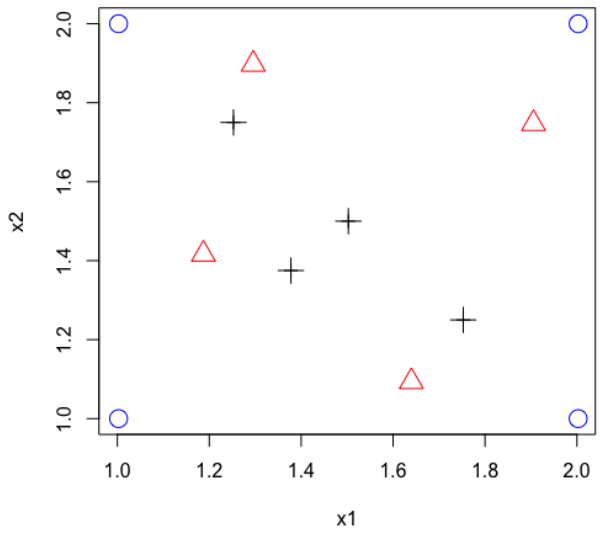

Fig. 1: Example of three different sampling methods in creating an initial design of size 4 in a two dimensional space: Sobol sequence (red triangles), Latin Hypercube (black crosses), Full factorial design (blue circles).

factorial, Box-Behnken, and central composite [37]. All these designs are designated to fit linear models for the response surface methodology, commonly with second-order and quadratic effects. For example, in a full factorial design for 2 variables with 2 levels for each design variable, a set of 2^2 evenly spaced points is determined. In case of continuous variables, they are determined by $[r^{\min}, r^{\max}]$. With this design, we are able to analyse main and second-order effects. For quadratic effects, center points need to be added. A full factorial design has the disadvantage of requiring an exponentially increasing number of experiments with rising number of variables. To prohibit an infeasible number of experiments, usually optimized fractional factorial or other screening designs are utilized.

From Fig. 1, one can see that, contrary to LHS and the Sobol sequence, a full factorial sampling exhibits a particular grid structure that eases distinguishing the effects of all design variables on the objective function.

4.2 Modeling

4.2.1 Modelling in Mixed-Integer Space

Once the first dataset has been created and observed, it is used to train a surrogate model that aims to replicate the behaviour of the objective function (see Algorithm 1). In SPOT, a surrogate model is used to determine promising candidate solutions. To that end, it aims to learn the relation between problem variables and the corresponding function's response.

Compared to their frequent use for real-valued problems, surrogate model driven approaches are less often used in mixed optimization [27]. According to Bartz-Beielstein and Zaefferer [9], few expensive, real-world optimization problems of this type have been brought to the science community's attention, e.g., in the engineering domain [54, 3, 50, 51], bioinformatics [41], or data science [47]. One reason for the scarce use of discrete surrogate model-based optimization is the availability of suitable methods. Bartz-Beielstein and Zaefferer [9] identified six strategies for surrogate modeling in mixed-discrete search spaces: the naive approach, customized models, inherently discrete models, feature extraction, mapping, and similarity-based models. These strategies explain how modeling techniques can be used in the general cases. The six strategies are not mutually exclusive. Some methods may belong to several categories, or combine different strategies. Here, we focus on three of the six strategies: the naive approach, inherently discrete models, and similarity based models since these are more commonly used.

4.2.2 The Naive Approach

The naive approach to discrete modeling is to ignore the discrete nature of the search space. Standard continuous methods are applied to solve the optimization problem. An application of this approach can be found in [8]. There, the authors faced an expensive parameter tuning problem and employed Kriging models. Especially if the discrete variables are of an ordinal nature, the naive approach may be successful. Indeed, this strategy could even be adopted to deal with categorical variables: Addressing an arbitrary order would create a one-to-one correlation between categorical variables and ordinal values. Several potential drawbacks can arise if this strategy is employed for problems that are too complex:

- Large areas of redundancy in the model's input space.
- Creation of infeasible solutions.
- Degeneration of performance due to bias caused by a misinterpretation of variables.

The naive approach is attractive, due to its ease of use and the ability to stick to continuous variable handling methods. Practitioners have to carefully evaluate if this option suits the characteristics of the problem under study.

4.2.3 Inherently Discrete Models

There are models that are discrete in their design, and hence need no further adaptation to discrete variables. For example, tree-based models, like regression trees or random forests are inherently discrete models. A representative application of this strategy can be found in [26], where an optimization process based on random forest models has been employed for a high dimensional algorithm tuning problem. On the one hand, inherently discrete models are easy to use, since they require no additional work to adapt them to discrete problems. On the other hand, in a mixed-variable case, a tree-based model would not be well-suited to represent the continuous parameters in the mixture. Also, tree-based models may not provide the useful features that models like Kriging have. For

example, uncertainty estimates can be derived from random forests, but unlike Kriging, these estimates do not go to zero at observed sample locations.

4.2.4 Similarity-based Models

Similarity-based modeling is a promising strategy that is gaining more and more interest. Here, suitable measures of similarity are used to model discrete data. With respect to their interpretation and use, the measures are referred to as similarity measures, dissimilarity measures, distance measures, correlation measures, or kernels. Although this approach is potentially very powerful, it requires the definition of proper measures. This may be problematic if these measures have to fulfil further requirements, e.g., definiteness, and if the problem involves different types of variables.

Fonseca et al. [19] defined similarity-based models that keep a memory of solutions and estimate the performance of new samples by comparing them to that memory. Three models from this class are of particular interest: Radial Basis Function Networks (RBFN), Support Vector Machine (SVM), and Kriging. Various model-based variants applied to different optimization problems can be found in the literature, e.g., [22, 32, 26, 38, 17, 23, 4, 58]. Several of these works involve the development of appropriate similarity measures for discrete or mixed search spaces.

As these previous developments indicate, similarity-based models like Kriging are very promising approaches towards handling mixed and discrete variables. Hence, we focus on Kriging in the following.

4.2.5 Handling Factor Variables in Kriging Model

Kriging is a similarity-based model and assumes that the data follows a multivariate Gaussian distribution, where errors are spatially correlated. A detailed and comprehensible description of Kriging is given by Forrester et al. [20].

We consider a simplified variant of the optimization problem defined in Eq. 1:

$$\min_{\vec{x}\in\mathbb{R}^n} f(\vec{x})$$

and no equality and inequality constraints are used. Importantly, the spatial correlation of the data is encoded within a kernel, or correlation function. A frequently employed correlation function that models the correlation between samples (or candidate solutions) is the Gaussian kernel $k(x, x') = \exp(-\sum_{i=1}^{n} \theta_i |x_i - x'_i|^2)$. Here, n is the number of modelled variables (search space dimension), and θ_i is a parameter of the kernel (determined by Maximum Likelihood Estimation (MLE)). Furthermore, x as well as x' are potential candidate solutions (or samples). Employing such a kernel, a Kriging model produces the following predictor:

$$\hat{y}(z^*) = \hat{\mu} + \mathbf{k}^T \mathbf{K}^{-1} (\mathbf{y} - \mathbf{1}\hat{\mu}), \tag{2}$$

where $\mathbf{y}$ are the training observations, $\hat{y}(z^*)$ is the predicted function value of a new sample z^*, $\hat{\mu}$ represents the process mean determined by MLE, $\mathbf{1}$ is a vector of ones, $\mathbf{K}$ is the matrix that collects all pair-wise correlations of the training samples $\mathbf{Z}$, and $\mathbf{k}$ is the column vector of correlations between the set of training

samples $\mathbf{Z}$ and the new sample z^*. After appropriate training, such a predictor may be employed to replace an expensive objective function.

The success of Kriging in the field of real world application problems mostly relies on the possibility to estimate the uncertainty of the predictor. This feature assumes a prominent role in applications in which the limited number of observations that can be performed inhibits an exhaustive exploration of the search space. In these cases, the estimate of the uncertainty can be used to balance exploration and exploitation by computing the expected improvement (EI) of candidate solutions [36]. The uncertainty of the model is computed with

$$\hat{s}^2(z^*) = \sigma^2_{process}(1 - \mathbf{k}^T\mathbf{K}^{-1}\mathbf{k}), \tag{3}$$

where $\sigma^2_{process}$ is the process variance, determined by MLE. If the uncertainty is zero, the EI is also zero. Else, the uncertainty is non-zero, and the EI is

$$\mathrm{EI}(z^*) = \mathbf{y}_{imp}\Phi\left(\frac{\mathbf{y}_{imp}}{\hat{s}(z^*)}\right) + \hat{s}(z^*)\phi\left(\frac{\mathbf{y}_{imp}}{\hat{s}(z^*)}\right),$$

where $\mathbf{y}_{imp} = \min(\mathbf{y}) - \hat{y}(z^*)$. $\Phi()$ indicates the normal cumulative distribution function. Respectively, $\phi()$ is the probability density function.

It has to be noted, that the above description of Kriging presents an interpolating model, which assumes zero error at already observed locations. Clearly, this does not take noise or uncertainty into account. One way to account for noise is to introduce the so-called nugget effect. This essentially adds a constant value η to the diagonal of the kernel matrix $\mathbf{K}$. The parameter η is determined by MLE. The nugget effect enables the model to regress the observed data, and hence smoothen noisy observations.

Until now, we discussed Kriging in the context of real valued search spaces. It is also applicable to "mixed" search space, where an appropriate kernel is available. With respect to mixed or discrete problems, Kriging is actually very flexible. By changing the kernel (or correlation) function, any search space may be modeled with Kriging [59, 9]. The flexibility of this modeling method renders Kriging one of the most promising mixed variable models. Take for example a typical problem characterization from algorithm tuning: parameters like mutation rates may be real valued, the choice between different mutation operators may be a categorical parameter. Hence, if x_i (the i-th dimension of a parameter configuration $\vec{x}$) is a factor variable, Hamming distance can be used, otherwise the absolute deviation may be used for real-valued variables. The reader is referred to the discussion in [9].

4.3 Optimization Algorithms for the Metamodel

As shown in Algorithm 1, the next step in the SPOT methodology, after constructing a model, consists in the employment of optimizers that search for promising candidate solutions. Standard techniques from mathematical programming [56], so-called Mixed-integer non-linear programming methods [18] are commonly not applicable to deterministic optimization of real-world application problems. These methods, such as outer approximation [16], branch and bound [11], and generalized Benders decomposition [21] have difficulties with the mixed design space, multi-modality, uncertainty in the observations, and unknown black-box properties. A consolidated alternative consists in the employment of metaheuristics for mixed-discrete optimization [31]. These strategies

propose to heuristically determine solutions that improve the objective function value.

In cases where mathematical programming techniques are not flexible enough to yield satisfying results, heuristic search for solutions that improve the objective function value can lead to interesting results. Metaheuristics for mixed-discrete optimization are generally categorized in two classes:

- *Hierarchical Approaches* solve problems with continuous variables together and discrete variables by considering the original optimization as a bi-level problem. The discrete variables are optimized by the upper level optimization process and the continuous parameters are optimized in the lower level. [34, 53]
- *Simultaneous Approaches* optimize discrete and continuous parameters simultaneously. In this approach, we consider that a similarity of parameter vectors due to an appropriate metric is equivalent to being positively correlated to the similarity in function values [44, 31].

In the following we will highlight the peculiarities of an algorithm from the class of simultaneous approaches. These algorithms are a better choice for our purposes. They need fewer observations and consider correlations between discrete and continuous variables. This is in contrast to the hierarchical approaches where variables of different type are strictly separated from each other. Particularly, we will discuss the Mixed Integer Evolution Strategy (MIES) proposed in [31].

4.4 MIES

Evolution Strategies (ES) are metaheuristics that follow the concept of natural evolution. An individual in an ES contains the information about one solution candidate. This individual is subject to recombination, mutation and selection operations. By evolving sets, or populations of individuals, the ES tries to find improved solutions.

In MIES, an individual contains information about real-valued variables, ordinal integer variables and categorical variables. Parameters of the probability distribution used in the mutation operator (such as standard deviations or step sizes) are also stored in the individual for the purpose of self-adaptive parameter control. The latter parameters are referred to as *strategy parameters*. As a consequence the domain of an individual $\mathbb{I}$ can be expressed as follows:

$$\mathbb{I} = R_1 \times ... \times R_{n_r} \times Z_1 \times ... \times Z_{n_z} \times D_1 \times ... \times D_{n_d} \times A_s \tag{4}$$

with $A_s = \mathbb{R}_+^{n_\sigma + n_\zeta} \times [0,1]^{n_p}$ being the domain of the *strategy variables*. Correspondingly, an individual of a population can be represented as:

$$\vec{a} = (r_1, ..., r_{n_r}, z_1, ..., z_{n_z}, d_1, ..., d_{n_d}, \sigma_1, ..., \sigma_{n_\sigma}, \zeta_1, ..., \zeta_{n_\zeta}, p_1, ..., p_{n_p}) \tag{5}$$

The so-called *design variables* $r_1, ..., r_{nr}, z_1, ..., z_{nz}, d_1, ..., d_{nd}$ determine the objective function value and thus the fitness of the individual. The *strategy variables* $\sigma_1, ..., \sigma_{nr}$ are standard deviations used in the mutation of the real valued variables, and $\zeta_1, ..., \zeta_{nz}$ denote mean step sizes in the mutation of the integer parameters. Finally, $p_1, ..., p_{np}$ denote mutation probabilities (or rates) for the

nominal discrete object parameters. MIES is considered a self-adaptive process because the strength of the mutation parameters continuously evolves during the optimization. Hence, the mutation strength itself is also governed by an evolutionary process. The philosophy behind self-adaptation is that the evolutionary process can solve two problems simultaneously: the determination of the best strategy variables, and the determination of the best object variables. More details on self-adaptation can be found in Chapter 11.

The first population P(0) of μ individuals is generated by uniform random sampling from $\mathbb{I}$. Then, the main loop of the MIES algorithm starts. In a first step, the algorithm generates the set of λ new offspring individuals with the following procedure. Two parents are randomly selected from the population and an offspring is generated by recombination and mutation. The recombination operator can be subdivided into two steps, selection of the parents and recombining the selected parents. The two parents $c_1, c_2 \in I$ are selected randomly, from the parental generation for each of the offspring individuals. In MIES, two different types of recombination are used: dominant and intermediate [45]. The first one is adopted for solution variables and consists of a random selection of one of the corresponding parental parameters for each offspring vector position. The latter is used for recombining the strategy parameters and computes the mean of both parental vectors. The mutation of the offspring relies on operators acting differently on real, integer and discrete variables, all respecting the requirements for a mutation strategy in the search spaces: *Accessibility*, *Feasibility*, *Symmetry*, *Similarity*, *Scalability*, and *Maximal Entropy* [31].

The MIES achieves this by adding normal distributed noise to real-valued variables. For integer variables, the distribution is based on the difference of two geometrical distributions. Categorical variables are simply re-sampled (uniform randomly) with some probability p [31].

In the next step of the iteration, the λ offspring individuals are ranked on the basis of the objective function. The μ best individuals out of the union of the λ offspring individuals and the μ parental individuals are selected. The generational loop is repeated until the number of evaluation exceeds the budget.

5 Case Study: Optimization of Composite Multi-layered Plate

In this section, a mixed-discrete problem based on a real-word application is discussed. The problem consists in the design optimization of a composite multi-layered plate. Our idea is to demonstrate the difficulties that researchers and practitioners encounter when facing black-box, time-consuming, mixed-discrete problems under uncertainty trough an illustrative example. A performance comparison between the MIES and a general purpose optimizer is also given.

5.1 Overview

The objective of the optimization problem is to find the materials (represented by categorical variables) and lamination angles (continuous variables) in order to minimize the bending of a loaded plate composed of 5 layers.

The available materials and their properties are reported in Table 1. As one can see, the list includes orthotropic and isotropic materials. In case of

orthotropic materials, the stiffness of the material is crucially affected by the lamination angle. Contrarily, isotropic materials have equal in-plane and out-of-plane Young's Modulus. Hence, the lamination angle of isotropic materials does not affect the material behavior. The use of both types of materials considerably increases the difficulty of the problem from the modeling perspective: The importance of the continuous parameters depends on the value of the categorical variables.

The plate has been loaded by a lifting load, which is linearly distributed along the length of the plate applied on the nodes in the centerline. An encastre at the root of the plate has been enforced as shown in Fig. 2. With the intent to reproduce uncertainty due to manufacturing tolerances and measurement conditions, perturbations to the nominal values have been added to both lamination angles and load magnitudes.

The purpose of this test case is to point out the difficulties of handling an "expensive" ($\approx$ 10 seconds for each run), multi-modal, mixed-integer, "high-dimensional" (5 continuous + 5 categorical variables) problem under uncertainty with an extremely limited budget (150 evaluations).

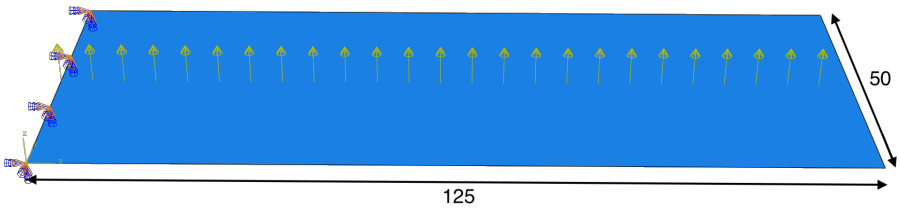

Fig. 2: Load and boundary constrains applied to the multi-laminate plate.

5.2 Optimization problem

The objective of the problem is the minimization of the displacement along the loaded axis of one of the vertices of the multi-layered plate tip s_t (red point in the corner of the plate in Fig. 3). The optimization problem is defined as:

$$\min_{x \in \mathbb{R}^5, y \in \mathbb{D}^5} f(x, y) = s_t$$
$$\text{subject to} \begin{cases} r_i \in [-89, 90], 1 \leq i \leq 5 \\ \mathbb{D}^5 = D_1 \times ... \times D_5 \\ d_i = \left\{ d_i^1, ..., d_i^{12} \right\}, 1 \leq i \leq 12 \end{cases},$$

where $d_i^1, ..., d_i^{12}$ are the 12 available materials [13]. The first 5 variables correspond to the lamination angles, the latter five describe the material of each ply as categorical variables. The categorical variables are mapped to integers from 1 to 12 to allow a numerical optimizer based on Differential Evolution (DE) [2] to handle them.

Since the mass of each ply has been fixed, the thickness will be dependent on the material density.

Table 1: Properties of the Material used in this study.

Properties		Young's Modulus 0	Young's Modulus 90	In-plane Shear Modulus	Poisson's Ratio	Density
Symbols		E1	E2	G12	v12	
Units		GPa	GPa	GPa		g/cc
CF	Vf 50%	70	70	5	0.10	1.60
HMCF	Vf 50%	85	85	5	0.10	1.60
E glass	Vf 50%	25	25	4	0.20	1.90
Kevlar	Vf 50%	30	30	5	0.20	1.40
Std CF	Vf 60%	135	10	5	0.30	1.60
HMCF	Vf 60%	175	8	5	0.30	1.60
M55**	Vf 60%	300	12	5	0.30	1.65
E glass	Vf 60%	40	8	4	0.25	1.90
Kevlar	Vf 60%	75	6	2	0.34	1.40
Boron	Vf 60%	200	15	5	0.23	2.00
Steel	S97	207	207	80	0.3	7.85
AL	L65	72	72	25	0.3	2.7

5.3 Methodology

To perform the optimization process, we take advantage of surrogate modeling techniques and rely on the SPOT R package. The experiments have been conducted employing MIES and DE as optimizers (on the surrogate model). As shown in the previous sections, MIES is dedicated to handling mixed variables, including categorical variables. Contrarily, DE is not designed to handle discrete or categorical variables. In order to compute an accurate estimation of the plate maximal displacement, the finite element analysis solver Abaqus [24] has been employed. Therefore, the optimization problem requires a coupling between SPOT and Abaqus.

The process follows the algorithm described in Algorithm 1: Initially a design is created by LHS. All the candidates in the design are then evaluated with the objective function (via Abaqus). This function receives the candidate solutions that describe the characteristics of the plate and feeds the corresponding displacement back to the optimizer. Nevertheless, to simulate the uncertainty, the nominal lamination angles and loads magnitudes are perturbed randomly by Gaussian perturbations with mean 0 and variance respectively 1 and 0.1. The candidates and the function responses are used to train a Kriging model. Then, an optimizer searches for the most promising candidate solution by optimizing an infill criterion based on the model. Here, the criterion is the expected improvement. Based on the assumption of expensive function evaluations, a very small budget of 150 function evaluations has been used.

Table 2: Optimal design achieved using DE and MIES.

Optimal design	Displacement	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
DE	169.49	0.35	15.04	-23.496	5.18	75.43	9	4	9	7	6
MIES	303.34	-5.79	-89.00	-89.00	-35.27	-7.15	7	1	7	9	7

5.4 Results

The obtained results clearly highlight that mixed-discrete problems present peculiar challenges. Such issues are unlikely to be resolved by continuous optimizers. A convergence plot is shown in Fig. 5. It can be seen that the best configuration found by MIES outperforms the one found by DE. Although the processes have been started from the same design, the results differ by around 80% (Tab. 2).However, to have a statistically meaningful comparison, we should repeat the analyses with the two best designs in order to quantify the effect of the uncertainties.

The displacements of the two best plate configurations are shown in Fig. 3 with the same scale and magnification factor. Figure 4 shows that the configurations differ significantly, concerning both the uous and categorical variables. Particularly, it is worth to focus on the effect of the lamination angles: The configuration found using DE presents materials that are mostly aligned with the plate. This configuration would lead to the maximal uniaxial stiffness. In this case, the displacement would be equal for all the nodes lying on the tip. However, the object of the optimization is the displacement in the corner of the plate, which is at some distance from the centerline of the plate. This means that another minimum is present, besides the local minimum resulting from the maximization of the stiffness in the direction of the plate. The second minimum is the global one. It consists in the perfect balance between the stiffness in both the directions in the plane of the plate. The balance is determined on the basis of the ratio between the length and the height of the plate. As a result, the overall displacement at the tip will be higher but the displacement at the corner will be lower.

In fact, in the configuration resulting from MIES, two plies are laminated with angles that are orthogonal to the direction of the plate. In light of this observation, we can say that this problem is clearly multi-modal in regard to the continuous variables. The multi-modality of the function derives from the dependence of both the mechanical properties and the ply thickness on the chosen material. This is also reflected in Fig. 6, where the interactions of four important variables are depicted. In each plot, two variables are varied while the remaining two are fixed to their optimal values. The red dots represent the observed values. The dataset used to train this model is composed of the observations made during the optimization process using MIES. In Fig. 6 a) and b), the interaction between the lamination angles and the materials for both the plies are shown. In both the cases two distinct regions of well-performing configurations are present. In Fig. 6 c) and d), the interaction between the materials and their lamination angles of the two plies are represented. The figures clearly show a complex and multi-modal search landscape. In light of these considerations, one can see the complexity involved in this, apparently, simple problem.

In the last section of this chapter, an example of the application of SPOT on a real-world optimization problem has been reported and analysed. The complexity of mixed-discrete real-world optimization problems has been addressed in this chapter. Despite the difficulties, more efficient ways to handled them have been developed. Nevertheless, despite the cutting-edge algorithms, these problems still appear very complex to tackle.

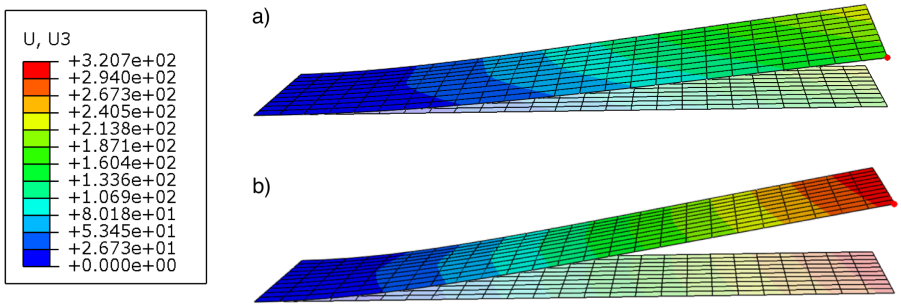

Fig. 3: Contour of the displacement in the loaded direction in the best configurations obtained employing DE (a) and MIES (b).

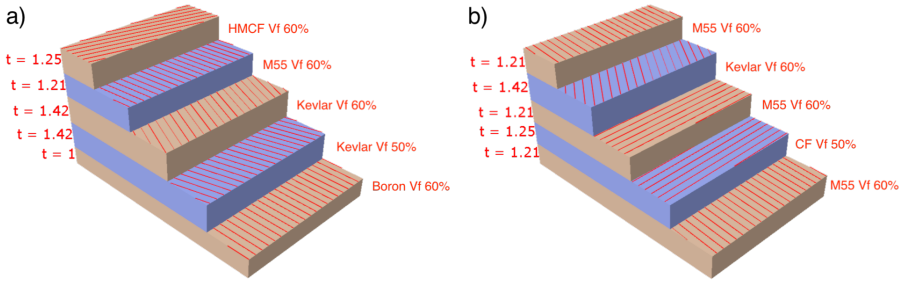

Fig. 4: Best lamination configuration (lamination angles, materials and thickness) obtained employing DE (a)and MIES (b).

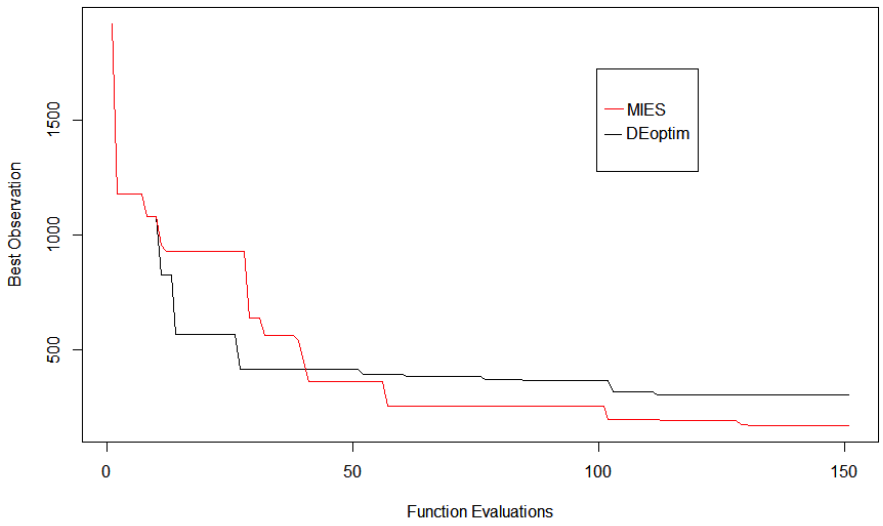

Fig. 5: Evolution of the best observation during the optimization processes.

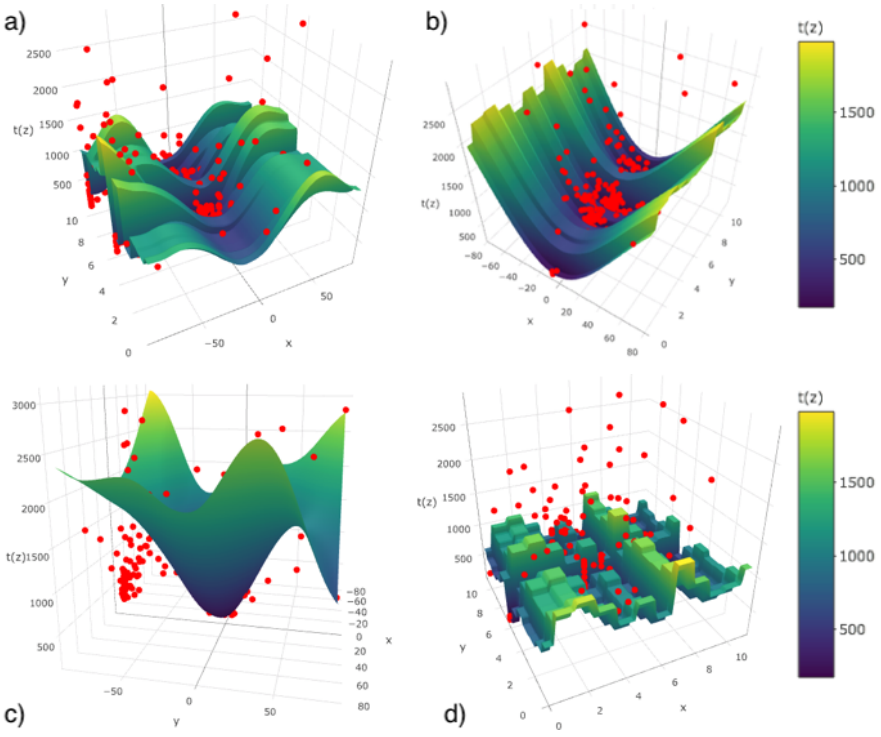

Fig. 6: Visualizations of the objective function landscapes in respect of variable combinations concerning Ply1 and Ply5. For each individual plots, variables that are not shown are fixed to the respective optimal values. a) Material and Lamination angle of Ply1. b) Material and Lamination angle of Ply5. c)Lamination angles of Ply1 and Ply5. d) Materials of Ply1 and Ply5.

Acknowledgement

This work is funded by the European Commission's H2020 program, through the UTOPIAE Marie Curie Innovative Training Network, H2020-MSCA-ITN-2016, Grant Agreement number 722734.

References

[1] C. C. Aggarwal, A. Hinneburg, and D. A. Keim. On the surprising behavior of distance metrics in high dimensional space. In *Database Theory ICDT 2001*, pages 420–434. Springer Science + Business Media, April 2001.

[2] D. Ardia, K. Boudt, P. Carl, K. M. Mullen, and B. G. Peterson. Differential Evolution with DEoptim: An application to non-convex portfolio optimization. *The R Journal*, 3(1):27–34, 2011.

[3] M. Asadi and J. Goldak. Combinatorial optimization of weld sequence by using a surrogate model to mitigate a weld distortion. *International Journal of Mechanics and Materials in Design*, 7(2):123–139, 2011.

[4] L. Bajer and M. Holeňa. Surrogate model for continuous and discrete genetic optimization based on rbf networks. In *Intelligent Data Engineering and Automated Learning – IDEAL 2010*, volume 6283 LNCS, pages 251–258, 2010.

[5] J. W. Bandler, R. M. Biernacki, S. H. Chen, P. A. Grobelny, and R. H. Hemmers. Space mapping technique for electromagnetic optimization. *IEEE Transactions on Microwave Theory and Techniques*, 42(12):2536–2544, 1994.

[6] T. Bartz-Beielstein. *Experimental Research in Evolutionary Computation—The New Experimentalism*. Natural Computing Series. Springer, Berlin, Heidelberg, New York, 2006.

[7] T. Bartz-Beielstein, L. Gentile, and M. Zaefferer. In a nutshell: Sequential parameter optimization. Technical report, TH Köln, 2017.

[8] T. Bartz-Beielstein and S. Markon. Tuning Search Algorithms for Real-World Applications: A Regression Tree Based Approach. In G. W. Greenwood, editor, *Proceedings 2004 Congress on Evolutionary Computation (CEC'04), Portland OR*, pages 1111–1118, Piscataway NJ, 2004. IEEE.

[9] T. Bartz-Beielstein and M. Zaefferer. Model-based methods for continuous and discrete global optimization. *Applied Soft Computing*, 55:154 – 167, 2017.

[10] A. Bergman and M. W. Feldman. Recombination dynamics and the fitness landscape. *Physica D: Nonlinear Phenomena*, 56(1):57–67, 1992.

[11] B. Borchers and J. E. Mitchell. An improved branch and bound algorithm for mixed integer nonlinear programs. *Computers & Operations Research*, 21(4):359–367, 1994.

[12] L. Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[13] P. composites Ltd. Mechanical properties of carbon fibre composite materials. `http://www.performance-composites.com/carbonfibre/mechanicalproperties_2.asp`. Url date: 2018-07-01.

[14] C. Ding, X. He, H. Zha, and H. D. Simon. Adaptive dimension reduction for clustering high dimensional data. In *Data Mining, 2002. ICDM 2003. Proceedings. 2002 IEEE International Conference on*, pages 147–154. IEEE, 2002.

[15] N. R. Draper and H. Smith. *Applied regression analysis*, volume 326. John Wiley & Sons, 2014.

[16] M. A. Duran and I. E. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical programming*, 36(3):307–339, 1986.

[17] R. Filomeno Coelho, M. Herrera, M. Xiao, and W. Zhang. On-line metamodel-assisted optimization with mixed variables. In J. Magalhães-Mendes and D. Greiner, editors, *Evolutionary Algorithms and Metaheuristics in Civil Engineering and Construction Management*, volume 39 of *Computational Methods in Applied Sciences*, pages 1–15. Springer International Publishing, 2015.

[18] C. A. Floudas. *Nonlinear and mixed-integer optimization: fundamentals and applications*. Oxford University Press, 1995.

[19] L. Fonseca, H. Barbosa, and A. Lemonge. A similarity-based surrogate model for expensive evolutionary optimization with fixed budget of simulations. In *Proceedings of the Congress on Evolutionary Computation (CEC'09)*, pages 867–874, New York, NY, USA, 2009. IEEE.

[20] A. Forrester, A. Keane, et al. *Engineering design via surrogate modelling: a practical guide*. John Wiley & Sons, 2008.

[21] A. M. Geoffrion. Generalized benders decomposition. *Journal of optimization theory and applications*, 10(4):237–260, 1972.

[22] T. Hemker. *Derivative Free Surrogate Optimization for Mixed-Integer Nonlinear Black Box Problems in Engineering*. PhD thesis, Technische Universität Darmstadt, December 2008.

[23] M. Herrera, A. Guglielmetti, M. Xiao, and R. Filomeno Coelho. Metamodel-assisted optimization based on multiple kernel regression for mixed variables. *Structural and Multidisciplinary Optimization*, 49(6):979–991, 2014.

[24] K. Hibbitt. Abaqus: User's manual: Version 6.13: Hibbitt. *Karlsson & Sorensen, Incorporated*, 2013.

[25] R. R. Hocking. A biometrics invited paper. the analysis and selection of variables in linear regression. *Biometrics*, 32(1):1–49, 1976.

[26] F. Hutter, H. H. Hoos, and K. Leyton-Brown. Sequential model-based optimization for general algorithm configuration (extended version). Technical Report TR-2010-10, University of British Columbia, Department of Computer Science, 2010. Available online: http://www.cs.ubc.ca/~hutter/papers/10-TR-SMAC.pdf.

[27] R. Jin, W. Chen, and T. W. Simpson. Comparative studies of metamodelling techniques under multiple modelling criteria. *Structural and Multidisciplinary Optimization*, 23(1):1–13, Dec. 2001.

[28] Y. Jin and J. Branke. Evolutionary Optimization in Uncertain Environments—A Survey. *IEEE Transactions on Evolutionary Computation*, 9(3):303–317, 2005.

[29] D. R. Jones, M. Schonlau, and W. J. Welch. Efficient Global Optimization of Expensive Black-Box Functions. *Journal of Global Optimization*, 13:455–492, 1998.

[30] T. Kohonen. The self-organizing map. *Proceedings of the IEEE*, 78(9):1464–1480, 1990.

[31] R. Li, M. T. Emmerich, J. Eggermont, T. Bäck, M. Schütz, J. Dijkstra, and J. H. Reiber. Mixed integer evolution strategies for parameter optimization. *Evolutionary computation*, 21(1):29–64, 2013.

[32] R. Li, M. T. M. Emmerich, J. Eggermont, E. G. P. Bovenkamp, T. Bäck, J. Dijkstra, and J. H. C. Reiber. Metamodel-assisted mixed integer evolution strategies and their application to intravascular ultrasound image analysis. In *2008 IEEE Congress on Evolutionary Computation (CEC)*, pages 2764–2771. IEEE, 2008.

[33] D. Liu, A. Litvinenko, C. Schillings, and V. Schulz. Quantification of airfoil geometry-induced aerodynamic uncertainties—comparison of approaches. *SIAM/ASA Journal on Uncertainty Quantification*, 5(1):334–352, 2017.

[34] R. Lohmann. Structure evolution and incomplete induction. *Biological Cybernetics*, 69(4):319–326, 1993.

[35] M. D. McKay, R. J. Beckman, and W. J. Conover. A Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, 21(2):239–245, 1979.

[36] J. Močkus. On Bayesian Methods for Seeking the Extremum. In *Optimization Techniques IFIP Technical Conference*, pages 400–404, 1974.

[37] D. C. Montgomery. *Design and analysis of experiments*. John wiley & sons, 2017.

[38] A. Moraglio and A. Kattan. Geometric generalisation of surrogate model based optimisation to combinatorial spaces. In *Proceedings of the 11th European Conference on Evolutionary Computation in Combinatorial Optimization*, EvoCOP'11, pages 142–154, Berlin, Heidelberg, Germany, 2011. Springer.

[39] M. D. Morris. Factorial sampling plans for preliminary computational experiments. *Technometrics*, 33(2):161–174, 1991.

[40] G. Papa. Parameter-less algorithm for evolutionary-based optimization. *Computational Optimization and Applications*, 56(1):209–229, 2013.

[41] P. A. Romero, A. Krause, and F. H. Arnold. Navigating the protein fitness landscape with Gaussian processes. *Proceedings of the National Academy of Sciences*, 110(3):E193–E201, 2013.

[42] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[43] A. Saltelli, S. Tarantola, F. Campolongo, and M. Ratto. *Sensitivity Analysis in Practice*. John Wiley & Sons, Ltd, 2004.

[44] M. Schütz and J. Sprave. Application of parallel mixed-integer evolution strategies with mutation rate pooling. In *Proceedings of the Fifth Annual Conference on Evolutionary Programming*. Citeseer, 1996.

[45] H.-P. Schwefel. Evolution and optimum seeking. sixth-generation computer technology series, 1995.

[46] S. Shan and G. G. Wang. Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization*, 41(2):219–241, 2010.

[47] J. Smith, C. Stone, and M. Serpell. Exploiting diverse distance metrics for surrogate-based optimisation of ordering problems: A case study. In *Proceedings of the 2016 on Genetic and Evolutionary Computation Conference*, GECCO '16, pages 701–708, New York, NY, USA, 2016. ACM.

[48] J. Sobieszczanski-Sobieski. Sensitivity analysis and multidisciplinary optimization for aircraftdesign-recent advances and results. *Journal of Aircraft*, 27(12):993–1001, 1990.

[49] I. Sobol and Y. L. Levitan. A pseudo-random number generator for personal computers. *Computers & Mathematics with Applications*, 37(4-5):33–40, 1999.

[50] C. Teixeira, J. Covas, T. Stützle, and A. Gaspar-Cunha. Optimization of co-rotating twin-screw extruders using pareto local search. In *Advances in Intelligent and Soft Computing*, pages 3–10. Springer Science + Business Media, 2010.

[51] C. Teixeira, J. A. Covas, T. Stützle, and A. Gaspar-Cunha. Multi-objective ant colony optimization for the twin-screw configuration problem. *Engineering Optimization*, 44(3):351–371, 2012.

[52] J. Tu and D. Jones. Variable screening in metamodel design by cross-validated moving least squares method. In *44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference*, page 1669, 2003.

[53] U. Utecht and K. Trint. Mutation operators for structure evolution of neural networks. In *International Conference on Parallel Problem Solving from Nature*, pages 492–501. Springer, 1994.

[54] I. Voutchkov, A. Keane, A. Bhaskar, and T. M. Olsen. Weld sequence optimization: The use of surrogate models for solving sequential combinatorial problems. *Computer Methods in Applied Mechanics and Engineering*, 194(30-33):3535–3551, Aug 2005.

[55] T. Weise, M. Zapf, R. Chiong, and A. J. Nebro. Why is optimization difficult? In *Nature-Inspired Algorithms for Optimisation*, pages 1–50. Springer, 2009.

[56] T. Westerlund and F. Pettersson. An extended cutting plane method for solving convex minlp problems. *Computers & Chemical Engineering*, 19:131–136, 1995.

[57] D. H. Wolpert and W. G. Macready. No Free Lunch Theorems for Optimization. *IEEE Transactions on Evolutionary Computation*, 1(1):67–82, Apr. 1997.

[58] M. Zaefferer, J. Stork, and T. Bartz-Beielstein. Distance measures for permutations in combinatorial efficient global optimization. In T. Bartz-Beielstein, J. Branke, B. Filipič, and J. Smith, editors, *Parallel Problem Solving from Nature–PPSN XIII*, pages 373–383, Cham, Switzerland, 2014. Springer.

[59] M. Zaefferer, J. Stork, M. Friese, A. Fischbach, B. Naujoks, and T. Bartz-Beielstein. Efficient global optimization for combinatorial problems. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 871–878, New York, NY, USA, 2014. ACM.