Am Fachbereich Informatik
der Technischen Universität Darmstadt
eingereichte Dissertation zur Erlangung des
akademischen Grades eines Doktor-Ingenieur (Dr.-Ing.)

Derivative Free Surrogate Optimization for Mixed-Integer Nonlinear Black Box Problems in Engineering

Dipl.-Math. Thomas Hemker

(geboren in Steinfurt, Westfalen)

Referenten der Arbeit:	Prof. Dr. Oskar von Stryk
	Prof. Dr. ir. Dick den Hertog, Tilburg University
Tag der Einreichung:	17. November 2008
Tag der mündlichen Prüfung:	19. Dezember 2008

Darmstädter Dissertation

D 17

meinen Eltern

Contents

Abstract

Optimization problems arise in many disciplines when trying to guarantee the best possible utilization of restricted resources and optimization denotes the determination of the best possible solution for defined problems.
Optimization problems based on black boxes as they often arise in engineering applications are the focus of this thesis. Such black boxes typically represent the simulated or experimentally obtained behavior of systems for which almost no internal, structural or analytical knowledge can be provided on a relevant level for the optimization's objective. Furthermore, the black boxes may produce noise which results in a disturbed response with low accuracy by only a small number of digits. Typically, no appropriate alternatives to these black boxes exist which would have a competitive accuracy and the ability to be coupled closely with special adapted optimization methods. Thus, the direct incorporation of the black box responses into the optimization problem formulation is required.
The price to overcome this lack of profound knowledge are the computational or experimental costs for evaluations of the black box to aggregate information from its response. A fitting expression for this situation is that: *There's no such thing as a free lunch!*
The optimization variables are defined on continuous and integer-valued domains and describe system internal parameters or even possible system topologies. The variables are explicitly included in the optimization problem's objective and constraints. They also affect the problem implicitly because changes in the optimization variables result in different black box responses.
Such non-relaxable mixed-integer nonlinear black box-based optimization problems cannot be carried out efficiently by today's optimization methods. Thus, there is a need for general and robust optimization methods to solve these optimization problems.
In this thesis, a new derivative free optimization approach is presented and surrogate functions will provide the main basics. The performance of this method will be demonstrated for several benchmark and real world problems from electrical engineering, environmental sciences, and robotics. It will be shown that huge improvements of the optimization's objectives can be achieved for all applications, simply by applying a reasonable number of black box evaluations.

1 Introduction

Numerical simulations are the most common tools for the development of technical systems and the design of technical processes [208]. Many off-the-shelf simulation programs have been developed over the last decades with considerable time and financial expenses, e.g., in mechanical, electrical, or chemical process engineering. Today, these simulations provide sufficiently accurate responses, but are often not compatible with the needs of modern mathematical optimization methods like sequential quadratic programming methods.
Underlying analytic models and further internal simulation characteristics are often not easily available for sensitivity analysis using derivatives. A response of the numerical simulation is given as the output of a black box for the chosen initial settings. Hence, a simulation in the loop scenario arises for optimization and mathematical methods which couple simulation and optimization more closely using derivative information cannot be realized (cf. Fig. 1.1).
Also, it may be mandatory for some cases to use the real systems response in addition to a good simulation model. Optimization may search for solutions especially in regions of limited validity of the model. The small differences between the real system's response and the simulation's response may become important for the optimization. If time restrictions, financial settings, laboratory equipment, or other reasons do not prevent approaches which couple the real system's response with the optimization procedure, a system in the loop scenario for optimization can be considered. Therefore, the real system is taken as a black box that generates a response based on the chosen input parameters.

1.1 Motivation

In engineering, the black box input parameters often describe continuous characteristics as well as discrete ones. Changes therein may result in changes of the structure or topology in the system of interest, and may lead to completely different initial settings for the black box. An example would be a parameter representing the non-relaxable number of installed coils for a magnet design problem. Optimization methods used for black box-based problems have to take into account the presence of continuous and integer-valued parameters to guarantee the efficient use of available computational resources for optimization procedures.
For the considered problem class, model-based optimization methods are not applicable because of the lack of access to information about the internal models and parameters of the black box. Furthermore, completely new implementations for the problems from scratch which are adapted for optimization purposes are often much too ambitious. This results in a situation where only existing black boxes are available for optimization purposes. Thus, information about the system's characteristics and response behavior can only be obtained by sampling different input parameter sets and observing the changes in the resulting responses. The usually vast amount of expert knowledge, which was once included in the

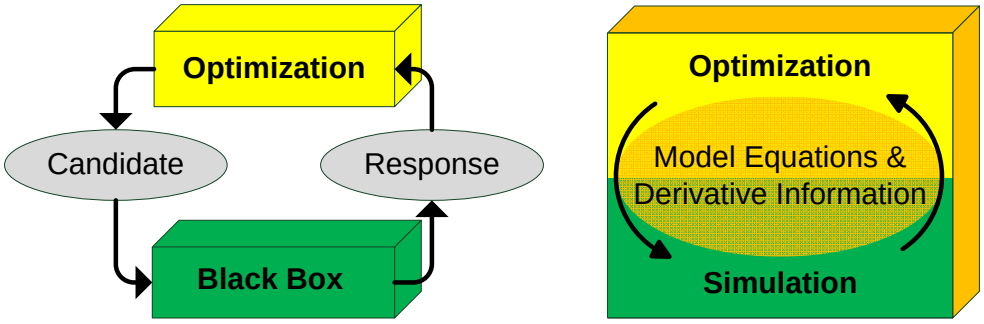


Figure 1.1: Simulation in a loop on the left vs. a closely coupled simulation and optimization approach on the right where model and derivative information is shared between both.

black boxes during their development, is still not available explicitly. Instead it can be utilized by sampling.
The optimization variables are given in parts by a number of or even all black box input parameters that can be changed. The optimization problem considered includes, in its objective and constraints, the responses from incorporated black boxes as components which implicitly depend on the optimization variables. Further less complicating components of the objective function and constraints which only depend explicitly on the optimization variables complete the problem formulation.
By incorporating the black box response in the otherwise, explicitly given problem formulation, certain issues must be considered for optimization, too. A common property of black box responses is that they are disturbed by noise and discontinuities, e.g., resulting from underlying iterative and other numerical methods. Even under these difficult conditions, the aim during the optimization process is to utilize as few black box evaluations as possible by the optimization method.
Finally, it has to be mentioned that the purpose for performing optimization is not always to find the exact minimum of the objective function subject to the constraints. In many engineering applications this is not even theoretically possible because the necessary assumptions, e.g., on smoothness, cannot be made due to the incorporated black boxes. In many cases, a suboptimal solution resulting in a major improvement of the objective function in only a few iterations is preferred over spending numerous resources for a possibly marginally better solution. The overall improvement from iteration to iteration in general decreases with an increasing number of performed iterations during an optimization run.
An example from water resources management discussed in this thesis illustrates best the emerging difficulties. An area with a contaminated subsurface close to drinking water extraction sides is considered. The distribution of the contamination, as well as the groundwater flow and pressure, are obtained from the output of an encapsulated numerical simulation. The decisions about the remediation installations are not fixed and can be selected freely for the simulation. Such decisions may include the location, number, type, and operation rate. Several of these are taken as optimization variables if the scenario is transformed to an optimization problem, and the objective function and constraints are defined by incorporating the response from the subsurface flow simulation. Hence, a constrained mixed-integer nonlinear optimization problem results and has to be solved. If

slight cost reductions for the real systems can be realized through better setup, the savings are enormous when considering the long term operation rates.

1.2 Problem Overview

Suitable optimization methods for the mentioned problem class have to rely on black box responses only and cope with high frequency and low amplitude noise, which disturbs evaluations of the objective function and the constraints. Hence, useful derivative approximations cannot be obtained easily, e.g., by finite differences. Derivative information is also not provided explicitly for the optimization problem. Gradient-based and Newton type methods therefore do not meet the problem's characteristics.
A wide range of derivative free optimization (DFO) methods are recommended for black box-based problems. These methods can be divided into three groups, (i) metaheuristics, summarizing random search methods or stochastic optimization methods, (ii) deterministic sampling approaches, and (iii) surrogate function-based optimization methods.
Almost all of these methods can handle noisy problems and some of them can even deal with problems including black box-based constraints. However only a few consider integer and continuous variables simultaneously. Thus, the development of suitable methods for black box-based mixed-integer nonlinear optimization problems was identified recently as a main topic for future research because many optimization problems are defined by variables of both kinds [249].
A first promising approach is based on the work of Audet and Dennis [8], implemented and applied as described in [1, 2]. The main component is a pattern search algorithm [265], which switches between search domains defined for each possible realization of the integer-valued variables. Apart from the curse of dimensionality of this approach, even for a small number of dimensions, a lot of problem adapted mappings between the different continuous-valued search domains have to be defined for any problem individually required.
A second group of approaches incorporates efficient methods known from mixed-integer nonlinear programming (MINLP). These methods originally rely on the relaxation of the integer-valued domains, or on the possibility of obtaining derivatives with respect to continuous and the integer-valued variables. These preconditions are not fulfilled for black box-based optimization problems. Noisy function values falsify any standard gradient approximations for the continuous-valued variables, Filter method may help to overcome this, but these problems do not provide responses for the continuously relaxed domains for integer-valued variables. Thus, no derivative information can be approximated for the integer-valued variables.
This deficiency of a non-relaxable domain for underlying black boxes can be overcome by applying surrogate optimization approaches. The black box components of objective function and constraints are replaced by functional approximations. The idea to use surrogate functions defined relaxed variable domain was applied within different surrogate optimization approaches [54, 127, 135], each of which was published independently.

1.3 Contents and Contributions

This thesis introduces a complete analytical formulation to solve black box-based, mixed-integer, nonlinear optimization problems in engineering. The developed problem formulation makes all explicitly given parts available for use by the applied optimization methods. On this basis it is possible to apply the developed sequentially updated surrogate function-based optimization approach to overcome the difficulties of existing methods described above. The resulting optimization procedure solves the addressed general class of mixed-integer black box-based problems. It is envisaged to obtain fast reduction of the objective function value even for a small and restricted number of black box evaluations.
Chapter 2 introduces the key elements of optimization problems in engineering, including the main notations for the elements involved as underlying systems, models, and simulation tools. The main concepts of simulation-based optimization are discussed as well as how they differ from black box optimization methods. It is outlined how the problems of interest result in black box optimization problems and provide a detailed overview of DFO as well as of MINLP methods. This is done from an optimization problem-oriented perspective, and it will be realized that both categories of optimization methods cannot cope with the emerging difficulties.
The general problem formulation is introduced in Chapter 3, followed by a discussion how surrogate optimization can help to overcome the deficiencies of DFO and MINLP by deriving a new technique from both fields. An approximation method is outlined in detail to generate appropriate surrogates for the black box response. Chapter 4 introduces the basic algorithmic framework realizing the ideas of the preceding chapter. This includes the strategy of how to choose candidates, for which the underlying black box should be evaluated at the beginning and during the update phase. The quality of the surrogate is improved from iteration to iteration by the newly obtained responses of the black boxes. We will discuss further options for parallelization as one possibility to reduce wall-clock-time of optimization runs.
Several engineering applications are presented in Chapter 5 and 6. The examples from electrical engineering and environmental engineering provide numerical simulation-based optimization problems, to which the proposed surrogate optimization approach for mixed-integer problems is applied. The arising optimization problems contain all the difficulties mentioned above and described in detail in Chapter 2.
Chapter 7 considers problems where real world systems are included as black boxes into the optimization problems. Results obtained from the developed surrogate optimization approach will be presented, namely for walking speed optimization problems of four-legged and two-legged robots. Finally, Chapter 8 concludes this thesis by summarizing the main results and describing further directions for research.

2 State of Research in Optimization Methods

Real systems are mapped to models for simulation purpose. The effects of interest of the system are usually formulated by mathematical equations. This introduces the need for models, modeling processes, and numerical simulations as much as this is possible [14]. Glanzhorn [112] stated already more than 25 year ago:

> *Models and simulation techniques are considered the most important area of future information processing in technology, economy, administration, society, and politics.*

Modeling and numerical simulation has become a 'third column' for the state of knowledge in almost every field [208]. If we know that simulation models fit to the underlying real system under the preassumptions that are made for modeling and for the transfer of the simulation response back to the real world system, it is only a small step to ask for the main input factors and its effects in optimizing the underlying system.
The simulation is often already built and can be considered for black box optimization as compared to the original system. The main difference between both is that the first one is typically deterministic, and the latter one includes also stochastic properties. In this thesis we consider the case where continuous-valued as well as integer-valued parameters are of interest for the optimization problem. We proceed with the current state of research and provide sufficient background information on black box-based, derivative free, and surrogate optimization as well as on MINLP.

2.1 Underlying Systems and Approved Optimization Approaches

Abstraction and models are the fundamentals of science. In astrophysics the modeling of stellar systems is standard, as in chemistry the research on proteins and the design of drugs. In biology or medicine the growth of plants and tumors are modeled, and nano-structures are simulated in material sciences. These are just a few well-known examples beside the ones we will consider in this thesis to show that modeling and abstraction is standard practice.

2.1.1 The Systems of Interest

A system of interest is a definable part of its surrounding environment with in- and outputs. The system states describe its status at a certain point of time. Typically, not all components of the states can be observed from the outside [32]. The system is the aggregation of different internal components, which interact closely and determine the system's

behavior. The important parts of a system of interest have to be classified separately and at an appropriate level of detail. A model fitting process to map the real system is only promising under these preconditions. More detailed introductions on the definition and classification of systems are given in [14, 211].

2.1.2 Relevant Simulation Models

An aim for using a model apart from observing true behavior is to collect information on the internal system states which are typically not observable. Knowledge to forecast (what-if) or to construct real systems ought to be achieved with models. Models are often used today to emulate properties of a technical system or a phenomenon of natural sciences. A definition is also given by Shannon in 1975 [240]:

> *Simulation is the process of designing a model of a real system and conduction experiments with this model for the purpose either of understanding the behavior of the system and its underlying causes or of evaluating various designs of an artificial system or strategies for operation of the system.*

A model is built under special assumptions and idealizations, it is a copy of a simplified partial reality. The level of detail of real world systems is too high to be identically mapped to a model for simulation [167]. We always have to deal with model simplifications [175]. The purpose of a model almost completely determines the level of detail required.Let us think about a model of a robot arm. Are we only interested in a representation of geometry and kinematics or in relevant forces to control the movement of it? The degree of abstraction has to be chosen in a way in which the model still describes the relevant properties on an appropriate level [32].

Classification of Models

Analytical models can be separated from simplified experimental setups [244]. They not just provide a mathematical description of the system, but they lead into numerical simulations.
A further distinction for analytical models is the internal representation of the state domain, discrete and combinatoric descriptions versus continuous descriptions of the modeled properties. Changes of the state are given by graphs and automats for discrete models and the states are described by binaries or integers. Continuous models use real values to describe the relevant states of the system and consist in general of algebraic as well as differential equations, physical units, and typically continuous changes over time.
The method of modeling a problem determines its representation. A simple traffic flow problem, for example, can be modeled in both ways. In the discrete case this can be done through the total number of cars, queues, and discretized roads. The continuous model would be more like a fluid dynamics model including conduits, branches, and barriers.
It is assumed that the states, as well as the simulation model internal time, may change only at the ends of an interval, or that changes are continuously valid. Such models contain discrete as well as continuously changes in the measurement scales. When a model is turned into a numerical simulation even the continuous components become discretized, e.g., by meshes. This may not even considered explicitly in the model.

A further property is the classification depending on the use of random numbers to describe state changes. A stochastic model uses stochastic elements to reproduce the systems behavior. Unlike these models a pure deterministic model will always produce the same output if the model assumptions do not change. These are the main characteristics of the input and output domains. Further aspects can be found in various textbooks on modeling complex systems like [175], [14], but the mentioned properties are the ones that have to be considered in this context.

Modeling Process

The classical method to set up a model is to use mathematical modeling by formal derivation and analysis. This process is subdivided into three steps, first, the informal description of the problem; second, a semi-formal description by standard formulations of the field application; and third, a strict formal and consistent mathematical description [175, 211].
These three steps of modeling are very akin to what is typically described as the process of abstraction, consisting of (i) the objective reality, unconscious abstraction of the system of interest, (ii) the subjective reality, modeling, conscious abstraction, and (iii) the consistent model of the system of interest. A main difficulty is the conflict of objectives between point one and two.
All relevant properties are not known a priori and the effect of idealization can be misjudging. Finally, it would lead to a model that does not fulfill the requirements for meaningful results. Bossel pointed out that the knowledge of structural information on the system enormously reduces the need for measured data from the system [32]. The simulation

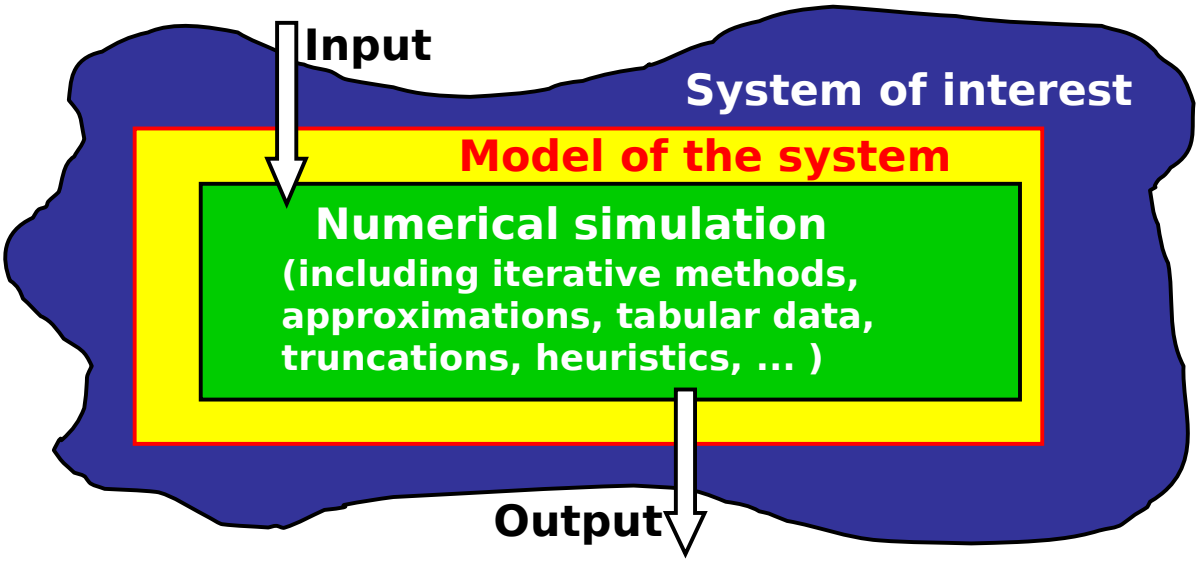


Figure 2.1: Simplifying the real system as part of the reality to set up a model to map it under the perspective on special properties.

model results finally from these processes and the properties of the underlying system. The model describes the relationship between the design parameters of the real system and properties of these parameters.

In some cases, it might be appropriate to generate more than just one model, maybe to build up a model hierarchy of different levels of abstraction or accuracy to validate one against another. Results obtained from optimization runs could be evaluated on a different model without using the real system in comparison.

Simulation

In many cases where the expression simulation is used, a final software to run a simulation model is intended. The simulation software consists of a physical-technical description named simulation model and implemented numerical methods. This combination leads to a software package for iterative numerical calculations about the underlying subsystems to get information about the system of interest (cf. Fig. 2.1). A broad overview on the evolution of simulation by computation in the past fifty years is given in [202].
Approved simulation software has often been left to accumulate data over years or even decades. Such simulation evolution almost definitely result in a very complex interior including heuristics and tabular data in their subroutines, obtained from real world observations by model parameter fitting, or just assumed by expert knowledge. The characteristics and classically known issues arising from the use of iterative numerical methods to solve the subproblems as truncations and approximation errors have to be respected. A lot of special knowledge and computational experience is often incorporated to provide the accuracy of these simulation packages. Furthermore, used simulation software in engineering applications is often only available in a compiled version. Hence, the explicit use of the internal components is almost impossible.

Simulation Verification, Validation, and Acceptance

The typical objective here is to obtain results that are transferable to real world problems. To test if simulation and models are meaningful, verification and validation of the simulation and its results are mandatory.
Verification has to answer the question, wether the conceptual simulation model with its model assumptions has been correctly translated into executable code. Proving this often consumes more time than the entire coding process. This is not done in just one step [167], but instead in parallel with the implementation and subsequent validation of model and simulation.
Validation means determining whether a model and its resulting simulation is an appropriate representation of the system for the particular chosen objectives [167]. This assumes a successfully passed verification test. However, validation is not just a single step in a simulation study, but rather, it is done repeatedly starting with the first available simulation model.
An important issue is the precision of the results, especially the dependence on the quality of input and model data is of interest. Often, validation goes hand in hand with the reformulation of preassumptions and tuning and changes of the model [175]. Furthermore, certain effects and physical properties of the underlying system may not enough apprehended to develop an appropriate simulation model for the allocated resources.
It is important that results obtained from the simulation are used for issues that are covered by the preassumptions made during modeling for the simulation. Otherwise results are taken with more confidence than it would be justified [14].

A posteriori observations are done for the validation by the real system's response. Sometimes the relevant systems are not available as required for validation. In such cases experimental setup of simplified parts of the underlying system can help to validate the simulation model. However, certain simulation models and software packages which increase the demand for optimization packages which easily can be coupled with the components that are already in use.

2.1.3 Adapted Optimization Approaches

A proverb of James Howell from 1659 describes the current situation where the simulation provide reliable responses,

> *Knowing all work and no play will make Don a dull boy.*

The resources to perform simulation runs repeatedly is provided by increasing available computational power. Thereby the optimization of simulation-based problems becomes reasonable. However, the costs for running simulations are, especially for complex problems, still an important factor. Further reasons for the optimization of system and simulation-based problems are manifold: short time to market periods, lost expert knowledge to design by hand and intuition, or the diversification of products impeding the growth expert knowledge.
Stuckman et. al. [259] differentiate between three approaches, (i) trial and error, (ii) intended variation in an area around an initial set, and (iii) approaches where the simulation is coupled with appropriate optimization methods. Today, most approaches belong to the third group, but almost exclusively for problems with either continuous-valued or integer-valued variables. No efficient optimization approaches are available for mixed-integer problems. Furthermore, today's simulation packages are criticized for their lack of providing necessary information for optimization [262].

Model-Based Optimization

Simulation-based optimization is equal to black box optimization for many users. However, this is only correct if the only output from the simulation software are values for the states without providing any further knowledge on the internal model and its structure.

Tailored Coupled Approach Problem specific approaches can be applied if we derive optimality conditions directly from the model equations. This requires more information about the underlying simulation model, typically the complete mathematical expression. One example of how this can be realized is given in [27] where optimal blade cross sections are determined by a tailored coupled simulation and optimization approach. A resulting program solves simulation and optimization simultaneously. The drawback is that such a program has to be built from scratch, based on explicit system information. Existing and user approved simulation software is eventually discarded in such a situation.

Symbolic Differentiation Mathematical models are often too complex for manual differentiation. A way out is to use symbolic differentiation, which automatically generates the derivatives.

We do not have to deal with truncation errors when symbolic differentiations are applied. The disadvantage is that even for straightforward input formulas, very complex expressions are generated, resulting from the strict use of the differentiation rules. A very high computational effort results if we apply it to complex systems, as is the case in engineering applications. Anyhow, when models and the obtained derivative information is given analytically, efficient gradient-based optimization can be applied and optimality conditions can be proved as well.

Variable Complexity-Based Approaches

Main characteristics of systems and simulations are often already covered by simplified models. These simplifications are still models, namely surrogate models, as Bandler et. al. [13] pointed out:

> [...] *the surrogate model is constructed using an available, low-fidelity (and physically meaningful) model of the object response (the model being a function of the design variables), rather than pure interpolation/ approximation. This is in keeping with the engineering tradition of developing for design purposes meaningful (not necessarily complex, often very simple) models of components of the physical world.*

This suggests that a model includes system information, maybe only on a coarse and abstract level, but is inherent. Two main ways for simplifications are known from literature, reducing the considered physical effects and reducing the computational effort by demanding less accuracy of the results. Madsen et. al. [183] describe that both ways came about naturally based on what is now, and what has been earlier, state of the art in engineering practice.

Space Mapping A less appropriate, and therefore less expensive, model is used to replace the original one. This surrogate model is calibrated to match the simulation model, called fine model, by parameter fitting through a set of function evaluations [12, 164]. Space mapping in its basic version maps the variable domain of the coarse model to the domain of the fine model. The mapping is given as the results of the least squares problem defined on the differences between both domains. In the easiest version it is linear function that is only on a trust region valid. Promising settings for the coarse model can be transferred through mapping to the fine model.
Apart from using local approximations as mappings, globally confident functions can also be used. Therefore, more sophisticated approximations than linear ones are necessary. Global convergence results for current space mapping approaches can be found in [184]. In literature regarding the space mapping framework, an implementation of the described proceeding, is unfortunately abbreviated the same way as done by Dennis and colleagues [29, 64] for their surrogate management framework (both by SMF).

Simplified Models Physical effects that are part of an underlying simulation model can often be replace by simplified simulation models. This is done by Alexandrov and Lewis [7] for an airfoil design problem with a first-order approximation model management optimization (AMMO) approach. The high fidelity model given by Navier-Stokes equations is replaced by a low fidelity model of Euler equations. Here, the difference is that a mapping

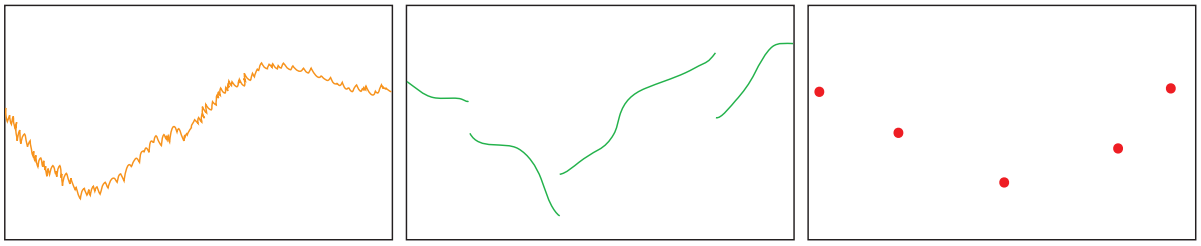
of the output spaces for the low and high fidelity models is performed and not a mapping between the input spaces of both.
The mapping in the output domains is done by additive or multiplicative correction functions. These function, predicting the difference or the quotient of both models responses, can be generated by functional approximations or even neural nets [168].
An important benefit of this approach is that the first and second order information can be used during optimization, as long as gradients are available by the high fidelity model. As pointed out in [7], models of this kind help to avoid the curse of dimensionality when compared to pure output data fitting methods.

Simplified Simulations In optimization for engineering problems, underlying simulation models are frequently based on the solution for partial differential equations (PDE) where the mesh size is a parameter that controls the level of refinements of the problem [7].
The coarser a mesh the faster a numerical evaluation run is performed. Varying complexity can be achieved by adjusting the mesh size. This can be utilized by multi fidelity approaches [64]. The basic idea is to start optimizing on an initial surrogate model and, if improvement is achieved, to switch to the next finer model to evaluate the obtained candidate. If improvement is realized the search is proceeded on that model. If no improvement is achieved, the approaches switches to the next finer model. The model is changed to a cheaper one if a lot more improvement than predicted is achieved. The model is not refined for improvement as expected by the initial model. A whole set of models with different fidelity is included in this approach. The difference between a fine model and a surrogate model could also be taken as an indicator for redefining the fine models.
The different fidelity levels of the models lead to hybrid approaches which couple surrogate models of different kinds. One surrogate model is used to calibrate another surrogate model at another level of detail, or even for the parameter fitting of the original model [266].

Code and Black Box-Based Optimization

Explicitly given analytical models may not be in any way accessible, but only included in a simulation software package. The introduced model-based approaches are not applicable in such a situation, and the preconditions for a successful use of any kind of first or second order information using optimization methods do not hold. Code-based approaches work directly on the simulation software code if available; black box-based ones are coupled in a loop to make use of the simulation's response [114].

Code-Based Derivative Information Automatic differentiation AD can be applied if the complete source code of a simulation software and of all its subroutines are available [26, 221]. AD is based on the fact that almost any complex function can be rewritten as a sequence of simple, fundamental functions of one or two arguments. Thus, complex functions can be broken down into an evaluation graph of fundamental ones for which derivatives can be calculated easily. A benefit of AD is that no approximation errors, only truncation errors occur.
AD is a growing field and already provides the evaluation of derivative information for increasingly complex software packages. Even problems given by ordinary differential equations (ODE) or partial differential equations (PDE) can be handled if the solver's source code is also available for AD. A wide introduction is given in [116].



(a) Noisy (b) Discontinuous (c) Discrete

Figure 2.2: Typical characteristics of black box-based objective functions.

Sampling-Based Approaches Black box optimization problems arise for applications where the complete source code, unlike the simulations software's responses, is not available. The simulation software becomes a pure black box response generator for the optimization. However, the approximation by finite differences will fail under the influence of the typical conditions of numerical simulations. The noise induced into the black box output is of high frequency and low amplitude [142] and will mislead any such approximated gradients. Furthermore no approximations can be made to obtain derivative information with respect to integer-valued variables. In [142] the arising problem characteristics are listed and visualizations are given in Figure 2.2.
Even if the derivative information, obtained directly from the response is as precise as required for gradient-based optimization, it requires a number of black box evaluations for each point of interest. Thus, sampling-based methods have to be considered for the reduction of the number of samplings during optimization.
The standard black box optimization problem finally can be defined by

$$\begin{aligned} \min \quad & f(\boldsymbol{\omega}) = f_{smooth}(\boldsymbol{\omega}) + f_{noise}(\boldsymbol{\omega}), \\ s.t. \quad & \boldsymbol{\omega} \in \Omega, \end{aligned} \tag{2.1}$$

where $\boldsymbol{\omega}$ represents optimization variables, and Ω the feasible domain for $\boldsymbol{\omega}$ [150]. The perturbations of underlying black boxes are induced by f_{noise}, and f_{smooth} represents the relevant parts of f. In contrast to [150] we consider the case where $\boldsymbol{\omega}$ summarizes not only continuous-valued but also integer-valued optimization variables. Thus, we have to carry out a noisy mixed-integer nonlinear black box-based optimization problem. In the following the expression candidate denotes one realization of $\boldsymbol{\omega}$.

2.2 Derivative Free Iterative Optimization

It is well-known how fast gradient-based optimization methods are [206]. However, the type of optimization problem impedes the direct use of any kind of gradient-based or (Quasi-)Newton type method, because no derivative information is available or of satisfying accuracy.
Anyhow, the basic idea of Newton type methods provides a way to classify DFO methods. Iterative numerical optimization methods can be differentiated simply by the number of terms they take from the objective function's Taylor expansion. Newton type methods use

it up to the second derivative. Approaches like the steepest descent use first derivative information. DFO methods make use only of the objective function value, so they are also called zero order methods [173].
The independence from derivative information makes DFO methods robust against small but discontinuous changes in the objective function stemming from infinitesimal changes in the variable space. Their structure is almost similar to the one of first or second order methods. They start with a beginning phase for initialization, followed by the iteration phase, and they end with a stopping phase. We will divide DFO methods into three main groups, the random search methods, deterministic sampling methods, and surrogate optimization methods.

2.2.1 Metaheuristic Search

The first group shortly merges random search methods and contains well-known representatives, such as genetic algorithms (GAs) and evolutionary algorithms (EAs). This group is sometimes also called stochastic optimization, however, stochastic optimization can also stand for optimization under uncertainties where the problem itself includes stochastic elements [68]. Random search methods utilize stochastic methods to generate sets of new promising candidates from earlier, already evaluated candidates. It follows that two runs of a metaheuristics search method starting from the same initial setting will not provide the same iterates nor the same solution. Most metaheuristics are inspirited by observed effects captured from physics, biology, or ethology [242].
One of the main characteristics is that occurring iterates, with increased objective function values in subsequent iterations, can be accepted. This provides the ability to escape from local valleys respective from local minima, and takes them to global search methods. A wide overview furthering the extension for random search methods are described in [71, 242]. Here we will introduce the main ideas and approaches.

Methods Using Integer/Discrete Variable Representations

Metaheuristics can be divided by the way they naturally encode the optimization variables values. First, those using discrete representations are discussed. Of course, these methods cannot only be applied to problems with discrete valued variables, but the space, on which the candidates are manipulated respectively updated, is a discrete valued one.

Genetic Algorithms The foundations for GAs were given by Holland in 1975 in [133]. Many further publications use and modify the idea of 'survival of the fittest'. GAs are population-based methods which apply selection and recombination as well as mutation, to generate out of elements from one generation the elements for the following generation. The elements in each generation are binary representations for candidates, defined from the variables domain of the underlying optimization problem. It is common that the objective function is typically defined as a fitness function that has to be minimized. Besides the objective function, penalty terms can also be included to handle constraints of almost all kind as reviewed in [200]. As for almost all random search methods, not much optimization problem information is needed. In addition, only a small number of parameters has to be set. However, this advance is typically paid for by high cost resulting from many function evaluations [100].

To overcome some of the drawbacks hybrid methods are proposed where e.g., local searches can be incorporated in each iteration [77] or in two stage approaches [176]. The handling of constraints is incorporated by a hybrid approach in [99]. These approaches use derivative information which makes them gradient-based and out of our choice. An approach for MINLP problems which also uses Newton type methods locally is presented by Gantovnik et. al. [100]. Further hybrid approaches are already applied, e.g., using surrogate function assisted optimization with trust region optimization on radial basis functions (RBF) to generate local approximations [283].

Tabu Search Tabu search is named after the methodology used to generate new candidates from a starting point [113]. First, a set of neighbors of a point is generated by elementary operations, called moves. As tabu search is introduced for integer-valued variables, such an elementary operation can be just a permutation of two components of the candidate.
The first step of a tabu search iteration is to generate the neighborhood of the initial candidate by applying all possible moves. Only a subset of the neighborhood is maybe randomly generated if the set would be otherwise too large. During the second step the objective function is evaluated for the generated points and the best out of the neighborhood is selected as the next candidate. This provides the ability to not get trapped in a local minimum. Inverting the "successive move" is prevented by listing the opposite of it onto the tabu list for a random number of further iterations, so that a return to earlier evaluated points is avoided. By saving the overall best found candidate, one iterations closes.

Simulated Annealing The effect that is observable when metal is cooling down is annealing [153, 218]. The use of this for a general optimization problem is done as follows. The objective function is treated as the total free energy of a physical system. The optimization variables are the positions of the particles, and the final solution, the optimum, is the state of the system with the lowest amount of free energy.
During each iteration, the locations of the elements change until a thermodynamic equilibrium is reached. For each modification step, the objective function value decreases. The modification is also accepted by a given probability if the objective function is increased. This enables it to get out of a local minimum. From an initial state, the temperature is decreased, iteration by iteration, until a solid state is reached and the variables values can be determined.

Methods Using Continuous Variable Representations

Evolutionary Algorithm The simplest version of an EA is to initialize it just by using one candidate from the variable domain. Hereupon, a number of elements for the first generation is built by mutation. Random vectors, possibly scaled by a further given scalars, are added. After evaluating the fitness function, only the best elements of the generation are kept and used to create the following generation for the next iteration. The basic idea is extended by various strategies for mutation and selection as described in [242].
Hence, almost all operators like the mutations and crossovers we know from GAs are also used by EAs. Often, it is not obvious if a binary representation or a real one is used inside an algorithm. Thus, GAs and EAs are often mentioned off the reel. However, the

representation of the optimization variable's values is not in binary but in real valued encoding. Especially if both, integer-valued and continuous-valued variables are involved, problems arise due to different encoding used for both types if special operators are not used inside [71].

Particle Swarm Optimization PSO is a population-based, heuristic minimization technique using the idea of social behavior to solve continuous-valued problems. PSO algorithms were first proposed by Kennedy and Eberhart [151] and are based on social communication within a group. A general PSO algorithm involves a set of particles representing design points used to explore the variable domain for minimizing the objective function. Each particle moves through space by updating its position with a velocity that accounts for the best position found for that particle and that of any particle. Convergence and efficiency are of great concern in the application of PSO [48, 86, 267, 279].
In [216] a further speed up of a particle swarm algorithm is obtained by incorporating surrogate functions within the algorithmic framework.

Further Aspects and Developments

We only mentioned some main ideas and methods in the area of metaheuristics because it is not in focus of this thesis. Besides, more ideas and new combinations of different optimization approaches can be found in literature, such as self organizing systems of agents with decentralized decisions like ant colony approaches [260], that are used to create new metaheuristics to solve optimization problems.
We have seen that all approaches include stochastic elements for generating new iterates respectively and new elements during one iteration. The representation of the variables is an important property and is owed to the underlying idea of each approach. Additional features like the type of memory are discussed for all kinds of heuristics to improve the performance [242].
What we have not seen are methods that naturally suit mixed-integer problems. All methods are sensitive to the used set of initializing parameters, which control main parts of the mechanism to generate new candidates. Algorithmic parameters are often set with expert knowledge and it is still an unsolved question how they are determined based on the problem that has to be solved. An approach for optimizing these initial parameters is given in [20]. Thus, the inclusion of difficult constraints is not naturally provided also by this methods.
However, metaheuristics can be expensive to use because of the large number of function calls required at each iteration. This is a significant drawback when function evaluations depend on the response from an expensive black box function value generator.

2.2.2 Deterministic Sampling

Deterministic sampling methods are the second group in our survey on DFO. Motivation for thinking about sampling is given by Powell in [218]:

> *[...] it seems more suitable, intuitively, to spread out the points at which the objective function is calculated, especially if sample values have to be taken from many parts of the space of the variables. Furthermore, when the values include*

> *some random noise, then the contributions from the noise to predict rates of change is less if the evaluation points are spread widely spaced.*

Thus, in relation to the used classification to separate the DFO methods herein, we propose that the expression deterministic sampling methods provides an appropriate frame for the following methods.

Direct Search

Direct search (DS) methods are originally defined as methods that do not make any use of the value obtained from the objective function, but instead using only ordinal information of the iterates to generate new iterates [136]. Lewis et. al. [173] provided a classification that separates them into three groups, pattern search methods, simplex search methods, and methods with adaptive sets of search directions. This goes along with the search direction property of methods mentioned in [264] and explains that the expression DS is today also used for directional search.

Basics The basic ideas were already developed in the 1960s and earlier [173]. However, the focus of the optimization community changed to Newton type methods until a revival in the 1990s, culminating in Torczon's important convergency conclusion for generalized pattern search (GPS) methods [265]. Famous GPS methods are the Hookes and Jeeves Algorithm, (multi-) directional search, and coordinate search [117, 173, 265].
New candidates for evaluation are selected by a finite number of linear independent search directions, e.g., a pattern provided around the starting point. After the evaluation of some or even all the generated points, the best one is selected as the new base point, around which a new pattern is generated. If no improvement is realized, the length on the search directions is reduced [150]. The different pattern search methods arise for the different strategies to perform these steps as well as for the different changes of the search directions from iteration to iteration.

Convergence Nelder and Mead's simplex algorithm [34, 203] is probably the most famous DS method, even if McKinnon shows in [198] convergence to non-stationary points. This occurs in special cases when degenerated simplices arise during the search. This can be resolved if congruent transformations are mandatory for the simplex in each iteration, resulting in the so-called multi-directional search approach [264]. Convergence to a stationary point is given if the used simplices remain bounded and form a regular lattice [150, 263, 264].
Beyond that, Torczon shows that GPS methods are descent and converge to a stationary point [265]. Multi-directional search or the basic Hookes and Jeeves algorithm [136] are special cases of GPS. Further results are shown for the convergence of grid-based pattern search [50] and for using local convergence to define stopping criteria [69]. Convergence is also proved for variants that came up for optimization problems with bound constraints [170], linear constraints [9, 171], and also for general nonlinear constraints [10, 172].

Extensions The limited number of search directions given in classical pattern search becomes critical when nonlinear constraints are involved [9]. This drawback is resolved by the mesh adaptive search as described by Audet and Abramson for general constraint

problems [3, 11]. In [162] further extensions are presented to improve exploration skills, e.g., the incorporation of oracle candidates with generalized set search (GSS).
In [8, 161] Audet and colleagues extend Torczon's GPS definition [265] to handle integer-valued variables, too. Unlike classical MINLP formulations they divide between those integer-valued variables, which change the optimization problem dimension and the impact of other variables on the objective discontinuously, and integer-valued variables that do not. The first ones are called categorical variables. Audet and Dennis favor branch and bound in those cases where it could be applied. However, no relaxation is typically possible if an underling black box expects integer values as input parameters [8]. It is essential for a successful adaption of this approach to define a neighborhood for each realization of the discrete variables, as well as a mapping from one to the next relevant variable space when categorical variables change [1, 182]. This fact bounds the usability for problems where the required information cannot be provided.

Using Function Value Information

The simplex gradient as defined in [150] can be derived directly from a number of objective function evaluations without any derivative information. It is the product of the matrix of the spanning set of the simplex in the variables space and a vector of the differences between function values of the resulting vertices and the generating vertex.
Kelley [150] also pointed out that even if methods only use ordering information about the vertices of each simplex generated e.g., by patterns, where the simplex gradient information is implicitly used, without knowing its explicit value.

Implicit Filtering The implicit filtering approach is a line-search-based algorithm and utilizes the simplex gradient in a direct way, usually the central difference simplex gradient [45, 107, 150]. Using an adapted Armijo line search rule, the definition of sufficient decrease arises straight forward for a basic version of implicit filtering [150]. This line search is iterated with a reduced vertex size of the simplex. It is controlled by stopping rules according to the number of iterations, the improvement, and the steepest descent size.
A more sophisticated version of implicit filtering is also introduced where the Quasi-Newton direction is calculated as the search direction based on the simplex gradient to accelerate optimization close to the minimum. For implicit filtering, no guarantee can be made to find a local minimum. In [150] it is proposed that the algorithm should be restarted from the result of a previous one for further improvement.

DIRECT The motivation is to measure the possible decrease of a Lipschitz continuous objective function in certain variable ranges, by combining function values and Lipschitzian constants as an indicator for promising regions of a minimum. Jones [148] describes his dividing rectangles algorithm (DIRECT) as a global derivative free nonlinear programming (NLP) method.
The basic version of DIRECT works on a variable space restricted by box constraints that is divided iteratively into subspaces. A ranking rise from the function value at the mid point and the size of the each subspace. The strategy is to combine both types of information. DIRECT takes those subspaces as candidates for further subdivisions which build the pareto front of center-to-vertex distance and lowest objective function value of

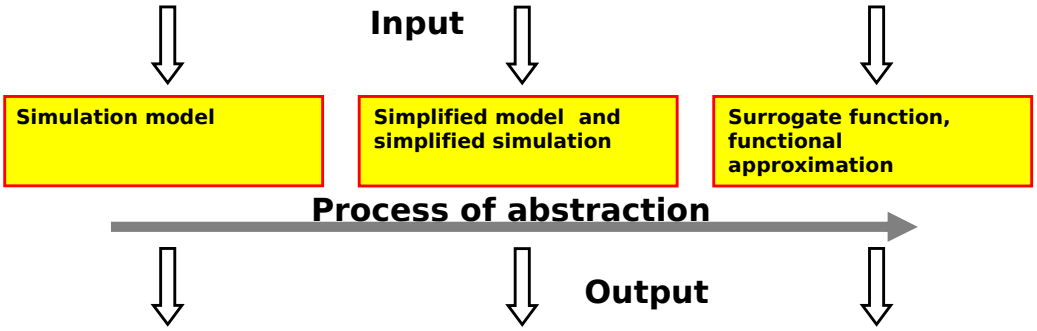


Figure 2.3: Process of abstraction, motivated by [155].

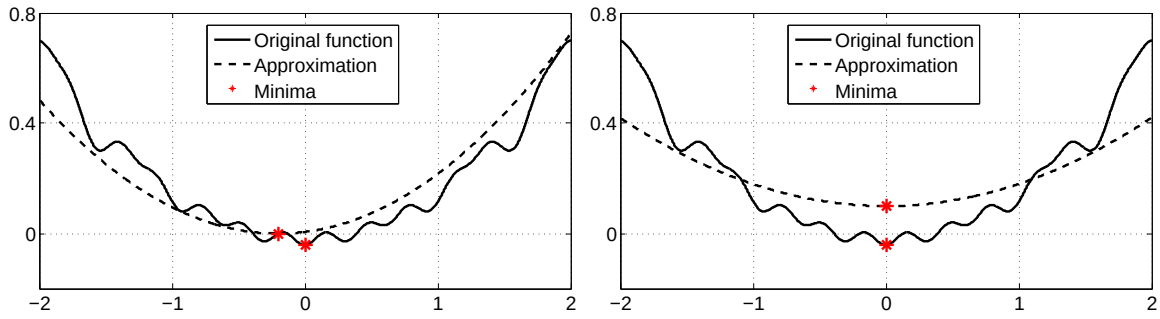
all known rectangles. Then, the algorithm chooses the mid point of the subspace with the lowest ratio for evaluation.
In [150] DIRECT is classified as between deterministic sampling methods and metaheuristic search methods, because it explores the variable space as aggressive as metaheuristics typically do. DIRECT applies function values directly for its search and explores the variable space, not based on search direction as previous methods and without requiring even one (feasible) starting point. On the other hand, DIRECT cannot use prior function evaluations in a hot start for a better performance, as other methods can.
Two extensions of the basic version are described in [146] to qualify DIRECT to handle general constraint problems and the presence of integer-valued variables. Constraints must be included as an auxiliary function, generated out of penalty functions so that only constraint violations are penalized. Only feasible points decrease the objective function. DIRECT still converges slowly to a minimum right on boundaries because the iterates are restricted to a grid defined by the center points of the subspaces. This is done in order to meet the integrality constraints for integer-valued variables.
In a mixed-integer problem the point for evaluation in each rectangle is moved to meet arising integrality constraints. Rectangles containing integer-valued variable dimensions are always subdivided such that at least one feasible value is contained.

Summary

Deterministic sampling methods make use of a kind of pseudo approximation through the simplex gradient, without using an internal model building procedure. Thus, these methods are not really designed to handle mixed-integer problems. Besides the DIRECT modification by Jones, only the approach of Audet and his colleagues can handle integer-valued variables. It, however, requires a defined neighborhoods for the integer-valued variables and a mapping between the variable domain for every possible integer-valued variable's state.

2.2.3 Surrogate-Based Optimization Techniques

Surrogate optimization represents solving surrogate problems generated with surrogate functions by approximation of function values that replaces the original objective function. The key feature of surrogate optimization approaches is that optimization avoids



(a) Surface fitting

(b) Surrogate-based optimization

Figure 2.4: Functional approximation for appropriate data fitting and for surrogate-based optimization, minima marked by stars, original function motivated by [150].

running directly in a loop with the original objective, but instead calls surrogate functions to determine new promising candidates. The response of each evaluation of the original function is stored, and the surrogates problems are generated from this storage. All other DFO methods do not utilize a memory that stores all earlier results from black box evaluations.
The terms surrogate model, surrogate approximation, approximation model, metamodel, response surface, compact model, or model of model are often used to describe the same idea in this context [61, 154, 224, 249, 253]. However, even if the term model is used, it stands for the approximation of response data in almost any case (cf. Fig. 2.3). We will use the expression surrogate function for a functional approximation of response value, and surrogate optimization for approaches based on such approximations.
Moreover, an overall fitting of the response values of a black box is not the purpose in surrogate optimization. An appropriate fitting on the entire domain would require a different approach for choosing basis points and different settings for the approximation technique. The approximation on the left in Figure 2.4 is much closer to the original function. The one to the right would, however, lead to the minimum of original problem with less iterations. The general interest is to find a candidate for minimizing the original problem quickly fast, and therefore the surrogate function only needs to catch the main characteristics of the underlying problem to lead to the minimum of the underlying problem. General overviews are given in [147, 219, 247, 248, 274]. Jones [149] declares that surrogate optimization often requires less function evaluation when it is compared to different other approaches:

> *[...] one can often interpolate and extrapolate quite accurately over large distances in the design space. Intuitively, the method is able to see obvious trends or patterns in the data and 'jump to conclusions' instead of having to move step-by-step along some trajectory.*

Building Surrogates

Barthelemy and Hafka [17] proposed a deviation between three groups of approximations, local, medium-range, and global approximations. Local approximation use Taylor series expansion information, derivative information respectively, and cannot be considered under

the assumed problem characteristics. Medium-range methods use function value information from the vicinity of an iterate. Global approximations provide a surrogate function for the entire variable domain. Between local and medium range methods we could set up approaches, provided by Alexandrov and Lewis [7], which are also designed for problems, where derivative information is available.
A well-known method to construct such surrogate functions is to use a stochastic approximation technique as described by Sacks et. al. [231]. Besides this, there are various other techniques for approximation known and already applied to find an approximation for evaluated function output data [19, 43, 143, 144, 251, 253]. Key factors for surrogate optimization approaches include the way to determine basis points, the internal parameter setting of the approximation methods [143].

Response Surface Methods The method of surrogate function building referred to as response surface methods (RSM) [34] is based on low order polynomials, mostly consisting of first and second order terms only [155, 199]. However, the term RSM is sometimes used as a synonym for surrogate optimization not depending on the kind of applied approximation method.
To fit a surrogate function to black box response we have to solve a least squares problem, or more general, a polynomial regression problem. However, surrogate functions built by low order polynomials are only accurate in a relatively narrow region of interest if the underlying black box is highly nonlinear [199]. Thus, the advantage is low computational costs, especially for local surrogates as used in trust region approaches, compared to other methods discussed later.
In [111] quadratic polynomials are applied as surrogates to smooth out noise that is induced into the objective function, and in [142] a branch and bound framework using RSM is utilized for a global optimization approach. It follows that RSMs are well-suited to get coarse impressions of the entire variable domain or to build just local surrogates [271].

Radial Basis Functions An overview on RBF can be found in [120]. Approximations of this kind consist of a polynomial part, as known from RSM, and a weighed sum of radial symmetric functions based on the euclidian distance between the basis points and the point of evaluation. The different classes of multivariate splines or multivariate adaptive regression splines belong also to the group of RBF methods [96].
Typically used types of radial symmetric functions are linear, cubic, thin plate spline, multiquadric, and Gaussian functions [224]. The way to determine the free parameters to approximate the underlying data makes the difference between the different RBF methods. The wise choice of the order of the basis polynomial and the type radial symmetric basis functions leads to special approximation functions like multivariate splines.
An application for a queueing system optimization using Gaussian RBFs is given in [44]. Turner et. al. [270] pointed out that the approximation accuracy is improved by smoothing cubic splines over RSM at the drawback of increased computational costs.

Design and Analysis of Computer Experiments This technique is often better known in its short form, DACE, which was finally established in 1989 for its use in numerical simulations [231]. It describes a statistical method used for approximation named Kriging [190] which is often used in geostatistical analysis [101]. Further expressions used are

spatial correlation models, frequency-domain approximations [19], or in optimization context Bayesian global optimization [169]. The main difference between DACE and Kriging is that instead of variograms, correlation functions are used to generate the covariance matrix [271].
The DACE approximation consists of two components, a global part to catch the trend for the whole domain and a local part driving the approximation to interpolate the basis points. The global part is often chosen as a constant or linear function, and rarely a low order polynomial. A combination of the functions values for the basis points weighted by the distance between basis point and point for evaluation build the local part. The closer a candidate is to an already evaluated point, the more positively correlated both function values are [158]. By maximum likelihood estimation, the process variance, the correlation parameters for the local part, and the regression parameters for the global part are estimated most consistently to the values to which the DACE approximation has to fit. Some similarities to RBF are given [149].
One benefit is that besides the function value estimation, an approximation quality indicator is provided by the expected mean squared error (MSE) of the surrogate. The MSE is often used in optimization approaches based on DACE as described below.
Surrogate functions generated by DACE usually become more accurate with more basis points, which is not the case for RSMs [271]. A disadvantage of DACE is that the process is computationally more expensive than the other methods described. This is due to the fact that, e.g., during the parameter estimation, an underlying nonlinear optimization problem has to be solved. A direct comparison between RSM and DACE is given by [226, 246]. More details on DACE will be discussed in the next chapter, and for a detailed discussion on the theory behind DACE we refer the reader to [158].

Trust Region Versus Global Approximations

A strong analogy between gradient-based trust region methods and surrogate-based methods exists, if surrogate functions are applied locally [80, 269]. Both types generate local approximations to obtain the search direction and the stepsize for the next iteration. Surrogate-based methods do not use first and higher order information to build an approximation but, rather, the function values of already evaluated candidates in a subregion around the current iterate [49].
The different steps for a trust region approach using surrogate functions are given in [109]. An approach where the location of points, which have been evaluated earlier, is used to define an adapted (ellipsoid) trust region [73]. Reliability is taken from the number and the location of the points, which are contained in the trust region. Furthermore, an approach using information of a sparse and band structured Hessian to reduce the computational costs is presented in [49].
The difference to approaches with global surrogate functions is that the surrogate function is not valid over the complete variable domain. A hybrid approach is described in [42], where the domain is separated a priori and local surrogate functions are built exclusively for each subregion to search for minima on each of them. Another hybrid combination of local and global surrogates can be found in [283].

Basis Point Selection for Approximation

If a set of initial candidates for evaluation cannot be obtained, through expert knowledge or otherwise, the selection should be done using methods which do not need prior information. Additionally, the choice of the basis point is as important as the approximation technique itself, in order to build a surrogate function [143]. Beyond this, Kleijnen et. al. propose to use robust designs rather than what is known as optimal designs. Often, the underlying black boxes do not hold the assumptions made a priori to derive such an optimal design [157].
We discuss three ways to find basis points, one requires no prior knowledge, a second where prior determined points are included to build bottom up a larger set but still without black box response data. Sequential methods is the third way which uses the points and the response values obtained earlier to build up surrogates to determine further points to be added.

Classical Design of Experiments Experimenters have to define a priori points for measurement, without knowledge of certain responses of the experimental system. Design of experiments (DOE) summarizes how initial sets of candidates for measurement are selected. It can be divided between the classical purpose in context with real experiments, and the purpose for computer experiments [33, 230]. Simpson et. al. point out that random sampling is a bad choice compared to any other initial sampling method, if no space-filling rule is applied [247]. A incomplete and little appropriate use of DOE techniques is even better than applying 'trial-and-error' or 'build-and-break' approaches to engineering design [110]. These approaches are also called 'one shot' approaches, because all candidates for evaluation are determined in one step [157]. Simpson gives a practical and important issue for an initial design for time-consuming evaluations [245]. The initial design should contain as many points as can be evaluated over night, because during this process evaluations are only done without the chance to engage.
Initial sets should be chosen as space-filling and non-collapsing if variable dimensions are removed. Further criteria are orthogonality, efficiency, the ability to handle constraints on factor-level combinations and ease of construction and analysis [157].
A main difference to real world experiments is the lack of random error and the repeatability of computer experiments. This was already mentioned in [180] as a special chance not just to use standard DOE methods developed for physical real world experiments but to derive design methods adopted to the special properties of computer experiments.
Classic DOE methods include Monte Carlo Sampling, Hammersley sampling, Sobolev sequences [110], and uniform designs to generate initial sets [87]. In computer experiments well-known examples include are Latin hypercube designs [31, 110, 257], Orthogonal arrays [210], and covariance matrix based designs like A- or D-optimal methods [201]. Several other DOE methods, as well as hybrids of them and further aspects, are described in various publications providing a wide overview [155, 157, 158, 247, 272].

Sequential Designs without Use of Response Data The second important group of methods to generate initial sets starts with the given set of points and adds further ones. These methods can extend sets so that costly, already obtained information is reusable for the generating surrogate functions. It enables the selection of numerous further points for evaluation, which can also improve the set of points to a more space-filling one. A generally

valid minimal size for an initial design was not found in literature, but in [180] ten times the number of active variables is proposed. Basic methods are maximal entropy designs and minmax designs, more on these methods can be found in [44, 52, 53, 158, 233, 270].

Response Allocating Sequential Designs These methods are also called response-adapted-designs [124]. An overview on the main update criteria for sets of candidates that are response-based is given in [270]. The general procedure is the same for most approaches. The first step is an initial set of points which is given by one of the two previously mentioned procedures, or just from given results from earlier evaluations that were made by any reason. A surrogate function is fitted to output values and, based on a criteria defined, the next candidates as the most promising point, is chosen [238, 270].
The standard approach iterates the following steps until kind of convergence is realized. A surrogate function is built using information obtained earlier from black box evaluations. Then, the minimum on this surrogate has to be determined and, finally, the resulting candidate is evaluated by the original objective function. In literature this is referred to as strawman approach.
In [224] a sequential approach is presented, which applies a strawman-like approach in combination with RBFs. The difference here is that additional constraints are added to the problem, providing a minimal distance a new candidate has to hold to other points of an existing set. This prohibits points from clustering too close together. Keys and Rees [152] proposed to select new points for evaluation based on the distribution of the quantiles of estimated second derivatives of the surrogate function. A rather different idea is to define a measure of bumpiness and minimizing this by assuming changes for added points [120].
Approaches using stochastic approximations like DACE often utilize information of uncertainty combined with the idea of the strawman approach, in order not to get strapped in a local minimum. The uncertainty information supports more global exploration properties [147].
Sacks et. al. [231] minimize the integrated MSE on the variable domain, whereas Trosset et.+al. [268] subtracts the quare root of the MSE from the DACE surrogate. Schonlau [237] merges surrogate functions and the probability of points having a lower function value for the so-called expected improvement criteria which is maximized to find the next point. Constraints can also be incorporated directly into an objective function for minimization this way [238]. The variance, respectively, the MSE, vanishes at already evaluated points because there is no uncertainty on the function value. The expected improvement function value becomes zero for these points. For the generalized version, an additional parameter balances the local and the global search.
Several other criteria are reviewed in [235], but no criteria has been identified that outperforms the others in terms of function value improvement and the number of used function evaluations. Thereupon, the heuristic is proposed to switch every five iterations of minimizing the surrogate function to maximizing the variance of the surrogate to obtain new candidates. In [233], this switching criteria is the most consistently high-performance method for a sequential design. A similar idea is also proposed in [29].
Sequential designs are well-suited for expensive black box-based optimization problems, but are still very sensitive to the initial set of points [72, 147]. The response data of an initial set of candidates could provide a misleading picture of the underlying function.

Beyond that, most methods that balance improvement and uncertainty simultaneously often do not explore the variable domain as aggressive as strictly strawman approaches do. Thus, they provide a better chance to find globally good candidates [233]. Local exploitation might be the better choice for problems with a small contingent of black box evaluations [169].

Optimization on Surrogate Problems

Optimization on surrogate problems does not have to deal with incomplete and misleading information, compared to the optimization directly on problems based on black boxes. The objective function of the surrogate problems is an explicitly given mathematical function. It enables the use of gradient-based first and second order optimization methods. However, DFO can also be used for any reason if the computational cost for optimization performed on the surrogate problem does not have to be taken into account.
EAs and GAs are used for optimization on DACE approximations in various approaches [76, 222]. In [100], it is reported that 90% of computational cost are saved by using the, therein proposed, approximation procedure. In [82] several EA frameworks using surrogate function are described. Within a hybrid framework an EA on a global surrogate function is used to generate new candidates as the starting point on a local surrogate function [282]. DIRECT is used for identifying the next candidate within the expected improvement criteria in [235].
Regis and Shoemaker [224] proposed, apart from using only standard Newton-Type methods, a hybrid method using DIRECT to generated starting points for Newton-Type methods [40]. Branch and bound approaches together with underestimators are proposed in [149] to add global search properties. A pattern search was applied for surrogates optimization approaches denoted, as the surrogate management framework SMF or model assisted pattern search [29, 64, 243, 269]. It integrates candidates identified on the surrogate problem into a pattern search algorithm, so that the convergence results for pattern search methods still hold. This way, promising candidates for a low objective function value are included, as well as candidates providing global exploration abilities.

Summary

Rules of thumb to decide which method to choose for a surrogate are given in [248]. DACE is suggested for problems up to 50 optimization variable dimensions, as well as for deterministic problems, because it is flexible and interpolates. RSM should be used with up to ten dimensions and only for non-highly nonlinear responses. It would also suit stochastic problems because it is a non-interpolating method. For a three-dimensional example, RSM and DACE performed almost equally [246]. Open questions are the way to set the free parameter and basis functions for any approximation method, as well as to define stopping rules. Standard choices for the latter are the number of iterations, the improvement per iteration, or the applied step size [28].
The validation of a surrogate once or in each iteration, is a special issue and is addressed by several publications [156, 156, 249]. Cross validation approaches provide just one, often used, tool for it [42, 149, 187, 209].
A surrogate function, once build for optimization purposes, can also provide additional information on sensitivity [149]. The analysis can be performed by screening [155], factor

interaction analysis [230] or analysis of variances (ANOVA) [31, 140, 145, 226]. Beyond this, a general introduction into sensitivity analysis is given in [232].
ANOVA can help to understand the main drivers of a problem when information is available about the importance of each variable dimension. In [28], the most important variables were identified by ANOVA to derive a ten-dimensional problem from a 31-dimensional one. A similar approach is presented in [254].
As we can imagine, the idea of using surrogate problems to solve optimization problems gives a lot possibilities to design frameworks using more than just strictly one optimization method. Examples are the use in combination with a GA [121], or within a PSO framework [216].

2.2.4 Constrained Optimization Problems

Gradient-based methods often incorporate linear and nonlinear constraints directly. However, most DFO methods are not made to incorporate the different kinds of possibly arising constraints. Hence, the optimization problem can be transformed into an appropriate formulation to close the arising gap for DFO methods.
The classical approaches are to use different types of penalty functions. These functions help to transform a constrained problem into unconstrained, by penalizing the violation of equality and inequality constraints with an extended objective function. An exterior penalty function penalizes if constraints are violated. Realized violations of constraints are added as a weighted sum to the objective function, multiplied by a factor that is often raised from iteration to iteration to identify a feasible region. The best known example for an exterior penalty function is the quadratic penalty function, which uses no special weights but adds the sum of squares of the violations. Another way would be to add a barrier term to the objective function. This guarantees feasibility in each iteration and is important for problems that otherwise cannot be evaluated. The most important function of this kind is the logarithmic barrier function, which provides continuous differentiability of the transformed problem. These approaches solve a sequence of optimization problems to find the optimum of the origin problem.
There are further approaches using different penalty terms added to the objection function, or even to the problem's lagrangian. They provide the solution of the origin problem under certain assumptions directly, but for more details we point the reader to standard textbooks on numerical optimization [104, 206]. Different filter approaches [89] to incorporate nonlinear problem constraints are applied to expensive black box problems in [9, 36, 65, 186]. Finally, it can be stated that methods to include general constraints still do not often found their way to the practitioners for the use combined with DFO methods.

2.3 Mixed-Integer Nonlinear Optimization

The last section has shown that existing DFO approaches do not have the capacity to handle mixed-integer problems considered. However, if the objective is given without induced noise and completely mathematical it is still a challenging task, because mixed-integer problems combine the difficulties from both continuous and discrete optimization. First, the integer-valued variable domain sets up the combinatorial problem that has to be solved, and with an increasing dimension of this component an NP-complete problem arises. Sec-

ond, nonlinearities and even the nonconvexity of the problem add further complexity [91]. A brief overview of optimization methods and software for problems with integer and continuous-valued variables is given in [39]; a detailed list of different applications can be found in [92]. In this section we will focus on the main advances in MINLP, considering they could help to solve black box-based problems.

2.3.1 Problem Statement

It has to be assumed that the problem defined in equation (2.2) is not disturbed by noise and provide exact derivative information. Additionally, it has to be defined on a variable domain that is relaxable according to the integer-valued subdomain. In addition, $f$and $\boldsymbol{g}$ need to be convex functions to find the global optimum with local MINLP methods.

A Problem Statement

Common formulations for MINLP problems are equivalent to the following one,

$$\begin{aligned} \min \quad & f(\boldsymbol{p}, \boldsymbol{s}), \\ s.t. \quad & g_i(\boldsymbol{p}, \boldsymbol{s}) = 0, \text{ for } i \in I_{eq}, \\ & g_j(\boldsymbol{p}, \boldsymbol{s}) \leq 0, \text{ for } j \in I_{neq}, \\ & \boldsymbol{p} \in \mathbb{R}^{n_p}, \ \boldsymbol{s} \in \mathbb{N}^{n_s}. \end{aligned} \tag{2.2}$$

The objective function is given by $f$as in equation (2.1), with $f : \overline{\Omega} \subseteq \mathbb{R}^{n_p} \times \mathbb{N}^{n_s} \to \mathbb{R}$. Bounds, linear, and nonlinear constraints are summarized by $\boldsymbol{g}$, with $\boldsymbol{g} : \overline{\Omega} \to \mathbb{R}^{n_g}$. Both functions are well-defined for $(\boldsymbol{p}, \boldsymbol{s}) \in \overline{\Omega}$, where $\Omega_{\boldsymbol{g}} \subseteq \overline{\Omega}$ is the domain for feasible tuple $(\boldsymbol{p}, \boldsymbol{s})$. The elements of the index sets I_{eq} denote the equality constraints, elements from I_{neq} the inequality constraints, where the total dimension of constraints is given by n_g. The solution for (2.2) is given by $(\boldsymbol{s}^*, \boldsymbol{p}^*) \in \Omega_{\boldsymbol{g}}$. A more detailed problem description including strategies for modeling these problems is given in [207].
Furthermore, starting with this formulation almost all other kinds of static optimization problems are special cases of it. The standard NLP problem arises by ignoring $\boldsymbol{s}$, the standard mixed-integer linear programming (MILP) problem by assuming $f$and $\boldsymbol{g}$ to be linear. Often $\boldsymbol{s}$ is introduced as a binary variable, but such a formulation is equivalent to the one introduced in (2.2). The same holds when $\boldsymbol{s}$ can take on only discrete values but no integer values. Transformations for both cases can change the dimension and size of the variable space $\overline{\Omega}$, but w.l.o.g. an integer-valued $\boldsymbol{s}$ is assumed in the following.

Problem Decomposition According to Developed Methods

In early publications like [102] a problem separation is used so that the integer-valued variables, namely the complicating variables, arise only linearly in the objective function. This also reflects the history of the devolvement of numerical solvers in two areas of research, NLP and MILP.
Grossmann describes in [118] a problem decomposition by formulating main subproblems that are generated and investigated by the developed numerical methods for MINLP. First he introduces the relaxed MINLP problem defined on the continuous relaxation of the variable domain according to its integer-valued subspace, as given in equation (2.2). The result of ignoring the integrality constraints for $\boldsymbol{s}$ is a NLP subproblem. The solution of

this problem typically provides no integer-valued candidate for $\boldsymbol{s}$, but a lower bound for the original MINLP is generated. Introducing and varying bounds for $\boldsymbol{s}$ will lead to branch and bound approaches, as we will see. By fixing the integer-valued variable $\boldsymbol{s}$ an NLP subproblem arises of the dimension n_p, because $\boldsymbol{p}$ is the only optimization variable. The solution provides an upper bound for the original problem. The third NLP subproblem formulated by Grossmann deals with nonlinear constraints. If we take the latter NLP with fixed value for $\boldsymbol{s}$, and assume the case that no feasible initial candidate for $\boldsymbol{p}$ is available, minimizing the constraint violation results in a further NLP subproblem.
Besides this, a linear optimization problem arises by replacing the nonlinear functions that are depending on $\boldsymbol{s}$ by supporting hyperplanes. Derived from linearization, the objective function is underestimated, and the feasible region overestimated. The variables of this MILP problem are $\boldsymbol{p}$ and $\boldsymbol{s}$, and a solution provides a valid lower bound for the MINLP problem.
Using this decomposition methods, Grossmann reduces MINLP problems to a series of NLP and MILP subproblems that have to be solved iteratively. MINLP strategies that apply this decomposition in MILP and NLP problems keep the benefit of efficient codes that have been developed in both areas.

2.3.2 Numerical Methods

In the following we summarize methods that apply the main ideas via decomposition in subproblems to solve MINLP problems [220].

Outer Approximation (OA) Methods

OA is originally designed for problems where the nonlinear part is convex and only depending on continuous-valued variable $\boldsymbol{p}$. The integer variable $\boldsymbol{s}$ is only a part of linear functions, as described [75]. The extension to general MINLP problems is given in [90]. It uses a relaxed MILP master problem combined with the NLP formulation, with fixed values for $\boldsymbol{s}$ combined in a cycle. The linearization to set up the MILP is consistently improved while the iterations go on. For the convex case, a sequence of non-decreasing lower bounds of the original problems is provided [118].
In [90], a further algorithmic extension is also discussed which includes a mixed-integer quadratic master problem that uses second order information. The NLP problem is solved in each cycle for the result of the MILP run, and the solution is an upper bound for the original MINLP problem.

Generalized Benders Decomposition (GBD)

Benders decomposition was first introduced for optimization problems with a linear part based on the integer-valued variables, and a nonlinear part containing all continuous-valued variables from the linear part besides more continuous-valued ones [23]. GBD extends the Benders decomposition and is introduced in [102] as a MINLP technique. The subproblem, parameterized by integer-valued variables, does not have to be linear any more.
OA and GBD share the fact, that the decomposition is not bounded to the integer variables. For GBD, it is assumed that the decompositions is realized by fixing the complicating variables, even if those are continuously valued. The NLP is similar to the one arising

in OA in the convex case. It provides an upper bound for the solution. The MILP master problem is the main difference between OA and GBD. GBD considers only active constraints [118, 220]. Thus, the dimension of the master problem increases with the number of iterations for both approaches.

Branch and Bound

The idea was already introduced for the use with MILP problems in the 1960s; an introduction is given in [91]. The nonlinear case is similar to the linear one. The branch and bound approach starts with an NLP problem which is defined on the continuously relaxed variable domain for $\boldsymbol{p}$ and $\boldsymbol{s}$. In the case that the solution already takes integer values for $\boldsymbol{s}$, no more work has to be done. Typically, this is not the case, and only a first lower bound is provided. A bunch of NLP subproblems is generated by varying bounds for the different components of $\boldsymbol{s}$. Each of them can be associated with a node of the branch and bound enumeration tree. The solutions provide lower bounds for the subproblems of the descendent nodes. A node and all descendent nodes can be eliminate as candidates for a solution if an associated lower bound is undercut by an upper bound from a different section of the enumeration tree. Upper bounds are found in nodes where $\boldsymbol{s}$ is completely fixed, or a solution is found with integer values for all components of $\boldsymbol{s}$.
The method's performance is directly correlated to the number of NLP problems, the complexity of them, and the computational costs to solve them. Thus, the strategy followed during generation of the enumeration tree and the node selection has an enormous influence on the overall performance. Already in 1985, Gupta and Ravindran [119] give a comparison of different branching strategies for convex MINLP problems. The results imply that those integer variable should be branched, having the most fractional value. In [74], an approach is presented that uses a probabilistic selection method for the subproblem to navigate through the branch and bound enumeration tree, and [163] describes a method where nodes selection is done using a GA. It is still an unsolved question which strategy outperforms others for the majority of problems.
In [174] the authors describe an integrated approach, where the separation between branch and bound and the resulting NLPs is removed. At each node of the branch and bound tree, only a quadratic program has to be solved, and not a complete NLP. This procedure is interpreted as an improved, lower bounding scheme for branch and bound.

Extended Cutting Planes

This is the first approach considered in this thesis that does not use problem decompositions including NLP subproblems to solve MINLP problems [277]. This is performed iteratively, each time adding linearizations of the most violated constraint to the problem. Extended Cutting Plane methods can only handle linear objective functions. Thus, some problem transformations have to be made to eliminate nonlinearities for the cost of additional variables and constraints.

Further Approaches

In [4, 92] wide ranges of different methods are presented in more detail, including references to a lot more publications in this field. Further methods are also developed based on the described approaches. The main intention of these approaches is the ability to

handle non-convex problems. Examples are generalized cross decomposition, feasibility approaches [192], recursive quadratic programming [134], different branch and bound extensions [91], and logic based approaches by generalized disjunctive programming [118].

2.3.3 Mixed-Variable Problems

Geoffrion [102] used the term complicating variables to describe the influence of the integer-valued variables when he introduced GBD. Benders [23] defined problems with integer-valued variables as mixed variable problems (MVP). Today these problems are referred to as MINLP problems. The expression MVP was picked up again in [1, 2, 8]. Therein, the integer-valued variables are divided into two different types, those that have no deep influence on the structure or characteristic on the optimization problems, and those that have. The latter ones are categorical variables and are the complication ones for black box-based MINLP problems.

Categorical Variables and Non-Relaxable Domains

Categorical variables arise when decisions like system topology or material to use are included as optimization variables [2, 127, 181, 182]. The result is that no relaxation on the variable domain of the categorical variables is possible. The use of MINLP methods is not possible anymore. Thus, these problems do not belong the group of relaxable problems [39] that can be solved by existing MINLP.
Abramson [2] mentioned that a linear connection in the categorical variables makes problems easier to solve, especially if system information is needed to generate mappings between variables space for changing value of the categorical variables.

Adopted Approaches for Non-Relaxable Problems

Extensions for Sequential Quadratic Programming A trust region sequential quadratic programming (SQP) method for non-relaxed problems with only smooth, almost linear influences resulting from the integer variables was adapted in [84]. Categorical variables are explicitly removed in this approach. We do not focus on the trust region methodology here. To include the integer-valued variable, an approximation of the first derivative with respect to the integer-valued variable is needed. This is obtained by calculating central finite differences, using the closest available point in each component of the integer-valued variable. This results in a linear approximation by three function evaluations for each component of the derivative. A comparable approach is also given in [41]. Especially for expensive function evaluations, such a proceeding generates high costs.

Variable Domain Relaxation The idea to relax the integer-valued variable domain by a surrogate optimization was discussed recently by three different independent groups [55, 127, 135] and applied within the surrogate optimization approach derived in the following chapters of this thesis. Holmström et. al. [135] considered a situation where the objective function is expensive to compute, and no derivative information available. He used MINLP on the surrogate objective function, which were generated by RBFs for smooth test problems. In [54] the idea of relaxation by surrogates is adapted for process synthesis

applications. The approach is embedded in a response surface methodology of local surrogate functions, where only a global surrogate function relaxes the integer-valued domain and enables the use of branch and bound to identify those promising regions where the local surrogate is applied [55].

2.4 Summary of Chapter 2

No further information beside a noisy function value of f is provided for the black box-based objective function. The same holds for the problem constraints that are also based on black box evaluations. This results typically into a problem of high nonlinearity and strict mathematical continuity of real-valued variables does not even have to be given. Derivative information is not available or cannot be generated or approximated in sufficient accuracy and consistency. In addition to continuous-valued variables, we also have to handle integer-valued variables. The considered problem does not allow any relaxation techniques, owing to the black box dependency of the problem. Additionally, the presence of integer-valued variables may result in a changing dimension of the optimization problem variables and constraints.
Dynamic optimization problems as described e.g., in [37] are completely out of reach. Current state of the art optimization methods can only handle problems with continuous variables of low and medium dimensions. The current situation for the considered problems was summarized by Tim Kelly in a talk as follows:

> *Theory is still behind practice, and practice is still behind applications.*

In the review of existing optimization methods we found that surrogate methods are promising for mixed-integer black box-based optimization problems, if they can be extended by approved MINLP methods to guarantee a feasible solution for the underlying problem.

3 Mixed-Integer Optimization for Surrogate Problems

The motivation of this thesis is to develop an derivative free approach for finding sufficient appropriate solutions for noisy mixed-integer optimization problems containing expensive black box components. However, in this Chapter we will outline how MINLP can be integrated within a surrogate optimization approach, based on stochastic approximation to replace all black box-based components.

3.1 General Optimization Problem Formulation

First, the arising black box optimization problem has to be defined in a way that any available explicit information is included. In many approaches the original objective function is replaced by a surrogate. However, this procedure discards explicit information which could be utilized otherwise by the optimization method. As a general guideline surrogate functions should only replace parts of the problem that really depend on black box evaluations [13], and not also the explicit ones.

3.1.1 Definition of the Black Box Components

The black box that defines the main components of the optimization problem usually depends on different external and internal influences and conditions. Typically, the black box is given as a function $\boldsymbol{y}$ which depends on the state $\boldsymbol{x}$, and controllable as well as uncontrollable external influences summarized as $\boldsymbol{\omega}$. The system state is only used inside the black box and is typically obtained as the numerical solution of a system of ordinary or partial differential equations. Therefore, we will focus on the parts that are directly incorporated in the resulting optimization problem.

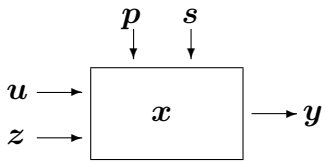
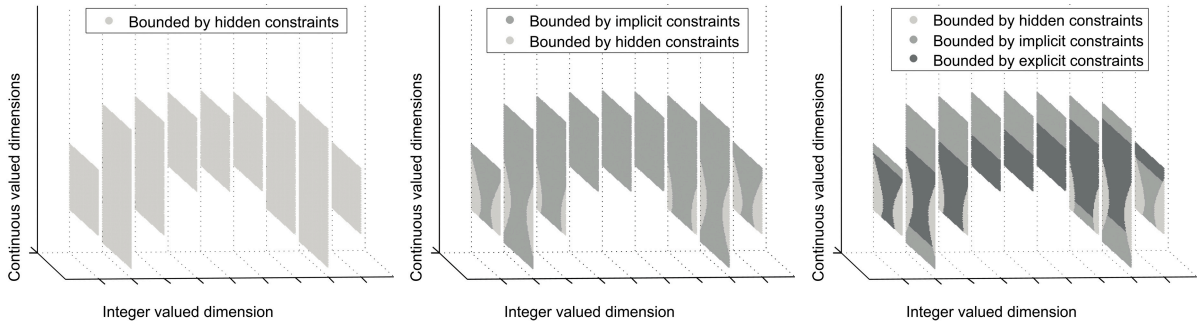


Figure 3.1: The relevant input $\boldsymbol{\omega}$, the black box state $\boldsymbol{x}$, and the black box output components $\boldsymbol{y}$.

The optimization variables, which will be of our special interest, are included in $\boldsymbol{\omega}$. Furthermore, $\boldsymbol{\omega}$ summarizes all external disturbances factors affecting the system. The system parameters are, in engineering problems, often system design parameters and become the optimization variables. However, all components of the initial system state, the controls, and other system influences can be selected as variables of the optimization problem. Thus, the relevant output arises as $\boldsymbol{y}(\boldsymbol{x}, \boldsymbol{\omega})$ (cf. Fig. 3.1).



(a) Variable domain bounded by hidden black box-based constraints.

(b) Variable domain bounded by implicit optimization problem constraints.

(c) Further bounding by explicit optimization problem constraints.

Figure 3.2: Visualization of a three-dimensional variable domain for an exemplary black box-based optimization problem.

Input Parameters Space

The influence on the system is summarized by $\boldsymbol{\omega}$ as proposed, for example, in [193]. In detail, $\boldsymbol{\omega}$ is a quadruple, $\boldsymbol{\omega} = (\boldsymbol{u}, \boldsymbol{p}, \boldsymbol{s}, \boldsymbol{z})$, and summarizes controls $\boldsymbol{u}$, continuous and discrete system parameters $\boldsymbol{p}$ and $\boldsymbol{s}$, and perturbations $\boldsymbol{z}$. Thus, a distinction between those components appears to be appropriate.

Controls The controls $\boldsymbol{u}$ given as input of the black box are time-dependent; this means $\boldsymbol{u}$ is a function that maps the time $t \in \mathbb{R}$ into the feasible domain of the controls. This infinite-dimensional variable typically ends up in a finite-dimensional one, if numerical methods are used. These are based on time discretization to identify optimal controls $\boldsymbol{u}^*$. The resulting large-scale optimization problem is solved usually by efficient gradient-based methods, e.g., [25, 273].

System Parameters In general, constant system parameters consist of a continuous-valued component $\boldsymbol{p} \in \mathbb{R}^{n_p}$, and an integer-valued one, $\boldsymbol{s} \in \mathbb{N}^{n_s}$. The feasible domains of $\boldsymbol{p}$ and $\boldsymbol{s}$ are constrained recursively by the output of the underlying black box, and are, therefore, not explicitly defined a priori. Several components of $\boldsymbol{p}$ and $\boldsymbol{s}$ can also be used to describe components of the initial state $\boldsymbol{x}$. The parameters $\boldsymbol{p}$ and $\boldsymbol{s}$ will be used as the optimization variables for the black box-based problems in this thesis.

Random System Perturbations Finally we divide those factors influencing the system that we are able to control, from those we are not. These time-dependent random factors are denoted by $\boldsymbol{z}$ and called perturbations. For the water resources management example discussed in Chapter 6, $\boldsymbol{z}$ could be used e.g., to describe rain fall as a kind of perturbation of hydraulic heads. Stochastic optimization methods are tailored for problems with random variables. For the following, we assume a fixed $\boldsymbol{z}$ for all black box evaluations during an optimization run.

Output State Space

The system state is described by $\boldsymbol{x}$, but not all components of $\boldsymbol{x}$ have a direct impact on the relevant black box output $\boldsymbol{y}$, which is included into the arising optimization problem.

Relevant Black Box Output The relevant output is collected by $\boldsymbol{y} \in \mathbb{R}^{n_y}$. Because we have only considered changes for the system parameters $\boldsymbol{p}$ and $\boldsymbol{s}$, $\boldsymbol{y}$ is a function of these two. For example, if we consider water resources management applications (Chapter 6), not all hydraulic heads in each knot need to be considered for the optimization problem in each time step of a numerical simulation. Just a few well chosen monitoring positions are incorporated. It is another task to select monitoring positions out of all system state information provided by $\boldsymbol{x}$ over the whole simulation time horizon. All further impacts on the black box output such as numerical noise, discontinuities, etc., are included by aggregating the relevant output information of the black box in $\boldsymbol{y}$.

Resulting Restrictions for the Input Space General optimization problem constraints on variables and controls have to be handled as well as state constraints. These constraints are formulated explicitly during the system definition, or later by formulating the optimization problem constraints accordingly. However, if we deal with optimization problems including black box components, three kinds of constraints emerge, the implicit and the explicit ones, and the hidden ones [40], which are a priori given by appropriate constraint functions depending on $\boldsymbol{p}$, $\boldsymbol{s}$, or $\boldsymbol{y}$. The first two will become part of the optimization problem by an appropriate modeling.
If hidden constraints are violated, the black box fails to return reasonable - or any - output. This means that a system state $\boldsymbol{x}$ and a there upon based $\boldsymbol{y}$ cannot be determined. The output of those simulation runs cannot be used quantitatively by an optimization approach, but only as qualitative information, so that a parameter set, $\boldsymbol{p}$ and $\boldsymbol{s}$, leads to an infeasible system state $\boldsymbol{x}$, respectively $\boldsymbol{y}$. In the case that no flags are set for unreasonable black box output, the evaluated $\boldsymbol{y}$ would be used as confident black box output. This would drive any process of information extraction or system adjustment into a wrong direction.
These considerations lead directly to the domain $\overline{\Omega}$ for the variables $\boldsymbol{p}$ and $\boldsymbol{s}$ that result in reasonable black box output $\boldsymbol{y}$,

$$\overline{\Omega}_{\boldsymbol{y}} = \bigcap_{i=1}^{n_y} \overline{\Omega}_{y_i}, \text{ with } \overline{\Omega}_{y_i} := \{(\boldsymbol{p}, \boldsymbol{s}) \in \mathbb{R}^{n_p} \times \mathbb{N}^{n_s} \mid y_i(\boldsymbol{p}, \boldsymbol{s}) \in \mathbb{R}^{n_y},\ y_i \text{ well-defined}\}\,.$$

As mentioned before, the different $\overline{\Omega}_{y_i}$ are a priori not known and cannot be defined without running the black box to evaluate $\boldsymbol{y}$ for the different values of $\boldsymbol{p}$ and $\boldsymbol{s}$. The definition of $\overline{\Omega}_{\boldsymbol{y}}$ as an intersection becomes relevant when a tuple $(\boldsymbol{p}, \boldsymbol{s}) \in \overline{\Omega}_{y_i}$, is evaluated and not also an element of $\overline{\Omega}_{\boldsymbol{y}}$.
Constraints on $\boldsymbol{p}$ and $\boldsymbol{s}$ that are not modeled using black box output, are not part of these constraints. They are taken directly into account as explicitly given constraints during the modeling of the optimization problem.

3.1.2 Modeling of the Optimization Problem

Solving the right problem - this should be the question during modeling of the optimization problem. If the use of a special optimization method is favored, it will influence the way of modeling the problem fundamentally. Besides, the way to formulate problem constraints can be very different. In Chapter 7.3, stability of a performed walking motion is included, just as an implicit constraint, by stopping the measurement of the objective function when the robot falls. An alternative would be to incorporate sensors detecting the stability and using this information to set up constraints explicitly.
W.l.o.g. we assume that all optimization variables are already included in the input parameter space of the black box by $\boldsymbol{p}$ and $\boldsymbol{s}$. Thus, a few parameters might be without influence on the black box. Even by assuming that all optimization variables are included in the feasible black box input space $\overline{\Omega}_{\boldsymbol{y}}$, we have to deal with different feasible domains for the variables of the optimization problem. A visualization of the different domains is given by Figure 3.2.

Problem Constraints and the Objective Function

We take the optimization variables directly from the feasible black box domain $\overline{\Omega}_{\boldsymbol{y}}$, namely $\boldsymbol{p}$ and $\boldsymbol{s}$. Furthermore, a distinction between explicitly given constraints depending only on $\boldsymbol{p}$ and $\boldsymbol{s}$, and implicitly given ones, depending on $\boldsymbol{y}$, has to be done. This is made analog to the problem definition presented in [13, 36, 64]. The first explicit ones are the bound constraints,

$$\boldsymbol{p}_{lb} \leq \boldsymbol{p} \leq \boldsymbol{p}_{ub},\ \boldsymbol{s}_{lb} \leq \boldsymbol{s} \leq \boldsymbol{s}_{ub},\ \text{with } \boldsymbol{p}_{lb}, \boldsymbol{p}_{ub} \in \mathbb{R}^{n_p},\ s_{lb}, \boldsymbol{s}_{ub} \in \mathbb{N}^{n_s}. \tag{3.1}$$

Here, $\cdot_{lb}$ indicates lower bounds and $\cdot_{up}$, respectively upper bounds. The linear constraints are given by

$$\boldsymbol{A} \begin{pmatrix} \boldsymbol{p} \\ \boldsymbol{s} \end{pmatrix} - \boldsymbol{b} \leq 0,\ \text{with } \boldsymbol{b} \in \mathbb{R}^{n_b},\ \boldsymbol{A} \in \mathbb{R}^{n_b \times (n_p + n_s)}. \tag{3.2}$$

Furthermore, we have to include the explicitly given nonlinear constraints by

$$g_i(\boldsymbol{p}, \boldsymbol{s}) \leq 0,\ \text{with } i \in I_{ex}. \tag{3.3}$$

The feasible domain for the optimization problem also depends on the relevant black box output $\boldsymbol{y}$. Thus, besides the explicit constraints, we have to handle the implicit ones, given by

$$g_i(\boldsymbol{y}, \boldsymbol{p}, \boldsymbol{s}) \leq 0,\ \text{with } i \in I_{im}, \tag{3.4}$$

where I_{im} and I_{ex} are the index sets numbering all nonlinear constraints. The dimension of the nonlinear constraints is given by the sum of the cardinal number of both index sets. The distinction between implicit and explicit constraints is used to derive the surrogate problem and was proposed for nonlinear problems in [234].
We summarize the box constraints in (3.1), the linear constraints in (3.2), and the nonlinear constraints from (3.3) and (3.4) by $\boldsymbol{g}$, a vector of n_g one-dimensional functions with

$$\boldsymbol{g} : \overline{\Omega}_{\boldsymbol{g}} \subseteq \{\mathbb{R}^{n_y} \times \mathbb{R}^{n_p} \times \mathbb{R}^{n_s}\} \rightarrow \mathbb{R}^{n_g},$$

which defines the feasible domain $\Omega_{\boldsymbol{g}}$ of the optimization problem,

$$\Omega_{\boldsymbol{g}} := \left\{(\boldsymbol{p}, \boldsymbol{s}) \in \overline{\Omega} \mid (\boldsymbol{y}(\boldsymbol{p}, \boldsymbol{s}), \boldsymbol{p}, \boldsymbol{s}) \in \overline{\Omega}_{\boldsymbol{g}},\ \boldsymbol{g}(\boldsymbol{y}, \boldsymbol{p}, \boldsymbol{s}) \leq 0\right\}. \tag{3.5}$$

This provides the last elements for a short formulation of the optimization problem by an objective function f,

$$\min_{(\boldsymbol{p},\boldsymbol{s})\in\Omega_{\boldsymbol{g}}} f(\boldsymbol{y}, \boldsymbol{p}, \boldsymbol{s}), \tag{3.6}$$

where $f$is defined on $\overline{\Omega}_f$, with

$$\overline{\Omega}_f \subseteq \left\{\mathbb{R}^{n_g} \times \mathbb{R}^{n_p} \times \mathbb{N}^{n_s}\right\} \to \mathbb{R}.$$

An optimal candidate $(\boldsymbol{p}^*, \boldsymbol{s}^*)$, with $\boldsymbol{y}^* = \boldsymbol{y}(\boldsymbol{p}, \boldsymbol{s})$ holds

$$f(\boldsymbol{y}^*, \boldsymbol{p}^*, \boldsymbol{s}^*) \leq f(\boldsymbol{y}, \boldsymbol{p}, \boldsymbol{s}), \quad \forall\, (\boldsymbol{p}, \boldsymbol{s}) \in \Omega_{\boldsymbol{g}}.$$

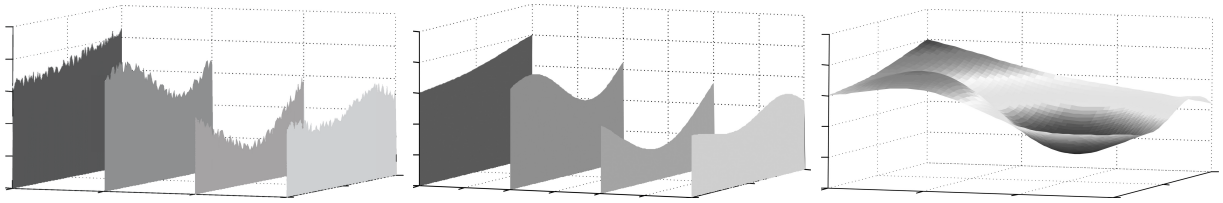
W.l.o.g. only inequality constraints are included into this formulation. Each equality constraint could be replaced by using two inequality constraints. Of course, this results in a higher dimension for the constraints. However, for the use of surrogate optimization such a formulation is more suitable as also discussed in [233].

Changing Dimension of Variables and Constraints

Up to here, the total dimension has always been taken as a constant during the optimization problem formulation. However, especially in engineering problems, where system design optimization is often favored, the integer-valued variables can express structural information. This typically has an influence on the optimization problem in terms of changing of the constraints given by $\boldsymbol{g}$ and added or removed components of the optimization variables $\boldsymbol{p}$ and $\boldsymbol{s}$. Considering, again, a water resources management problem, no pumping rate has to be set for a removed well.
Modeling challenges arise if the, initially assumed constant, dimensions of the variables and constraints given by n_p, n_s, and n_g, are implicitly controlled by some of the integer-valued variables. If structural variables are modeled by binaries variables, they can be used as on/off switches for those variables, parts of the objective function, and constraints on which they have impact. Structural changes through variations of the structural variables are covered this way. The price is an increasing problem dimension because of additional decision variables of all kinds. An overview on methods to formulate appropriate constraints is given in [22]. An example for such a proceeding is given in Chapter 6.

3.2 Surrogate Functions for Mixed-Integer Black Box Type Problems

The key feature of surrogate optimization approaches is that the selected optimization method is not running directly in a loop with the black box, whose output is part of the underlying optimization problem, but is applied to a problem formulation where the black box is replaced by an intermediate surrogate approximation.



(a) Original simulation-based function optimization surface. (b) Smoothed surface for the original variable domain. (c) Surface additionally relaxing the integer-valued domain.

Figure 3.3: Surrogate function for non-relaxable mixed-integer problems.

3.2.1 Extension of the Optimization Variables' Domain

The variables $\boldsymbol{p}$ and $\boldsymbol{s}$ have to be out of $\overline{\Omega}_{\boldsymbol{y}}$ for any evaluation of the optimization problem's underlying black box. The user just receives an error back for the evaluation of an infeasible tuple $(\boldsymbol{p}, \boldsymbol{s})$. This problem is overcome by defining the surrogate problem on an extended variable domain.

Relaxing the Integrality Conditions

The first restriction on $\overline{\Omega}_{\boldsymbol{y}}$ is the integrality constraint affecting $\boldsymbol{s}$. This impedes a relaxed problem formulation which is the first step to applying a MINLP method according to [39]. Hence, $\overline{\Omega}_{\boldsymbol{y}}$ is extended to its continuous relaxation,

$$\widehat{\Omega}_{relax} = \left\{ (\boldsymbol{p}, \boldsymbol{s}) \in \mathbb{R}^{n_p} \times \mathbb{R}^{n_s} \mid (\boldsymbol{p}, \lfloor \boldsymbol{s} \rfloor), (\boldsymbol{p}, \lceil \boldsymbol{s} \rceil) \in \overline{\Omega}_{\boldsymbol{y}} \right\},$$

where

$$\begin{aligned} \lfloor \boldsymbol{s} \rfloor &:= (..., \lfloor s_i \rfloor, ...)^T,\ i = 1, ..., n_s, \quad \lfloor s_i \rfloor := \max \left\{ s_i \in \mathbb{N} \mid \lfloor s_i \rfloor \leq s_i \right\}, \\ \lceil \boldsymbol{s} \rceil &:= (..., \lceil s_i \rceil, ...)^T,\ i = 1, ..., n_s, \quad \lceil s_i \rceil := \min \left\{ s_i \in \mathbb{N} \mid \lceil s_i \rceil \geq s_i \right\}. \end{aligned}$$

The underlying black box does not provide any output due to a non-integer-valued $\hat{\boldsymbol{s}}$. However, we assume that a surrogate function which is defined on the relaxed domain $\widehat{\Omega}_{relax}$, might capture the main effects of the underlying black box for the response $\boldsymbol{y}$ (cf. Fig. 3.3).

Extending the Feasible Domain

A further extension besides relaxing $\overline{\Omega}_{\boldsymbol{y}}$ from the integrality constraints by introducing $\widehat{\Omega}_{relax}$ seems reasonable. Hidden constraints may create holes or other awkward boundaries for $\overline{\Omega}_{\boldsymbol{y}}$. For those cases, the use of extended domain would ease the use of iterative optimization methods. Such a domain $\widehat{\Omega}$ has at least to be a superset of $\widehat{\Omega}_{relax}$, but may also be without any restriction,

$$\overline{\Omega}_{\boldsymbol{y}} \subseteq \widehat{\Omega}_{relax} \subseteq \widehat{\Omega}_{\boldsymbol{y}} \subseteq \mathbb{R}^{n_p + n_s}$$

Typically, numerical optimization methods do not match the constraints during all iterations, so that candidates $(\boldsymbol{p}, \boldsymbol{s})$ not in $\overline{\Omega}_{\boldsymbol{y}}$ might have to be evaluated. This would lead

to program aborts for the original problem. If we imagine a $\overline{\Omega}_{\boldsymbol{y}}$ relaxed from integrality constraints and local undefined areas, then, metaphorically speaking, we are able to walk from one tuple $(\boldsymbol{p}, \boldsymbol{s})$ to another without leaving the domain.
A new integer-valued candidate $\boldsymbol{s}$ will be provided using MINLP methods. How the implicit constraints are held becomes important for the case where $\overline{\Omega}_{\boldsymbol{y}}$ is a real subsets of the feasible domain of the surrogate problem.

Modeling of the Surrogate Problem

To get a complete, explicitly given optimization problem we have to replace the implicitly given parts of the original problem (3.6). The black box output $\boldsymbol{y}$ is replaced by a surrogate function output $\hat{\boldsymbol{y}}$, where $\hat{\boldsymbol{y}}$ is defined on $\widehat{\Omega}_{\boldsymbol{y}}$. All other components of the original problem are already defined explicitly, and can be included directly into the surrogate problem. The surrogate optimization problem follows as

$$\min_{(\boldsymbol{p},\boldsymbol{s})\in\Omega_{\hat{\boldsymbol{g}}}} f(\hat{\boldsymbol{y}}, \boldsymbol{p}, \boldsymbol{s}), \tag{3.7}$$

where

$$\Omega_{\hat{\boldsymbol{g}}} := \left\{ (\boldsymbol{p}, \boldsymbol{s}) \in \widehat{\Omega}_{\boldsymbol{y}} \mid (\hat{\boldsymbol{y}}(\boldsymbol{p}, \boldsymbol{s}), \boldsymbol{p}, \boldsymbol{s}) \in \widehat{\Omega}_{\boldsymbol{g}},\ \boldsymbol{g}(\hat{\boldsymbol{y}}, \boldsymbol{p}, \boldsymbol{s}) \leq 0 \right\}. \tag{3.8}$$

Of course, a candidate $(\boldsymbol{p}, \boldsymbol{s})$ has to be out of the $\Omega_{\boldsymbol{g}}$ to be feasible for the original problem, but during the optimization process on the surrogate problem errors are excluded resulting from missing black box output for $\boldsymbol{y}$
We can assume, as a special case, that the objective function $f$and the implicit constraints g_i, with $i \in I_{im}$, are equal to the black box output $\boldsymbol{y}$. This simplification is often used in literature to formulate the surrogate problem. Ignoring the implicit components and all other constraints apart from the box constraints would lead to the original formulation of Sacks et. al. [231]. Hence, the explicit components of the optimization problem are included in the underlying black box and the provided information is ignored.

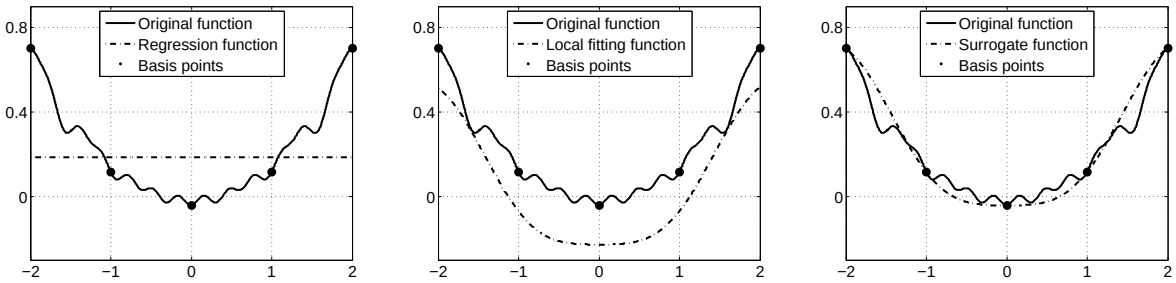
3.2.2 Generating Surrogate Functions by Stochastic Approximations

The optimization approach we use is the statistical method DACE as mentioned in Chapter 2. The standard approach designed for deterministic black box optimization problems described by Sacks et. al. [231] is widely used for surrogate-based optimization methods. A benefit of DACE is that it is not bounded to a special kind of initial set of basis points. This makes DACE easy applicable to existing sets of black box output data.

DACE for Relaxed Mixed-Integer Problems

We use an extension for DACE to handle not only the continuous-valued $\boldsymbol{p}$, but also the integer-valued $\boldsymbol{s}$ in such a way that the black box output $\boldsymbol{y}$ is approximated [126]. An underlying stationary Gaussian stochastic process with a constant mean is assumed to provide a good surrogate $\hat{\boldsymbol{y}}$ for the black box. W.l.o.g. we assume a one-dimensional y to describe DACE. For a given set of system designs $\mathcal{B} = \{(\boldsymbol{p}_i, \boldsymbol{s}_i)\}_{i=1,\ldots,n}$ and black box outputs $y(\boldsymbol{p}_i, \boldsymbol{s}_i)$ the surrogate function $\hat{y}$ to be built by DACE has to meet

$$y(\boldsymbol{p}_i, \boldsymbol{s}_i) = \hat{y}(\boldsymbol{p}_i, \boldsymbol{s}_i), \quad \forall \ \ (\boldsymbol{p}_i, \boldsymbol{s}_i) \in \mathcal{B}.$$



(a) Regression by a constant basis function.

(b) Stochastic process part.

(c) Complete approximation.

Figure 3.4: Decomposition of a DACE approximation.

The main effects of y should be covered by $\hat{y}$, but it is obvious that $\hat{y}$ is exact only for $(\boldsymbol{p}, \boldsymbol{s}) \in \mathcal{B}$. It results

$$\hat{y} = \boldsymbol{h}(\boldsymbol{p}, \boldsymbol{s})\boldsymbol{\beta}^T + Z(\boldsymbol{p}, \boldsymbol{s}). \tag{3.9}$$

The first and more global part on the right side in (3.9) approximates the global trend of the unknown function y by a linear combination of a scalar vector $\beta \in \mathbb{R}^k$, with k known fixed functions depending on $\boldsymbol{p}$ and $\boldsymbol{s}$, summarized by the k-dimensional $\boldsymbol{h}$ (cf. Fig. 3.4 a). The second part Z guarantees that $\hat{y}$ fits to y for all elements of $\mathcal{B}$ (cf. Fig. 3.4 b and c). This "lack of fit" part is assumed to be a realization of a stationary Gaussian random function with a mean of zero, $E[Z(p)] = 0$, and a covariance,

$$Cov[Z(\boldsymbol{p}_i, \boldsymbol{s}_i), Z(\boldsymbol{p}_j, \boldsymbol{s}_j)] = \sigma_Z^2 \boldsymbol{R}((\boldsymbol{p}_i, \boldsymbol{s}_i), (\boldsymbol{p}_j, \boldsymbol{s}_j)),$$

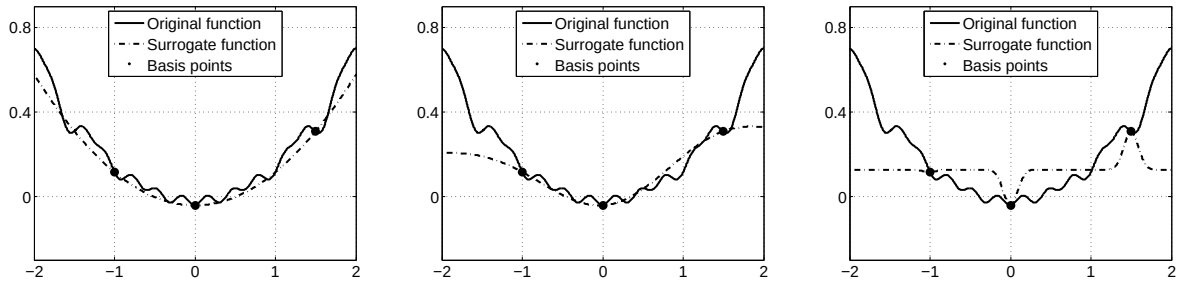
with $i \neq j$, and $i, j \in \{1, ..., n\}$. The smoothness and property of differentiability are controlled by the chosen correlation function R, here as the product of one-dimensional correlation functions,

$$R\left(\begin{pmatrix} \boldsymbol{p}_l \\ \boldsymbol{s}_l \end{pmatrix}, \begin{pmatrix} \boldsymbol{p}_k \\ \boldsymbol{s}_k \end{pmatrix}\right) = \prod_{j=1}^{(n_p+n_s)} e^{-\theta_j d_j^2},$$

with

$$d_j = \begin{cases} |p_{l,j} - p_{k,j}| & \text{for } j = 1, ..., n_p \\ |s_{l,i} - s_{k,i}| & \text{for } j - n_p = i = 1, ..., n_s \end{cases}$$

More theoretical details and assumptions can be found in [158]. However, issues, such as stability of the covariance matrix [143, 149], or the increased computational effort needed when $\mathcal{B}$ contains a larger number of basis points, have to be considered for practical applications. However, Srivastava et. al. [256] pointed out that even if the fitting of a DACE surrogate takes some computational time, compared to the cost of evaluation of the underlying black box problems, this property can be disregarded.
In a direct comparison with RSM, which assumes a white noise for regression, DACE enables the use of correlation information, additionally. It also becomes more accurate with an increasing number of basis points. However, close to the boundary, the DACE approximation is like many other methods just an extrapolation, which makes it less ac-



(a) $\theta = 0.01$ (b) $\theta = 1$ (c) $\theta = 100$

Figure 3.5: Influence of the correlation parameter θ on the DACE approximation, motivated by [158].

curate. Then, it has to be pointed out that DACE does not provide the possibility to identify spikes, large discontinuities, or gaps for black boxes apart from regions with a smooth output y. These phenomena are often the result if characteristics resulting from integer-valued or binary-valued variables are neglected during the optimization problem modeling [93]. Now we can address the integer-valued variables directly.

Parameter Estimation

The free regression parameter vector $\boldsymbol{\beta}$, the process variance σ_Z, and the correlation parameter θ have to be adjusted. Again, we follow the original idea of Sacks and estimate the DACE parameters by a maximum likelihood approach most consistent to the present black box values y which $\hat{y}$ have to match (cf. Fig. 3.5).
The optimal $\boldsymbol{\beta}$ solves the regression problem and σ_Z depends on the correlation [178]. By appropriate setting of $\boldsymbol{\theta}$, as the only unknown component of R, $\boldsymbol{\beta}$ and σ_Z are defined. If a Gaussian process is assumed, the optimal θ is given as the maximizer of the likelihood function.

Basis Functions for Regression

Different approaches can be considered for the global regression part of the surrogate function generated by DACE as described in [159, 188]. Universal Kriging is predicting the mean of the underlying function as a realization of a stochastic process by a linear combination of k known basis functions, summarized by $\boldsymbol{h}$. Detrended Kriging applies a classical linear regression model. Sacks proposed ordinary Kriging that we will use here. A constant mean is assumed on the entire domain with $k = 1$, one constant h, and a scalar β.
The use of different basis functions for the surrogate $\hat{y}$ should not be mixed up with the integration with explicitly given components of the original problem. However, the choice for certain basis functions in universal Kriging approaches could be motivated by prior knowledge of typical characteristics of the underlying black box output y. This would of course improve the precision of the approximation.
A further modification would be to use different basis regression functions depending on the number of basis points that are used for approximation. If a inadequate number of basis

points is available for quadratic basis functions, only linear basis functions are used. The basis functions can be chosen to the best advantage during each iteration independently.

3.3 Mixed-Integer Nonlinear Programming for Surrogate Problems

The use of efficient Newton type methods for optimization is now enabled since the complete analytical, explicitly defined surrogate problem is introduced. In this section we will only briefly discuss the two main components many common MINLP codes are based on. The first is to branch and to bound the relaxed integer-valued domain of the problem, and the second is the use of NLP techniques to solve arising subproblems. In [8] branch and bound methods are also recommend for problems where continuous relaxation of discrete variables is possible.
We would like to mention that the user is not restricted to the methods suggested in this section. Any optimization methods introduced in Chapter 2 is applicable, if it provides feasible candidates for the underlying problem. Typically, gradient-based methods can provide a faster convergence than other methods, which results in lower computational costs. For the following considered applications and scenarios the costs can be ignored, but when we are thinking about online methods even the resources used for the optimization on the surrogate problem become important. The more important reason is that Newton type methods are naturally enabled for use with problems including nonlinear constraints. Since the surrogate problem is completely explicitly given, we can make use of gradient information.

3.3.1 Branch and Bound

Also, the use of decomposition methods developed for MINLP, like branch and bound [92, 118], is now enabled. As introduced, MINLP methods cannot be applied to the original objective function because a relaxed MINLP problem can not be defined for the underlying black box. However, the surrogate problem is defined on the relaxed domain of $\boldsymbol{s}$, and as this is the most common and effective method as proposed in the previous chapter, it is our method of choice.
Beside general branching strategies, problem specific strategies can also make sense. As it is shown by [74], the use of prior problem information can accelerate the tree search. A lot of useable additional information is provided by the surrogate problem. Firstly, the correlation parameter $\boldsymbol{\theta}$ can be taken to indicate which variables to branch first. This information is of course only available for the implicitly given components. The explicit ones are not replaced by surrogates. Results from techniques as screening or ANOVA provide additional sensitivity information of the surrogate problem [31]. The branch and bound approach is not only useful to guarantee a candidate with an integer-valued s. With almost no modifications it is a tool for global optimization according to continuous-valued variables. Thereupon, the resulting bunch of MINLP subproblems is left to be solved.

3.3.2 Sequential Quadratic Programming

We suggest SQP to solve the resulting subproblems. For instance any NLP method could also be applied to these subproblems, but in our approach, explicitly given nonlinear constraints and their gradients are easy to incorporate apart from the linear and box constraints of the considered original optimization problem. Furthermore, the noise induced by numerical simulation into the original objective function is already smoothed out, and thereby, these subproblems can be solved efficiently by SQP methods [104]. Additionally, it is a highly efficient optimization method for NLP problems, which include nonlinear constraints problems. In our case, even only small scale NLP problems arise so that the main advantages is not as obvious as it would be for large-scale problems.
The idea behind SQP is to solve a NLP problem by using a sequence of quadratic programming subproblems. The search direction of each subproblem fulfills the necessary condition for an optimum of the quadratic subproblem. It is built from the quadratic Taylor approximation of the objective function plus its Hessian, weighted by the Lagrange multipliers of the active constraints. The latter components include the information on curvature of the active constraints. The quadratic problem is solved under linear equality constraints representing Taylor approximations of the set of active constraints. For more details we would like to refer to [106] and the very efficient implementation SNOPT [105]. Some process parameters for SQP have to be chosen by the user. An important factor according to the performance is the starting point. Three different strategies were tested. We used the best known feasible candidate, the last evaluated feasible candidate, or the last evaluated candidate. If the SQP algorithm aborts or stops without a solution, we restart the optimization process with the next best found, namely the previous evaluated feasible or just evaluated candidate. A superior strategy, out of these, for the choice of a starting point was not identified.

3.4 Summary of Chapter 3

The approximation quality of the surrogate functions is the linchpin of the derived approach and makes surrogate optimization a heuristic method. The fitting quality in the regions of interest is a priori not provable and can only be tested with additional effort by black box evaluations. However, the derived extension now enables one to address mixed-integer nonlinear black box-based problems. The basic elements for solving surrogate problems were introduced in this chapter. How the information is obtained to build the surrogates, and how the algorithmic framework of the optimization procedure looks like, will be discussed in the next Chapter.

4 Sequential Optimization for Surrogate Problems

Underlying black boxes are part of the objective function and the constraints of the original problem statement. Knowledge from these is only obtained by evaluating the black box for certain parameter settings $\boldsymbol{p}$ and $\boldsymbol{s}$. In the previous chapter we assumed that the information to build the surrogate function approximation is given. However, the choice of candidates $(\boldsymbol{p}, \boldsymbol{s})$ is not a trivial question.
DOE is well-known for real-world experiments [33]. The difference to our situation is that we do not have to determine all elements of an initial base $\mathcal{B}$ for evaluation a priori. A priori information can be used to determine new candidates $(\boldsymbol{p}, \boldsymbol{s})$ which leads to an adaptive extension of the prior bases. Such an approach matches better to the characteristics of black box-based optimization problems, because evaluations are typically undertaken separately for each candidate $(\boldsymbol{p}, \boldsymbol{s})$.
Often, because of limited available computational resources the immediate evaluation of a space-filling set of candidates is not possible. A more efficient sequential optimization procedure is suggested. During the first step of a sequential surrogate optimization procedure, initial points are used to build the first surrogate problem; in our case, a surrogate for the black box response $\boldsymbol{y}$.

4.1 Selection of Initial Design Points

Building surrogate functions to approximate the black box output is mostly done for two reasons. The first is to use surrogates for visualization of systems or function responses as mentioned in [33]. Using a surrogate for plots is much cheaper than evaluating the black box on a complete grid. This would be done first to get an impression of the overall system characteristics. Therefore, a space-filling design, which covers almost the complete feasible domain, would be favored.
Visualization is definitely an essential tool to get a better understanding of optimization problems. However, surrogate functions are used in this approach for the optimization of problems including black box parts for the same reason of low computational costs. However, candidates for evaluation which build the basis $\mathcal{B}$ for a surrogate are not chosen for visualization, but for the purpose of an efficient optimization procedure. Response surfaces are only fitted in regions of $\Omega_{\boldsymbol{g}}$ where an optimal candidate $(\boldsymbol{p}^*, \boldsymbol{s}^*)$ is expected to be located. If the number of possible simulation runs to find a good solution for an optimization problem is particularly small, all efforts should be spent on optimization and not for generating an appropriate basis for visualization purposes. However, after a completed optimization run, many data from black box responses is available to generate global or local response surfaces to visualize system characteristics, or the objective function and the implicit constraints of the optimization problem.

During the whole selection process of new candidates the following (as pointed out in [33]) should be kept in mind:

> *Don't forget nonstatistical knowledge - When you are doing statistics do not neglect what you and your colleagues know about the subject matter field. Statistical techniques are useless unless combined with appropriate subject matter knowledge and experience. They are an adjunct to, not a replacement for, subject matter expertise.*

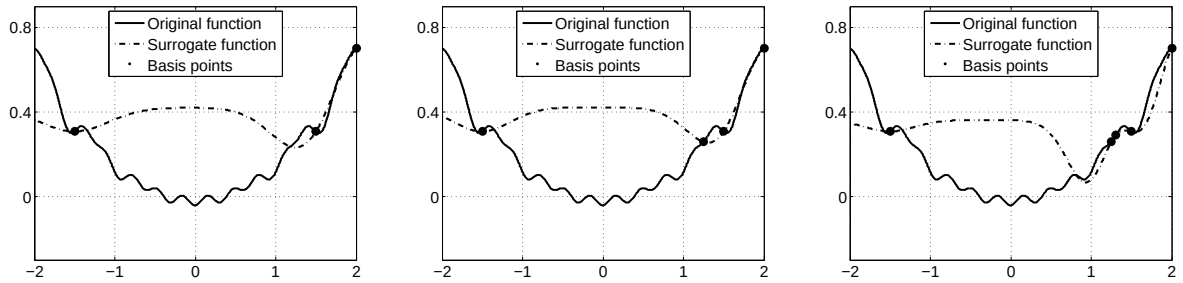
First a small initial set of system designs $\mathcal{B}_0 = \{(\boldsymbol{p}^{(l)}, \boldsymbol{s}^{(l)})\}_{l=1,\dots,k}$ is selected and evaluated as initial base of the first iteration of the surrogate problem. We assume that all elements of $\mathcal{B}_0$ and of all following extended bases provide a black box output for each component of $\boldsymbol{y}$. Even if they do not, the only difference is that the surrogate functions for each component of $\hat{\boldsymbol{y}}$ are built on different bases according to the definition of $\overline{\Omega}_{y_i}$. New response information is obtained as long as just one component of $\boldsymbol{y}$ is provided, and an updated surrogate problem is generated which uses this information.

4.1.1 Candidate Selection without Black Box Responses

The typical method to find initial candidates building $\mathcal{B}_0$ is to use DOE methods. A good overview is given in [158]. Beyond this, is must be decided how many candidates should be evaluated before the black box response is incorporated for the selection of further candidates, especially if parallel computing infrastructure is available.
However, standard approaches usually do not consider that besides components of the objective function, also constraints are partially implicitly given. DOE methods are usually not even designed for handling explicit linear and nonlinear constraints. It follows that by a classical DOE method, not even one chosen candidate has to be feasible. Thus, computational costs would arise for evaluating candidates $(\boldsymbol{p}, \boldsymbol{s})$, which do not even eventually provide a black box output.
Of special interest in this context are DOE methods which do not build a base $\mathcal{B}_0$ instantaneously, like Latin hypercube designs which build complete bases $\mathcal{B}_0$. Some classical DOE methods are able to update earlier initial bases with new elements, or generate further elements for $\mathcal{B}_0$ based on already given points, independently from any optimization approach. These methods evaluate additional elements of $\mathcal{B}_0$ based on an a priori defined scheme. The only information to generate new elements are the given elements of $\mathcal{B}_0$. This is the main difference to sequential update methods in surrogate optimization; produced black box output information $\boldsymbol{y}$ is not used. This property can also become a feature when investigating problems with hidden constraints. Typically, in a situation where, after the evaluation of a somehow defined initial $\mathcal{B}_0$, almost no simulation output is returned or no feasible point is found, optimization on a surrogate problem may not provide any reasonable result. In such a case a DOE that is not depending on response information can help to explore the domain for feasible candidates to extend $\mathcal{B}_0$. Even the fact that no output is provided is useful information.

4.1.2 Using Expert's Guesses and Available Information

If expert knowledge is available like a given set of feasible candidates for $(\boldsymbol{p}, \boldsymbol{s})$, it should be included as initial information to start the optimization procedure. $\mathcal{B}_0$ can be generated



(a) First iteration with surrogate and found minimizer.

(b) Second iteration surrogate.

(c) Third iteration with updated surrogate.

Figure 4.1: Visualization of the idea of the strawman method.

around the known feasible candidate by varying each single dimension p_i and s_i on its own, as known from [35, 263]. Of course, each $\boldsymbol{s}$ has to be rounded at least to the next integer for a feasible $\mathcal{B}_0$.
Results from earlier optimization runs can also help to reduce the computational cost for getting all information needed for an initial $\mathcal{B}_0$. This would be the starting situation if such a surrogate optimization approach is applied as the subsequent component in a hybrid optimization approach. With an initial surrogate function $\hat{\boldsymbol{y}}^{(0)}$ based on $\mathcal{B}_0$, the initial phase is closed.

4.2 A Distance-Based Update Criteria

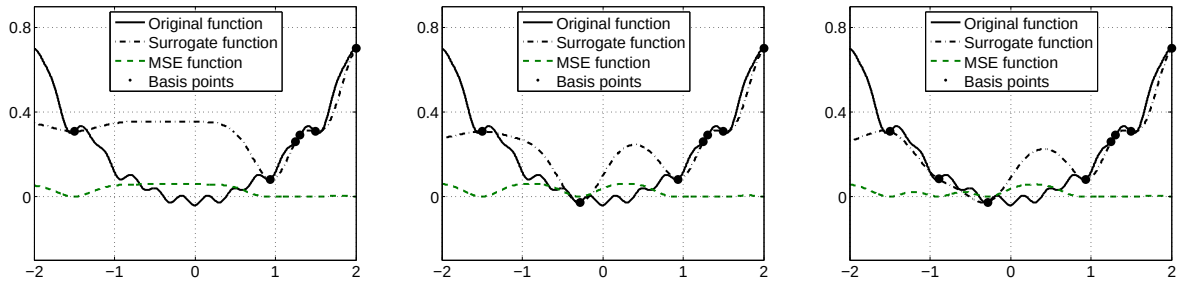
The characteristics of computer experiments are utilized after all available information on the underlying black box is included. Each new candidate is chosen using all prior available information, especially obtained from earlier evaluations of the original underlying black box problem. Torczon and Trosset pointed out in [266] that:

> *There is merit in balancing the goal of finding minimizers of the current approximation against the goal of constructing better approximations for future iterations.*

The sequential update starts by adding further points obtained from searching for candidates with certain properties on the surrogate problems during each iteration. After each iteration a new surrogate is built using the extended basis of responses. Different sequential update strategies were already discussed in [233].

4.2.1 Local Update, Strawman Approach

The iteration is triggered by searching for a minimizer of the initial surrogate problem $(\boldsymbol{p}^{*0}, \boldsymbol{s}^{*0})$ and by adding it as the next candidate for the evaluation of the underlying black box. This point is added to $\mathcal{B}_0$ for an updated basis $\mathcal{B}_1$ of a new surrogate $\hat{\boldsymbol{y}}^{(1)}$ (cf. Fig. 4.1). This is repeated in each iteration i to build new surrogate problems by extending the previous basis $\mathcal{B}_{i-1}$. If candidates evaluated do not provide a response for



(a) Minimizer of the surrogate is rejected and for the first time the MSE is evaluated.

(b) Surrogate function with the first added MSE-maximizer.

(c) Surrogate function of the following iteration with a second added MSE maximizer.

Figure 4.2: Including the MSE into the update criteria.

each component of $\boldsymbol{y}$, the components of the surrogate $\hat{y}_j^{(i)}$ are defined on different bases ${}^j\mathcal{B}_i$, with ${}^j\mathcal{B}_i \subset \overline{\Omega}_{y_i}$.

4.2.2 Mean Squared Error as Uncertainty Predictor

The MSE provides a measure to predict the approximation quality of $\hat{\boldsymbol{y}}$ and is given as the difference between the output of the black box to be approximated, and its approximation. This valuable information is a byproduct of using DACE to determine a surrogate as introduced in Chapter 3. Using the notation proposed by [233], the MSE for a component y_i of the surrogate is given as

$$MSE\left[\hat{y}_i(\boldsymbol{p},\boldsymbol{s})\right] = \sigma^2(\boldsymbol{p},\boldsymbol{s}) = \sigma_Z^2\left(1 - \boldsymbol{r}_{(p,s)}^T R^{-1} \boldsymbol{r}_{(p,s)}\right), \tag{4.1}$$

where σ^2 describes the Kriging variance of the surrogate, which can be evaluated once the regression factor β and the process variance σ_Z^2 are already determined [237]. The remaining elements in Equation (4.1) are the matrix $\boldsymbol{R}$ describing the correlation between all candidates of $\mathcal{B}_i$, and a vector $\boldsymbol{r}$ describing the correlation between the set ($\boldsymbol{p}$,$\boldsymbol{s}$), at which the response of the black box is unknown, and all elements of $\mathcal{B}_i$,

$$\boldsymbol{r}_{(p,s)} = \left(R\left((\boldsymbol{p},\boldsymbol{s}),(\boldsymbol{p}^{(1)},\boldsymbol{s}^{(1)})\right),...,R((\boldsymbol{p},\boldsymbol{s}),(\boldsymbol{p}^{(n)},\boldsymbol{s}^{(n)}))\right)^T, \tag{4.2}$$

where n denotes again as the magnitude of $\mathcal{B}_i$.
The main idea is to balance between searching for a new minimum of the surrogate problem and to improve the quality for unexplored regions of the variable domain. In almost every approach where DACE is used to build the surrogate problem, a MSE equivalent information is somehow used to realize this idea.
However, in contrast to the update strategies discussed by Sasena [233], we use convergence to candidates determined previously as criteria to stop minimizing $f(\hat{\boldsymbol{y}},\boldsymbol{p},\boldsymbol{s})$, and to search for a new candidate with a modified objective. This is partly motivated by the work of Regis and Shoemaker. In [224] the search for a point that minimizes the surrogate problem extended by an additional constraint is proposed. A new minimizer has to be outside ϵ-balls surrounding all elements of the current basis $\mathcal{B}_i$.

We use this constraint as a switch for our optimization procedure. If a minimizer $(\boldsymbol{p}^{*i}, \boldsymbol{s}^{*i})$ is, during iteration i, inside of an ϵ-ball around elements of $\mathcal{B}_i$, the process is forced to find a candidate $(\boldsymbol{p}^{MSE*i}, \boldsymbol{s}^{MSE*i}) \in \Omega_{\hat{\boldsymbol{g}}}$, which maximizes the MSE of the component $\hat{y}_j$ with the most impact on $f$or a user defined aggregation for different components of $\hat{\boldsymbol{y}}$, given by $MSE_f(\hat{\boldsymbol{y}})$. The new candidate is still obtained subject to the unchanged constraints of the surrogate problem. The effect of this switching criteria is that the optimization cannot get stuck into a local minimum (cf. Fig. 4.2). It ensures that all information obtained earlier is included for the selection of new promising candidates. This continues until a stopping rule is satisfied.
This procedure has two further benefits. First, by ensuring a minimal distance between two elements of any base $\mathcal{B}_j$, numerical stability is enhanced. The elements of the covariance matrix $\boldsymbol{R}$ are dependent on the distance between elements of $\mathcal{B}_j$. If the distance approaches zero, the entries of $\boldsymbol{R}$ do the same. This leads to numerical issues during the surrogate building process [79, 143, 149]. Second, from a practice-oriented perspective, the minimal distance criteria can be used to reflect manufacturing tolerances of the underlying systems. The outlined criteria behave as described in Algorithm 1.

Algorithm 1 Distance-based update criteria for surrogate optimization

1: $\mathcal{B}_0$ is given, set $j = 0$.
2: Evaluate $f\left(\boldsymbol{y}(\boldsymbol{p}, \boldsymbol{s}), \boldsymbol{p}, \boldsymbol{s}\right)$ and $\boldsymbol{g}\left(\boldsymbol{y}(\boldsymbol{p}, \boldsymbol{s}), \boldsymbol{p}, \boldsymbol{s}\right)\ \forall\ (\boldsymbol{p}, \boldsymbol{s}) \in \mathcal{B}_j$.
3: Build a surrogate $\hat{\boldsymbol{y}}^{(j)}$ by running DACE on $\mathcal{B}_j$.
4: Find $(\boldsymbol{p}^{*j}, \boldsymbol{s}^{*j}) = \arg\min f(\hat{\boldsymbol{y}}, \boldsymbol{p}, \boldsymbol{s})$ subject to $(\boldsymbol{p}, \boldsymbol{s}) \in \Omega_{\hat{\boldsymbol{g}}}$.
5: If $||(\boldsymbol{p}^{*j}, \boldsymbol{s}^{*j}) - (\boldsymbol{p}, \boldsymbol{s})|| \geq \epsilon,\ \forall (\boldsymbol{p}, \boldsymbol{s}) \in \mathcal{B}_j,\ i = 1, ..., n$, go to 8, else go to 6.
6: Find $(\boldsymbol{p}^{MSE*j}, \boldsymbol{s}^{MSE*j}) = \arg\max\left(MSE_f\right)$ subject to $(\boldsymbol{p}, \boldsymbol{s}) \in \Omega_{\hat{\boldsymbol{g}}}$.
7: Set $(\boldsymbol{p}^{*j}, \boldsymbol{s}^{*j}) = (\boldsymbol{p}^{MSE*j}, \boldsymbol{s}^{MSE*j})$.
8: Evaluate $\boldsymbol{y}(\boldsymbol{p}^{*j}, \boldsymbol{s}^{*j})$, and define $\mathcal{B}_{j+1} := \mathcal{B}_j \bigcup \{(\boldsymbol{p}^{*j}, \boldsymbol{s}^{*j})\}$.
9: Stop, or set $j = j + 1$ and go to 2.

4.2.3 Stopping Rules and Convergence

There is still the question when to stop the search for new minimizer. Some stopping rules exist which can be followed, but a general rule cannot be given. In practice the underlying black box evaluations consume many computational resources. To bound the cost in total or in relation to the achieved improvement is a simple consecutive rule. Total cost means in this context, the overall consumed resources, and relative cost means the resources compared to the achieved improvement. During the first iterations there is usually a large improvement. An objective function improvement is typically not found in each iteration, but only after a number of black box evaluations in iterative optimization processes by sampling. The length of such unsuccessful periods can be bounded. For most of the applications presented in Chapters 5 to 7 a total number of black box evaluations is applied as a stopping rule.
How many resources are finally spent is difficult to predict. The desirable aim, five times the dimension of the optimization variable is mostly not a realistic goal for large improvements. But this reflects the especial interest to find good solutions and not optimal solutions many iterations later. However, if prior knowledge is available in the sense of theoretically known objective function minimum values, or given reference solutions, it should be incorporated

directly. The procedure is stopped if an objective value is obtained close to a known optimal value respecting a user-defined tolerance. On the other hand, a yet missing systematic reasonable stopping rule may enforce more interest in interactive optimization methods, where new system information is under the user's review directly during each iteration as in some of the so-called simulated reality approaches.

4.3 Options for Parallelization

A way to reduce the required wall-clock-time for computationally exhaustive, or other resource-consuming applications is parallelization, e.g., applied in [108] for a surrogate optimization approach. Parallelization of simulations in engineering is on its own a huge area of current research. The evolution of workstations and desktop computers to multi-core CPUs in background even accelerates the progress in this field. Indeed, the measurement of wall-clock-time does not provide an appropriate possibility for comparison of the efficiency of different optimization approaches, but rather, enables to reduce the user-time needed to obtain a proper solution. Therefore, hardware used in total weighted by total time consumed would also provide a certain degree of comparison [24]. W.l.o.g., we assume in the following that the evaluation of the underlying black box is done as efficiently as possible as proposed in [78].
The starting point for parallelization approaches is the distribution of the objective function evaluations, the distribution through the optimization framework, and the distribution of each optimization instance itself. The following aspects have to be considered under the assumption that the computational costs for optimization can almost be neglected, in comparison to the black box evaluation costs.

4.3.1 Sequential Optimization Procedures

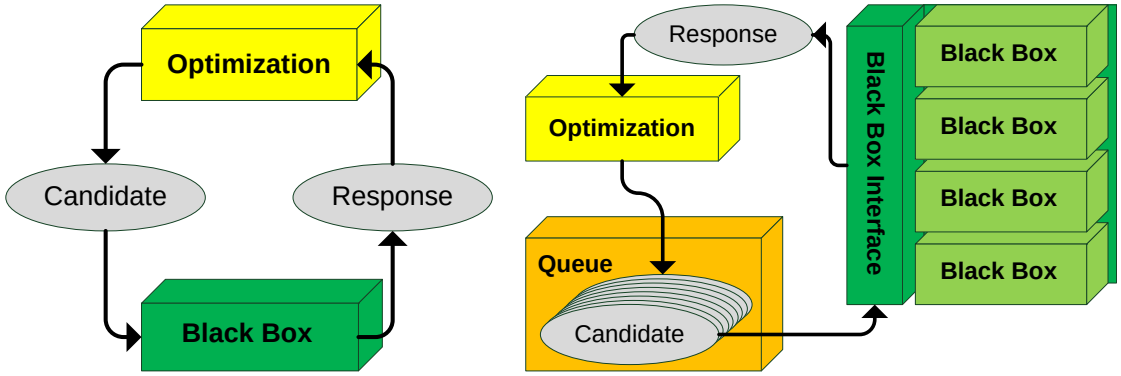
The pure sequential procedure is the classical approach where optimization and black box evaluation are coupled in a simple loop as shown in Figure 4.3 a. The applied optimization method uses the provided output $\boldsymbol{y}$ to determine the next candidate, and switches back for the black box evaluation. The resources needed to obtain new candidates are only a small part by definition, and the black box evaluation environment is only for a short period not in use. This is the simplest example of a queueing system, one queue and one exit point.

4.3.2 Synchronous Parallelization Approaches

For the following considerations we assume that the black box allows n evaluation runs in parallel, so that the total used time can be reduced if the applied optimizers make use of it.

Simultaneous Parallelization of Objective Functions Calls

For simultaneous use of the available parallel computing infrastructure, optimization methods can only be adopted efficiently if they aggregate the output of n objective function evaluations to generate n new candidates for simulation. Methods like GAs or EAs are naturally enabled for these kinds of parallel function evaluations. If the optimization parameters are well-suited for the hardware that is used, all available resources are used.



(a) Simple coupled optimization and black box evaluation in a loop.

(b) General layout for using parallel black box evaluations.

Figure 4.3: Problem parallelization frameworks approaching the different procedure levels. New candidates are only generated by the used optimization procedure on demand when resources for black box evaluations are available.

Depending on the size of the problem, usually only sizes of 20, at the most of 30, elements per generation are used for expensive black box-based optimization.
The number of elements per generation is just one approach for parallelization of these methods; most parallelizable methods are not freely scalable according to available hardware. A trade-off arises between more applied hardware resources and total efficiency so that the wall-clock-time cannot be reduced linearly by simply providing more resources. Especially for the proposed surrogate optimization methods, synchronous parallelization is a way to speed up the process. The initial DOE is typically evaluated in parallel. After an initial DOE, the use of multipoint methods provide further simple methods to generate a number of candidates, as is needed to exploit the capacity of the parallel infrastructure. In [252] this is performed by running the black box for as many local minima of the surrogate problem as instances for evaluations are available.

Parallelization of Optimization Methods

Additionally, a simultaneous procedure of parallelized objective function evaluation enables the use of the resources between two steps to determine the next candidates. This does not provide a significant improvement of total computational time; these costs are negligible for the problems considered here. It becomes interesting for hybrid and interactive optimization methods. As a simple example of how this can be applied, just assume that each of the n candidates that are generated on separate nodes during one iteration of the optimization approach, so that all resources are allocated in rotation by optimization and by black box evaluations. The described type of parallelization is a sequential simultaneous parallel procedure. On example is given by Dennis for the multi-directional search algorithm in [63]. A hybrid optimization approach could place multiple optimization methods on different nodes, each generating a new candidate.

4.3.3 Asynchronous Parallelization Approaches

Besides simultaneous approaches for parallelization, we can also consider situations where each of the objective function evaluations for different candidates do not consume the same time, or where the evaluations are not started simultaneously (cf. Fig. 4.3 b). This may occur after the initial phase of the optimization procedure, or even at the very beginning when the information from previous black box evaluations is incorporated.

Asynchronous Distribution of the Simulations

Optimization methods, which generate new candidates for simulation without getting new responses for candidates cause an asynchronous proceeding. As a very simple example of such a method, we can think of a number of existing evaluated points, and each new candidate should minimize the maximal distance to all other points. This way, parallel systems with any number of nodes can be used. However, there is no guarantee that the available computational power in such a net will always be used at a certain level.
For a detailed description for a special procedure we refer to an implementation of Gray and Kolda [115], where a queue of candidates is also unattached by new obtained responses. Methods using all available problem information are more efficient for surrogate optimization. As soon as new results are returned, all previous candidates from the queue for simulation are replaced by new candidates, generated using all obtained information. This means that new candidates for evaluation should be generated only on demand.

Distribution and Parallelization of Simulation and Optimization

In an asynchronous parallelization scenario, the optimization distributes the objective function evaluations and the black box evaluations. These jobs run on different available nodes. If we especially consider mixed-integer optimization problems that are solved by branch and bound, different nodes of the branch and bound tree can easily be distributed. A parallel tree search helps to decrease the total time to get new candidates. A first approach to utilize parallelization for simulation and optimization issues is given in [81], where distributed simulation, the use of surrogates and different optimization methods are applied.

4.3.4 Problems with Multiple Black Boxes

Finally, we can assume a situation where all components of $\boldsymbol{y}$ are not available at the same time, which corresponds in parts with the situation where not all components of $\boldsymbol{y}$ are provided with one black box response. This is the typical resulting scenario if more than one black box has to be run for a function evaluation of $f$and $\boldsymbol{g}$. Multiple black box problems arise, for example, if different numerical simulation programs are used to evaluate different components of $\boldsymbol{y}$. Barton discusses a layout for a multi black box scenario [18], and presents a general layout to deal with multiple black boxes, needed to evaluate the underlying objective function (cf. Fig. 4.4).
A more complex situation occurs in multidisciplinary design optimization, a field of engineering that uses optimization methods to solve design problems incorporating a number of disciplines, which results in optimization problems aggregating various black boxes [6, 30, 137, 272]. Such a multi black box layout allows, in the first place, integrated

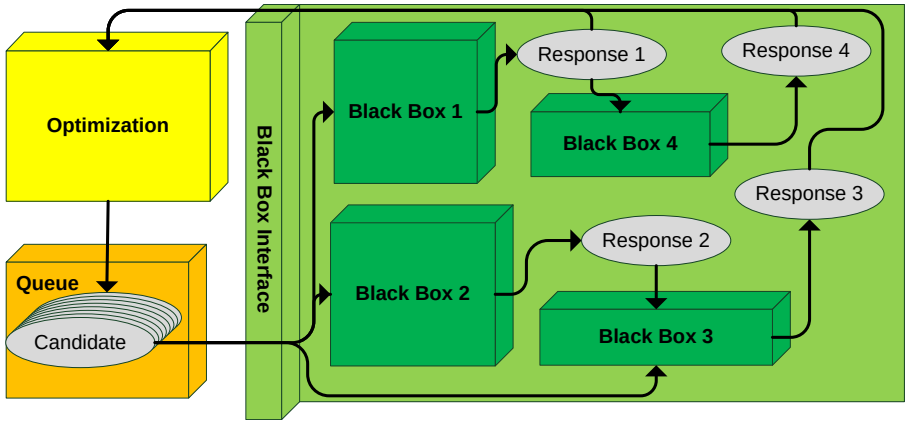


Figure 4.4: Multi black box problems layout.

optimization approaches. However, on an abstract level, the situation is well-suited to the derived approach, as it is not mandatory for $f$and $\boldsymbol{g}$ to be evaluated completely. As soon as just a response value of one component of $\boldsymbol{y}$ is returned, new candidates are generated. Another problem often arises when different black boxes are coupled directly, and not only aggregated by the objective function, but depend partly on the output from another. An approach presented in [141] is called collaborative metamodeling, where dependencies of the underlying black boxes are identified and used to build up a system of surrogate functions for each single black box or for even closely related sets of black boxes. On these sets of surrogate functions, an optimization run can be carried out. Foremost, the derived approach allows the adaption of such closely-coupled and dependant surrogate functions as part of the optimization according to integer-valued and real-valued variables.

4.4 Summary of Chapter 4

We described how the surrogate problem, which replaces the original problem as introduced in Chapter 3, is applied into a sequential update procedure. The candidates for black box evaluations are determined by the optimization procedure and all the information from each obtained black box evaluation is used to find new candidates. In this way, the derived surrogate optimization approach generates a framework for a iterative black box-based optimization procedure, which es even applicable to parallel environments.

5 Electrical Engineering Applications

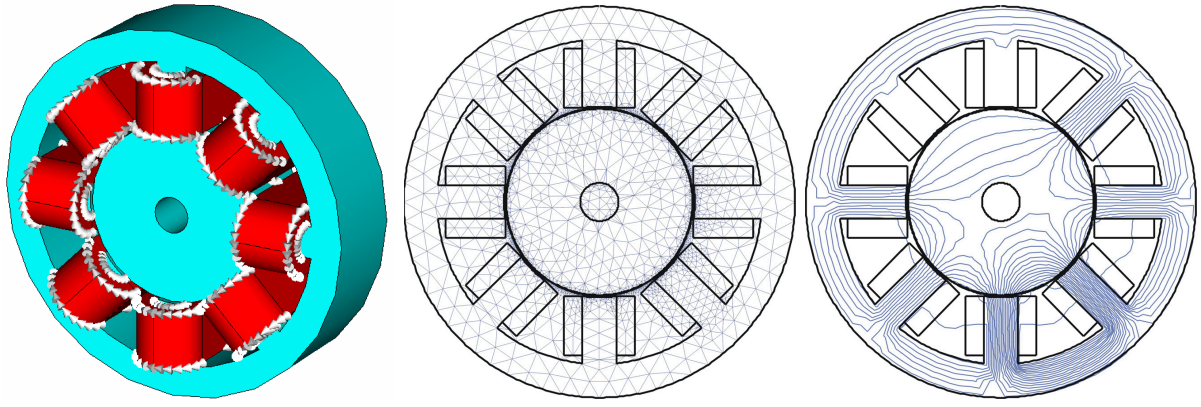
In this Chapter we apply the derived surrogate optimization approach to solve typical optimization problems arising for two electrical engineering applications. We will retrieve all the characteristic optimization problems attributes described when we introduced general black box based problems.

5.1 Characteristics of Magnet Design Optimization

Both design problems following are based on a finite element (FE) simulation of electromagnetic fields but are used for the design of quite different devices. First, we take a look at a design optimization problem of a magnetic bearing which is of lower complexity than the latter design problem of a superconductive magnet. Therefore, a number of implicit constraints create of nonconvex and disconnected feasible domain as we will see. Besides continuous-valued variables also integer-valued ones have to be handled for the second problem.

5.2 Cost Minimal Design of a Magnetic Bearing

Magnet bearings provide properties needed for certain applications as high surface speeds, low energy consumption, non-contacting, low vibration, and further more. Here we do not focus on applications for this devices, but on the optimal design under certain assumptions for the magnetic bearing properties.



(a) Geometry model. (b) Meshed model. (c) Electromagnetic field.

Figure 5.1: The different steps form design to simulation of a magnetic bearing.

Table 5.1: Numerical simulation response.

$y_{costs}(n_p, p_{pl}, p_{pw}, p_{yw})$	Total costs of copper and steal	in €
$y_{direct}(n_p, p_{pl}, p_{pw}, p_{yw})$	Force in direct direction	in N
$y_{quad}(n_p, p_{pl}, p_{pw}, p_{yw})$	Force in direction of quadrature	in N^2

5.2.1 Design Details for the Magnetic Bearing

The technical problem in designing a magnetic bearing is to respect the ferromagnetic saturation, the eddy current, and the hysteresis losses in the numerical simulation. Especially the exact calculation of the forces has to be guaranteed.
For the case considered here, the layout is parameterized by the number, the length, and the width of the poles, as well as the thickness of the yoke, and the inner radius of the stator. A 3-D model is generated (cf. Fig. 5.1 a) for which a mesh is built (cf. Fig. 5.1 b), that provides the base for the FE code to simulate the electro magnetic field (cf. Fig. 5.1 c). All steps are summarized by one simulation package [212]. Three factors calculated from the simulation output are taken into consideration, the total costs of copper and steel to build the bearing, and the force in the direct direction as well as in the direction of quadrature to guarantee the expected bearing properties. Table 5.1 summarizes the output $\boldsymbol{y} = (y_{costs}, y_{direct}, y_{quad})^T \in \mathbb{R}^3$ of the used simulation package which is being coupled in a loop with the optimization.

5.2.2 Optimization Problem Formulation

In Table 5.2 the parameters of the system design problem are given. A reduction to two continuous-valued variables is motivated by initial examinations where these two optimization variables were identified as the most important ones, the poles width and length. The number of poles is fixed to nine, because in contrast to prior guesses, only a few feasible designs with a different number of poles exist. This was confirmed during an expensive grid search of high resolution with different number of poles to proof the results from the optimization process. The aim is to reduce the costs of copper and steal for such a device,

$$\min f(p_{pl}, p_{pw}) := y_{costs},$$

subject to constraints $\boldsymbol{g} = (g_1, ..., g_4)^T$, where the first two are given implicitly

$$\begin{array}{rcrcr} g_1(p_{pl}, p_{pw}) & := & (y_{direct} - y_{quad})^2 & \leq & 10^4 \\ g_2(p_{pl}, p_{pw}) & := & -y_{direct} - y_{quad} & \leq & -200. \end{array}$$

Table 5.2: Design problem parameters.

n_p	$\in \mathbb{N}$	Number of poles	p_{pl}	$\in \mathbb{R}$	Pole length [mm]
p_{pw}	$\in \mathbb{R}$	Pole width [mm]	p_{yw}	$\in \mathbb{R}$	Yoke width [mm]
r_s	$\in \mathbb{R}$	Inner radius of stator [mm]	p_{pd}	$= \frac{2\pi}{n_p}$	Pole distance [rad]
ϵ_{tol}	$= .001$	Manufacturing tolerance [mm]			

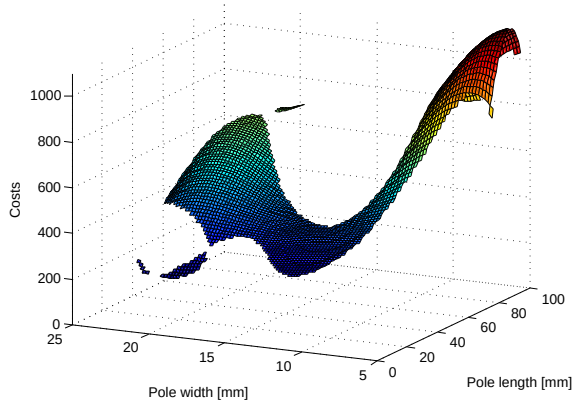
Another two explicit constraints,

$$\begin{array}{rcccl} g_3(p_{pl}, p_{pw}) & := & -r_s \sin(\frac{p_{pd}}{2}) + \frac{p_{pw}}{2} + \epsilon_{tol} & \leq & 0 \\ g_4(p_{pl}, p_{pw}) & := & \epsilon_{tol} - p_{pl} & \leq & 0, \end{array}$$

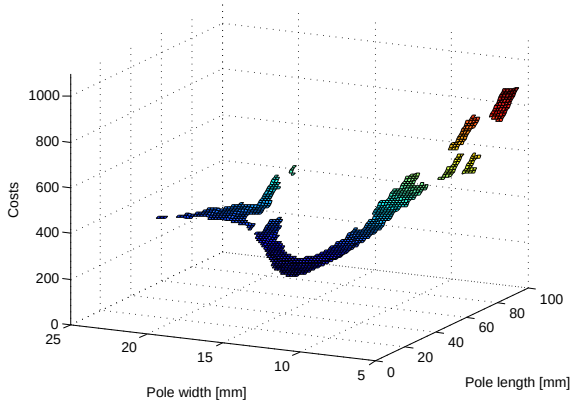
and simple bounds on the optimization variables for the geometric model,

$$10 \leq \quad p_{pl} \quad \leq 100, \quad 5 \leq \quad p_{pw} \quad \leq 25.$$

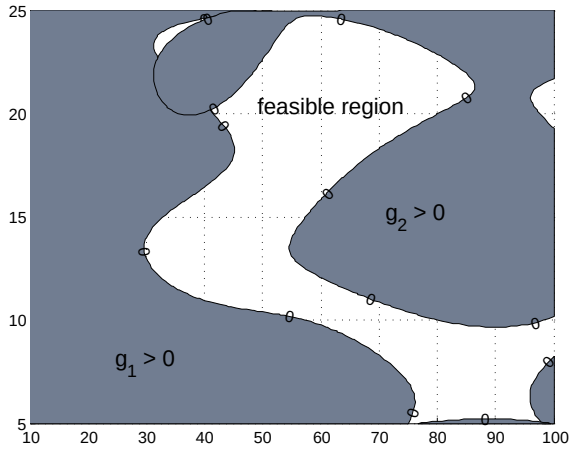
have to be hold. The constraints g_3 and g_4 guarantee the geometric integrity for the design of the device.



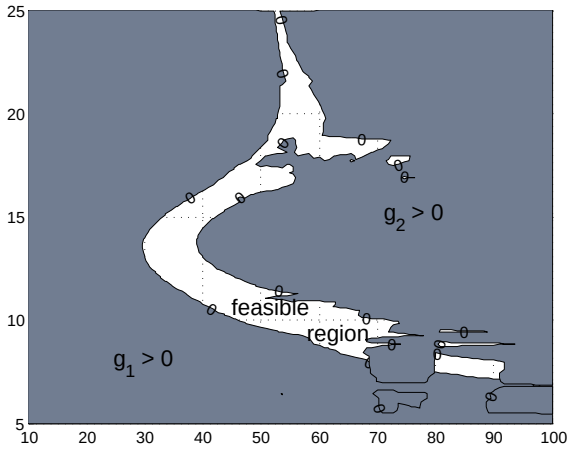
(a) Surface of the objective function surrogate after 35 simulation runs.



(b) Surface of the objective function.



(c) Surrogates of the constraints after 35 simulation runs



(d) Original constraints.

Figure 5.2: Surrogate-based versus simulation-based optimization surface.

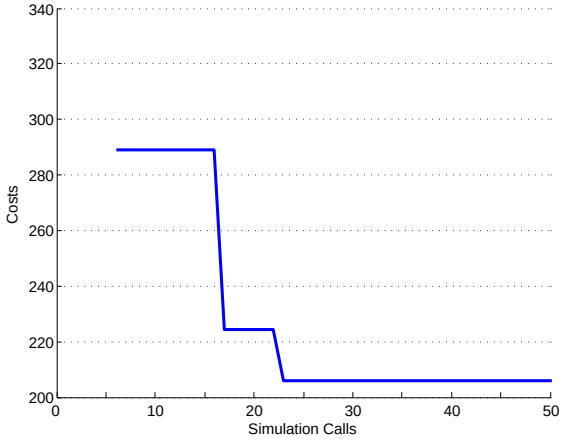
5.2.3 Implementation and Numerical Results

The code of the electro magnetic field simulation is a stand alone black box. The whole optimization procedure is implemented in Matlab, which calls the DACE toolbox [177] to

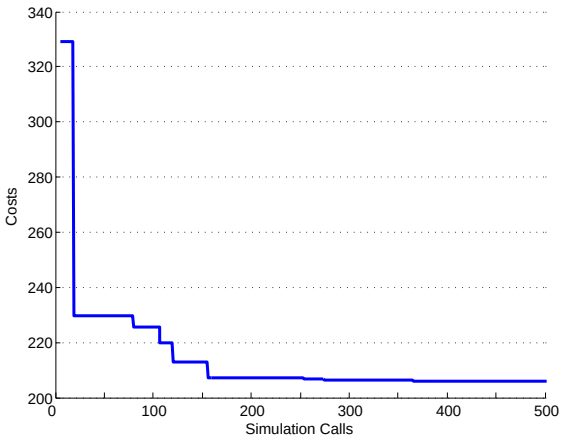
Table 5.3: Best found design of the magnetic bearing during the optimization process.

Number of iteration	(p_{pl}, p_{pw}) SurOpt	Costs SurOpt	(p_{pl}, p_{pw}) DIRECT	Costs DIRECT
6	(40, 11.6667)	289.1745	(55, 21.6667)	329.1087
17	(32.2393, 13.3958)	224.2539	(55, 21.6667)	329.1087
23	(29.8582, 13.8585)	205.9889	(35, 15)	229.7703
24	(29.8796, 13.8954)	205.8434	(35, 15)	229.7703
33	(29.8794, 13.8955)	205.8407	(35, 15)	229.7703
36	(29.8792, 13.8956)	205.8395	(35, 15)	229.7703
419	-	-	(29,8148, 13.8294)	205.9421
2579	-	-	(29.8880, 13,9026)	205.8404

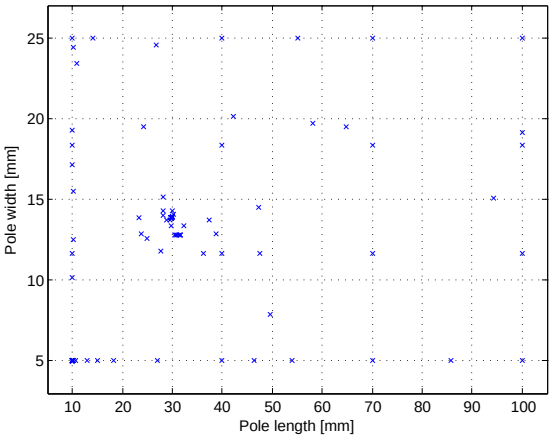
generate the multivariate approximations of the simulation output. The SQP-method of the Matlab optimization toolbox is used for minimizing the surrogate problem during each iteration, as well as for maximizing the MSE_f function when needed.
Because of the absence of prior knowledge we start with the evaluation of a four times four grid on the variable domain of equally spaced points, after smaller grids have not carried out even one feasible design candidate. This forms also the base for approximating the problem by a DACE surrogates. The objective function f and the constraints g_1 and g_2 are replaced by surrogate functions, the explicit constraints are included during the whole procedure directly. An instant decrease is achieved right after the sequential update process is started based just on the information of the previous 16 simulation runs. The following evaluated design candidates do not provide a feasible design, but they reduce the constraints violation of g_1, which is during the whole procedure the most active one.
The ϵ-ball criteria introduced in Chapter (4) becomes active and some new candidates are evaluated that maximize $MSE(\hat{y}_{costs})$. This step provide kind of a restart. However, the candidate for evaluation start quickly to cluster in one region (cf. Fig. 5.3). The best found feasible candidate is found after only 36 simulation runs, after only 19 updates of the surrogate problem. Even after 100 iterations no better design was found, only design candidates from the entire design space were evaluated. This small example illustrates all parts of the in Chapter 4 developed update procedure.
To give an impression on the optimization surfaces, the original and the surrogate problem are compared to Figure 5.2 a and b. The approximation accuracy of the surrogate functions building the constraints are visualized in Figure 5.2 c and d. The combination between the feasible region and the objective function is interesting. The surrogates of the constraints are almost exact around the best found design, but do not provide a good approximation of the total feasible region of the original problem. A scenario using DIRECT as the optimizer was set up for comparison reasons. The Matlab implementation of Finkel [88] was used out of the box. The implicit constraints are handled as hidden ones by DIRECT. The stopping criteria used is the objective function value of the best found design during optimization by the proposed surrogate-based approach. After 2579 simulation runs a function value is found close to the minimum function value found by the first approach. The steps of improvement during the optimization procedure is given in Table 5.3 as well as in Figure 5.3, where the results of both applied methods are visualized to point out the



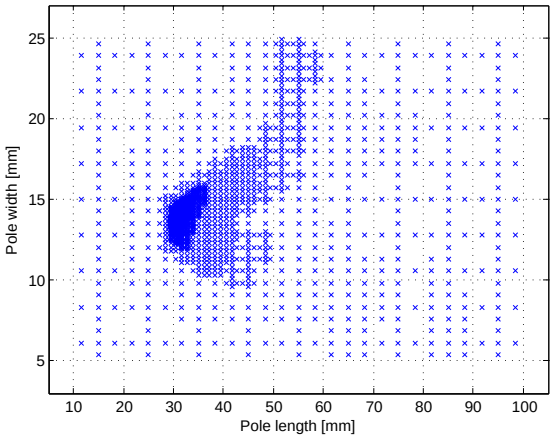
(a) Costs for the best found feasible design versus the number of used objective function evaluations during the surrogate optimization approach.



(b) Costs for the best found feasible design versus the number of used objective function evaluations during optimization with DIRECT.



(c) Scatterplot of 100 evaluated magnetic bearing designs determined by the surrogate optimization approach.



(d) Scatterplot of 2579 evaluated magnetic bearing designs determined by DIRECT.

Figure 5.3: Results of the optimization run for the magnet bearing design.

differences. During the optimization procedure with DIRECT, a better design was found 97 times. During an earlier analysis with other optimization methods no better designs were found than tho one determined using surrogate optimization. A grid exploration of 100 times 100 points equally spaced over the whole variable domain has only provided a feasible candidate with an objective function value of 206.63.

5.2.4 Summary

We have seen that handling of nonlinear constraints in an efficient way leads to enormous advantages in comparison to methods not really adopted to deal with them. Of course, DIRECT can be combined with a more sophistic constraints handling approach, but here we applied it out of the box, as a typical user would have done it. Thus, the surrogate

optimization approach determined the cheapest identified geometry of the device with least function evaluation.

5.3 Efficiency Optimization of a Superconductive Synchrotron Magnet

Figure 5.4: Example for the type of used syncotron magnet for the new SIS100/300 accelerator [103].

The numerical optimization of continuous parameters in electrotechnical design using electromagnetic field simulation as seen above is already standard. The surrogate optimization approach is now applied for the design of an efficient superconductive magnet for the use in a particle accelerator facility (cf. Fig. 5.4), where real-valued as well as integer-valued variables are used to determine the design of the device [127, 129, 130].This application is motivated by the planned new facility for antiproton and ion research (FAIR) (heavy ion acceleration) at the Gesellschaft für Schwerinonenforschung (GSI) in Darmstadt (cf. Fig. 5.5). The field quality in the aperture of the magnet heavily depends on the geometrical distribution of the turns. The optimal placement of the turns and the distribution of the number of turns along the turns blocks is our optimization task.

5.3.1 Problem Description

The homogeneity of the magnetic field in the aperture of the superconductive magnet is determined by the geometry of the coil. Especially, the position of the coil blocks and the number of turns on each are influencing the quality of the aperture field. The layout of the coil has to obey mechanical constraints such as, a minimal distance between two adjacent coil blocks. Invoking a separate continuous-valued optimization for every possible distribution of the integer number of turns over the coil blocks is not feasible by the underlying numerical simulation. The function evaluations involve 2D nonlinear magnetostatic field simulations. The PDE

$$-\frac{\partial}{\partial x}\left(n_u \frac{\partial A_z}{\partial x}\right) - \frac{\partial}{\partial y}\left(n_u \frac{\partial A_z}{\partial y}\right) = J_z \tag{5.1}$$

where n_u denotes the reluctivity, $A_z(x, y)$ is the z-component of the magnetic vector potential and $J_z(x, y)$ is the z-component of the applied current density, is discretized by linear finite-element (FE) shape functions. Due to symmetry, only a quarter of the magnet cross-section has to be modeled (cf. Fig. 5.6). The FE solver is equipped with adaptive mesh refinement controlled by a heuristic error estimator based on the locally stored magnetic energy [212]. This is absolutely necessary to guarantee a reliable function evaluation for

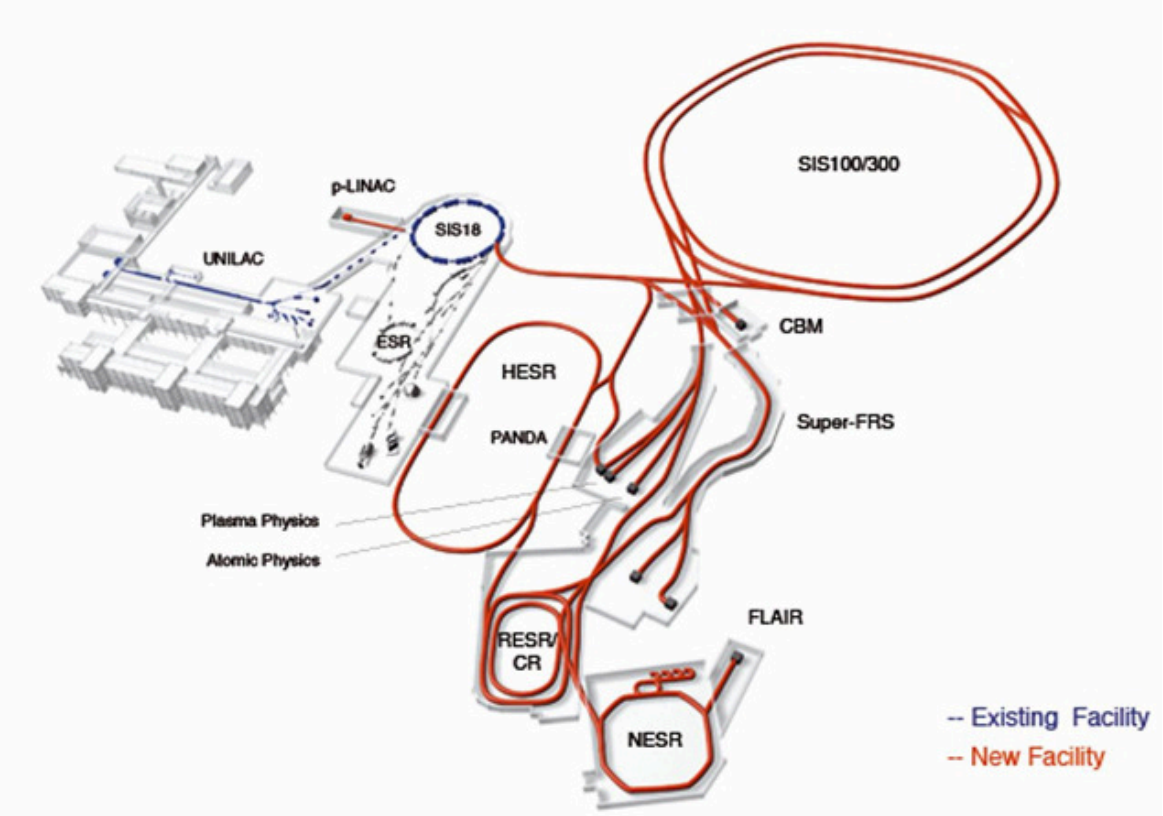


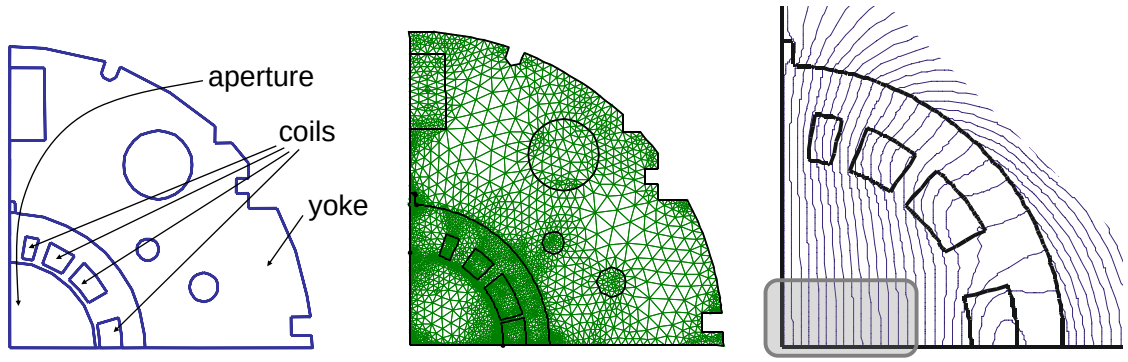
Figure 5.5: Newly planned heavy ion accelerator facility at GSI in Darmstadt [103].

every geometry suggested by the optimizer. The nonlinearity caused by the ferromagnetic saturation in the iron yoke is resolved by the Newton method [213].
The quality of the design depends on the homogeneity of the magnetic field in the center of the magnet aperture. The radial component B_r of the magnetic flux density at a reference radius r_{ref} is represented by the series expansion, i.e.,

$$B_r\left(r_{\mathrm{ref}}, \theta\right) = B_1 \sum_{n=1}^{\infty} b_n \sin(n\theta) + a_n \sin(n\theta) \; . \tag{5.2}$$

Here, B_1 denotes the magnitude of the vertically oriented dipole magnetic flux density and b_n and a_n are the relative normal and skew components, evaluated at r_{ref}, respectively [165]. By construction, $b_1\left(r_{\mathrm{ref}}\right)$ equals 1. For an exact dipole field, all other components are 0. Acceptable values for b_n and a_n are in the range of 10^{-4}. These field properties are extracted from the FE solution by evaluating the magnetic vector potential $A_z(r_{\mathrm{ref}}, \theta)$ at a circle with the reference radius.
Thus, the field quality factor is defined by

$$y = \sqrt{\sum_{n \in \mathcal{S}_b} b_n^2 + \sum_{n \in \mathcal{S}_a} a_n^2} \tag{5.3}$$



(a) Parts of the device. (b) 2D Mesh. (c) Area of interest.

Figure 5.6: 2D design optimization problem of the computed electromagnetic field in the aperture of the magnet.

where $\mathcal{S}_b \subset \mathcal{S}_0$ and $\mathcal{S}_a \subset \mathcal{S}_0$ select the relevant components. The aim is the minimization of y subject to the geometrical constraints mentioned above. Hence, a constrained, mixed-integer nonlinear optimization has to be carried out.

5.3.2 Optimization Problem Formulation

Typically, only continuous-valued variables describing the azimuthal distance for each of the n_p coils by $\boldsymbol{p}$, with $\boldsymbol{p} \in \mathbb{R}^{n_p}$, are considered for optimization. Additionally, the optimization algorithm has to decide on the number of turns applied in each of the coil blocks, which determines the size of the individual blocks. These data is represented by integer numbers $\boldsymbol{s}$, with $\boldsymbol{s} \in \mathbb{N}^{n_s}$.
The sum of turns on all coil blocks is fixed, and furthermore a gap of the size of one winding has to be respected between two neighboring coil blocks. These geometric restrictions lead to lower and upper bounds as well as linear constraints on the optimization variables $\boldsymbol{p}$ and $\boldsymbol{s}$, which are summarized by

$$\boldsymbol{p}_{lb} \leq \boldsymbol{p} \leq \boldsymbol{p}_{ub}, \quad \boldsymbol{s}_{lb} \leq \boldsymbol{s} \leq \boldsymbol{s}_{ub}, \tag{5.4}$$

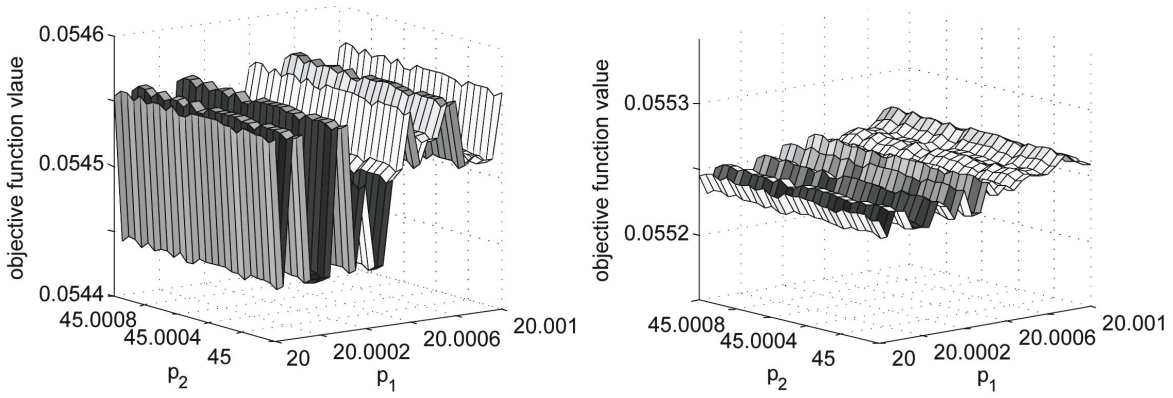
with $\boldsymbol{p}_{lb}, \boldsymbol{p}_{ub} \in \mathbb{R}^{n_p}$, $\boldsymbol{s}_{lb}, \boldsymbol{s}_{ub} \in \mathbb{R}^{n_s}$, and by

$$\boldsymbol{A}(\boldsymbol{p}^T, \boldsymbol{s}^T)^T \leq \boldsymbol{b}, \quad \boldsymbol{b} \in \mathbb{R}^{n_b}, \quad \boldsymbol{A} \in \mathbb{R}^{n_b \times (n_p + n_s)}. \tag{5.5}$$

The resulting domain of feasible design candidates $(\boldsymbol{p}, \boldsymbol{s})$ for the device is defined by $\Omega_{\boldsymbol{g}}$. Equation (5.3) provides the simulation-based objective function value for a certain combination of $(\boldsymbol{p}, \boldsymbol{s})$, which has to minimized for a good design. Finally,

$$\min f(\boldsymbol{p}, \boldsymbol{s}) := y(\boldsymbol{p}, \boldsymbol{s}), \text{ subject to } (\boldsymbol{p}, \boldsymbol{s}) \in \Omega_{\boldsymbol{g}}. \tag{5.6}$$

defines the resulting mixed-integer nonlinear optimization problem.



(a) Without mesh adaption.

(b) Mesh adaptation applied.

Figure 5.7: Comparison of the optimization surface smoothness.

5.3.3 Implementation and Numerical Results

The derived surrogate optimization approach is combined with an electromagnetic field simulation software [212]. The optimization problem is invoked for a fixed number of four coil blocks and a total number of 32 turns, equivalent to the variables $\boldsymbol{p} \in \mathbb{R}^3$ and $\boldsymbol{s} \in \mathbb{N}^3$. The number of linear constraints resulting from the geometric requirement given by $\boldsymbol{A}$ and $\boldsymbol{b}$ is 5. The minimal distinction allowed between two considered design candidates controlled by the size of the ϵ-ball introduced in Chapter 4 is motivated by the manufacturing tolerance of the real device, which is assumed to be met by $\epsilon = 10^{-4}$.
In this case only the objective function f is replaced by the surrogate $\hat{f} := \hat{y}$, all other parts are given explicitly. During our numerical experiments, it turns out that best found designs of different applied optimization methods without mesh adaptation in the numerical simulation are unstable with regards to small changes in the positions of the coil blocks, which indicates the necessity of mesh adaptation. An impression of the discontinuity in the optimization surface without mesh adaption is given in Figure 5.7, where function values of f are plotted for a grid of 26 variations of the second and the third coil by steps of 4ϵ. The resulting change in f is ten times higher without mesh adaptation, and more than ten times higher than the lowest found objective function value.
Figure 5.8 illustrates the quality improvement of the magnetic field according to the applied number of simulation calls. The progress of three optimization runs with an EA is also shown. Three expert's guesses for the distributions $\boldsymbol{s}$ of turns over the coil blocks are used to optimize position of the coils given by $\boldsymbol{p}$. Around these three initial expert's guesses the initial set of points $\mathcal{B}_0$ for approximation is built. It is done by varying each dimension of each of expert's guesses separately in both directions as proposed in [128].
During three out of the four different optimization runs a design with an objective function value of less than 10^{-4} is found in less than 300 simulation calls. It is to emphasize that the EA was run on the three-dimensional NLP, whereas the surrogate approach was run on a six-dimensional MINLP. In other word, the EA solved within each run one of the NLP problem that arise at the final knots of the branch and bound tree during optimization by the surrogate optimization approach. All generated design candidates, given by tuples $(\boldsymbol{p}, \boldsymbol{s})$, during all iterations of the surrogate optimization approach are plotted in Figure 5.9.

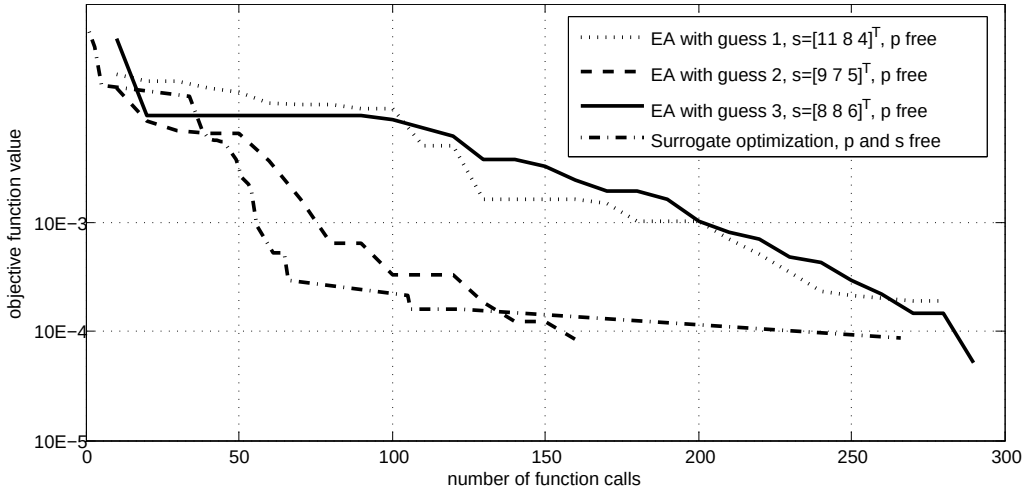
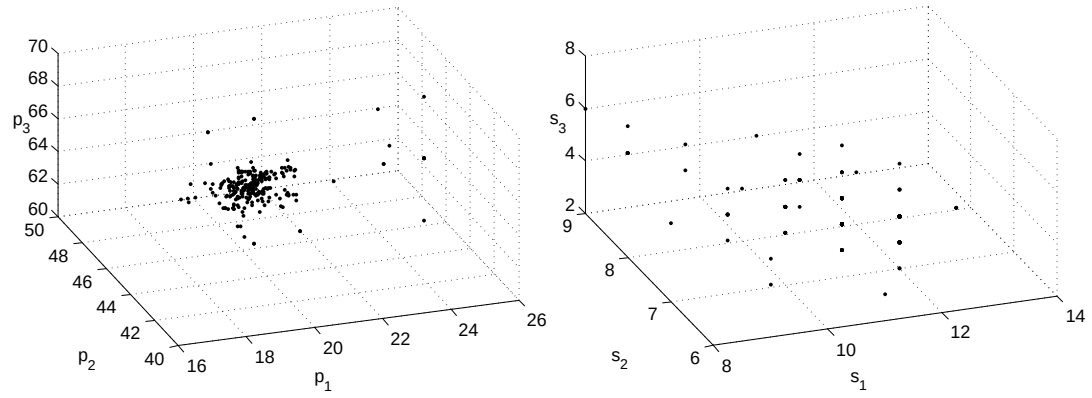


Figure 5.8: Progress of three different optimization methods, three runs with an EA, each with a different discrete design and only free continuous design variables, and one run with the proposed surrogate optimization method, with free discrete and free continuous design variables.

The finally obtained best design has a discrete part $\boldsymbol{s}$ which was not suggested as one of the expert's guesses and was not a part of a tuple out of the set $\mathcal{B}_0$.
Further applied DOE methods to generate a space-filling initial bases $\mathcal{B}_0$ like Latin Hypercube sets, grids, or just random sets of points have not carried out any candidate $(\boldsymbol{p}, \boldsymbol{s})$ with an objective function even close to 10^{-4}. As a further optimization approach, a test version of a commercial EA OptiY version 2.3 was applied [217]. This optimization package is able to handle mixed-integer nonlinear optimization problems even with linear and nonlinear constraints in its default settings. The attempts to find results for a comparison with the surrogate optimization approach on the six-dimensional MINLP problem were not successful. Even after thousands of simulation calls no design of comparable quality is



(a) Evaluated variation of $\boldsymbol{p}$ (b) Evaluated variation of $\boldsymbol{s}$

Figure 5.9: Different design candidate separated in **p** and **s** evaluated during all iterations the proposed MINLP approach by surrogate functions.

found by the commercial EA. Changes in the parameters settings of the EA have not led to any further improvement.

5.3.4 Summary

The aperture-field quality of the considered superconductive magnet can be significantly improved on the basis of the derived surrogate optimization approach: The determination of the optimal coil geometry only consumed an acceptably small number of computationally expensive electromagnetic field simulations, especially if it is considered that the applied optimization approach solved a six-dimensional MINLP problem, in comparison with the EA results of a three-dimensional NLP problem obtained by fixing the integer-valued design parameters. An expert's guess is still helpful to start in a region of good candidates for the magnet design.

6 Water Resources Management Applications

The focus of this chapter is the formulation of simulation-based optimal design problems in the context of water resources management. The importance of subsurface flow problem in real application can e.g. be discovered in [281] and [250], and results for mixed-integer approaches of the author are already published in [126, 131].
The flow, diffusion, and reactions processes are typically modeled in a different rate of complexity [67] according to the purpose of investigation. However, the increase in available computational power lead to a trend of complex, nonlinear, mostly three-dimensional geometrically detailed models, which are simulated based on well-adapted mathematical and numerical methods. A considerable challenge in combining simulation and optimization for this scenario arises, because widely-used subsurface flow simulators, e.g., FEFLOW [66] and MODFLOW [196], are not specifically designed for the needs of optimization software. However, the demand for efficient handling of water resources problems can only be met by applying optimization methods, as pointed out in [204].

6.1 Problem Description

We consider two different simulation scenarios of the ones proposed in the literature as benchmark problems [193], a water supply problem and a hydraulic capture problem. These problems are no standard benchmark problems, because in contrast to many others these include a numerical simulation output in objective function and constraints.
Both problems require to determine the appropriate number of wells, well locations, and pumping rates to operate a well-field at minimal costs. In [195], these types of problems are studied with classical gradient-based methods for NLP and optimal control theory approaches. In particular, the coupling of numerical simulation and optimization yields an objective function that is nonconvex, non-differentiable, and discontinuous. The feasible region may also be disconnected. To summarize, these problems cover almost all of the typical problems which we introduced already. Other problems in the field of groundwater engineering and management were considered earlier in several works of Barth et. al. [15, 16]. The problems resulting from discretized optimal control or optimal design problem formulations therein are solved by a distributed simplex-based optimization procedure.
Here we are interested in the modeling of the optimization problem and how it can change the range of applicable optimization methods and how the results can differ. The cost model considered depends on the number of wells in the final design by including a fixed installation cost for each well in addition to an operational cost over the simulation period. Inclusion of fixed costs leads to a discontinuous objective function, and one must consider how to add or remove wells from the design space. The problems would naturally be formulated as mixed-integer, simulation-based, nonlinear programs, as it is mentioned in

e.g., [51]. In the context of water resources, a method based on OA for handling simulation-based constraints in the context of subsurface flow is described in [214].
We apply three different optimization methods to solve the resulting problem formulations. Beside the surrogate optimization approach we have chosen an implicit filtering algorithm [107] as a frequently used deterministic sampling method in the optimization community. Additionally a GA as a random sampling approach is applied, since GAs have gained popularity in the optimal design community and have been successfully applied in the groundwater engineering community [47, 139, 189]. The results presented here are based on [126] and [131].

6.2 The Community Problems: A Benchmark Set

The so-called community problems (CPs) posed in [193] were developed after an extensive literature search and serve as a suite of benchmark applications for both the optimization and water management communities. The CPs consist of models, physical domains, objective functions, and constraints resulting in thirty design applications. In [194], there is a complete set of data to define the physical domains which range from simple homogeneous, confined aquifers to complicated, unconfined aquifers with hydraulic conductivities that are represented using correlated random fields corresponding to typical values observed in nature. The CPs provide an opportunity to apply recent advances in subsurface simulators, numerical methods, optimization algorithms, and computing capacities to understand the solutions to these applications better.
We consider two of the CPs, a well-field design problem and a hydraulic capture problem. The applications will be defined in terms of an objective function that measures the cost to install and operate a set of wells and constraints that define the goal of each application. These are described in detail in Section 6.3 below. For both problems, the decision variables are the number of wells, the pumping rates, and well locations.
The objective of the well-field design problem is to supply a specified amount of water while minimizing the cost to install and operate the set of wells. The importance of using a detailed objective is discussed in [225]. We consider the problem in two hydrological settings described in [193]. The first is a homogeneous confined aquifer, while the second is a homogeneous, unconfined aquifer, which leads to additional challenges and complications as pointed out in [166].
The objective of the hydraulic capture application is to prevent an initial contaminant plume from spreading by using wells to control the direction of flow. Several approaches exist to control the migration of a contaminant plume including particle tracking advective control, flow based gradient control, and constraining a target concentration contour [280]. In this thesis, we use the gradient control approach, which uses only flow information since it is the most straightforward to implement and is common in practice [5]. To capture the plume with the gradient control method, we impose constraints on hydraulic head differences at certain points around the plume. The hydrological setting for the hydraulic capture problem is the same homogeneous, unconfined aquifer used in the well-field problem. A detailed description about the groundwater flow model and hydrological settings can be found in [93–95, 194].
To simulate groundwater flow, we use the U.S. Geological Survey code, MODFLOW [122, 196]. MODFLOW is a block-centered finite difference code that is well-supported and

widely used. In this context, MODFLOW is the black box simulator that provides the system state (hydraulic head values) of for a certain system design. At the end of the simulated time horizon, the hydraulic heads at installed wells and at monitoring wells are used to calculate the objective function and the constraints.

6.3 Optimization Problem Formulation

A closer look at the formulation of the underlying design problem reveals that a mixed-integer optimization problem arises naturally. This can be identified not only by the arising total number of wells installed in the system, which is integer, but also in determining which well is to "de-install" in order to reduce the number of wells in the design. Such approaches were earlier proposed in the context of subsurface flow problems by [193, 227, 275].
The used notation and general aspect of the problem formulation are motivated by Mayer et. al. [193], and were also applied earlier by Fowler et. al. [93, 94]. Part of this is that the general optimization problem is formulated as

$$\min_{\boldsymbol{\omega}\in\Omega_{\boldsymbol{g}}} \; f(\boldsymbol{y}(\boldsymbol{\omega}),\boldsymbol{\omega}), \text{ with } \boldsymbol{y}\in\overline{\Omega}_{\boldsymbol{y}}.$$

$\Omega_{\boldsymbol{g}}$ is the feasible domain for the system parameter vector $\boldsymbol{\omega}$, and for $\boldsymbol{y}$ it is defined as $\overline{\Omega}_{\boldsymbol{y}}$. The arising differences depend on the general problem formulation introduced in Chapter 3.

6.3.1 Modeling of the Optimization Problem

First the new mixed-integer formulation that arises quite naturally is discussed, and thereafter the surrogate optimization problems is derived.

Decision Variables

The decision variables are given by the vector of the control parameters $\boldsymbol{\omega}$, but in a different way than they are used in the reference formulation from [93]. The location of each installed well is given by its real-valued x- and y-positions on the domain given by $\boldsymbol{p}_x$ and $\boldsymbol{p}_y$, the operating rate of each well is given by $\boldsymbol{p}_Q$, and the total number of wells in the system is n_s. The last is not included as an optimization variable cause to earlier approaches.
In the reference formulation, the number of potential wells in the design is fixed at its maximum number and the decision of which wells are installed or de-installed out of the set of all wells is directly connected to $\boldsymbol{p}_Q$. We will give more details about the reference approach in the subsequent sections.
We introduce a switching vector $\boldsymbol{s}$, with $\boldsymbol{s}\in\{0,1\}^{n_s}$, that controls whether or not a well is installed and ultimately included in the simulation to stay conform with the numerical experiments that are made in [93, 94]. If we now unite the integer-valued $\boldsymbol{s}$ and the real-valued part $\boldsymbol{p}=(\boldsymbol{p}_x^T,\boldsymbol{p}_y^T,\boldsymbol{p}_Q^T)^T$ to get the variable vector $\boldsymbol{\omega}$, the optimization problem arises as a problem according to

$$\boldsymbol{\omega}=(\boldsymbol{p},\boldsymbol{s}), \text{ with } \Omega_{\boldsymbol{g}}\subset\{\mathbb{R}^{3n_s}\times\{0,1\}^{n_s}\}.$$

The constraints formulated below will define the boundary of the feasible domain for $\boldsymbol{\omega}$ explicitly and implicitly based on the system state $\boldsymbol{y}$.

Objective Function

The definition of the optimization variables given above leads to a modified objective function in comparison to the proposed formulation of [193]. We define the total cost of a system design as a sum of the installation cost, f^c, and the operational cost, f^o, given by

$$f(\boldsymbol{y},\boldsymbol{\omega}) = \underbrace{\sum_{i=1}^{n_s} s_i c_0 d_i^{b_0} + \sum_{p_{Q,i}<0.0} s_i c_1 |1.5 p_{Q,i}|^{b_1} (z_{gs}-h_{min})^{b_2}}_{f^c} + \underbrace{\int_0^{t_f} \left(\sum_{i,p_{Q,i}\leq 0}^{n_s} s_i c_2 Q_i (y_i - z_{gs}) + \sum_{i,p_{Q,i}\geq 0}^{n_s} s_i c_3 p_{Q,i} \right) dt}_{f^o}, \tag{6.1}$$

where the total number of possible wells is given by n_s and the minimal allowed hydraulic head is h_{min}. Note that $\boldsymbol{p}_Q < 0$ m^3/s for extraction wells and $\boldsymbol{p}_Q > 0$ m^3/s for injection wells.
In (6.1), the first term accounts for drilling and installing each well and the second term is an additional cost for an extraction well pump. In f^o the first term accounts for the cost to lift the water to surface elevation while the second term accounts for operation of the injection wells, which are assumed to operate under gravity feed. In Equation (6.1), if $\boldsymbol{s}$ is equal to one in all components then the objective function matches the one in the reference approach and an alternate modeling approach for removing a possible well from the design space must be applied. The values for the constants used in (6.1) are given in Table 6.1. As pointed out above, f can be divided into two parts, one depending explicitly on the optimization variables,

$$f^{ex}(\boldsymbol{\omega}) = f^c + \int_0^{t_f} \left(\sum_{i,p_{Q,i}\geq 0}^{n_s} s_i c_3 p_{Q,i} \right) dt, \tag{6.2}$$

and one depending implicitly on the optimization variables given by,

$$f^{im}(\boldsymbol{y},\boldsymbol{\omega}) = \int_0^{t_f} \left(\sum_{i,Q_i\leq 0}^{n} s_i c_2 Q_i (y_i - z_{gs}) \right) dt. \tag{6.3}$$

Note that f^{im} depends on the simulation output via $\boldsymbol{y}$ and can be highly nonlinear due to the subsurface flow simulation. This term also leads to a noisy, black box-based problem. In our case the relevant system state $\boldsymbol{y}$ is provided by MODFLOW as hydraulic heads at n_s specified locations in the physical domain given by $\boldsymbol{y}$, with $\boldsymbol{y} \in \mathbb{R}^{n_y}$, $n_y = n_s + 2m$. Specifically, $y_i, i = 1, ..., n_s$ are the evaluated hydraulic heads at the position of all installed wells. The hydraulic heads at $2m$ monitoring positions y_i, $i = (n_s+1), ..., (n_s+2m)$, with $m \in \mathbb{N}$, are used to define additional hydraulic gradient constraints for the hydraulic capture application.

Constraints

Constraints are enforced during the optimization so that the wells are located appropriately in the physical domain and are operating at reasonable levels. The location constraints and well capacities are defined by each component by

$$(s_{lb}, p_{x,lb}, p_{y,lb}, p_{Q,lb})^T \leq (s_i, p_{x,i}, p_{y,i}, p_{Q,i}) \leq (s_{ub}, p_{x,ub}, p_{y,ub}, p_{Q,ub}). \tag{6.4}$$

It is also required that two wells cannot be placed at the same grid cell,

$$\min(\max(|\overline{p_{xi}} - \overline{p_{xj}}|, |\overline{p_{y_i}} - \overline{p_{y_j}}|)) \geq s_i s_j \delta, \ \forall \ i, j = 1, ..., n, \ i < j, \tag{6.5}$$

where $\overline{p_{x,i}} = \text{int}(p_{x,i}/\delta)$, $\overline{p_{y,i}} = \text{int}(p_{y,i}/\delta)$ is a grid cell identifier along the x and the y axes of the domain simulated by MODFLOW. The total net pumping rate is also bounded,

$$p_{Q,T,ub} \leq p_{Q,T} = \sum_{i=1}^{n_s} s_i p_{Q,i} \leq p_{Q,T,lb}. \tag{6.6}$$

Besides the explicit constraints, implicit constraints for the system state $\boldsymbol{y}$ arise. The hydraulic head is bounded for both applications by

$$y_{lb} \leq y_i \leq y_{ub}, i = 1, ..., n_s, \tag{6.7}$$

to control the water level in the aquifer. In addition, for the hydraulic capture problem, we bound hydraulic gradients to control the migration of the plume. This is given by

$$y_i - y_{i+m} \geq \Delta, \quad i = n + j, \ j = 1, ..., m. \tag{6.8}$$

where y_i is the hydraulic head inside the plume and $i + m$ is a neighbor just outside the allowed plume boundary. The constraint values are given in Table 6.2. The constraints (6.4) to (6.6) bound the feasible design space $\Omega_{\boldsymbol{g}}$ explicitly, (6.7) and (6.8) implicitly through feasible system states.

6.3.2 Modeling of the Surrogate Problem

The resulting surrogate problem consists of a mathematical part given by the original objective function and an approximated surrogate function replacing the simulation-based part.
The simulation-based components of the problem, $(f^{im}, \boldsymbol{g}^{im\ T})^T$ are approximated by a multivariate surrogate $(\hat{f}^{im}, \hat{\boldsymbol{g}}^{im\ T})^T$ to build the surrogate. The explicit components $(f^{ex}, \boldsymbol{g}^{ex\ T})^T$ stay in the surrogate problem (6.9) as obtained from the original optimization problem. The starting initial values are taken from reference approaches [93, 94] for the sake of comparability. The surrogate optimization approach generates a completely analytically given MINLP problem,

$$\min \hat{f} = f^{ex} + \hat{f}^{im}, \text{ subject to } \hat{\boldsymbol{g}} = \begin{pmatrix} \boldsymbol{g}^{ex} \\ \hat{\boldsymbol{g}}^{im} \end{pmatrix} \leq 0. \tag{6.9}$$

Table 6.1: Constants included in the objective function f.

constant	well-field confined	well-field unconfined	hydraulic capturing	units
b_0	0.3	0.3	0.3	-
b_1	0.45	0.45	0.45	-
b_2	0.64	0.64	0.64	-
c_0	5.5×10^3	5.5×10^3	5.5×10^3	$\$/\mathrm{m}^{b_0}$
c_1	5.75×10^3	5.75×10^3	5.75×10^3	$\$/[(\mathrm{m}^3/\mathrm{s})^{b_1} \cdot \mathrm{m}^{b_2}]$
c_2	2.9×10^{-4}	2.9×10^{-4}	2.9×10^{-4}	$\$/\mathrm{m}^4$
c_3	1.45×10^{-4}	1.45×10^{-4}	1.45×10^{-4}	$\$/\mathrm{m}^3$
d_i	60	30	30	m
z_{gs}	60	30	30	m
t_f	5	5	5	years
y_{lb}	40	10	10	m

$\boldsymbol{\omega}$ is the optimization variable and is defined on a domain just bounded by the box-constraint (6.4). For all computations, the variable domain was scaled in all dimension to an interval from 0 to 1, to avoid influence because of different sizes and ranges of the variables.

6.4 Reference Approaches

The reference problem formulation was introduced to solve the optimization problem because mixed-integer black box optimization methods were not existing. We do now briefly describe results of earlier published modeling approaches because we will compare our results in with them [93, 94, 126, 131].

6.4.1 Implicit Filtering applied to Alternative Formulations

Implicit filtering is used for the penalty coefficient approach as well as for the inactive-well threshold approach. Here, we use IFFCO, with the symmetric rank one quasi-Newton update [46]. We used the default optimization parameter settings. IFFCO has been successfully applied to other groundwater management problems [21, 93, 94].
The following two optimization formulations use the objective function directly as defined in (6.1), but minimization is performed according only to the r optimization variables. The constraints are included by a penalty function value f_{pen}, where f_{pen} is the function value of the initial design set plus 20%. If a design set is not feasible according to the explicit constraints (6.5) or (6.6) no simulation run is obtained. The two approaches differ in how values for $\boldsymbol{s}$ are assigned.

Table 6.2: Additional constants in use for the constraints.

constant	well-field confined	well-field unconfined	hydraulic capturing	units
s_{lb}	0	0	0	-
s_{ub}	1	1	1	-
$p_{x,lb}$	0	0	0	m
$p_{x,ub}$	800	800	800	m
$p_{y,lb}$	0	0	0	m
$p_{y,ub}$	800	800	800	m
$p_{Q,lb}$	-6.4×10^{-3}	-6.4×10^{-3}	-6.4×10^{-3}	m^3/s
$p_{Q,ub}$	6.4×10^{-3}	6.4×10^{-3}	6.4×10^{-3}	m^3/s
$p_{Q,T,lb}$	-3.2×10^{-2}	-3.2×10^{-2}	n.d.	m^3/s
$p_{Q,T,ub}$	n.d.	n.d.	-3.2×10^{-2}	m^3/s
δ	20	20	10	m
y_{ub}	60	30	30	m
Δ	n.d.	n.d.	10^{-4}	m/s
m	n.d.	n.d.	5	-

Inactive-Well Threshold

The reference approach from [93] to determine the number of wells in the design is to set an inactive-well threshold. If a well rate becomes low enough, the well is removed from the design space thereby avoiding any integer variables,

$$s_i = \begin{cases} 0 & \text{for } |p_{Q,i}| < 10^{-6}\text{m}^3/\text{s}, \\ 1 & \text{otherwise.} \end{cases} \tag{6.10}$$

This means that the well rate is set to zero and well i is not included in the installation cost. Incorporating (6.10) leads to large discontinuities in the minimization landscape and a drastic decrease in cost once the well rate falls into this region of the design space, but $\boldsymbol{s}$ is removed from the optimization problem and only continuous variables have to be taken into account.

Multiplicative Penalty Coefficients

To reformulate a mixed-integer water management application as an NLP, the authors of [197] introduce a polynomial penalty coefficient method. Here the penalty coefficient β is given by

$$\beta_i = p_{Q,i}/(p_{Q,i} + \xi), \tag{6.11}$$

where $0 < \xi << 1$ is a small number. This penalty term is then multiplied by the fixed costs for each well in f^c. Note that in (6.11), if $p_{Q,i} = 0$, then $\beta_i = 0$ and the fixed cost for well i does not contribute to the objective function at all. We used $\xi = 10^{-6}$.

Table 6.3: GA parameters for simulations.

30	size of population	20	distribution index for real-coded crossover
30	max number of generations	10	distribution index for real-coded mutation
0.9	crossover probability	0.5	binary-coded mutation probability
0.1	real-coded mutation probability		

6.4.2 A Genetic Algorithm

For comparison purposes, we also use a GA for both applications, since GAs are popular derivative-free approaches for black box optimization problems. We use the non-dominated sorting GA called NSGA-II here. The NSGA-II has performed well in comparisons to a number of other GAs for multi-objective optimization problems [60, 285]. As its name suggests, NSGA-II is a multi-objective GA based on non-dominated sorting [60] that includes elitism, does not require a sharing parameter for maintaining solution diversity in multi-objective problems, and can account for multiple constraints in a straightforward manner without the use of penalty parameters [56, 58]. Like most multi-objective GAs, NSGA-II uses a binary tournament operator for selection. It includes crossover and mutation operators for both real and binary-coded variables and uses simulated binary crossover for real-coded problems [57, 60].
Parameters like the population size, number of generations, as well as the probabilities and distribution indexes chosen for the crossover and mutation operators effect the performance of a GA [193, 223]. The population size of 30 was the lower bound of the suggested range, while a maximum of 30 generations were allowed. The crossover and mutation operator parameters were chosen based on the performance of NSGA-II for a multi-objective test problem with several local pareto-optimal fronts [59]. A limited number of experiments with other crossover and mutation operator parameter settings were performed, but no combination were found that gave better performance across the test problems considered here. The values used are listed in Table 6.3.
An approach based on [58, 59] is used to include constraints without the use of penalty parameters. Box constraints such as those in equation (6.4) are enforced automatically in the generation of candidate design variables, while a constraint such as those in equations (6.5) to (6.8) is formulated as a non-negative function $\boldsymbol{g}(\boldsymbol{y}(\boldsymbol{\omega}), \boldsymbol{\omega}) \geq 0$. The GA tournament selection process is then modified to account for the three scenarios: 1. when two feasible solutions are compared, the one with lower objective value is preferred; 2. when a feasible and infeasible solution are compared, the feasible one is taken; and 3. when two infeasible solutions are compared the one with lower overall constraint violation is preferred [59]. To be concrete, (6.6) becomes

$$g_1 = \left(p_{Q,T,lb} - \sum_{i=1}^{n_s} p_{Q,i} \right) / |p_{Q,T,lb}| \tag{6.12}$$

for the well-field problems, and

$$g_1 = \left(\sum_{i=1}^{n_s} p_{Q,i} - p_{Q,T,ub} \right) / |p_{Q,T,ub}| \tag{6.13}$$

for the hydraulic capture problem, respectively. Constraint (6.5) is translated to

$$g_2 = -1 + 2 \min_{i,j \neq i} \left(\max \left(|\bar{p}_{x,i} - \bar{p}_{x,j}|, |\bar{p}_{y,i} - \bar{p}_{y,j}| \right) \right).$$

The implicit constraints (6.7) and (6.8) are included as

$$g_3 = \left[\sum_{i=1}^{n_s} \min(y_i - y_{lb}, 0) + \sum_{i=1}^{n_s} \min(y_{ub} - y_i, 0) \right] / |y_{ub}|$$

and

$$g_4 = \left[\sum_{i=n_s+1}^{n+m} \min(y_i - y_{i+M} - \Delta, 0.) \right] / |h^{max}|.$$

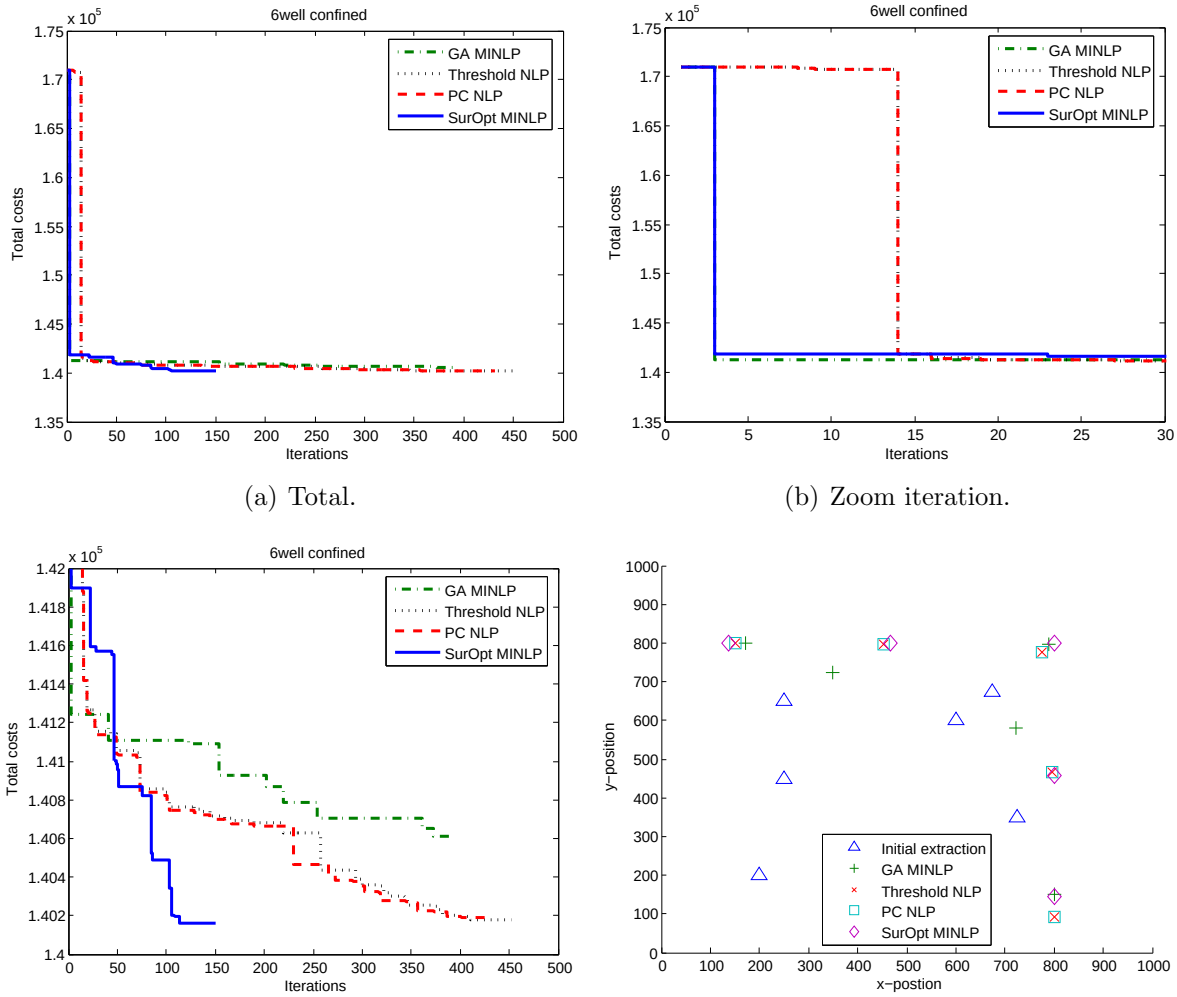
In the calculations, the GA uses $\sum_i g_i$ to determine the overall constraint violation. If a design variable $\boldsymbol{\omega}$ violates constraint (6.12) or (6.13), there is no need to run the groundwater flow simulation, and a value $g_3 = 0$ is assigned.
The GA formulation also differs in its treatment of the integer variable s in some respect. For the hydraulic capture problem, a switch $s_i \in \{0, 1\}$ is used to determine if well i was active or not. In the well-field design problem, the active-inactive well information in s is collapsed into a single integer variable, $\tilde{s}$ in the range $1, \ldots, \tilde{s}_{ub}$, with $\tilde{s}, \tilde{s}_{ub} \in \mathbb{N}$. A value of $\tilde{s} \in 1, \ldots, n_s$ corresponds to shutting off the associated well, while a value greater than n_s means all wells are active. In the numerical experiments below, $\tilde{s}_{ub}$ is set to 8 for $n_s = 6$.

6.5 Newly Obtained Best Designs

All applied approaches provide feasible solutions with expected characteristics in the numerical experiments. The number of active variables is bounded by the number of maximal installed wells, which is four for the hydraulic capturing problem, and six for both well-field design problem. The unequal ranges associated with five and six well designs skews the GA's formulation to favor five-well designs. This reflects a heuristic that installing

Table 6.4: Final objective function values and obtained simulation calls in brackets.

problem formulation	optimization method	well-field confined	well-field unconfined	hydraulic capturing
Initial	-	\$170,972	\$152,878	\$80,211
MINLP	NSGA-II	\$140,610 (391)	\$125,226 (273)	\$24,854 (659)
Threshold/NLP	IFFCO	\$140,175 (362)	\$124,527 (320)	\$24,032 (363)
PC/NLP	IFFCO	\$140,190 (402)	\$124,512 (316)	\$23,640 (580)
MINLP	SurOpt	\$140,159 (113)	\$124,387 (87)	\$23,491 (22)



(a) Total.

(b) Zoom iteration.

(c) Zoom improvement.

(d) Initial and final well positions.

Figure 6.1: Objective function reduction versus MODFLOW calls for the confined well-field problem and a comparison of the final positions.

the minimum number of wells is likely to be cheaper given the relative magnitudes of the installation and operational costs. In addition, the designs for both problems were also subject to the inactive-well threshold given in (6.10), regardless of the values of $\boldsymbol{s}$. It follows for the well-field problems a number of non-box constraints of n_g is 14, for the hydraulic capture problem n_g is 15.

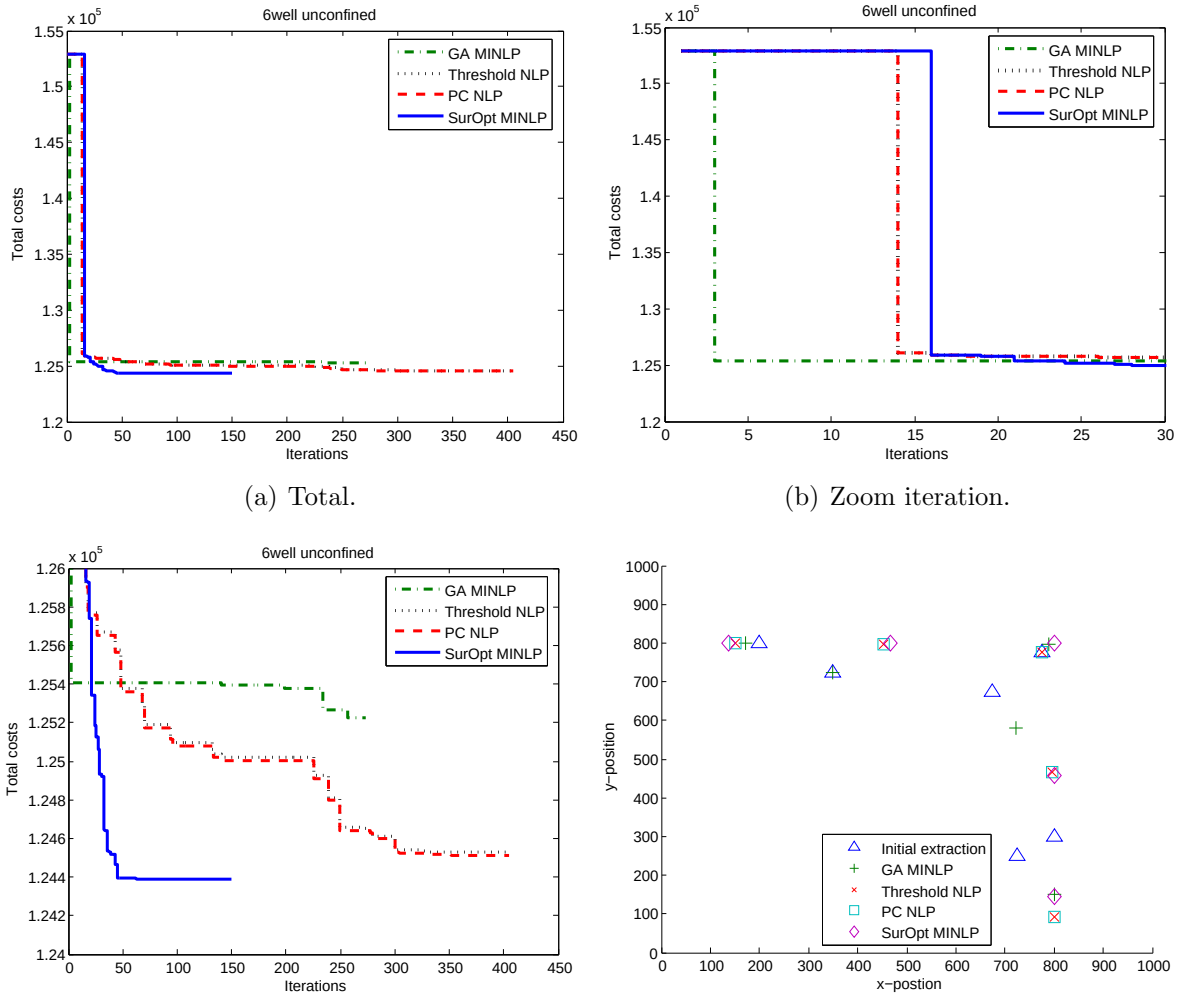
6.5.1 Comparison of the Applied Approaches

The number of installed wells is reduced by all applied approaches to the minimum possible number and the results are comparable for all examples to those found in the literature. The solutions differ only in the real-valued variables which are of minor influence for the total objective function value. Table 6.4 shows the cost of the well design at the initial iterate and at the final design for each optimization model and for each application. The number of calls to the simulator is also used to measure performance and is in parentheses.

The surrogate optimization approach determined system designs using the least number of simulation calls for all considered benchmark examples. However, the rising number of points for interpolation and the computational time during each iteration should be considered. We also observed that even with more than the applied 150 simulation calls as the stopping criteria, an improved system design was not found for any of the problems.
Two interesting observations can be made with the penalty coefficient and the threshold problem formulation for the well-field design problems as solved with IFFCO. The original penalty formulation of McKinney and Lin [197] is not able to handle positive and negative pumping rates for a well, because the penalty coefficient becomes negative and drives the iterates of the optimization into a bad direction. IFFCO had nearly the same convergence pattern for both formulations because the stencil landed on an exactly zero well rate. Secondly, if the bounds on the pumping rate are varied so that 0 is not the middle of the range of possible pumping rates, the PC combined with IFFCO, is not able to find a solution where a well is de-installed from the design, in contrast to the threshold method which turns a well off. Additionally we tried to apply the pseudo integer approach from [197], but after extensive parameter testing not a single optimization run could be performed that lead to a decrease of the objective function. In the reference pseudo integer approach of McKinney only switches for a bigger number of fixed wells were included as decision variables, so that the resulting optimization problem is of a different characteristic than the ones considered here.
Feasibility is a challenge for these applications and adds complexity in choosing an efficient optimization algorithm. During all tested optimization approaches a quota of system design of infeasible system state are simulated. The explicit inclusion of the constraints of the flow direction by surrogate functions could be a second point, besides the direct handling of integer variables to explain the least number of simulation runs the third approach requires. As mentioned above we only provide numerical results for one initial system design, which was determined from an engineering perspective.
However, we should note that the results of each optimizer are sensitive to both the initial design and the technique for handling infeasible points due to constraint violation.

Table 6.5: Solutions for confined well-field problem.

Location (meters)	Initial	NSGA-II MINLP	IFFCO Threshold	IFFCO PC	SurOpt MINLP
$p_{x,1}$, $p_{y,1}$	350, 725	366.0, 791.8	341.0, 798.1	350.0, 798.1	135.6, 800
$p_{x,2}$, $p_{y,2}$	775, 775	788.6, 799.0	799.4, 775.0	799.4, 775.0	800, 22.7
$p_{x,3}$, $p_{y,3}$	675, 675	769.6, 676.5	656.6, 794.6	668.9, 772.5	399.8, 800
$p_{x,4}$, $p_{y,4}$	200, 200	181.1, 376.4	102.5, 797.2	102.5, 797.2	800, 20
$p_{x,5}$, $p_{y,5}$	725, 350	788.3, 302.6	792.0, 300.0	792.0, 300.0	800, 800
$p_{x,6}$, $p_{y,6}$	600, 600	674.4, 602.3	600.0, 600.0	600.0, 600.0	800, 567.7
Integer decision	-	$\tilde{s} = 5$	$q_6 = 0$	$q_6 = 0$	$s_2 = 0$



(a) Total.

(b) Zoom iteration.

(c) Zoom improvement.

(d) Initial and final well positions.

Figure 6.2: Objective function reduction versus MODFLOW calls for the unconfined well-field problem and a comparison of the final positions.

6.5.2 Well-Field Design Problem

For the well-field design problem all methods found solution where the number of wells is reduced to five for both hydrological settings. To fulfill the constraints on the total extraction rate (6.6), $Q = -0.0064m^3/s$ for each of the five remaining wells.
In Figure 6.1 and Figure 6.2 we see that all optimization methods reduce the objective function value by the installation costs of one well in less than 20 optimization iterations. Afterward as is given in the subfigures, the rate of improvement varies for the different approaches. The characteristic is similar for both well-field problems. Figures 6.1 d and 6.2 d illustrate the final well designs for each approach for the confined and unconfined aquifers. The final positions of the wells are nearly the same with the exception of the wells from the GA solution. Tables 6.5 and 6.6 give the specific points found and explains why the total costs only barely differ.

Table 6.6: Solutions for the unconfined well-field problem.

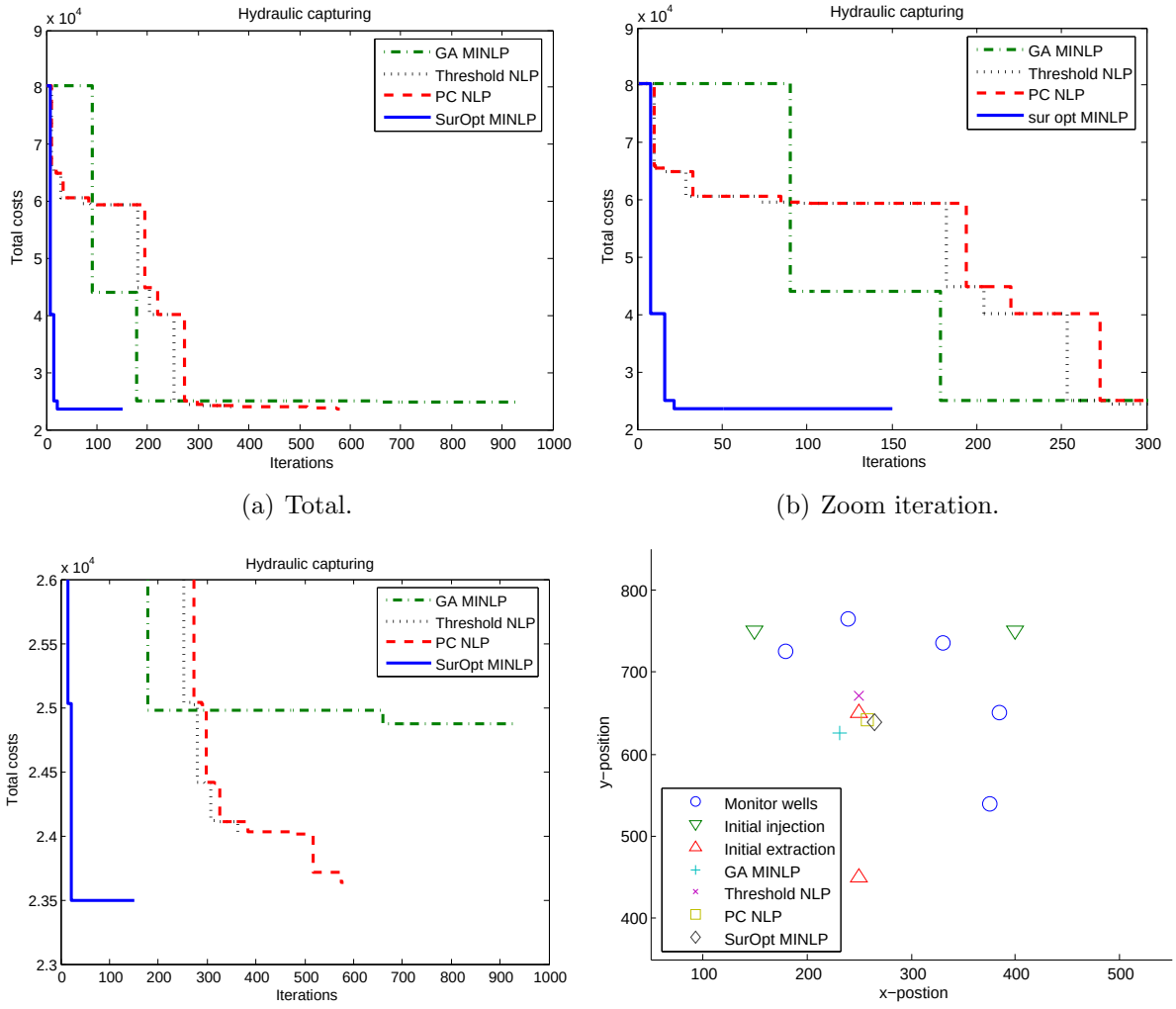
Location (meters)	Initial	NSGA-II MINLP	IFFCO Threshold	IFFCO PC	SurOpt MINLP
$p_{x,1}$, $p_{y,1}$	350, 725	350.1, 725.0	453.6, 798.1	453.6, 798.1	466.9, 800.0
$p_{x,2}$, $p_{y,2}$	775, 775	788.3, 797.1	775.0, 775.0	775.0, 775.0	800.0, 800.0
$p_{x,3}$, $p_{y,3}$	675, 675	722.2, 579.7	796.9, 467.8	796.9, 467.8	800, 456.2
$p_{x,4}$, $p_{y,4}$	200, 800	170.8, 800.0	151.3, 800.0	151.3, 800.0	136.7, 800
$p_{x,5}$, $p_{y,5}$	725, 250	710.0, 324.2	725.0, 250.0	725.0, 250.0	800, 144.2
$p_{x,6}$, $p_{y,6}$	800, 300	800.0, 152.0	800.0, 92.8	800.0, 92.8	800, 143.7
Integer decision		$\tilde{s} = 6$, $p_{Q,5} \leq 10^{-6}$	$p_{Q,5} \leq eps$	$p_{Q,5} = 0$	$s_6 = 0$

Table 6.7: Solutions for hydraulic capturing problem.

Location (meters)	Initial	NSGA-II MINLP	IFFCO Threshold	IFFCO PC	SurOpt MINLP
$p_{x,1}$, $p_{y,1}$	150, 750	183.5, 342.2	165.3, 750.0	165.3, 750.0	10.0, 746,8
$p_{x,2}$, $p_{y,2}$	400, 750	143.3, 528.8	400.0, 750.0	400.0, 750.0	411.9, 717.4
$p_{x,3}$, $p_{y,3}$	250, 650	272.2, 652.2	250.0, 671.0	257.7, 642.3	264.3, 638.9
$p_{x,4}$, $p_{y,4}$	250, 450	231.5, 626.1	250.0, 450.0	250.0, 450.0	266.9, 413,9
Well rates (m^3/s)					
$p_{Q,1}$, s_1	6.4E-3, 1	6.4E-3, 0	0.0, -	0.0, -	6.4E-3, 0
$p_{Q,2}$, s_2	6.4E-3, 1	-3.362E-3, 0	0.0, -	0.0, -	6.2969E-3, 0
$p_{Q,3}$, s_3	-6.4E-3, 1	-6.185E-3, 0	-5.7499E-3, -	-5.4999E-3, -	-5.3984E-3, 1
$p_{Q,4}$, s_4	-6.4E-3, 1	-6.306E-3, 1	0.0, -	0.0, -	-0.0064, 0

6.5.3 Hydraulic Capture Problem

The hydraulic capture example is the most challenging of this set of problems. The unconfined hydrological setting is sensitive to the positions of the wells with respect to the drawdown constraint (6.7) and the additional capture constraint on the flow direction at the monitoring positions. The capture constraints are particularly sensitive to variations of the well positions within the catchment area, especially if only one well is placed on the whole domain. With only one well in the design, if the well moves outside the capture zone, the head gradient constraints are violated. Feasibility is a main difficulty to overcome, even in generating an initial feasible solutions to start the optimization. As in the two well-field problems the solution with the different optimization approaches are clustered around one location, but where found in one order of magnitude less simulations evaluations than other approaches.



(a) Total.

(b) Zoom iteration.

(c) Zoom improvement.

(d) Initial and final well positions.

Figure 6.3: Objective function reduction versus MODFLOW calls for the hydraulic capture problem and a comparison of the final positions.

6.6 Summary of Chapter 6

A first conclusion drawn from this chapter is that the modeling of an analytical formulation is important according to (i) the properties of the problem that has to be solved and (ii) the optimization methods that can be applied. Another point in this context is that if the initial number of candidate wells n_s is large, the complexity of the integer part of the optimization problem increases if the optimal number of wells is significantly smaller than n_s. Note that as the dimension of the continuous-valued variables of the problem gets bigger it will take more simulation calls for a DFO method to succeed.
Besides the formulation and modeling, new optimal designs for this set of benchmark problems were found and each applied optimization method has returned an acceptable system design of equal characteristic. The only difference between solutions is in the continuous-valued problem variables. However, despite the small differences in the final

costs, the methods' iterations histories differed significantly in the number of function evaluations required for convergence. Although for these problem formulations we are only using flow information, optimization with fewer function calls is even more attractive when more sophisticated physical models and computationally expensive simulators must be used. The benchmark set provides not only a framework for optimization on subsurface flow problems, but also serves as a set of problems for benchmarking any optimization method for black box-based problems.

7 Applications in Robotics and Control

The speed and postural stability of dynamically walking robots is a critical factor in many applications, especially in autonomous robot soccer games. Walking of humanoid as well as 4-legged robots is modeled by high-dimensional nonlinear dynamic multi-body systems (MBS) with changing contact situations (impacts) and an underlying control of the joint motors [228]. Many different approaches have already been investigated for improving the walking speed of bipedal and quadrupedal robots. For example, numerical optimal control techniques enable the computation of stable and fast walking motions based on three-dimensional computational multi-body-dynamics models of the legged robot dynamics [38, 258].
However, all model-based optimization approaches have in common that their outcome critically depends on the quality and accuracy of the robot model [284]. The derivation of highly accurate enough robot models to achieve the best possible walking speed may require too many efforts considering, e.g., the effects of gear backlash, elasticity and temperature dependent joint friction or of different ground properties. With a reasonable effort a MBS dynamics simulation of a humanoid robot can only achieve an error of about 5 to 10 % compared to the real system. Also a lot of additional work has to be spent in the transformation of results from the simulated model to the real system before they can be used. Furthermore, robot prototypes with the identical technical design differ significantly in their motion even if the same motion control software is used. This is due to inevitably small differences in the many robot components. For this reason, the final improvement can only be obtained by working on the real robot.

7.1 Directly Coupled System: Hardware in the Loop Optimization

An alternative approach to the optimization of walking motions based on detailed MBS dynamics simulations is to start with a reasonable, initial walking motion and then to use online hardware in the loop optimization of the physical robot prototype. This leads to the question which optimization methods can be applied to find a good walking motion with reasonable efforts.
Besides the overall accuracy of MBS simulations it is still a question how to formulate an optimization problem for bipedal respecting the issue of instability and other not measurable properties of walking motions like slipping, falling, or drifting from an aspired direction.
Here, all walking optimization experiments are made under the assumption, that the used robot carries enough energy supply itself to perform the found walking motions in permanent operations during a robot-soccer-game of two times 10 minutes as required.

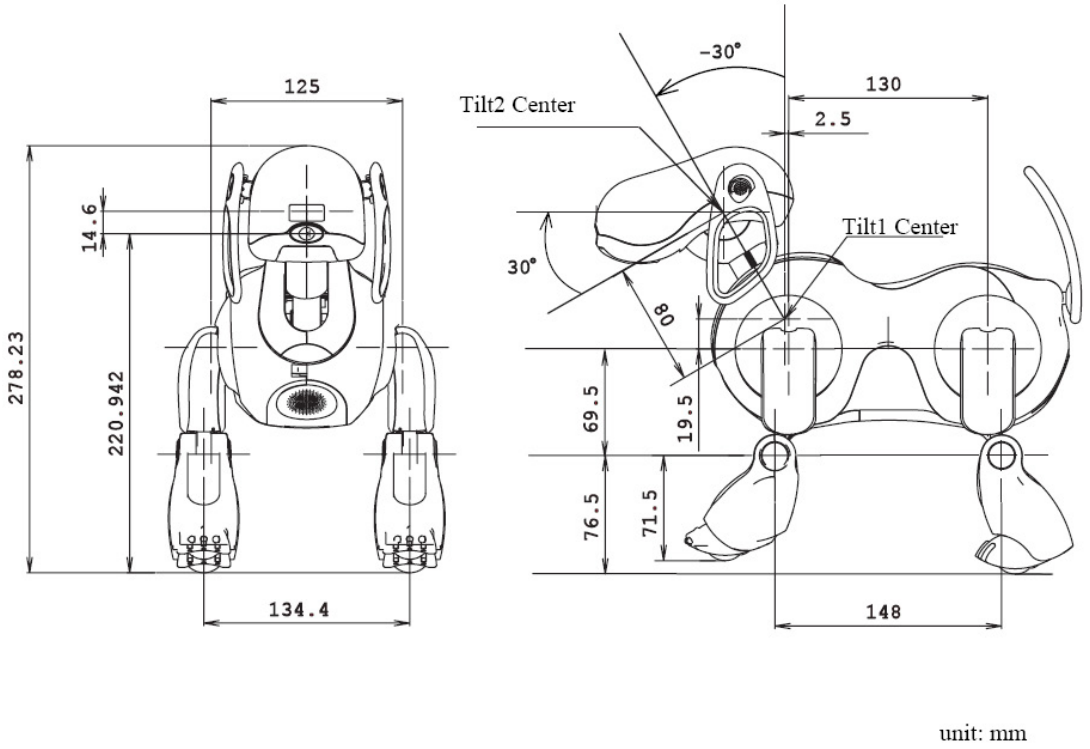


Figure 7.1: Sony Aibo OPEN-R SDK, Model Information for model ERS-7 [255].

7.2 Four-Legged Sony Aibos

Our aim was to increase the walking speed of robots of type Sony Aibo in two scenarios, (1) straight forward walking without ball as done in [70], (2) turning around (its vertical axis) without losing the ball, whereby the ball is placed in between the front legs of the robot, and it has to stay there during the turning motion. These two cases often arise during a robot soccer game, the first when the robot has to reach the ball, the second one when it is aligning for passing the ball to another robot, or when it tries to shoot a goal. The walking trajectories of the four legs are determined in space and time by a 31-dimensional parameter vector.

7.2.1 Applied Turning and Walking Speed Optimization Setup

This low-dimensional parametrization is achieved by approximating the trajectory by polygons and by taking advantage of symmetry and redundancies of a four legged walking robot. The walking optimization is performed on the standard RoboCup four-legged league field, where the walking or turning speed of the robot is measured by a ceiling camera (cf. Fig. 7.2). The position and orientation of the robot is detected using circular markers attached to the robot's back. The speed of the robot is then approximated from two consecutive measurements over a fixed time interval. The average speed measured over a fixed number of runs is returned to the optimizer as the objective function value. When optimizing turning while holding the ball, a distance sensor in the chest of the robot is used in order to determine whether the ball was lost while walking. For such an unsuccessful

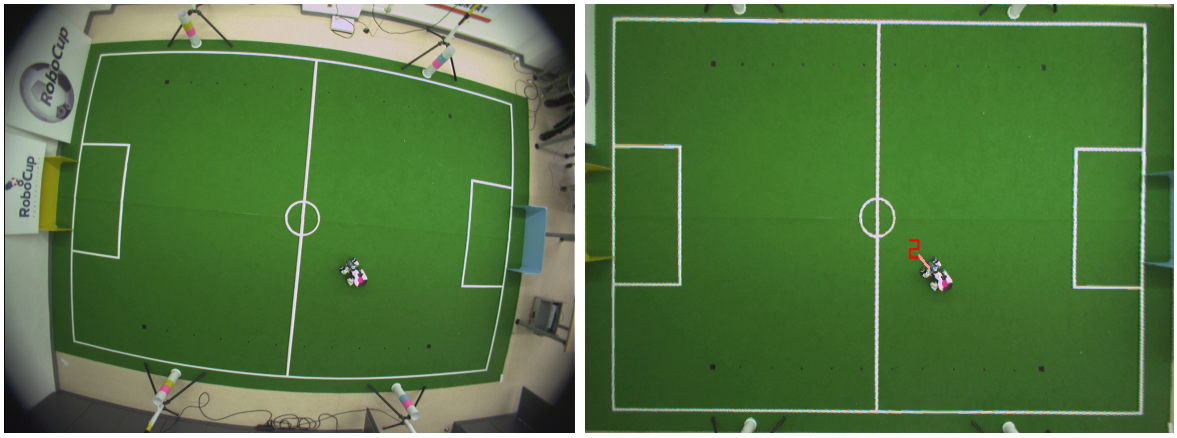


Figure 7.2: Ceiling camera output and the revised version for measurements.

run, an evaluation value of zero is returned to the optimizer. Therefore, walking motions are penalized where the ball is not held securely.
This way a hardware in the loop layout is applied, where the robot control is coupled with the optimization tool. In our case APPSPACK in version 4.0.2 [115] is used, which is an implementation of an asynchronous parallel pattern search as described in Chapter 2. It is originally written for deterministic nonlinear optimization problems, but for the here considered problem with averaged walking speed and turning speed, respectively, the chosen optimization method performed well. Besides, the restricting factor in this application is the number of walking experiments, so that no implemented stopping criteria of APPSPACK is used, which would also be disturbed by the underlying stochastic character of this application.
The starting parameter set for the forward walking speed optimization was obtained by an optimization experiment with an EA, for the turning speed optimization we start with a hand tuned initial parameter set. Besides bounds on the variable space representing reachable leg positions, we are using the default parameters of the optimizer.

7.2.2 Obtained Experimental Results

The measured forward walking speed of the initial parameter set was 40 cm/sec, the resulting parameters provided an average speed of 43 cm/sec. Thus, yielding an improvement in the walking speed of about 7.5%. This result was achieved after evaluating 83 parameter sets in about half an hour optimization time. When optimizing the turning motion of the robot, the turning speed is increased about 50% from 120 deg/sec to 180 deg/sec without losing the ball while turning. This result required 206 black box evaluations in terms of robot experiments in the lab, that took not more than 45 minutes.

7.3 Humanoid Robot Prototype Bruno

The following example describes a successful approach undertaken for obtaining the fastest walking humanoid robot in the humanoid robot league of RoboCup 2006 based on [128, 132]. The development of optimized motions of humanoid robots that guarantee a fast and

also stable walking is an important task especially in the context of autonomous soccer playing robots in RoboCup.
This work is done with the humanoid robot prototype Bruno which is equipped with a low-dimensional parameterized walking trajectory generator, joint motor controller and an internal stabilization. The hardware and software components of Bruno will be described, and based there upon the optimization problem by objective function, variable domain, and the implementation details of the applied surrogate optimization method are introduced, as well as the experimental setup and finally the results obtained from the online optimization. Depending on the problem modeling, the described optimization approach could be applied also to other types of humanoid robot locomotion like turning or walking sidewards as well. In contrast to previously performed walking optimization approaches we apply a sequential surrogate optimization approach using stochastic approximation of the underlying objective function and SQP to search for a fast and stable walking motion. This is done under the conditions that only a small number of physical walking experiments should have to be carried out during the online optimization process. For the identified walking motion trajectories for the considered 55 cm tall humanoid robot we measured a forward walking speed of more than 30 cm/sec. With a modified version of the robot even more than 40 cm/sec could be achieved in permanent operation.

7.3.1 Humanoid Locomotion

Walking of humanoid robots is much more difficult than with four-legged robots because during walking there must be phases where only one foot (or even no foot at all) is in contact with ground. Promising results for a hardware in the loop scenarios are presented in [276], where a simplex approach is applied successfully to improve the walking speed of a six-legged robot. Therefore, stability control is much more an issue than with four- and more legged robots. Sensors are needed to detect instabilities in a very early phase. Sophisticated software must evaluate the sensors and modify the motion if needed to guarantee stability.

Humanoid Robots

The legs of all current humanoid robots which are able to reliably perform a variety of different walking motions in experiments (as Honda ASIMO, HRP-2, Johnnie or Sony QRIO) consist of rigid kinematic chains with 6 or 7 revolute joints using electrical motors of high performance and with rigid gears for rotary joint actuation. Only few initial approaches exist to insert and exploit elasticities in humanoid robot locomotion [239]. Commonly servo motors are used in low- and medium-cost humanoid robots. However, there exist specialized joint motors, gears and controllers designed for high-cost humanoid robots (e.g. ASIMO, QRIO). A large variety of humanoid robots is involved in the Humanoid Robot League of the RoboCup competitions.

Walking Optimization

Walking optimization generally may be done in two different ways: on computational models or on the real robot. Optimization on computational models using optimal control techniques for both four-legged and humanoid robots needs careful adaption of the

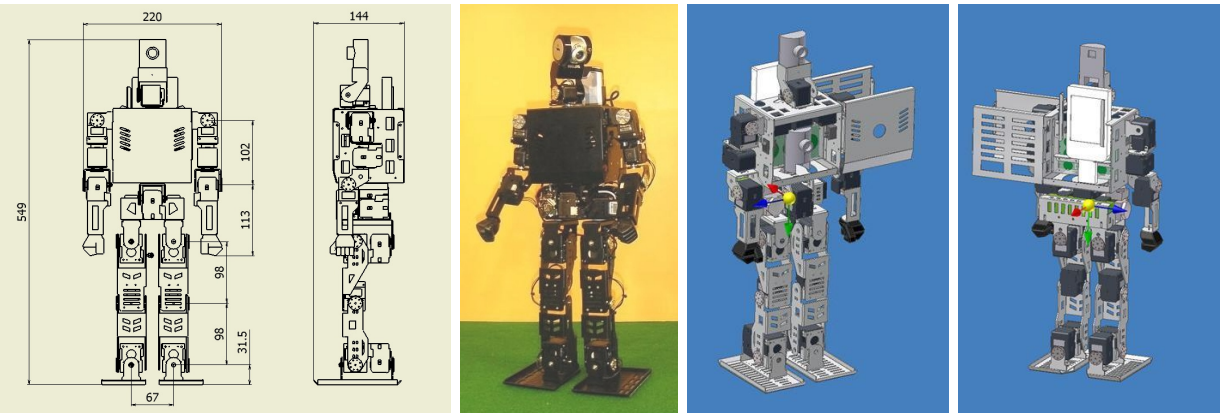


Figure 7.3: The humanoid robot prototype Bruno.

model to the real robot and successive refinements of the model so that the computed trajectories may be implemented to the real robot [38, 62, 123, 258, 278]. The iterations of the optimization procedure however can be done without human assistance. Stability criteria may be used in the optimization; therefore, methods that proved to be useful for four-legged robots may be used for humanoid robots as well. Besides only considering parameterized walking motion trajectories, in [215] additionally construction issues are also included as free parameters to be optimized, here, the centers of mass of the links. During a pre-construction phase a lot more design parameters would be interesting to consider. When optimization is done on the real robot, methods used for four and more legged robots may not directly be used for humanoid robot as instabilities may lead to severe damage of the robot. E.g. if the robot repeatedly falls down during automated walking experiments for optimization. Therefore, special approaches must be chosen. For four-legged robots, approaches based on parametrization of the walking motion and optimization of the parameters by evaluation of the walking speed have been used successfully [125, 160, 229]. For biped robots, stability is much more critical than for four-legged robots. However, more walking optimization approaches special adapted can be found [85, 185, 205, 261].

7.3.2 Design Details of the Robot Prototype

The autonomous humanoid robot prototype Bruno ($<$ 4 kg weight, 55 cm height, cf. Fig. 7.3) was developed by a cooperation of the Hajime Research Institute and TU Darmstadt.

Hardware Selection and Design

Bruno consisted during the first undertaken experiments of 24 actuated rotational joints, two cameras and onboard computing; 14 joints are the most relevant ones for walking: six in each leg in the standard configuration for six-DOF humanoid robot legs (cf. Fig. 7.4) and two in the waist for forward/backward (pitch) and left/right turning motion of the waist. Motion of the four-DOF arms may be used for stabilization of locomotion. The head joint with the articulated camera has two-DOF.

Metal frames for links and body are only used where really needed, i.e. mainly to connect the motors and to protect the controller boards from impacts in case of falling down. For all joints Robotis DX-117 servo motors are used which are operated at 14.8 V, have a maximum speed of 0.138 sec/60° and a maximum torque of 33.4 kg-cm. Besides controlling the position of the joints with adjustable control parameters, the servo motors are able to monitor their operation environment, e.g. temperature of the motor or voltage of the power supply. The servos are connected via a RS485 bus to a controller board equipped with a Renesas SH7145 32bit microcontroller running at 50 MHz and 1 MByte of RAM. This controller board is used for real-time motion generation. Every 10 ms new desired positions are generated and sent to the servos. Furthermore, the controller is used to gather and evaluate data from the inertial sensors of the robot. The controller is connected to the main CPU of the system using a RS232 connection running at 57.6 kbits/s.

Lithium polymer rechargeable battery packages are used for onboard power supply because of their good ratio between mass and capacity, a four-cell 14.8 volt battery for actuation of the motors, and a two-cell battery for the controller board. According to the design rules of the RoboCup Humanoid Robot League the foot has a rectangular size of 125 mm times 90 mm. It is a flat one without any ground contact sensors. Depending on the surface the robot shall walk upon, different materials like rubber or paper may be added to the ground contact area of the feet to avoid or give rise to a favored level of friction properties. The robot is equipped with three 1-axis gyroscopes and one three-axes accelerometer, all working with 100 Hz and attached in the hip. They are used to stabilize the walking motion by correction motions of the arms and the legs.

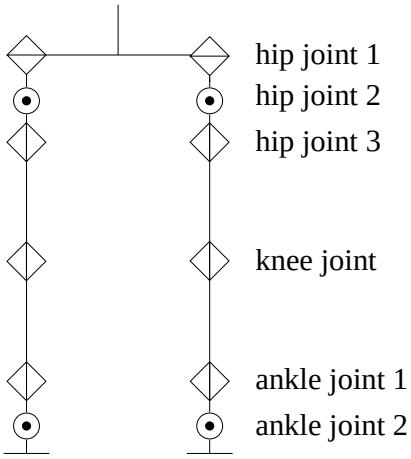


Figure 7.4: Kinematic chain of the legs, axis of hip joint 1 and 2 intersect, the axis of hip joint 2 and 3, as well as the axis of ankle joint 1 and ankle joint 2 in each leg.

For the needs of acting in a highly dynamically environment like it is arising for autonomous soccer games in RoboCup, Bruno has been equipped at TU Darmstadt with two off-the-shelf webcams and a pocket PC to enable autonomous soccer playing.

High- and Low-Level Software

Two levels of software are used on the robot. On a pocket PC the high-level software is running. It consists of a framework [98] that coordinates the control architecture with sensor data processing, self localization, world modeling, hierarchical finite state machine for behavior control [179], and finally the motion generation and requests. The motion requests are sent via the described serial port to the controller board.

The low-level software on this board distinguishes between two different kinds of motions: Walking motions and so-called special motions. The first ones are parameterized as described in the next section, and the latter ones are teach-in-motions that are stored by joint angle trajectories. All motions are PD-controlled on joint angle level by the servo-motors.

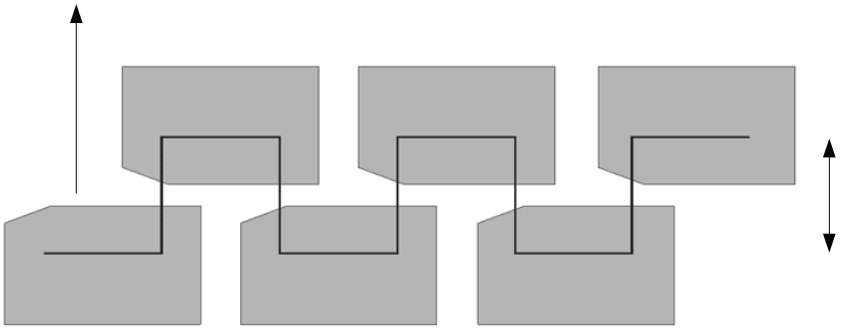


Figure 7.5: Footprint of the walking motion.

The inertial sensor values are used to reduce vibration of unbalance during walking or other motions with a PD controller. The vibration is caused by a fast, but simplified calculation of the zero moment point (ZMP) or by uneven terrain. The PD control is useful to stabilize especially during the fast walking of the robot. It is calculated by

$$\theta_{new} = \theta + K_p * \omega_{gyro} + K_d * \frac{d}{dt}\omega_{gyro}$$

for the joint motors of the foot pitch, the foot roll, the hip pitch, the hip roll, the waist pitch, the shoulder pitch, and finally the shoulder roll, herein described as θ for the angle calculated by inverse kinematics and θ_{new} the controlled angle; K_p and K_d are the PD control parameter and ω_{gyro} is the angular velocity of the gyro. The values for K_p and K_d are identified by expert knowledge and experiments. Direct access to the memory of the robot exists, i.e. all parameters may be changed during run time which allows easy alteration of walking or sensory parameters and debugging.

Parametrization of Humanoid Walking Trajectories

Like any motion, walking motion may be described by joint angle trajectories. These however are infinite-dimensional and therefore hard to handle. In contrast to special motions like kicking the ball, arm waving, getting down or standing up, walking is a periodical and comparatively smooth motion. Therefore, the walking motion may be generated by prescribing trajectories for the hip and the feet and solving the inverse kinematics for the joint angles. The inverse kinematics model can be calculated analytically due to the special kinematic structure of the leg (cf. Fig. 7.4). The trajectories for x-, y-, and z-position of feet relatively given to the hip are parameterized by geometric parameters, which leads depending on the number of time steps to a high-dimensional (finite, however) discretization of the walking motion. The three-dimensional walking trajectories are calculated by the following formulas.
To control the y-position of the hip the ZMP is used:

$$\begin{aligned} y(t) &= c_1 * \exp(\sqrt{(g/H_g)} * t) \\ &+ c_2 * \exp(-\sqrt{(g/H_g)} * t) + y_{ZMP} \end{aligned} \tag{7.1}$$

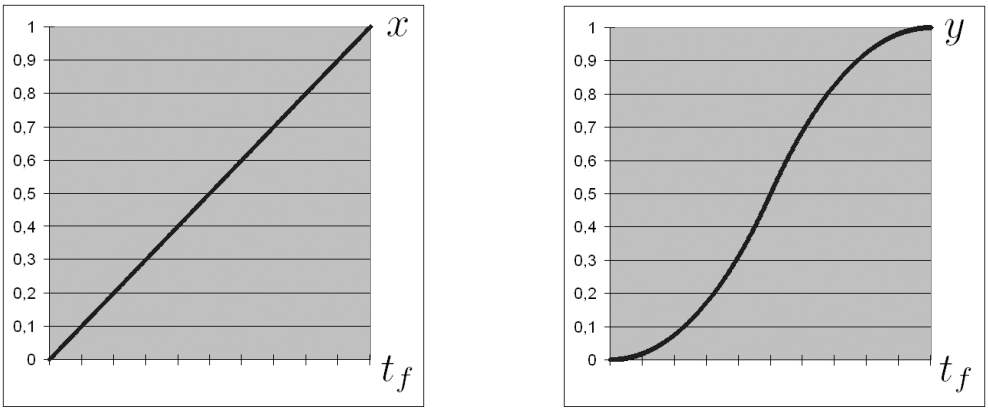


Figure 7.6: Parameterized x- and y-trajectories of for the walking motion.

where g describes the gravity, H_g the height of center of gravity of the robot standing, c_1 and c_2 are constants, and y_{ZMP} is precalculated so that the ZMP moves on a rectangular trajectory between the feet (cf. solid line in Fig. 7.5). The x-position of the hip is just a simple straight line that is used to control the center of gravity movement of x-axis. When the robot is walking at constant speed, the ZMP of the x-axis is 0 and is neglected.
The z-position is the distance between hip and foot, and described by two curves, one for lifting (z_{up}) and a second for dropping (z_{dw}). Both are moving fast close to the ground to reduce the influence of the ground error and to reduce inertia during swing of the leg. For lifting and dropping the foot we use

$$z_{up}(t) \quad = \quad \frac{z_{height} \cdot c_4}{\pi} \left(\frac{\pi}{2} - \arcsin\left(1 - \frac{t}{t_f}\right) - c_3 \right), \tag{7.2}$$

respectively

$$z_{dw}(t) \quad = \quad z_{up}(t_f - t), \tag{7.3}$$

as basis functions, with z_{height} as maximum wanted height in z direction of the foot, c_3 and c_4 are constants, and t_f as final time of one step. By aggregating these curves for a walking motion we obtain completely parameterized walking motion trajectories. Parameter tuning by hand identified the important parameters for stable walking. Those are the relation of the distances of the front and of the rear leg to the center of mass, the lateral position, the roll angle and the height above ground of the foot during the swing phase, and the pitch of the upper body. These parameters scale and shift the defined walking trajectories in Equations (7.1) to (7.3) during the optimization process, where the other parameters are assumed to be constant with suitable values which are set by expert knowledge and experiments before the optimization process starts.
Walking motions symmetry allows a further problem reduction. The parameter for the lateral position of the right foot can be set to the negative lateral position of the left foot during the swinging phase, and the roll angle of the right foot has to be the negative roll angle of the left foot. A six-dimensional parametrization of a forward walking trajectories at a fixed step frequency is obtain finally.

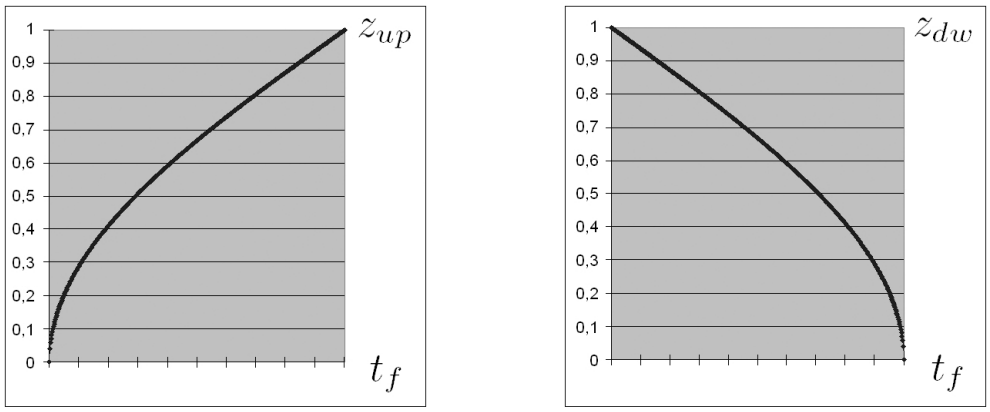


Figure 7.7: Parameterized z-trajectories for the walking motion.

7.3.3 The Derived Optimization Problem

Even for this non-deterministic problem the surrogate may help to catch the main characteristics of the problem if the deviation is in comparison to the mean not too high to mask the main effects of the underlying black box $\boldsymbol{y}$.

Modeling of the Objective Function

For the objective function value $f$of a parameter set $\boldsymbol{p}$ describing a walking motion the distance which the robot covers on the experimental field until it stops or falls is measure. It starts with a small step length 110 mm and the step length is increased every 2 steps by 5 mm, so that the final step length of 240 mm would be reached after 52 steps, always with a constant frequency of about 2.85 steps per second. It follows that the walking speed is only influenced directly by the applied step length.
This definition of the objective function includes constraints for maintaining walking stability of the walking motion implicitly, which otherwise would have to be formulated analytically with a high effort to incorporate them explicitly into the definition of the here considered optimization problem. Only robust walking motions for all step lengths between 110 mm and 240 mm have a chance to reach a high objective function value, because during the measurement 27 different step lengths are applied. This ensure that a found walking motion is a fast but also robust one for different velocities.

Feasible Domain of Optimization Variables

The above introduced number of relevant parameters for the walking motion is reduced with this objective function definition by another dimension to finally five optimization variables. The variation of the step length during the walking experiment, the x-position of one foot in front of the upper body and the x-position of the other foot behind the upper body during a step are reduced to only one variable. This variable describes the proportion between both. For all applied values for the step length, both parameters are described uniquely. During the online optimization the five real-valued variables that are influencing the main characteristics of the walking behavior of the robot are being varied

to maximize the defined objective function only subject to box constraints. These bounds for the optimization parameters must be strictly met by the optimization method for each trial walk of the robot occurring during one evaluation of the objective function. The bounds are set to values for which a wide range of variations of the walking motion is possible. However, the bounds also guarantee that no self-colliding occurs.

Classification of the Optimization Problem

The robot is included as hardware in the loop for the walking experiment to evaluate the objective function f measuring the walking speed. In this context a non-deterministic black box optimization problem with box constraints arises, where besides a noisy function value of f no further information is provided. Especially no gradient information can be provided which would be compulsive for effective gradient-based optimization. Even the approximation of gradients by finite differences is not applicable because of the non-deterministic experimental results. Additionally we are also not able to prove properties like continuity or differentiability of the objective function because of the black box characteristic of the problem. The resulting mathematical formulation of the optimization problems reads as follows:

$$\max_{\boldsymbol{p}\in\Omega_{\boldsymbol{g}}} \; f(\boldsymbol{p}),\; f : \mathbb{R}^5 \to \mathbb{R},$$

with

$$\Omega_{\boldsymbol{g}} = \{\boldsymbol{p} \in \mathbb{R}^5 | \boldsymbol{p}_{lb} \leq \boldsymbol{p} \leq \boldsymbol{p}_{ub},\; \boldsymbol{p}_{lb}, \boldsymbol{p}_{ub} \in \mathbb{R}^5\}.$$

This leads to the question which optimization methods can be applied to find a good walking motion with reasonable efforts.

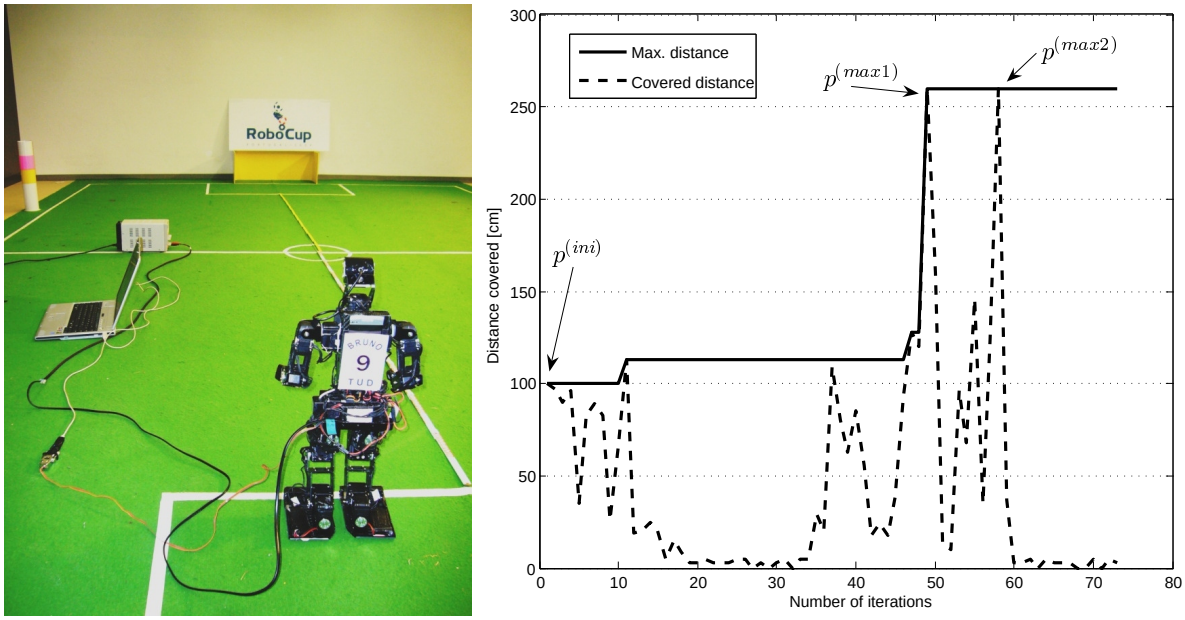
Adapted Optimization Procedure

The optimization process is started with an experiment, where the basis of the first surrogate function is chosen by prior obtained expert knowledge to provide a stable but possibly slow, initial humanoid robot walking motion. A set of candidates is generated around the initial motion by varying each single variable dimension p_i on its own by 10 % of the total range of the feasible domain. In our case this leads to a basis with eleven sampled points. The further procedure is as described in Chapter 4.

7.3.4 Experimental Setup and New Fast Walking Motions

Setup and Procedure

All walking experiments with the humanoid robot HR 18 are performed on a standard RoboCup soccer field (cf. Fig. 7.8 a). To guarantee constant conditions during the experiments the joint motors are supplied by a continuous external power supply where the batteries are only used as weights to provide the operating condition the robot meets in the intended soccer application. The Pocket PC is only carried in all walking experiments for the same reason as the batteries are. Only the low-level software is running on the robot during the experiments with its joint motor control and the postural stabilization.



(a) Setup at the beginning of a walking experiment.

(b) Walking experiment results during optimization.

Figure 7.8: Experiment setup and measured distances.

The controller board is serially connected to an external computer, on which the above described high-level software and the optimization procedure is running. The walking requests generating module therein is extended for the special needs of the modeled objective function and the walking experiments that have to be performed, so that the step length can be increased during runtime. Additionally the joint motor temperatures are observed manually to guarantee no overheating at any time.
The motion generation obtains new walking parameter candidates from the optimization procedure as a text file and sends walking requests for walking experiments together with the walking parameters to the robot.
Of course, not every new parameter set that is generated by the optimization procedure results in a walking motion where the robot is really walking forward or is even able to walk one step without falling. For such candidates $\boldsymbol{p}$ we assign zero as the objective function value. All walking experiments start at the same position heading into the same direction. The distance covered by the robot during walking is measured by visual inspection of the human operator in an accuracy of about ± 5 cm. In principle the covered distance can also be measured by one of the onboard cameras using a standard geometrical object with given shape, color, and known constant position. Additionally the temperature of the joint motors as a possible critical factor is observed manually during all experiments, to guarantee no overheating at any time. The power cord, the serial connection, and a belt to protect the robot from damage in the case that it falls down are carried by a human helper, but it is keenly observed that the robot's motions are not influenced by the cables and the belt.

Table 7.1: Definition of the optimization variable's components.

p_1	backward / forward	each in mm
p_2	step height	in mm
p_3	lateral position	in mm
p_4	roll angle	in degree
p_5	body pitch	in degree

Table 7.2: Bounds, initial expert's guess, and two candidates for an optimal $\boldsymbol{p}$.

Description	Notation	p_1 backward/ forward mm	p_2 step height mm	p_3 lateral position mm	p_4 roll angle mm	p_5 body pitch degree
Lower bound	$\boldsymbol{p}^{(lb)}$	-2.000	0	50.000	-8.000	0
Initial set	$\boldsymbol{p}^{(0)}$	-1.000	25.000	75.000	-4.000	15.000
Candidate 1	$\boldsymbol{p}^{(max1)}$	-0.958	17.344	81.641	-0.018	20.000
Candidate 2	$\boldsymbol{p}^{(max2)}$	-2.000	27.761	50.000	-0.155	17.331
Upper bound	$\boldsymbol{p}^{(ub)}$	-0.500	50.000	100.000	0	20.000

Optimization Results

The initial candidate $\boldsymbol{p}^{(0)}$ has been obtained by tuning of the parameters by hand and the definition of $\boldsymbol{p}$ as described above is summarized in Table 7.1. During the eleven initial walking experiments during the optimization process, the results approve a nonlinear connection to the parameters as we can see from the objective function values (cf. Fig. 7.8 b). Directly after the initial experiments a $\boldsymbol{p}$ is already found which results in a larger objective function value. The following iterations of the surrogate optimization approach can be taken as a kind of exploring to catch the characteristics of the black box function $f$better.

Also the maximizers of the surrogate functions $\hat{y}^{(i)}$ during the first iterations are not likely to be maximizers of the underlying black box function as can be observed from iteration 18 to 32 in Figure 7.8. Promising areas for improvements are identified in form of two candidates $\boldsymbol{p}^{(max1)}$ and $\boldsymbol{p}^{(max2)}$, where the robot covers the same distance after performing all 52 steps of the walking experiments including to stop at the end without falling. Both candidates are identified during maximizing the surrogate and not the MSE. The motion resulting from the set $\boldsymbol{p}^{(max1)}$ is visualized in Figure 7.9.
Even after exploring promising areas of the variable domain, no continuous improvement is guaranteed as observed during the last 10 walking experiments. An appropriate approximation quality for the entire five-dimensional variables domain cannot be obtained by the number of the so far performed walking experiments.
As a more technical result affecting the hardware we observed that at no time the external power supply generated more than 3.2 ampere at a voltage of 14.8. During the whole number of walking experiments the joint motor temperature was never becoming critical with average temperatures of 55 to 60 degree Celsius. The knee joint motor as the most stressed

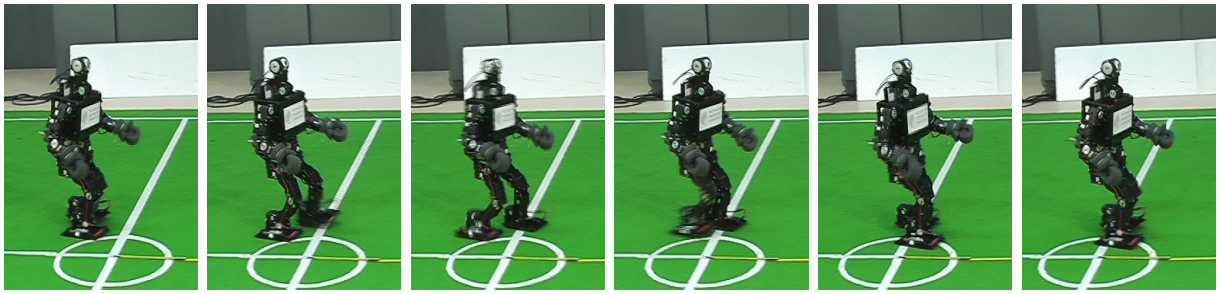


Figure 7.9: Optimized walking motion of the humanoid robot prototype Bruno.

one has been observed during all iterations carefully. The here described optimization run was paused only for one time when the motor temperature exceeded 65 degree Celsius after about 40 runs. This was done to guarantee that all walking experiments are performed under roughly comparable conditions. The shut down temperature of the joint motors is 85 degree Celsius.

Adaption for a Fast Walking Motion

The solutions found during the optimization are successively adjusted by setting up a constant step length and step time dependent on different floor coverings. The result of this first approach is a stable humanoid robot walking motion with a speed of 30 cm/sec. This motion provides a robustness that allows to start with the found maximum speed and to perform an abrupt stop without unbalancing the robot, even on different floor coverings. All walking experiments during the optimization run have been performed on a single time slot of about four hours, including breaks of about one hour, and less than 100 walking experiments including a warming up of the robot.
The robot is right from the start designed to be very lightweight. Additional work was spent on the lightweight aspect, one DOF was reduced in each arm so that 2 DX-117 motors could be removed as well as the left/right turning waist joint. By the same reason the steal frame of the upper body was replaced by a slimmer carbon frame of 0.2 kg less weight. To lower the center of mass, the battery for the joint motors was replaced by two 2-cell batteries of the same kind like it is used for the controller board. These two batteries are placed on the sides of the both feet (cf. Fig. 7.10).
Another change was done due to the relatively high temperatures in the knee joints, so that during the normal operation time of robot soccer games no heating problems arise. In exchange for the first used DX-117, a motor of type Robotis RX-64 with a higher maximum torque of 53 kg-cm as well as a higher maximum speed of 0.198 sec/60° (but also larger weight and size) was built in.
The modified version of the humanoid robot has a final weight of 3.3 kg and was the basis of the humanoid robot Bruno. For a more detailed report on the modified robot we refer the reader to [97].
For the modified robot the walking motion needed some steps for acceleration with increasing step length to get into a stable walking motion of 40 cm/sec. It also performs a stable instant stop in one step. In further experiments, it was observed that the obtained

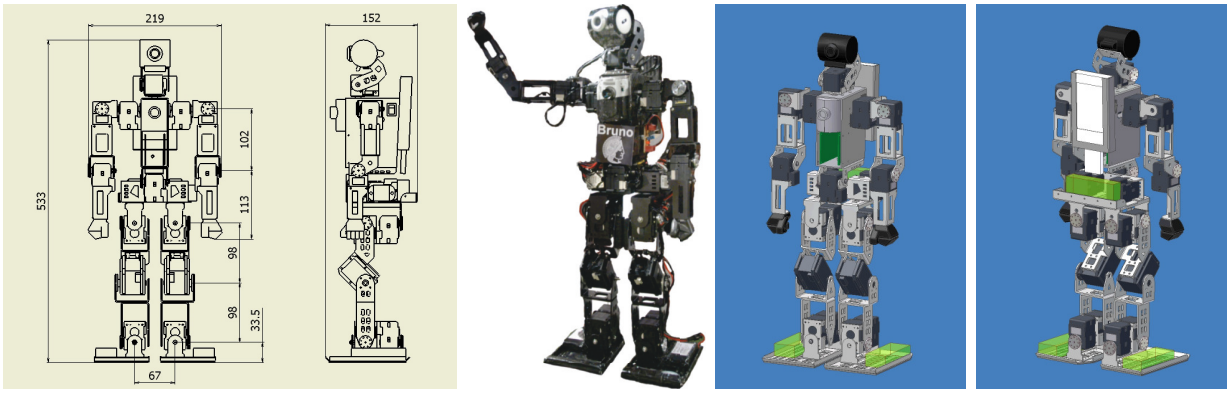


Figure 7.10: Technical layout with measurements in mm and 3D view including the design modifications.

walking motion is also robust in a sense that the performance does not change much on different floor coverings.

7.4 Summary of Chapter 7

The first small example, the walking optimization of the Sony Aibos, has shown, that hardware in the loop optimization of walking movements is a promising approach, which leads us to the approach with the two-legged humanoid robot. Beside a case study for optimization directly coupled with hardware, the measured improvements have shown, that pure expert knowledge does not guarantee a good walking behavior. Even by a standard pattern search implementation within an hour of lab time new trajectories were found.
For the newly developed, 55cm tall autonomous humanoid robot prototype Bruno a hardware in the loop walking optimization approach has been presented. The optimization problem formulation is based on a suitable parametrization of the walking trajectories, where a small number of parameters are used as optimization variables. During fast walking, the underlying joint servo motor control is stabilized using inertial sensing which is part of the low-level software running on the robot's controller board. Under this preconditions it was demonstrated that efficient walking speed optimization can be obtained even with a quite small number of functions evaluations. In about 50 walking experiments a 30 cm/sec fast walking motion has been obtained using the surrogate optimization approach. In further experiments, it was observed that as an interesting side effect the obtained walking motion is also robust in a sense that the performance is also well and does not change much on different floor coverings which have not been used during the optimization. A hardware design update enabled improvements by reducing the weight of the upper body. With a short starting phase for acceleration from stand still a further increase of the forward walking speed to more than 40 cm/sec was achieved. The described optimization procedure can be applied to other types of humanoid robot locomotion like turning or walking sidewards as well.

8 Conclusion

For the first time it has been shown in this thesis that mixed-integer nonlinear optimization for non-relaxable black box problems arising in different engineering disciplines can be solved efficiently. This was achieved by a newly developed surrogate optimization approach. This general approach is not restricted to applications from the three specific fields of engineering discussed in this thesis.
The results presented here demonstrate that and how integer variables can be handled in an efficient way and that optimization subject to explicit and implicit constraints can be carried out directly. The performance of the presented approach is investigated for different applications, where besides numerical simulation packages also lab experiments with robots are included in the optimization problem formulation. For all of the considered applications, no other optimization method was able to provide results of similar quality within a competitive effort.

8.1 Summary

Integer valued variables should be taken as what they are: integer. In a mixed-integer scenario continuous-valued and integer-valued variables have to be taken into consideration simultaneously. Existing optimization methods could not address this problems directly in a satisfactory manner. The presented results, especially from the water resources management applications, show that transforming a mixed-integer optimization problem to a real-valued one does not have to be a better way to solve it, just because efficient optimizers are available. It is important to solve the right problem with the right tool.
Furthermore, all available information, especially that related to black box-based hidden (black box evaluation failure) and implicit constraints (infeasible system state) should be taken into account during the optimization process. Considering all information from the constraints is as important as the pure objective function value for optimization. Not to make use of it can result in repeated failures of black box evaluations which just allocate resources without providing any benefit. Information about black box responses obtained during optimization should be used at once for improved performance.

Methodology We derived in this thesis a surrogate optimization approach that really take into account that only parts of the optimization problem formulation are given explicitly. Stochastic approximations are used to replace the implicit parts of the problem, to keep all explicit information. Gradient-based optimization methods are applied to the resulting smooth surrogates problems, which are defined on relaxed and extended domains. Promising regions and feasible candidates for the initial optimization problem are identified at low computational costs this way.
With any new response from evaluation of candidates by the incorporated black boxes, the functional approximations are updated instantaneously, and the quality of the surrogate

problem is improved sequentially. This procedure is enabled by a newly developed efficient framework that handles black box evaluations, stochastic approximations, the surrogate building processes, and the use of optimization methods. The developed iterative update strategy guarantees that numerical stability is kept for the stochastic approximation method. The chosen strategy causes, coincidentally, the exploration of the whole optimization variable domain. Furthermore the approach does not have to start from scratch, but is a hot start method and makes it possible to integrate prior response information easily. The transition from surrogate models to surrogate functions is fluent. Knowledge-based low fidelity models are fitted to output data, basis functions for approximation are chosen by system characteristics. To improve and speed up the search for optimal candidates all kind of problem information has to be used. Typically, methods which use model knowledge of the underlying system are faster than pure output approximating methods. The surrogate optimization approach provides the possibility to incorporate prior knowledge, e.g., by adapting the different basis functions for the approximation of the black box responses. This can be done during each iteration in different ways and highlights the flexibility of the developed approach.

Applications The design optimization examples from electrical engineering have shown that the presented approach for solving mixed-integer nonlinear problems determines solutions of high quality for the system design. The benefit of a direct handling of implicit constraints has been demonstrated for the design of a magnetic bearing. For the design of a superconductive synchrotron magnet all earlier applied methods have not been able to deal with the integer-valued variables that define the devices' structure. With the amount of simulation evaluations that other methods needed to solve the inherent continuous-valued nonlinear problems, a competitive design close to the theoretical minimum was identified. The second type of applications discussed is taken from environmental engineering. In a reply to Lucas (both in [180]), Sacks and his colleagues pointed out that first, simulation problems often do not behave like stochastic problems and second, that finally, even the test problems should be some kind of simulation application. The presented applications, called Community Problems, consist not only of a number of optimization problems based on a subsurface flow simulation. They are also established as a benchmark set of problems for any derivative free optimization method, as was claimed by Sacks. It has been shown earlier, for the Community Problems, that deterministic sampling methods can outperform metaheuristic search methods [93–95]. The presented surrogate optimization approach, combined with a newly introduced mixed-integer nonlinear problem formulation, resulted in an even better solution obtained by only a small fraction of black box evaluations needed by other approaches.
Finally, a real system in the loop scenario for maximizing the walking speed of humanoid robots was investigated. It is a very challenging task to capture all characteristics which are important to achieve a walking speed with postural stability at the limits of the real robot's capabilities by standard multibody system dynamics models for humanoid robots. This was overcome by the modeling of a low-dimensional black box-based problem, where each black box evaluation represents the laboratory experiment to measure the humanoid robot's walking performance. The walking speed of the robot was increased by a factor of four up to more than 40 cm/sec which is one of the fastest and stable bipedal walking gaits obtained so far for autonomous humanoid robots of this size.

8.2 Perspectives

Finally, some perspective on further improvement of optimization methods and their applications in engineering are outlined. Methods that handle implicit but calculable black box type constraints efficiently can additionally help with the determination of hidden and closed constraints which only lead to black box evaluation failures. A fundamental difficulty was identified in this work if "engineering-perspective" designs are not available as initial iterates for the optimization. In general, new methods to predict unfeasible areas of the variable domain because of hidden constraints are needed, as they will help to reduce the number of unsuccessful black box evaluations.
To avoid potential weakness of one approach, hybrid approaches of combined derivative free optimization methods are beginning to emerge, e.g., in the area of water resources management [83, 138, 191, 216, 236, 241]. However, there are still many open questions regarding how to optimally combine two different methods, e.g. how to define switching criteria for changing from one method to another method. These problems increase the degree of complexity, but in a well-suited framework hybrid approaches will help to overcome the specific weaknesses of a single optimization approach.
Furthermore, optimization methods should make use of the resources provided by the ongoing development of multi-core CPU and GPU computing architecture. Parallelization of iterative optimization approaches and even more, of hybrid approaches needs an algorithmic foundation suitable for these emerging computing environments. The algorithms should be freely scalable and not restricted to a certain number of cores or nodes. The increasing computational power and a greater number of available options for parallelization will offer a further speed up for numerical optimization and black box evaluations.
New concepts for user interaction will grow coincidentally, as soon as well adapted presentation and visualization concepts become available to illustrate intermediate data during optimization. Expert knowledge can then be incorporated not only by setting initial variables values, but also during all iterations to add information that possibly was not available at the beginning. Optimization could made use of this for the interpretation of hidden constraints, to introduce or discard optimization variables, or even to realize necessary changes in the objective function and constraints.
During the plenary "The Forward Looking Session" at the SIAM Conference on Optimization in 2008 simulation-based mixed-integer nonlinear problems were mentioned explicitly as one of the most relevant fields for further research. The presented results of this thesis contribute to further progress in this direction and may also point to future research topics in this field.

Zusammenfassung

Optimierungsprobleme spielen in vielen wissenschaftlichen Disziplinen eine immer wichtigere Rolle, z.B. zur optimalen Nutzung begrenzter Ressourcen oder zur Auswahl optimaler Lösungen aus einer großen, endlichen oder unendlichen Menge von Alternativen. Die existierenden Methoden und Werkzeuge für die rechnergestützte Optimierung werden zur Lösung der unterschiedlichsten Probleme angewandt. Optimierung soll problemspezifisch die bezüglich eines definierten Gütekriteriums bestmögliche Ausnutzung vorhandener Ressourcen garantieren. In den Ingenieurwissenschaften ergibt sich allerdings häufig die Situation, dass die Standardoptimierungsverfahren nicht geeignet sind, um die sich ergebenden Problemstellungen effizient zu lösen. Im Rahmen dieser Arbeit werden speziell Fragestellungen betrachtet, in die neben einer expliziten analytischen Formulierung auch das Verhalten von nicht einsehbaren abgeschlossenen Systemen („Black Box“) wie komplexen ingenieurwissenschaftlichen Simulationsprogrammen oder reale Roboter (z.B. Roboter) eingeht.
Häufig sind Ergebnisse komplexer numerischer Simulationen bei der Optimierung zu berücksichtigen, welche zusätzlich meist hohe Rechenzeiten einfordern. Komplexe Simulationssoftware ist in den Ingenieurwissenschaften über Jahre und Jahrzehnte entstanden und ermöglicht nun in vielen Anwendungen eine realitätsnahe Simulation. Es ist daher wünschenswert, diese auch für eine systematische, iterative Optimierung von Systemparametern und Systemverhalten einzusetzen. Ein entscheidender Faktor für den praktische Einsatz ist die Anzahl verfügbarer Auswertungen aufgrund nur beschränkt verfügbarer Ressourcen an Entwicklungs- und Rechenzeit.
Die innerhalb von komplexer numerischer Simulationssoftware verwendeten Methoden, unterlagerte und miteinander gekoppelte iterative Berechnungsverfahren, Approximationen tabellarischer Daten, führen zu Optimierungsproblemstellungen, die häufig vordergründig deterministisch sind, jedoch gewissen Störungen und sehr niedrigen Differenzierbarkeitsordnungen (z.B. Unstetigkeiten in Funktion oder Ableitungen) bezüglich der Optimierungsvariablen unterliegen. Neben den reinen Ausgabewerten der Simulationssoftware stehen meist keine weiteren Systeminformationen zur Verfügung, und können auch mit größerem Aufwand nicht explizit generiert werden. Insbesondere können gradienten- oder modellbasierte Optimierungsverfahren nicht oder nur mit sehr begrenzten Erfolg eingesetzt werden. Weiterhin muss angenommen werden, dass die Optimierungsvariablen sowohl kontinuierlich als auch rein diskret sein können. Für die entstehenden gemischt ganzzahligen, Black Box basierten, nichtlinearen Optimierungsprobleme existieren noch sehr wenige allgemein anwendbare und effiziente Optimierungsverfahren. Auf diese Problemklasse zielt die vorliegende Arbeit ab.
Zunächst ist die Einführung einer neuen Problemformulierung nötig, die zwischen explizit analytischen und implizit nur durch Black Box Auswertungen gegebenen Problemkomponenten unterscheidet. Dies ermöglicht die Anwendung eines neuen, auf Ersatzfunktionen basierenden Optimierungsverfahrens. Etwaige explizit gegebene Teile der Optimierungsproblemformulierung bleiben damit komplett erhalten und können durch

dieses Optimierungsverfahrens vorteilhaft ausgenützt werden. Die impliziten, Black Box basierten Komponenten werden durch geeignete stochastische Approximationsfunktionen ersetzt, und führen so zu einem Ersatzproblem für dessen Auswertung im Rahmen des Optimierungsverfahrens die impliziten Komponenten nicht mehr nötig sind. Durch geeignete Wahl der Ersatzfunktionen ergibt sich ein glattes (d.h. mehrfach differenzierbares), relaxierbares, sowie numerisch hoch effizient auswertbares Optimierungsproblem.
Zur Berechnung von Lösungskandidaten für das Originalproblem werden effiziente gradientenbasierte Verfahren der gemischt ganzzahligen nichtlinearen Optimierung auf den Ersatzproblemen angewendet. Mit Hilfe von Fehlerschätzern der Approximationsfunktionen wird gewährleistet, dass der Raum der zulässigen Optimierungsvariablen dabei weiträumig exploriert wird. Die eigentliche Qualität der Kandidaten wird anhand der Gütefunktion des Originalproblems durch Auswertung der Black Box Komponenten bestimmt. Die Ergebnisse jeder dieser Auswertungen fügen weitere Approximationsinformationen für die Ersatzfunktionen hinzu. Eingebunden in einen iterativen Prozess werden so immer neue Kandidaten zur Auswertung auf schrittweise verfeinerten Ersatzproblemen bestimmt. Das entwickelte Verfahren reduziert die zur Optimierung benötigte Anzahl von Black Box Auswertungen erheblich, so dass im Rahmen dieser Arbeit erstmals eine effizient rechnergestützte Optimierung für eine Reihe schwieriger gemischt ganzzahlige Black Box basierte Anwendungsprobleme möglich wird.
Für die Designoptimierung von Magnetlagern und supraleitenden Synchrotronmagneten können die theoretisch erreichbaren Zielkriterien in nur einem Bruchteil des Aufwands, verglichen mit früheren verwendeten Ansätzen, erfüllt werden. Die bisherige Lösung ohne Berücksichtigung und damit ohne Anpassung der ganzzahligen Variablen erforderte bei der Optimierung des Synchrotronmagneten vergleichbar viele numerische Simulationen wie die Lösung des Gesamtproblems mit dem in dieser Arbeit hergeleiteten Optimierungsverfahren.
Für eine aktuelle Sammlung von Benchmarkproblemen aus dem Bereich des Grundwassermanagements, die auf komplexen Strömungsdynamiksimulationen beruhen, können neue kostenminimale Systemauslegungen berechnet werden. Dabei ist eine erhebliche Reduktion der benötigten Simulationsauswertungen und damit der benötigten Rechenzeit verglichen mit publizierten Lösungen zu verzeichnen.
Mit dem Ziel die Laufgeschwindigkeit eines humanoider Roboterprototyps zu maximieren, werden Messungen am realen technischen System direkt mit dem entwickelten Optimierungsverfahren gekoppelt. Auch bei dieser Anwendung stellt sich eine erhebliche Verbesserung der Laufstabilität und eine Geschwindigkeitsteigerung von zu Beginn unter 10 cm/s auf über 40 cm/s am Ende der Optimierung ein.
Die direkte Kopplung des hergeleiteten Optimierungsansatzes mit Black Box basierten Problemen ermöglicht es nun auch Problemstellungen erfolgreich zu behandeln, die bisher entweder nur mit sehr viel höheren Rechenaufwand, oder gar nicht lösbar waren. Auf diese Weise kann das in die Entwicklung der Black Box Komponenten über Jahre und Jahrzehnte eingeflossene Expertenwissen für eine systematische Optimierung von Systemparametern und Systemverhalten nutzbar gemacht werden.

Bibliography

[1] M. A. Abramson. *Pattern Search Algorithms for Mixed Variable General Constrained Optimization Problems*. PhD thesis, Rice University, 2002.

[2] M. A. Abramson. Mixed variable optimization of a load-bearing thermal insulation system using a filter pattern search algorithm. *Optimization and Engineering*, 5:157177, 2004.

[3] M. A. Abramson, C. Audet, and J. E. Dennis jr. ORTHOMADS: A deterministic MADS instance with orthogonal directions. Technical report, Rice University, Houston, Texas, 2008.

[4] C. S. Adjiman, C. A. Schweiger, and C. A. Floudas. *Handbook of Combinatorial Optimization*, chapter Mixed-Integer Nonlinear Optimization in Process Synthesis. Kluwer Academic Publishers, 1998.

[5] D. P. Ahlfeld and A. E. Mulligan. *Optimal Design of Flow in Groundwater Systems*. Academic Press, San Diego, 2000.

[6] N. Alexandrov. Editorial - Multidisciplinary design optimization. *Optimization and Engineering*, 6:5–7, 2005.

[7] N. Alexandrov and R. M. Lewis. An overview of first-order model management for engineering optimization. *Optimization and Engineering*, 2:413–430, 2001.

[8] C. Audet and J. E. Dennis jr. Pattern search algorithms for mixed variable programming. *SIAM Journal on Optimization*, 11:573–594, 2000.

[9] C. Audet and J. E. Dennis jr. Analysis of generalized pattern searches. *SIAM Journal on Optimization*, 13:889–903, 2003.

[10] C. Audet and J. E. Dennis jr. A pattern search filter method for nonlinear programming without derivatives. *SIAM Journal on Optimization*, 14:980–1010, 2004.

[11] C. Audet and J. E. Dennis jr. Mesh adaptive direct search algorithms for constrained optimization. *SIAM Journal on Optimization*, 17:188–217, 2006.

[12] M. H. Bakr, J. W. Bandler, K. Madsen, and J. Søndergaard. An introduction to the space mapping technique. *Optimization and Engineering*, 2:369384, 2001.

[13] J. W. Bandler, S. Koziel, and K. Madsen. Space mapping for engineering optimization. *SIAG/OPT Views-and-News*, 17:19–26, 2006.

[14] J. Banks and J. Carson, II. *Discrete-event system simulation*. Prentice-Hall, 1984.

[15] T. Barth, B. Freisleben, M. Grauer, and F. Thilo. Distributed solution of simulation-based optimization problems on networks of workstations. In *International Conference on Parallel Computing Systems*, 1999.

[16] T. Barth, B. Freisleben, M. Grauer, and F. Thilo. A scalable algorithm for the parallel solution of simulations-based optimization problems. In *Proceedings of the International Conference on Parallel and distributed Processing Techniques and Applications PDPTA'2000*, 2000.

[17] J.-F. M. Barthelemy and R. T. Haftka. Approximation concepts for optimum structural desing - a review. *Structural Optimization*, 5:129– 144, 1993.

[18] R. R. Barton. Design of experiments for fitting subsystem metamodels. In S. Andradttir, K. J. Healy, D. H. Withers, and B. L. Nelson, editors, *Proceeding of the 1997 Winter Simulation Conference*, pages 303–310, 1997.

[19] R. R. Barton. Simulation metamodels. In D. J. Medeiros, E. F. Watson, J. S. Carson, and M. S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 167–174, 1998.

[20] T. Bartz-Beielstein, K. E. Parsopoulos, and M. N. Vrahatis. Design and analysis of optimization algorithms using computational statistics. *Applied Numerical Analysis & Computational Mathematics*, 1:413–433, 2004.

[21] A. Battermann, J. Gablonsky, A. Patrick, C. T. Kelley, and K. R. Kavanagh. Solution of a groundwater control problem with implicit filtering. *Optimzation and Engineering*, 3: 189–199, 2002.

[22] A. Bemporad and M. Morari. Control of systems integrating logic, dynamics, and constraints. *Automatica*, 35:407–427, 1999.

[23] J. F. Benders. Partioning procedures for solving mixed-variables programming problems. *Numerische Mathematik*, 4:283–252, 1962.

[24] D. P. Bertsekas and J. N. Tsitsiklis. *Parallel and Distributed Computation: Numerical Methods*. LIDS Technical Reports. Athena Scientific, 2003.

[25] J. T. Betts. *Practical methods for Optimal Control Using Nonlinear Programming*. SIAM, 2001.

[26] C. H. Bischof and H. M. Bücker. Welchen Einfluss besitzen Modellparameter? Bewertung und Auslegung von Computermodellen durch automatisches Differenzieren. In *Computational Engineering Science — Ein neues Ingenieurprofil*, volume 2 of *RWTH-Themen*, pages 18–20. RWTH Aachen University, 2007.

[27] H. G. Bock, W. Egartner, W. Kappis, and V. Schulz. Pratical shape optimization for turbine and compressor blades by the use of PRSQP methods. *Optimization and Engineering*, 3: 395–414, 2002.

[28] A. J. Booker, J. E. Dennis jr., P. D. Frank, D. B. Serafini, and V. Torczon. Optimization using surrogate objectives on a helicopter test example. *Computational Methods for Optimal Design and Control*, 24:49–58, 1998.

[29] A. J. Booker, J. E. Dennis jr., P. D. Frank, D. B. Serafini, V. Torczon, and M. W. Trosset. A rigorous framework for optimization of expensive functions by surrogates. *Structural Optimization*, 17:1–13, 1999.

[30] A. J. Booker, J. Brown, P. Frank, and G. Miller. Reduced design cycle time and improved design space exploration. In *Space 2004 Conference and Exhibit, San Diego, California*, number AIAA-2004-6125, Sep. 28-30 2004.

[31] A. J. Booker, M. Meckesheimer, and T. Torng. Reliabilty based design optimization using design explorer. *Optimization and Engineering*, 5:179–205, 2004.

[32] H. Bossel. *Modellbildung und Simulation.* Vieweg, 1994.

[33] G. E. Box, J. S. Hunter, and W. G. Hunter. *Statistics for Experimenters: Design, Innovation, and Discovery.* Wiley-Interscience; 2nd edition, 2005.

[34] M. J. Box. A new method of contrained optimization and a comparison with other methods. *The Computer Journal*, 8:42–52, 1965.

[35] R. C. M. Brekelmans, L. T. Driessen, H. J. M. Hamers, and D. den Hertog. Gradient estimation schemes for noisy functions. *Journal of Optimization Theory and Applications*, 126:529–551, 2005.

[36] R. C. M. Brekelmans, L. T. Driessen, H. J. M. Hamers, and D. den Hertog. Constrained optimization involving expensive function evaluations: A sequential approach. *European Journal of Operational Research*, 127:121–138, 2005.

[37] C. Büskens. *Optimization Methods and Sensitivity Analysis for Optimal Control Problems with Control and State Constraints.* PhD thesis, Institute for Numerical and Instrumental Mathematics, University of Münster, 1998.

[38] M. Buss, M. Hardt, J. Kiener, J. Sobotka, M. Stelzer, O. von Stryk, and D. Wollherr. Towards an autonomous, humanoid, and dynamically walking robot: Modeling, optimal trajectory planning, hardware architecture, and experiments. In *Proc. IEEE/RAS Humanoids 2003.* Springer-Verlag, 2003.

[39] M. Bussieck and A. Pruessner. Mixed-integer nonlinear programming. *SIAG/OPT Newsletter: Views & News*, 14:19–22, 2003.

[40] R. G. Carter, J. Gablonsky, C. T. Kelley, and O. J. Eslinger. Algorithms for noisy problems in gas transmission pipeline optimization. *Optimization and Engineering*, 2:139–157, 2001.

[41] J. Z. Cha and R. W. Mayne. Optimization with discrete variables via recursive quadratic programming: Part 1 - Concepts and definitions. *Transactions of the American Society of Mechanical Engineers*, 111:124–129, 1989.

[42] P. K. Chandila. *Strategy For Global Optimization And Post-Optimality Using Local Kriging Approximations.* PhD thesis, University of Notre Dame, Indiana, 2004.

[43] R. C. H. Cheng and C. S. M. Currie. Optimization by simulation metamodelling methods. In *Proceedings of the 2004 Winter Simulation Conference*, 2004.

[44] R. Chin, W. Chen, and A. Sudjianto. On sequential sampling for global metamodeling in engineering design. In *ASME 2002 Design Engineering Technical Conferences And Computers and Information in Engineering Conference*, Montreal, Canada, September 29-October 2 2002.

[45] T. D. Choi and C. T. Kelley. Superlinear convergence and implicit filtering. *SIAM J. Optimization*, 10:1149–1162, 2000.

[46] T. D. Choi, P. Gilmore, O. J. Eslinger, A. Patrick, C. T. Kelley, and J. Gablonsky. IFFCO: Implicit filtering for constrained optimization, version 2. Technical Report CRSC-TR99-23, Center for Research in Scientific Computation, July 1999.

[47] S. E. Cieniawski, J. W. Eheart, and S. Ranjithan. Using genetic algorithms to solve a multiobjective groundwater monitoring problem. *Water Resources Research*, 31:399–409, 1995.

[48] M. Clerc and J. Kennedy. The particle swarmexplosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6: 58–73, 2002.

[49] B. Colson and P. L. Toint. Exploiting band structure in unconstrained optimization without derivatives. *Optimization and Engineering*, 2:399–412, 2001.

[50] I. D. Coope and C. J. Price. On the convergence of grid-based methods for unconstrained optimization. *SIAM Journal on Optimization*, 11:859–869, 2001.

[51] M. Cuncha. On solving aquifer management problems with simulated annealing algorithms. *Water Resources Management*, 13:153–169, 1999.

[52] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. A bayesian appraoch to the design and analysis of computer experiments. Technical Report ORNL-6498, ORNL Oak Ridge National Laboratory (US), January 1988.

[53] C. Currin, T. Mitchell, M. Morris, and D. Ylvisaker. Bayesian prediction of deterministic functions, with applications to the design and analysis of computer experiments. *Journal of the American Statistical Association*, 86:953–963, 1991.

[54] E. Davis and M. Ierapetritou. A kriging based method for the solution of mixed-integer nonlinear programs containing black-box functions. *Journal of Global Optimization*, 43: 191–205, 2007.

[55] E. Davis and M. Ierapetritou. Adaptive optimisation of noisy black-box functions inherent in microscopic models. *Computers and Chemical Engineering*, 31:466–476, 2007.

[56] K. Deb. An efficient constraint handling method for genetic algorithms. *Computer Methods in Applied Mechanics and Engineering*, 186:311–338, 2000.

[57] K. Deb and R. B. Agrawal. Simulated binary crossover for continuous search space. *Complex Systems*, 9:115–148, 1995.

[58] K. Deb and T. Goel. Controlled elitist non-dominated sorting genetic algorithms for better convergence. In E. Zitler, K. Deb, L. Thiele, C. A. Coello-Coello, and D. Corne, editors, *Proceedings of the First International Conference on Evolutionary Multi-Criterion Optimization EMO 2001*, Lecture Notes on Computer Science, pages 67–81, 2001.

[59] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. KanGal Report 200001, Indian Institute of Technology Kanpur, Kanpur, India, 2000.

[60] K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multi-objective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6:182–197, 2002.

[61] D. den Hertog, J. P. C. Kleijnen, and A.-Y. D. Siem. The correct kriging variance estimated by bootstrapping. *Journal of the Operational Research Society*, 57:400–409, 2006.

[62] J. Denk and G. Schmidt. Synthesis of a walking primitive database for a humanoid robot using optimal control techniques. In *Proceedings of IEEE-RAS International Conference on Humanoid Robots (HUMANOIDS2001), Tokyo, Japan, November 2001*, pages 319–326, 2001.

[63] J. E. Dennis jr. and V. Torczon. Direct search methods on parallel machines. *SIAM Journal on Optimization*, 1:448–474, 1991.

[64] J. E. Dennis jr. and V. Torczon. *Multidisciplinary Design Optimization: State of the Art*, chapter Managing approximation models in optimization. Soc for Industrial & Applied Math, 1996.

[65] J. E. Dennis jr., C. J. Price, and I. D. Coope. Direct search methods for nonlinearly constrained optimization using filters and frames. *Optimization and Engineering*, 5:123–144, 2004.

[66] H.-J. Diersch. *Interactive, Graphics-Based Finite-Element Simulation System FEFLOW for Modeling Groundwater Flow Contaminant Mass and Heat Transport, Users Manual.* WASY GmbH, Berlin, 1998.

[67] H.-J. Diersch. *Wasserbewirtschaftung im neuen Jahrtausend*, chapter Wie komplex sollen Grundwassermodelle sein?, pages 99–112. Verlag Bauwesen Berlin, 2001.

[68] U. Diwekar. Optimization under uncertainty: An overview. *SIAG/OPT Views-and-News*, 13:1–8, 2002.

[69] E. D. Dolan, R. M. Lewis, and V. Torczon. On the local convergence of pattern search. *SIAM Journal on Optimization*, 14:567–583, 2003.

[70] H. Dong, M. Zhao, J. Zhang, Z. Shi, and N. Zhang. Gait planning of quadruped robot based on third-order spline interpolation. In *Proceedings of the 2006 IEEE/RSJ, International Conference on Intelligent Robots and Systems, October 9 - 15, 2006, Beijing, China*, 2006.

[71] J. Dréo, A. Pétrowski, P. Siarry, and E. Taillard. *Metaheuristics for Hard Optimization.* Springer, 2005.

[72] L. T. Driessen, H. P. Stehouwer, and J. J. Wijker. Structural mass optimization of the engine frame of the ariane 5ESC-B. In *Proc. Eur. Conf. Spacecraft, Structures, Materials Mechanical Testing*, 2002.

[73] L. T. Driessen, R. C. M. Brekelmans, H. J. M. Hamers, and D. den Hertog. On d-optimality based trust regions for black-box optimization problems. *Structural and Multidisciplinary Optimization*, 31:40–48, 2006.

[74] M. Dür and V. Stix. Probalilistic subproblem selection in brach-and-bound algorithms. *Journal of computational and applied mathematics*, 182:67–80, 2005.

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

[76] M. A. El-Beltagy, P. B. Nair, and A. J. Keane. Metamodeling techniques for evolutionary optimization of computationally expensive problems: Promises and limitations. In W. Banshaf, J. Daida, E. Eiben, M. Garzon, V. Honavar, M. Jakiela, and R. Smith, editors, *Proceedings of the Genetic and Evolutionary Computation Conference*, pages 196–203, 1999.

[77] T. A. El-Mihoub, A. A. Hopgood, L. Nolle, and A. Battersby. Hybrid genetic algorithms: A review. *Engineering Letters*, 13:124–137, 2006.

[78] M. S. Eldred, W. E. Hart, B. D. Schimel, and B. G. van BloemenWaanders. Multilevel parallelism for optimization on mp computers: theory and experiment. In *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, number 2000-4818, Long Beach, 2000.

[79] M. S. Eldred, A. A. Giunta, B. G. van Bloemen Waanders, S. F. Wojtkiewicz, jr., and W. E. H. andM. P. Alleva. *DAKOTA, A Multilevel Parallel Object-Oriented Framework for Design Optimization, Parameter Estimation, Uncertainty Quantification, andSensitivity Analysis Version 3.1 Users Manual*, April 2003. SAND2001-3796.

[80] M. S. Eldred, A. A. Giunta, and S. S. Collis. Second-order corrections for surrogate-based optimization with model hierarchies, 2004.

[81] M. S. Eldred, A. A. Giunta, and B. G. van Bloemen Waanders. Multilevel parallel optimization using massively parallel structural dynamics. *Structural and Multidiscipilnary Optimization*, 27:97–109, 2004.

[82] M. Emmerich, A. Giotis, M. Ozdemir, T. Back, and K. Giannakoglou. *Parallel Problem Solving from Nature PPSN VII*, chapter Metamodel-assisted evolution strategies, pages 361–370. Lecture Notes in Computer Science. Springer Berlin / Heidelberg, 2002.

[83] F. P. Espinoza, B. S. Minsker, and D. E. Goldberg. Adaptive hybrid genetic algorithm for groundwater remediation design. *Water Resources and Planning Management*, 131(1): 14–24, 2005.

[84] O. Exler and K. Schittkowski. A trust region sqp algorithm for mixed-integer nonlinear programming. *Optimization Letters*, 1:269–280, 2007.

[85] F. Faber and S. Behnke. Stochastic optimization of bipedal walking using gyro feedback and phase resetting. In *IEEE-RAS 7th International Conference on Humanoid Robots (Humanoids)*, Pittsburgh, USA, Dezember 2007.

[86] H. Fan. A modification to particle swarm optimization algorithm. *Engineering Computations*, 19:970–989, 2002.

[87] K.-T. Fang, D. K. J. Lin, P. Winker, and Y. Zhang. Uniform design: Theory and application. *Technometrics*, 42:237–248, 2000.

[88] D. E. Finkel. *DIRECT Optimization Algorithm User Guide*. Center for Research in Scientific Computation North Carolina State University, Raleigh, NC 27695-8205, US, March 2003.

[89] R. Fletcher. Nonlinear programming without a penalty function. *Mathematical Programming*, 91:239–269, 1997.

[90] R. Fletcher and S. Leyffer. Solving mixed-integer nonlinear programs by outer approximation. *Mathematical Programming*, 66:327–349, 1994.

[91] C. A. Floudas. *Nonlinear and mixed-integer optimization.* Oxford University Press, 1995.

[92] C. A. Floudas. *Deterministic Global optimization Theory, Methods and Applications.* Kluwer Academic Press, 1999.

[93] K. R. Fowler, C. T. Kelley, C. E. Kees, and C. T. Miller. A hydraulic capture application for optimal remediation design. In Miller, Farthing, Gray, and Pinder, editors, *Proceedings of the 15th International Conference on Computational Methods in Water Resources (CMWR XV), 13-17 June 2004, Chapel Hill, North Carolina, Computational Methods in Water Resources, Volume 2, Developments in Water Science 55*, pages 1149–1157. Elsevier Science, Amsterdam, The Netherlands, 2004.

[94] K. R. Fowler, C. T. Kelley, C. T. Miller, C. E. Kees, R. M. Darwin, J. P. Reese, and M. W. F. an M. S. C. Reed. Solution of a well-field design problem with implicit filtering. *Optimzation and Engineering*, 5:207–233, 2004.

[95] K. R. Fowler, J. P. Reese, C. E. Kees, J. E. Dennis jr., C. T. Kelley, C. T. Miller, C. Audet, A. J. Booker, G. Couture, R. W. Darwin, M. W. Farthing, D. E. Finkel, J. M. Gablonsky, G. Gray, and T. G. Kolda. A comparison of derivative-free optimization methods for water supply and hydraulic capture community problems. *Advances in Water Resources*, 31: 743–757, 2008.

[96] J. H. Friedman and C. B. Roosen. An introduction to multivariate adaptive regression splines. *Stat Methods Med Res*, 4:197–217, 1995.

[97] M. Friedmann, J. Kiener, S. Petters, H. Sakamoto, D. Thomas, and O. von Stryk. Versatile, high-quality motions and behavior control of humanoid soccer robots. In *Proc. Workshop on Humanoid Soccer Robots of the 2006 IEEE-RAS Int. Conf. on Humanoid Robots*, pages 9–16, 2006.

[98] M. Friedmann, J. Kiener, S. Petters, D. Thomas, and O. von Stryk. Reusable architecture and tools for teams of lightweight heterogeneous robots. In *Proc. 1st IFAC-Symposium on Multivehicle Systems*, pages 51–56, Salvador, Brazil, October 2-3 2006.

[99] R. Y. K. Fung, J. Tang, and D. Wang. Extension of a hybrid genetic algorithm for nonlinear programming problems with equality and inequality constraints. *Computers and Operations Research*, 29:261–274, 2002.

[100] V. B. Gantovnik, Z. Gürdal, L. T. Watson, and C. M. Anderson-Cook. Genetic algorithm for mixed integer nonlinear programming problems using separate constraint approximations. *AIAA Journal*, 43:1844–1849, 2005.

[101] C. Gebhardt. *Bayessche Methoden in der geostatischen Versuchsplanung.* PhD thesis, Universität Klagenfuhrt, 2003.

[102] A. M. Geoffrion. Generalized Benders decomposition. *Journal of Optimization Theory and Applications*, 10:237–260, 1972.

[103] Gesellschaft für Schwerionenforschung mbH, GSI. FAIR - facility for antiproton and ion research. http://www.gsi.de/fair, 2008.

[104] P. E. Gill, W. Murray, and M. H. Wright. *Practical optimization*. Academic Press, London, 1981.

[105] P. E. Gill, W. Murray, and M. A. Saunders. User's guide for SNOPT 7.1: a Fortran package for large-scale nonlinear programming. Technical Report Report NA 05-2, Department of Mathematics, University of California, San Diego., 2005.

[106] P. E. Gill, W. Murray, and M. H. Wright. SNOPT: An SQP algorithm for large-scale constrained optimization. *SIAM Review*, 47:99–131, 2005.

[107] P. Gilmore and C. T. Kelley. An implicit filtering algorithm for optimization of functions with many local minima. *SIAM Journal on Optimization*, 5:269–285, 1995.

[108] A. A. Giunta. Use of data sampling, surrogate models, and numerical optimization in engineering design. In *Proceedings of the 40th AIAA Aerospace Sciences Meeting and Exhibit*, number AIAA-2002-0538, Reno, NV, January 2002.

[109] A. A. Giunta and M. S. Eldred. Implementation of a trust region model management strategy in the DAKOTA optimization toolkit. In *Proceedings of the 8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, Long Beach, CA., September 6-8 2000. AIAA-2000-4935.

[110] A. A. Giunta, S. Wojtkiewicz jr., and M. S. Eldred. Overview of modern design of experiments methods for computational simulations. In *Proceedings of the 41st AIAA Aerospace Sciences Meeting and Exhibit Reno NV: American Institute of Aeronautics and Astronautics*, number AIAA-2003-0649, 2003.

[111] T. Glad and A. Goldstein. Optimization of functions whose values are subject to small errors. *BIT*, 17:160–169, 1977.

[112] K. Glanzhorn. Industrielle Innovation und Informationstechnik. *IBM Nachrichten*, pages 21–27, 1982.

[113] F. Glover. Future paths for integer programming and links to artificial intelligence. *Computers and Operations Research*, 13:533–549, 1986.

[114] F. Glover, J. P. Kelly, and M. Laguna. New advances for wedding optimization and simulation. In *Proceedings of the 1999 Winter Simulation Conference*, 1999.

[115] G. Gray and T. G. Kolda. Algorithm 856: Appspack 4.0: asynchronous parallel pattern search for derivative-free optimization. *ACM Trans. Math. Softw.*, 32:485–507, 2006.

[116] A. Griewank. *Evaluating Derivatives*. SIAM, 2000.

[117] J. D. Griffin and T. G. Kolda. Asynchronous parallel generating set search for linearly-constrained optimization. Technical report, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, July 2006.

[118] I. E. Grossmann. Review of nonlinear mixed-integer and disjunctive programming techniques. *Optimization and Engineering*, 3:227–252, 2002.

[119] O. K. Gupta and A. Ravindran. Branch and bound experiments in convex nonlinear integer programming. *Management Science*, 31:1533–1546, 1985.

[120] H.-M. Gutmann. A radial basis function method for global optimization. *Journal of Global Optimization*, 19:201–227, 2001.

[121] B. Guyaguler and R. Horne. Optimization of well placement. *Journal of Energy Resources Technology*, 122:64–70, 2000.

[122] A. Harbaugh and M. McDonald. *Users Documentation for MODFLOW-96, an update to the U.S. Geological Survey Modular Finite-Difference Ground-Water Flow Model.* U.S. Geological Survey, Reston, Virginia, US, open-file report 96-485 edition, 1996.

[123] M. Hardt and O. von Stryk. Dynamic modeling in the simulation, optimization and control of legged robots. *ZAMM: Zeitschrift für Angewandte Mathematik und Mechanik*, 83:648–662, 2003.

[124] J. P. Hardwick and Q. F. Stouta. Response adaptive designs that incorporate switching costs and constraints. *Journal of Statistical Planning and Inference*, 137:2654–2665, 2007.

[125] M. Hebbel, W. Nistico, and D. Fisseler. Learning in a high dimensional space: Fast omnidirectional quadrupedal locomotion. In *RoboCup 2006: Robot Soccer World Cup X, Lecture Notes in Artificial Intelligence, Springer, 2006*, 2006.

[126] T. Hemker, K. R. Fowler, and O. von Stryk. Derivative-free optimization methods for handling fixed costs in optimal groundwater remediation design. In *Proc. of the CMWR XVI - Computational Methods in Water Resources*, 19-22 June 2006.

[127] T. Hemker, M. Glocker, H. D. Gersem, O. von Stryk, and T.Weiland. Mixed-integer simulation-based optimization for a superconductive magnet design. In *Proc. of the Sixth International Conference on Computation in Electromagnetics*. VDE, 4-6 April 2006.

[128] T. Hemker, H. Sakamoto, M. Stelzer, and O. von Stryk. Hardware-in-the-loop optimization of the walking speed of a humanoidrobot. In *CLAWAR 2006: 9th International Conference on Climbing and Walking Robots*, pages 614–623, Brussels, Belgium, September 11-14 2006.

[129] T. Hemker, H. D. Gersem, O. von Stryk, and T. Weiland. Mixed-integer nonlinear design optimization of a superconductivemagnet. *IEEE Transactions on Magnetics*, 44:1110–1113, 2007.

[130] T. Hemker, O. von Stryk, H. De Gersem, and T. Weiland. Simulation-based design improvement of a superconductive magnet by mixed-integer nonlinear surrogate optimization. In *16th Intl. Conf. on the Computation of Electromagnetic Fields - Compumag 2007*, pages 449–450, Aachen, Germany, June 24-28 2007.

[131] T. Hemker, K. R. Fowler, M. W. Farthing, and O. von Stryk. A mixed-integer simulation-based optimization approach with surrogate functions in water resources management. *Optimization and Engineering*, 9:341–360, 2008.

[132] T. Hemker, H. Sakamoto, M. Stelzer, and O. von Stryk. Efficient walking speed optimization of a humanoid robot. *The International Journal of Robotics Research*, 28:303–314, 2009.

[133] J. Holland. *Adaption In Natural and Artificial Systems.* University of Michigan Press, 1975.

[134] K. Holmberg. On the convergence of cross decomposition. *Mathematical Programming*, 47: 269ff, 1990.

[135] K. Holmström, N.-H. Quttineh, and M. M. Edvall. An adaptive radial basis algorithm (ARBF) for expensive black-box mixed-integer constrained global optimization. *Optimization and Engineering*, 9:311–339, 2008.

[136] R. Hooke and T. A. Jeeves. "Direct search" solution of numerical and statistical problems. *Journal of the Association of Computing Machinery*, 8:212–229, 1961.

[137] S. Hosder, L. T. Watson, B. Grossman, W. H. Mason, R. T. Haftka, and S. E. Cox. Polynomial response surface approximations for the multidisciplinarydesign optimization of a high speed civil transport. *Optimization and Engineering*, 2:431–452, 2001.

[138] C. T. Hsiao and L. C. Chang. Dynamic optimal groundwater management with inclusion of fixed costs. *Journal of Water Resources Planning and Management*, 128:57–64, 2002.

[139] C. Huang and A. S. Mayer. Pump-and-treat optimization using well locations and pumping rates as decision variables. *Water Resources Research*, 33:1001–1012, 1997.

[140] J. Z. Huang. Functional ANOVA models for generalized regression. *Journal of Multivariate Analysis*, 67:49–71, 1998.

[141] B. Husslage, E. van Dam, D. den Hertog, P. Stehouwer, and E. Stinstra. Collaborative metamodeling: Coordinating simulation-based product design. *Concurrent Engineering*, 11:267–278, 2003.

[142] W. Huyer and A. Neumaier. SNOBFIT stable noisy optimization by branch and fit. *ACM Transactions on Mathematical Software*, 35:9:1 – 9:25, 2008.

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

[144] R. Jin, X. Du, and W. Chen. The use of metamodeling techniques for optimization under uncertainty. *Structural and Multidiscipilnary Optimization*, 25:99–116, 2003.

[145] R. Jin, W. Chen, and A. Sudjianto. Analytical metamodel-based global sensitivity analysis and uncertainty propagation for robust design. *SAE Transactions: Journal of Materials & Manufacturing*, 113:121–128, 2005.

[146] D. R. Jones. Direct global optimization algorithm. In C. A. Floudas and P. M. Pardalos, editors, *Encyclopedia of Optimization*, pages 431–440. Kluwer Academic Publishers, 2001.

[147] D. R. Jones. A taxonomy of global optimization methods based on response surfaces. *Journal of Global Optimization*, 21:345–383, 2001.

[148] D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *J. Optim. Theory Appl.*, 79:157–181, 1993. ISSN 0022-3239.

[149] 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.

[150] C. T. Kelley. *Iterative Methods for Optimization*. SIAM, 1999.

[151] J. Kennedy and R. Eberhart. Particle swarm optimization. *IEEE Conference on Neural Networks*, 4:1942–1948, 1995.

[152] A. C. Keys and L. P. Rees. A sequential-design metamodeling strategy for simulation optimization. *Computers and Operations Research*, 31:19111932, 2004.

[153] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by simulated annealing. *Science*, 220:671–680, 1983.

[154] J. P. C. Kleijnen. *Statistical Tools for Simulation Practitioners*. Marcel Dekker, New York, 1987.

[155] J. P. C. Kleijnen and W. van Groenendaal. *Simulation a statistical perspective*. Wiley, 1992.

[156] J. P. C. Kleijnen, D. den Hertog, and E. Angün. Response surface methodology steepest ascent and step size revisited. *European Journal of Operational Research*, 159:121–131, 2004.

[157] J. P. C. Kleijnen, S. S. Sanchez, T. W. Lucas, and T. M. Cioppa. State-of-the-art review: A users guide to the brave new world of designing simulation experiments. *Informs Journal on Computing*, 17:263–289, 2005.

[158] J. R. Koehler and A. B. Owen. Computer experiments. *Handbook of Statistics*, 13:261–308, 1996.

[159] J. R. Koehler, A. A. Puhalskii, and B. Simon. Estimating functions evaluated by simulation: A bayesian/analytic approach. *The Annals of Applied Probability*, 8:1184–1215, 1998.

[160] N. Kohl and P. Stone. Policy gradient reinforcement learning for fast quadrupedal locomotion. In *Proceedings of the IEEE International Conference on Robotics and Automation*, volume 3, pages 2619–2624, May 2004.

[161] M. Kokkolaras, C. Audet, and J. E. Dennis jr. Mixed variable optimization of the number and composition of heat intercepts in a thermal insulation system. *Optimization and Engineering*, 2:5–29, 2001.

[162] T. G. Kolda, K. Lewis, and V. Torczon. Optimization by direct search: New perspectives on some classical and modern methods. *SIAM Review*, 45:385–482, 2003.

[163] K. Kostikas and C. Fragakis. *Genetic Programming*, volume 3003/2004 of *Lecture Notes in Computer Science*, chapter Genetic Programming Applied to Mixed Integer Programming, pages 113–124. Springer Berlin / Heidelberg, 2004.

[164] S. Koziel and J. W. Bandler. Coarse and surrogate model assessment for engineering design optimization with space mapping. In *Microwave Symposium, IEEE/MTT-S International*, pages 107–110, 2007.

[165] S. Kurz, S. Russenschuck, and N. Siegel. Accurate calculation of fringe fields in the LHC main dipoles. *IEEE Trans. Appl. Supercond.*, 10:85–88, 2000.

[166] U. Lall and M. D. Santini. An optimization model for unconfined stratified aquifer systems. *Journal of Hydrology*, 111:145–162, 1989.

[167] A. M. Law and W. D. Kelton. *Simulation Modeling and Analysis*. McGraw-Hill, 1999.

[168] S. J. Leary, A. Bhaskar, and A. J. Keane. A constraint mapping approach to the structural optimization of an expensive model using surrogates. *Optimization and Engineering*, 2: 385–398, 2001.

[169] S. J. Leary, A. Bhaskar, and A. J. Keane. A derivative based surrogate model for approximating and optimizing the output of an expensive computer simulation. *Journal of Global Optimization*, 30:39–58, 2004.

[170] R. M. Lewis and V. Torczon. Pattern search algorithms for bound constrained minimization. *SIAM Journal on Optimization*, 9:1082–1099, 1999.

[171] R. M. Lewis and V. Torczon. Pattern search methods for linearly constrained minimization. *SIAM Journal on Optimization*, 10:917–941, 2000.

[172] R. M. Lewis and V. Torczon. A globally convergent augmented lagrangian pattern search algorithm for optimization with general constraints and simple bounds. *SIAM Journal on Optimization*, 12:1075–1089, 2002.

[173] R. M. Lewis, V. Torczon, and M. W. Trosset. Direct search methods: then and now. *Journal of Computational and Applied Mathematics*, 124:191–207, 2000.

[174] S. Leyffer. Integrating SQP and branch-and-bound for mixed-integer nonlinear programming. *Computational Optimization and Applications*, 18:295–309, 2001.

[175] F. Liebl. *Simulation. Problemorientierte Einführung*. Oldenbourg, 1995.

[176] G. Lombardi, G. Mengali, and F. Beux. A hybrid genetic based optimization procedure for aircraft conceptual analysis. *Optimization and Engineering*, 7:151–171, 2006.

[177] S. N. Lophaven, H. B. Nielsen, and S. Søndergaard. DACE, a matlab kriging toolbox. Technical Report IMM-TR-2002-12, IMM, August 2002.

[178] S. N. Lophaven, H. B. Nielsen, and S. Søndergaard. Aspects of the Matlab toolbox DACE. Technical Report IMM-TR-2002-13, IMM, August 2002.

[179] M. Lötzsch, M. Risler, and M. Jüngel. XABSL - a pragmatic approach to behavior engineering. In *Proceedings of IEEE/RSJ International Conference of Intelligent Robots and Systems (IROS)*, pages 5124–5129, Beijing, China, October 9-15 2006.

[180] J. M. Luca, W. J. Welch, R. J. Buck, J. Sacks, H. P. Wynn, M. D. Morris, and M. Schonlau. Comments on computer experiments, letter to the editor: Response to James M. Lucas. *Technometrics*, 38:197–203, 1996.

[181] S. Lucidi and V. Piccialli. A derivative-based algorithm for a particular class of mixed variable optimization problems. *Optimization Methods and Software*, 19:371–387, 2004.

[182] S. Lucidi, V. Piccialli, and M. Sciandrone. An algorithm model for mixed variable programming. *SIAM Journal on Optimization*, 15:1057–1084, 2005.

[183] K. Madsen and M. Langthjem. Multifidelity response surface approximations for the optimum designof diffuser flows. *Optimization and Engineering*, 2:453–468, 2001.

[184] K. Madsen and S. Søndergaard. Convergence of hybrid space mapping algorithms. *Optimization and Engineering*, 5:145–156, 2004.

[185] P. Manoonpong, T. Geng, T. Kulvicius, B. Porr, and F. Wörgötter. Adaptive, fast walking in a biped robot under neuronal control and learning. *PLoS Computational Biology*, 3:1–16, 2007.

[186] A. L. Marsden, M. Wang, J. E. Dennis jr., and P. Moin. Optimal aeroacoustic shape design using the surrogate managementframework. *Optimization and Engineering*, 5:235–262, 2004.

[187] J. D. Martin and T. W. Simpson. Use of adaptive metamodeling for design optimization. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, number AIAA 2002-5631, Atlanta, Georgia, 4-6 September 2002.

[188] J. D. Martin and T. W. Simpson. A study on the use of kriging models to approximate deterministic computer models. In *Proceedings of DETC'03*, 2003.

[189] S. Maskey, A. Jonoski, and D. P. Solomatine. Groundwater remediation strategy using global optimization algorithms. *Journal of Water Resources Planning and Management*, 128:431–440, 2002.

[190] G. Matheron. Principles of geostatistics. *Econom. Geol.*, 58:1246 – 1266, 1963.

[191] S. L. Mattot, S. L. Bartelt-Hunt, A. J. Rabideau, and K. R. Fowler. Application of heuristic optmization techniques and algorithm tuningto multi-layered sorptive barrier design. *Environmental Science and Technology*, 40:6354 –6360, 2006.

[192] H. Mawengkang and B. A. Murtagh. Solving nonlinear programs with large-scale optimization software. *Annals of Operations Research*, 5:425–437, 1985.

[193] A. S. Mayer, C. T. Kelley, and C. T. Miller. Optimal design for problems involving flow and transport in saturated porous media. *Advances in Water Resources*, 12:1233–1256, 2002.

[194] A. S. Mayer, C. T. Kelley, and C. T. Miller. Optimal design for problems involving flow and transport in saturated porous media, electronic supplement. *Advances in Water Resources*, 12:1233–1256, 2002.

[195] L. W. Mays. *Optimal Control of Hydrosystems*. M. Dekker, 1997.

[196] M. G. McDonald and A. W. Harbaugh. A modular three dimensional finite difference groundwater flow model. *U.S. Geological Survey Techniques of Water Resources Investigations*, 6:568, 1988.

[197] D. C. McKinney and M.-D. Lin. Approximate mixed-integer nonlinear programming methods foroptimal aquifer remediation design. *Water Resources Research*, 31:731–740, 1995.

[198] K. I. M. McKinnon. Convergence of nelder mead simplex method to a nonstationary point. *SIAM Journal on Optimization*, 9:148–159, 1999.

[199] M. Meckesheimer, R. R. Barton, T. Simpson, F. Limayem, and B. Yannou. Metamodeling of combined discrete/continuous responses. *AIAA Journal*, 39:1950–1959, 2001.

[200] Z. Michalewicz. Genetic algorithms, numerical optimization, and constraints. In L. Eshelman, editor, *Proceedings of the Sixth International Conference on Genetic Algorithms*, pages 151–158. Morgan Kaufmann, 1995.

[201] T. J. Mitchell. An algorithm for the construction of "D-optimal" experimental design. *Technometrics*, 16:203–210, 1974.

[202] R. E. Nance and R. G. Sargent. Perspectives on the evoluation of simulation. *Operations Research*, 50:161–172, 2002.

[203] J. A. Nelder and R. Mead. A simplex method for function minimization. *The Computer Journal*, 7:308, 1965.

[204] J. W. Nicklow. Discret-time optimal control for water resources engineering andmanagement. *International Water Resources Association*, 25:89–95, 2000.

[205] J. Niehaus, T. Röfer, and T. Laue. Gait optimization on a humanoid robot using particle swarm optimization. In *Humanoids*, 2007.

[206] J. Nocedal and S. J. Wright. *Numerical optimization*. Springer New York, 2006.

[207] I. Novak. Relaxation and decomposition methods for mixed integer nonlinearprogramming. Habilitationsschrift zur Erlangung der Lehrbefähigung für das Fach Mathematik eingereicht an der Mathematisch-Naturwissenschaftlichen Fakultät II Humboldt-Universität zu Berlin, 2004.

[208] J. T. Oden, T. Belytschko, J. Fish, H. T. J. R., C. Johnson, D. Keyes, A. Laub, L. Petzhold, D. Srolovitz, and S. Yip. A report of the national science foundation blue ribbon panel on simulation-based engineering science - revolutionizing engineering science through simulation. Technical report, National Science Foundation Blue Ribbon Panel on Simulation-Based Engineering Science, 2006.

[209] J. C. Otto, M. Paraschivoiu, S. Yeş, and A. T. Patera. Computer-simulation surrogates for optimization: Application to trapezoidal ducts and axisymmetric bodies. In *Proc of the Forum on CFD for DEsign and Optimization, ASME InternationalMechanical Engineering Conference and Exposition*, 1995.

[210] A. B. Owen. Orthogonal arrays for computer experiments, integration and visualization. *Statistica Sinica*, 2:439–452, 1992.

[211] B. Page, H. Liebert, and A. Heymann. *Diskrete Simulation Eine Einfhrung mit Modula-2*. Springer, 1991.

[212] U. Pahner, R. Mertens, H. De Gersem, R. Belmans, and K. Hameyer. A parametric finite element environment tuned for numerical optimization. *IEEE Transactions on Magnetics*, 34:2939–2939, 1998.

[213] U. Pahner, R. Belmans, and K. Hameyer. Optimizing ferromagnetic characteristics to reduce Newton iterations. *COMPEL*, 18:629–637, 1999.

[214] M. P. Papadopoulou, G. F. Pinder, and G. P. Karatzas. Enhancement of the outer approximation method for the solution of concentration constrained optimal-design groundwater-remediation problems. *Water Resources Research*, 39:1185, 2003.

[215] J. H. Park and M. Choi. Generation of an optimal gait trajectoriy for biped robots usinga genetic algorithm. *Japan Society of Mechanical Engineers International Journal*, 47:715–721, 2004.

[216] M. D. Parno, K. R. Fowler, and T. Hemker. Particle swarm optimization with surroagate functions. Technical Report TUD-CS-2009-0139, Technische Universität Darmstadt, Department of Computer Science, 2009.

[217] T. Q. Pham. Optiy - demo und dokumentation, version 2.3. www.optiy.de, 2007.

[218] M. J. D. Powell. Direct search algorithms for optimization calculations. *Acta Numerica*, 7: 287–336, 1998.

[219] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and R. P. K. Tucker. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences*, 41:1–28, 2005.

[220] I. Quesa and I. E. Grossmann. An LP/NLP based branch and bound algorithm for convex MINLP optimizationproblems. *Computers and Chemical Engineering*, 16:937–947, 1992.

[221] A. Rasch. *Efficient Computation of Derivatives for Optimal Experimental Design.* Dissertation, Department of Computer Science, RWTH Aachen University, 2007.

[222] A. Ratle. Kringing as a surrogate fitness landscape in evoluationary optimization. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, 15:37–49, 2001.

[223] P. Reed, B. Minsker, and D. E. Goldberg. Designing a competent simple genetic algorithm for search and optimization. *Water Resources Research*, 36:3757–3761, 2000.

[224] R. G. Regis and C. A. Shoemaker. Constrained global optimization of expensive black box functions using radial basis functions. *Journal of Global Optimization*, 31:153–171, 2005.

[225] X. Ren and B. Minsker. Which groundwater remediation objective is better: A realistic one or a simple one? *Journal of Water Resources Planning and Management*, 131:351–361, 2005.

[226] J. J. M. Rijpkema, L. F. P. Etman, and A. J. G. Schoofs. Use of design sensitivity information in response surface and kriging metamodels. *Optimization and Engineering*, 2: 469–484, 2001.

[227] B. J. Ritzel, J. W. Eheart, and S. Ranjithan. Using genetic algorithms to solve a multiple objective groundwater pollution containment problem. *Water Resources Research*, 30:1589–1604, 1994.

[228] R. E. Robertson and R. Schwertassek. *Dynamics of Multibody Systems.* Springer, 1988.

[229] T. Röfer. Evolutionary gait-optimization using a fitness function based onproprioception. In *RoboCup 2004: Robot Soccer World Cup VIII, Lecture Notes in Artificial Intelligence. Springer*, pages 310–322, 2005.

[230] R. K. Roy. *Design of Experimence using the Taguchi Approach.* Wiley-Interscience, 2001.

[231] J. Sacks, S. B. Schiller, and W. J. Welch. Design for computer experiments. *Technometrics*, 31:41–47, 1989.

[232] A. Saltelli, K. Chan, and E. M. Scott. *Sensitivity Analysis.* John Wiley, 2000.

[233] M. Sasena. *Flexibility and Efficiency Enhancements for Constrained Global Design Optimization with Kriging Approximations*. PhD thesis, University of Michigan, 2002.

[234] M. Sasena, P. Y. Papalambros, and P. Goovaerts. Global optimization of problems with disconnected feasible regions via surrogate modeling. In *9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization*, number AIAA-2002-5573, Atlanta, Georgia, September 2002.

[235] M. Sasena, P. Y. Papalambros, and P. Goovaerts. Exploration of metamodeling sampling criteria for constrained global optimization. *Engineering Optimization*, 34:263–278, 2002.

[236] M. Sayeed and G. K. Mahinthakumar. Efficient parallel implementation of hybrid optmization approaches for solving groundwater inverse problems. *Journal of Computing in Civil Engineering*, 19(4):329–340, 2005.

[237] M. Schonlau. *Computer Experiments and Global Optimization*. PhD thesis, University of Waterloo, Waterloo, Canada, 1997.

[238] M. Schonlau, W. J. Welch, and D. R. Jones. Global versus local search in constrained optimization of computer models. Technical Report Technical Report RR-97-11, Institute for Improvement in Quality and Productivity, University of Waterloo, Waterloo, Ontario, Canada, December 1997.

[239] A. Seyfarth, R. Tausch, M. Stelzer, F. Iida, A. Karguth, and O. von Stryk. Towards bipedal jogging as a natural result of optimizing walking speed for passively compliant three-segmented legs. In *CLAWAR 2006: International Conference on Climbing and Walking Robots, Brussels, Belgium, Sept. 12-14*, 2006.

[240] R. E. Shannon. *Systems simulation: the art and science*. Prentice-Hall, 1975.

[241] H. J. Shieh and R. Peralta. Optimal in situ bioremediation design by hybrid genetic algorithm and simulated annealing. *Water Resources and Planning Management*, 131(1): 67–78, 2005.

[242] P. Siarry and Z. Michalewicz. *Advances in Metaheuristics for Hard Optimization (Natural Computing)*. Springer, Berlin, 2008.

[243] C. Siefert, V. Torczon, and M. W. Trosset. Model-assisted pattern search methods for optimizing expensive computer simulations. In *Proceedings of the Section on Physical and Engineering Sciences,American Statistical Association*, 2002.

[244] H.-J. Siegert. *Simulation zeitdiskreter Systeme*. Oldenbourg, 1991.

[245] T. Simpson, T. M. Maurery, J. J. Korte, and F. Mistree. Kriging models for global approximation in simulation-based multidisciplinary design optimization. *AIAA Journal*, 39: 2233–2241, 2001.

[246] T. M. Simpson, T. W. Maurery, J. J. Korte, and F. Mistree. Comparison of response surface and kriging models for multidisciplinary design optimization. In *7th AIAA/USAF/ISSMO Symposium on Multidisciplinary Analysis & Optimization*, number AIAA-98-4755. Institute of Aeronautics and Astronautics, September 1998.

[247] T. W. Simpson, D. K. J. Lin, and W. Chen. Sampling strategies for computer experiments: design and analysis. *International Journal of Reliability and Application*, 2:209–240, 2001.

[248] T. W. Simpson, J. D. Peplinski, P. N. Koch, and J. K. Allen. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers*, 17: 129–150, 2001.

[249] T. W. Simpson, A. J. Booker, D. Ghosh, A. A. Giunta, P. N. Koch, and R. J. Yang. Approximation methods in multidisciplinary analysis and optimization: a panel discussion. *Structural and Multidisciplinary Optimization*, 27:302–313, 2004.

[250] E. Sinha, B. Minsker, and M. Babbar. Multiscale island injection genetic algorithm for ground water remediation. In *Proceedings of the 2004 World Water and Environmental Resources Congress*, Salt Lake City, UT, US, June 27.July 1, 2004 2004.

[251] M. Smith. *Neural Networks for Statistical Modeling*. John Wiley & Sons, Inc., New York, NY, USA, 1993.

[252] A. Sóbester, S. J. Leary, and A. J. Keane. A parallel updating scheme for approximating and optimizing highfidelity computer simulations. *Structural and Multidisciplinary Optimization*, 27:371–383, 2004.

[253] J. Søndergaard. *Optimization Using Surrogate Models - by the Space Mapping Technique*. PhD thesis, Technical University of Denmark, 2003.

[254] W. Song and A. J. Keane. Parameter screening using impact factors and surrogate-based ANOVA techniques. In *11th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference*, number AIAA 2006-7088, 2006.

[255] Sony Entertainment Robot Europe. Aibo support homepage. http://support.sony-europe.com, 2007.

[256] A. Srivastava, K. Hacker, K. Lewis, and T. W. Simpson. A method for using legacy data for metamodel-based design of large-scalesystems. *Structural and Multidiscipilnary Optimization*, 28:146–155, 2004.

[257] H. P. Stehouwer and D. den Hertog. Simulation-based design optimization: methodology and applications. In *Proc. 1st ASMO UK/ISSMO Conf. Engineering Design Optimization*, pages 349–355, Ilkley, U.K., 1999.

[258] M. Stelzer, M. Hardt, and O. von Stryk. Efficient dynamic modeling, numerical optimal control and experimental results for various gaits of a quadruped robot. In *CLAWAR 2003: International Conference on Climbing and Walking Robots, Catania, Italy, Sept. 17-19*, pages 601–608, 2003.

[259] B. Stuckman, G. Evans, and M. Mollaghasemi. Comparison of global search methods for design optimization using simulation. In *Proceedings of the 1991 Winter Simulation Conference*, 1991.

[260] T. Stützle. *Local Search Algorithm for Combinatorial Problems - Analysis, Algorithms, and New Applications*. dissertation, Technische Universität Darmstadt, 1998.

[261] W. Suleiman, E. Yoshida, J.-P. Laumond, and A. Monin. On humanoid motion optimization. In *IEEE-RAS 7th International Conference on Humanoid Robots - IEEE-RAS 7th International Conference on Humanoid Robots*, Pittsburgh, Pennsylvania, United States, 2007.

[262] E. Tekim and I. Sabuncuoglu. Simulation optimization: A comprehensive review on theory and applications. *IIE Transactions*, 36:1067–1081, 2004.

[263] V. Torczon. *Multi-Directional Search: A Direct Search Algorithm for ParallelMachines*. PhD thesis, Department of Mathematical Sciences, Rice University, Houston, Texas, 1989.

[264] V. Torczon. On the convergence of the multidirectional search algorithm. *SIAM Journal on Optimization*, 1:123–145, 1991.

[265] V. Torczon. On the convergence of pattern search algorithms. *SIAM Journal on Optimization*, 7:1–25, 1997.

[266] V. Torczon and M. W. Trosset. Using approximations to accelerate engineering design optimization. In *AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis andOptimization, 7th, St. Louis, MO, Sept. 2-4, 1998, Collection of Technical Papers.*, 1998.

[267] I. C. Trelea. The particle swarm optimization algorithm: convergence analysis and parameter selection. *Information Processing Letters*, 85:317–325, 2003.

[268] M. Trosset, N. Alexandrov, and L. Watson. New methods for robust design using computer simulations. In *American Statistical Association. Proceedings of the Section on Physicaland Engineering Science.*, 2003.

[269] M. W. Trosset and V. Torczon. Numerical optimization using computer experiments. Technical Report TR-97-38, Institute for Computer Applications in Science and Engineering (ICASE), 1997.

[270] C. J. Turner, M. I. Campbell, and R. H. Crawford. Generic sequential sampling for metamodel approximations. In *Proceedings of DETC03: ASME 2003 Design Engineering Technical Conferences and Computers and Information in Engineering Conference*, Chicago, Illinois, September 2-6 2003.

[271] W. van Beers and J. P. C. Kleijnen. Kriging interpolation in simulation: A survey. In *Proceedings of the 2004 Winter Simulation Conference*, 2004.

[272] R. Vitali. *Response Surface Methods For High Dimensional Structural Design Problems*. PhD thesis, University of Florida, 2000.

[273] O. von Stryk and R. Bulirsch. Direct and indirect methods for trajectory optimization. *Annals of Operations Research*, 37:357–373, 1992.

[274] G. G. Wang and S. Shan. Review of metamodeling techniques in support of engineering design optimization. *Transactions of the American Society of Mechanical Engineers*, 129: 370–380, 2007.

[275] D. Watkins jr. and D. C. McKinney. Decomposition methods for water resources optimization models with fixed costs. *Advances in Water Resources*, 21:261–324, 1998.

[276] J. D. Weingarten, G. A. D. Lopes, M. Buehler, R. E. Groff, and D. E. Koditschek. Automated gait adaptation for legged robots. In *Proceedings of IEEE Int. Conf. Robotocs and Automation (ICRA)*, 2004.

[277] T. Westerlund and F. Pettersson. An extended cutting plane method for solving convex minlp problems. *Computers and Chemical Engineering*, 19:131–135, 1995.

[278] E. R. Westervelt and J. W. Grizzle. Design of asymptotically stable walking for a 5-link planar biped walker via optimization. In *IEEE Int. Conf. on Robotics and Automation*, pages 3117–3122, Washington D.C., 2002.

[279] C. Zhang, A. Sheng, and R. O. nez. Notes on the convergence and applications of surrogate optimization. *Discrete and Continuous Dynamical Systems*, Supplement:947–956, 2005.

[280] C. Zheng and P. Wang. *MT3DMS: A Modular Three-Dimensional Multispecies Transport Model for Simulation of Advection, Dispersion, and Chemical Reactions of Contaminants in Groundwater Systems; Documentation and Users Guide.* Department of Geological Sciences, University of Alabama, Tuscaloosa, AL 35487, US, 1999.

[281] C. Zheng, M. Hill, and P. Hsieh. *MODFLOW-2000, The U.S. Geological Survey Modular Ground-Water ModelUserGuide to the LMT6 Package, the Linkage with MT3DMS For Multi-Species Mass Transport Modeling.* U.S. Department of the Interior, U.S. Geological Survey, 2001.

[282] Z. Zhou, Y. S. Ong, and P. B. Nair. Hierarchical surrogate-assisted evolutionary optimization framework. *Evolutionary Computation*, 2:1586–1593, 2004.

[283] Z. Zhou, Y. S. Ong, P. B. Nairy, A. J. Keaney, and K. Y. Lum. Combining global and local surrogate models to accelerate evolutionary optimization. *IEEE transactions on systems, man and cybernetics. Part C, Applications and reviews*, 37:66–76, 2007.

[284] J. Ziegler. *Evolution von Laufrobotersteuerungen mit Genetischer Programmierung.* PhD thesis, TU Dortmund, 2003.

[285] E. Zitzler, K. Deb, and L. Thiele. Comparison of multiobjective evolutionary algorithms: Empirical results. *Evolutionary Computation Journal*, 8:173–195, 2000.

Glossary

AD Abbr., Automatic Differentiation, 11

AMMO Abbr., First-Order Approximation/Model Management Optimization, 10

ANOVA Abbr., Analysis of Variances, 24, 25, 40

CPs Abbr., Community Problems, the CPs are a set of subsurface flow benchmark problem for derivative free optimization, 63

DACE Abbr., Design and Analysis of Computer Experiments, 20, 21, 23, 24, 37–39, 45, 46, 53, 54

DFO Abbr., Derivative Free Optimization, 3, 4, 12, 13, 15, 16, 18, 24, 25, 74

DIRECT Abbr., Dividing Rectangular Algorithm, 17, 18, 24, 54, 55

DOE Abbr., Design of Experiments, 22, 42, 43, 48, 60

DOF Abbr., Degree of Freedom, 81, 89

DS Abbr., Direct Search, 16

EA Abbr., Evolutionary Algorithm, 13, 14, 24, 47, 59–61, 79

FEFLOW Commercial subsurface flow simulation, 62

GBD Abbr., Generalized Benders Decomposition, 27, 29

GA Abbr., Genetic Algorithm, 13, 14, 24, 25, 28, 47, 63, 68–70, 73

GPS Abbr., Generalized Pattern Search, 16, 17

GSS Abbr., Generalized Set Search, 16

IFFCO IFFCO is an implicit filtering implementation made for constrained derivative free optimization, available in Matlab and Fortran, 67, 70, 72, 73

Matlab Software package for numerical operations, `http://www.mathworks.com`, 53, 54

MBS Abbr., Multi-Body System, 77

MINLP Abbr., Mixed-Integer Nonlinear Programming, 3–5, 13, 17, 25–31, 36, 37, 40, 59–61, 66, 70, 72, 73

MILP Abbr., Mixed-Integer Linear Programming, 26–28

MODFLOW U.S. Geological Survey code for subsurface flow simulation, 62, 63, 65, 66, 70, 73, 74

MSE Abbr., Mean Squared Error, 21, 23, 45, 88

MVP Abbr., Mixed Variable Problem, 29

NLP Abbr., Nonlinear Programming, 17, 26–28, 40, 41, 59, 61, 62, 68, 70

ODE Abbr., Ordinary Differential Equation, 11

OA Abbr., Outer Approximation, 27, 62

PSO Abbr., Particle Swarm Optimization, 15, 25

FE Abbr., Finite Element, 51, 52, 56, 57

PDE Abbr., Partial Differential Equation, 11, 56

RBF Abbr., Radial Basis Function, 13, 20, 21, 23, 29

RoboCup For more information on the different leagues and robots see `www.robocup.org`, 78–80, 82, 86

RSM Abbr., Response Surface Methods, 20, 21, 24, 38

SMF Abbr., Surrogate Management Framework or Space Mapping Framework, 10, 24

SNOPT SNOPT is an SQP implementation in Fortran for large-scale NLP problems, 41

SQP Abbr., Sequential Quadratic Programming, 29, 40, 41, 53, 80

ZMP Abbr., Zero Moment Point, 82, 83