

**A stochastic algorithm for the identification of solution spaces in
high-dimensional design spaces**

Inauguraldissertation

zur

Erlangung der Würde eines Doktors der Philosophie

vorgelegt der

Philosophisch-Naturwissenschaftlichen Fakultät

der Universität Basel

von

Lavinia Graff

aus Neunkirchen (Saar), Deutschland

München, 2013

Originaldokument gespeichert auf dem Dokumentenserver der Universität Basel

edoc.unibas.ch



Dieses Werk ist unter dem Vertrag "Creative Commons Namensnennung-Keine kommerzielle Nutzung-Keine Bearbeitung 2.5 Schweiz" lizenziert. Die vollständige Lizenz

kann unter

creativecommons.org/licences/by-nc-nd/2.5/ch

eingesehen werden.

Genehmigt von der Philosophisch-Naturwissenschaftlichen Fakultät
auf Antrag von

Prof Dr. Helmut Harbrecht und Prof. Dr. Rolf Krause

Basel, den 18.06.2013

Prof. Dr. Jörg Schibler



Namensnennung-Keine kommerzielle Nutzung-Keine Bearbeitung 2.5 Schweiz

Sie dürfen:



das Werk vervielfältigen, verbreiten und öffentlich zugänglich machen

Zu den folgenden Bedingungen:



Namensnennung. Sie müssen den Namen des Autors/Rechteinhabers in der von ihm festgelegten Weise nennen (wodurch aber nicht der Eindruck entstehen darf, Sie oder die Nutzung des Werkes durch Sie würden entlohnt).



Keine kommerzielle Nutzung. Dieses Werk darf nicht für kommerzielle Zwecke verwendet werden.



Keine Bearbeitung. Dieses Werk darf nicht bearbeitet oder in anderer Weise verändert werden.

- Im Falle einer Verbreitung müssen Sie anderen die Lizenzbedingungen, unter welche dieses Werk fällt, mitteilen. Am Einfachsten ist es, einen Link auf diese Seite einzubinden.
- Jede der vorgenannten Bedingungen kann aufgehoben werden, sofern Sie die Einwilligung des Rechteinhabers dazu erhalten.
- Diese Lizenz lässt die Urheberpersönlichkeitsrechte unberührt.

Die gesetzlichen Schranken des Urheberrechts bleiben hiervon unberührt.

Die Commons Deed ist eine Zusammenfassung des Lizenzvertrags in allgemeinverständlicher Sprache: <http://creativecommons.org/licenses/by-nc-nd/2.5/ch/legalcode.de>

Haftungsausschluss:

Die Commons Deed ist kein Lizenzvertrag. Sie ist lediglich ein Referenztext, der den zugrundeliegenden Lizenzvertrag übersichtlich und in allgemeinverständlicher Sprache wiedergibt. Die Deed selbst entfaltet keine juristische Wirkung und erscheint im eigentlichen Lizenzvertrag nicht. Creative Commons ist keine Rechtsanwalts-gesellschaft und leistet keine Rechtsberatung. Die Weitergabe und Verlinkung des Commons Deeds führt zu keinem Mandatsverhältnis.

Abstract

The volume of an axis-parallel hyperbox in a high-dimensional design space is to be maximized under the constraint that the objective values of all enclosed designs are below a given threshold. The hyperbox corresponds to a Cartesian product of intervals for each input parameter. These intervals are used to assess robustness or to identify relevant parameters for the improvement of an insufficient design.

A related algorithm which is applicable to any non-linear, high-dimensional and noisy problem with uncertain input parameters is presented and analyzed. Analytical solutions for high-dimensional benchmark problems are derived. The numerical solutions of the algorithm are compared with the analytical solutions to investigate the efficiency of the algorithm. The convergence behavior of the algorithm is studied. The speed of convergence decreases when the number of dimensions increases. An analytical model describing this phenomenon is derived. Relevant mechanisms are identified that explain how the number of dimensions affects the performance. The optimal number of sample points per iteration is determined depending on the preference for fast convergence or a large volume. The applicability of the method to a high-dimensional and non-linear engineering problem from vehicle crash analysis is demonstrated. Moreover, we consider a problem from a forming process and a problem from the rear passenger safety.

Finally, the method is extended to minimize the effort to turn a bad into a good design. We maximize the size of the hyperbox under the additional constraint that all parameter values of the bad design are within the resulting hyperbox except for a few parameter values. These parameters are called key parameters because they have to be changed to lie within their desired intervals in order to turn the bad into a good design. The size of the intervals represents the tolerance to variability caused, for example, by uncertainty. Two-dimensional examples are presented to demonstrate the applicability of the extended algorithm. Then, for a high-dimensional, non-linear and noisy vehicle crash design problem, the key parameters are identified. From this, a practical engineering solution is derived which would have been difficult to find by alternative methods.

Kurzfassung

Das Volumen einer achsenparallelen Hyperbox in einem hochdimensionalen Designraum soll maximiert werden unter der Nebenbedingung, dass die Zielfunktionswerte aller enthaltenen Designs kleiner als ein vorgegebener Grenzwert sind. Die Hyperbox entspricht einem kartesischen Produkt von Intervallen für jeden Eingangsparameter. Diese Intervalle werden verwendet, um Robustheit zu bewerten oder um relevante Parameter zur Verbesserung eines Designs, dessen Zielfunktionswert grösser als der vorgegebene Grenzwert ist, zu identifizieren.

Ein entsprechender Algorithmus, der auf beliebige, nichtlineare, hochdimensionale und verrauschte Probleme mit unsicheren Eingangsparametern anwendbar ist, wird präsentiert und analysiert. Analytische Lösungen für hochdimensionale Benchmarkprobleme werden hergeleitet. Die numerischen Lösungen des Algorithmus werden mit den analytischen Lösungen verglichen, um die Effizienz des Algorithmus zu bewerten. Das Konvergenzverhalten des Algorithmus wird untersucht. Die Konvergenzgeschwindigkeit nimmt mit ansteigender Dimensionsanzahl ab. Ein analytisches Modell wird entwickelt, welches dieses Phänomen beschreibt. Relevante Mechanismen werden identifiziert, die erklären, wie die Dimensionsanzahl die Performance beeinflusst. Die optimale Anzahl an Stichproben pro Iteration wird bestimmt, abhängig davon, ob man schnelle Konvergenz oder ein grosses Volumen bevorzugt. Die Anwendbarkeit der Methode auf ein hochdimensionales und nichtlineares Ingenieursproblem aus der Fahrzeugcrashanalyse wird gezeigt. Zudem betrachten wir ein Problem des Tiefziehprozesses und des Schutzes der Insassen im Fond.

Schliesslich erweitern wir die Methode, um den Aufwand dafür zu reduzieren, ein schlechtes in ein gutes Design zu ändern. Wir maximieren das Volumen der Hyperbox unter der zusätzlichen Nebenbedingung, dass alle Parameterwerte des schlechten Designs in der Lösungshyperbox enthalten sind bis auf wenige Parameterwerte. Diese Parameter werden Stellhebel genannt, da sie so geändert werden müssen, dass ihre Werte in ihren gewünschten Intervallen liegen, um das schlechte Design in ein gutes zu ändern. Die Intervallbreite repräsentiert die Toleranz gegenüber Variabilität, die zum Beispiel durch Unsicherheit

erzeugt wird. Zweidimensionale Beispiele werden präsentiert, um die Anwendbarkeit des erweiterten Algorithmus zu zeigen. Wir identifizieren Stellhebel für ein hochdimensionales, nichtlineares und verrauschtes Fahrzeugcrashproblem. Daraus wird eine praktische Ingenieurslösung abgeleitet, die mit anderen Methoden schwierig zu finden wäre.

Contents

Abstract	VI
1 Introduction	1
2 Motivation and problem statement	7
2.1 Motivation	7
2.2 Problem statement	8
2.3 Known approaches	9
2.3.1 Cellular evolutionary strategy	10
2.3.2 Cluster analysis	14
2.3.3 Support vector machines	16
3 Algorithm	21
3.1 Identifying the initial hyperbox	22
3.2 Exploration phase	22
3.2.1 Sampling methods	24
3.2.2 Statistical evaluation	24
3.2.3 Cutting algorithm	25
3.2.4 Growing	33
3.3 Consolidation phase	37
3.3.1 Sampling methods	37
3.3.2 Statistical evaluation	37
3.3.3 Cutting algorithm	40

3.4	Extensions of the algorithm	40
3.4.1	Measures for the hyperbox	40
3.4.2	Sensitivity of the solution hyperbox	43
4	Results of the algorithm	47
4.1	Problem 1. Restricted Rosenbrock function as boundary	48
4.1.1	Analytical solution	48
4.1.2	Numerical solution	49
4.2	Problem 2. A convex polytope as boundary	51
4.2.1	Analytical solution	51
4.2.2	Numerical solution	56
4.3	Problem 3. A hyperbox as boundary	60
4.3.1	Analytical solution	60
4.3.2	Numerical solution	61
4.4	Problem 4. A tilted hyperplane as boundary	63
4.4.1	Analytical solution	64
4.4.2	Numerical solution	65
4.5	Corner problem	67
4.5.1	Numerical results	69
4.5.2	Dependence on the boundary of the good space	70
5	Convergence behavior in the consolidation phase	75
5.1	Convergence coefficient	76
5.2	Analytical model	77
5.3	Influence of the dimensionality	83
5.4	Influence of the number of sample points	84
5.4.1	Number of sample points versus convergence speed	84
5.4.2	Number of sample points versus volume	86
5.5	Convergence speed versus hyperbox volume	89

6 Applications	91
6.1 Front vehicle crash design	91
6.1.1 Evaluation	91
6.1.2 Crash simulation models	93
6.1.3 Finite element method	94
6.1.4 Example 1	96
6.1.5 Example 2	99
6.2 Forming process	108
6.2.1 Motivation	108
6.2.2 Evaluation	108
6.2.3 Response surface model	110
6.2.4 Example	111
6.3 Rear passenger safety	115
6.3.1 Motivation	115
6.3.2 Evaluation	116
6.3.3 Example	118
7 Identifying key parameters	123
7.1 Motivation	123
7.2 A simple example problem	125
7.3 General problem statement	128
7.4 Computing solution spaces with constraints	128
7.5 Analytical examples	129
7.5.1 Problem 1: Rosenbrock function	129
7.5.2 Numerical results of the simple example problem	132
7.6 High-dimensional crash problem	135
7.6.1 Why vehicle crash design is difficult	137
7.6.2 Application in crash design	139

8 Conclusion	145
A Theory of the optimization under constraints	149
A.1 Definitions	149
A.2 Theorems	152
B Extension of the analytical model	155
Bibliography	158

Chapter 1

Introduction

In many engineering problems, uncertainty is naturally present, especially in the early development phase. Uncertainty arises because some parameters cannot yet exactly be specified or they may be changed over the course of development. There is in general a lack of knowledge about the engineering system under consideration. Moreover, there is no knowledge about the variability of the input parameters. This type of uncertainty is called epistemic uncertainty since it is reducible if greater knowledge is provided, see [29, 46, 57, 75].

Classical optimization methods seek an optimum in the design space. Typically, they do not consider the variability of design variables and do thus not take into account uncertainty. Consequently, optimal designs may be non-robust and quite sensitive to parameter variabilities, and, therefore, infeasible for practical applications. Some authors even believe that optimization is actually just the opposite of robustness, see [48].

As reliability is required in industrial engineering, developers of engineering designs have to look for robust designs which avoid unexpected deviations from the nominal performance, see [61]. To this end, more advanced methods have been developed to include uncertainties of the parameters and robustness criteria in the optimization.

Robust design optimization (RDO), as introduced in [76], includes robustness measures in the optimization problem. RDO helps to obtain a design that is less sensitive to variations of uncontrollable input variables without eliminating the source of the uncertainty, see [28, 32, 63]. The impact of uncertainty or variation in the design parameters to the objective function value of a design is considered. RDO creates a robust design for problems whose objective function value is insensitive to uncertainties, see [2].

Reliability-based design optimization (RBDO) is a method to scale down the probability

of failure of the classical optimum. RBDO minimizes an objective function subject to probabilistic constraints which leads to a design feasibility under uncertainty. RBDO provides thus an optimal design in the presence of uncertainty. Methods of RBDO are, for example, the first and second order reliability method (FORM/SORM), see [86]. In RBDO, it is assumed that the complete information of the input uncertainties is known, see [28, 55, 72]. This means, if there exists an inherent randomness in the non-deterministic behavior of the physical system, i.e., aleatoric uncertainty, this uncertainty must be known and described. Aleatoric uncertainty is known to be irreducible, except through design modifications [46, 57, 75].

Sensitivity analysis (SA) provides another approach to deal with uncertainty. Sensitivity analysis is a method which estimates the variability of the objective function value, affected by the variability of the input parameters. It is a method to identify the parameters which have significant effects on the results, see [67]. Sensitivity analysis will give information about the effects of the uncertainty but requires appropriate sensitivity measures. Methods of determining such measures for each input parameter are, for example, ANOVA (analysis of variance) and the Sobol' method where Sobol' indices are calculated, see [59, 70].

Uncertainty also arises when more than one design team is involved in the design of an engineering development process and every design team must optimize their subsystem without full information about the other subsystems. Every team has its own individual subsystem with goals and constraints which must match the goals of the overall design. Furthermore, the different disciplines (e.g. in vehicle crash development, vibration analysis, durability, aerodynamic, etc.) may have conflicting objectives and the subsystems are often coupled, see [1, 42]. Some authors postulate that an appropriate method to solve such problems is *multidisciplinary design optimization (MDO)* because different disciplines are simultaneously optimized in MDO, see [10, 36].

Unfortunately, MDO, RBDO, RDO and SA suffer from certain disadvantages. For MDO, a model which comprises all relevant disciplines must be provided. Nevertheless, such a model is usually not available for the design of complex engineering systems where different teams are involved in the development process. RBDO and RDO deal with data where the variability of input parameters is known. However, if the uncertainties of input parameters are not completely known, other methods have to be used. When applying SA, information on how to improve a non-robust or critical solution is limited: what parameter needs to be adjusted and what value it should admit is unknown.

The method presented in this thesis identifies a maximum solution space for any high-dimensional, non-linear, and noisy system. The computed solution space is such that it

guarantees a subcritical objective function value (or performance/output) with a defined probability for all enclosed designs. The solution space is expressed by intervals for each input parameter. Therefore, the solution space will be a hyperbox, given by the Cartesian product of all the intervals to the input parameters. For a design to be good, the choice of a parameter value within its assigned interval does not depend on the values of the other parameters as long as they are within their respective intervals. In this sense, the parameters are decoupled from each other. The intervals may be used to assess robustness and sensitivity to uncertain input parameters which can be measured by the widths of the associated intervals. Moreover, a hyperbox helps to identify relevant parameters to improve a non-robust or bad design. They also may be combined with intervals of other disciplines – their cross sections are global solution spaces.

In the literature, there are already approaches which can be applied to identify a maximum hyperbox which includes only designs with subcritical objective function value. The first approach, which is studied in [66], identifies the sought hyperbox by a method which combines a cellular evolutionary strategy and interval arithmetic. However, this approach is not applicable to objective functions which are not given analytically. In the second approach, the sought hyperbox is identified by cluster analysis and fuzzy set theory (see [6, 60]). The drawbacks which arise with this approach are, first, that the fuzzy set theory needs some additional information like the membership function of the parameters, which is often not available in the engineering design development, and, second, that a very large number of sample points is required – especially in high dimensions – to identify the solution space due to only a single sampling procedure in the design space. The third approach which is proposed in [22] uses support vector machines to identify the maximum hyperbox within the solution space. However, hyperboxes can be only identified if the data are linearly separable. The three approaches are reviewed in detail in Subsections 2.3.1–2.3.3.

An iterative algorithm, consisting of two phases, is presented in this thesis for the iterative identification of the hyperbox described above, see Chapter 3. The algorithm was introduced in [87] and improved and analyzed in [24]. The algorithm is applicable to any high-dimensional, non-linear and noisy problem and requires no access to the analytical expression of the objective function.

The algorithm starts from a candidate hyperbox built around a design with subcritical objective function value, see Section 3.1. This design is identified by an algorithm called differential evolution (see [73]). Then, this candidate hyperbox is iteratively evaluated and modified.

In the first phase, called the *exploration phase*, the landscape of the optimization problem under consideration is explored as described in Section 3.2. This phase consists of four steps. In the first step, a design of experiments is performed (e.g. by Monte Carlo sampling, see [69]). The second step consists of a statistical evaluation of the candidate hyperbox by computing the ratio of the number of good sample points and the total number of sample points. In the third step, a subset is identified which contains only good designs of the original design space. For the third step, we propose two different algorithms. The first algorithm is called the *optimal cutting algorithm* because it identifies the maximum hyperbox which contains only good designs within the sample. Unfortunately, the computational costs are very expensive, especially in high dimensions. Therefore, we implement another algorithm which is very cheap, but not optimal. However, the agreement of the numerical results and the optimal solutions are reasonable which is confirmed in Chapter 4. This algorithm is called the *fast cutting algorithm*. In the fourth step, the hyperbox is moved through the design space in order to find the hyperbox with maximum volume. This is done by extending the candidate hyperbox in all parameter directions. The new boundaries are calculated by adding to the upper boundary the widths of the intervals multiplied by a given factor, and by subtracting from the lower boundary the widths of the intervals multiplied by the same factor. The factor is chosen such that the new candidate hyperbox is expected to contain a desired fraction of good sample points. The first phase is iterated until the hyperbox does not move any more.

Then, the second phase starts, called the *consolidation phase*. This phase consists of the application of the third step of the first phase, and an evaluation of the hyperbox by Bayesian statistics which estimates the fraction of good design points. Especially, it provides a confidence level of this estimate, see [45]. These steps are repeated until a hyperbox is identified which contains only subcritical outputs with a predefined probability. The similarity of the algorithm to on-line learning and query learning is discussed in [87]. The algorithm is similar to on-line learning because the candidate hyperbox is relocated in each iteration step and new sample points are created within the modified candidate hyperbox. In [18], on-line learning is introduced, and, in [15, 49], an example of on-line learning is given. Sample points are added iteratively which successively improves the support vector machines. In [13], a strategy for the efficient selection of support patterns by support vector machines is presented. Such a strategy is called query learning, cf. [18]. Our method is similar to query learning because we zoom into the good space, and we use only a few sample points in each iteration step. But there is a major difference between on-line learning or query learning and our method because our method does not only seek the boundary of the good space. Our method identifies the largest hyperbox within the

solution space.

The rest of this thesis is organized as follows. In Chapter 2, an example problem from the engineering practice is presented which motivates the need for maximum hyperboxes which guarantee a subcritical performance. The related problem statement is formulated. An overview on known approaches for the numerical solution is given, and the drawbacks are discussed.

In Chapter 3, the solution algorithm is proposed to identify the sought hyperboxes as described before for arbitrary non-linear, high-dimensional and noisy problems. Moreover, different measures of the resulting hyperbox are introduced, and a measure for the sensitivity of the solution hyperbox is shown.

In Chapter 4, four illustrative examples are given to demonstrate the efficiency of the proposed algorithm. The analytical solutions for these optimization problems are calculated. The numerical results are then compared with these analytical solutions. Moreover, the so-called *corner problem* is identified and investigated.

To analyze the reliability of the algorithm, the consolidation phase is studied in Chapter 5. The convergence behavior of the consolidation phase is studied. We derive an analytical model which describes the behavior of the speed of convergence for a benchmark problem. The relevant mechanisms which are related to the influence of the dimensionality and of the number of sample points are identified, and the optimal number of sample points per iteration is determined in dependence of the preference for speed or volume size.

In Chapter 6, different applications of the algorithm in the automotive industry are presented. The first application of the algorithm is an engineering problem from vehicle front crash design which confirms the applicability to high-dimensional and non-linear engineering models. Different hyperbox measures and the resulting sensitivity of the solution hyperbox are demonstrated at hand of this example. Then, the algorithm is applied to a forming process whose simulation is based on a response surface model. Finally, the application to the rear passenger safety is shown.

A procedure to identify key parameters with the aid of hyperboxes is presented in Chapter 7. A design which fails the design goals is improved by changing the key parameters in order to lie within the associated intervals.

In Chapter 8, some concluding remarks are given.

The appendix consisting of the Chapters A and B contains the theory of the optimization under constraints and an extension of the analytical model which is introduced in Chapter 5.

Finally, we remark that some parts of this thesis are already published. Some parts of the Chapters 2–6 are published in [24]. In [25], some parts of the Chapter 7 are published.

Acknowledgment

My hearty gratitude goes to my supervisor Prof. Dr. Helmut Harbrecht for his invaluable support, patience and trust regarding my work. He spent endless hours to discuss all different aspects of my thesis and to generate brilliant ideas. Moreover, he spared no effort to take it upon himself to travel monthly the long way between Basel and Munich. For this unbelievable support, I heartily thank. I have been amazingly fortunate to have a supervisor like him.

I would like to thank Prof. Dr. Rolf Krause for taking over the role of the co-referee.

Also, I thank Dr. Markus Zimmermann for his intense supervision at BMW. He spent a lot of time to discuss and to generate valuable ideas to approach both theoretical and applied problems. He helped me to connect theory and practice. Especially, I would like to thank for his engagement to succeed in ensuring that the developed method was applied in the projects of vehicle safety at BMW.

I am also very grateful to my colleagues at BMW who supported me by answering any of my questions about vehicles, crash tests and vehicle simulations: Dr. Frank Moeller and Dominik Schuster (my group leaders at BMW), Dr. Martin Doernfelder, Johannes Fender, Dr. Kathrin Grossenbacher, Johannes von Hasselbach, Johannes von Hoessle, Franz Hoiss, Patrick Semrau, Dr. Martin Unger, Florian Woelfle and all other supporting colleagues.

I would like to thank my friend and my friends. I greatly value their friendship and I deeply appreciate their belief in me.

My sincere and hearty gratitude goes to my mother Maria, my sister Geno and my brother Philipp for their love, support and patience.

Chapter 2

Motivation and problem statement

In this chapter, an example problem from the engineering practice is given to illustrate our problem setting which is then stated in Subsection 2.2. The problem statement is to identify a maximum hyperbox which guarantees a subcritical performance. For identifying numerically the maximum hyperbox which contains only subcritical designs, there are already approaches in the literature. We overview on these approaches and the drawbacks which arise with these approaches are discussed.

2.1 Motivation

As example problem, a model of a full-width front impact crash is considered, see Figure 2.1. The vehicle crashes head-on into a rigid concrete barrier at 56 km/h. In the vehicle development, the maximum deceleration of the vehicle generated by the vehicle structure is a relevant parameter to minimize the injury of car occupants in a front crash, see [83]. The deceleration time history is measured at the rocker panel and the B-pillar of the vehicle, see [34].

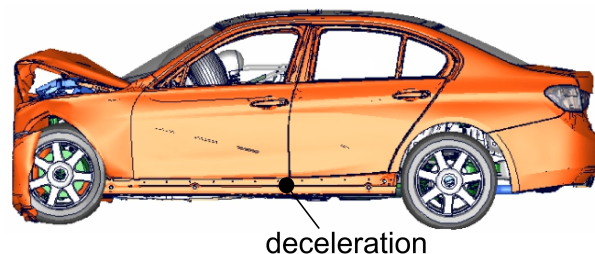


Figure 2.1: Simulation of a vehicle front crash.

The maximum deceleration is entirely determined by the force-deformation characteristics of the elements of the car structure, parametrized by F_i , $i = 1, 2, \dots, d$, see [38]. A visualization of a force-deformation curve of a part of a front structure is shown in Figure 2.2. Crash simulations show an inherently non-linear physical behavior with respect to structural parameters. For this reason, the maximum deceleration is difficult to design. It holds

$$a_{pulse} = f(F_1, F_2, \dots, F_d).$$

For this function, an optimization could be run in order to find an optimum for the maximum deceleration. Unfortunately, this solution cannot be realized exactly due to uncertainties. Therefore, rather than computing one optimum, a range of solutions $F_i^{low} \leq F_i \leq F_i^{up}$, $i = 1, 2, \dots, d$, is sought. This can be expressed by a hyperbox which is obtained by the Cartesian product of the d ranges. The hyperbox represents permissible intervals for the degrees of freedom F_i , $i = 1, 2, \dots, d$.

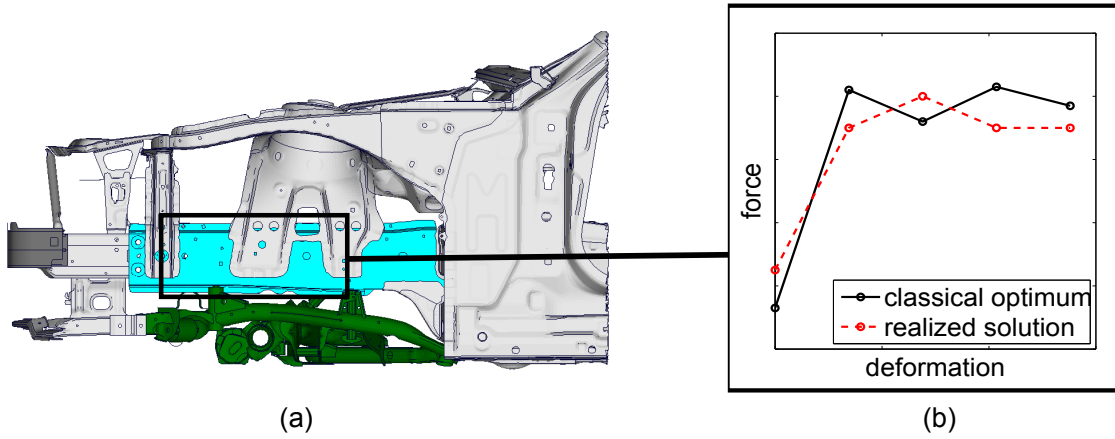


Figure 2.2: Vehicle front crash: (a) vehicle front structure and (b) force-deformation characteristic of a structural component of a front car structure with a classical optimum and the realized solution due to uncertainties.

2.2 Problem statement

Let $\Omega_{DS} \subseteq \mathbb{R}^d$ be a closed and convex set of admissible designs, called the design space.

Definition 2.2.1 (Hyperbox). Consider $\mathbf{x}^{low} = (x_1^{low}, x_2^{low}, \dots, x_d^{low})$, $\mathbf{x}^{up} = (x_1^{up}, x_2^{up}, \dots, x_d^{up}) \subseteq \Omega_{DS}$ such that $\mathbf{x}^{low} \leq \mathbf{x}^{up}$ component-by-component. Then, the hyperbox $\Omega_{box} = \Omega_{box}(\mathbf{x}^{low}, \mathbf{x}^{up})$

is the Cartesian product

$$\Omega_{box} := I_1 \times I_2 \times \cdots \times I_d \subseteq \Omega_{DS}$$

of intervals

$$I_i := [x_i^{low}, x_i^{up}] \subseteq \mathbb{R} \quad \text{for all } i = 1, 2, \dots, d.$$

A hyperbox is an axis-parallel, simply connected, and compact subset of Ω_{DS} . If we denote the width of the i -th interval $I_i = [x_i^{low}, x_i^{up}]$ by $\ell_i := x_i^{up} - x_i^{low}$, then $\ell_1, \ell_2, \dots, \ell_d$ are the lengths of the edges of the hyperbox Ω_{box} . Especially, $\boldsymbol{\ell} = (\ell_1, \ell_2, \dots, \ell_d)$ is given by $\boldsymbol{\ell} := \mathbf{x}^{up} - \mathbf{x}^{low}$. The volume $\mu(\Omega_{box})$ of the hyperbox Ω_{box} is thus given by

$$\mu(\Omega_{box}) := \prod_{i=1}^d \ell_i.$$

Let $f : \Omega_{DS} \rightarrow \mathbb{R}$ be an objective function which denotes a scalar quantity of interest. In practical applications, it represents a numerical simulation producing a result $f(\mathbf{x})$ from input parameters \mathbf{x} . For the system $f(\mathbf{x})$ and a given critical value $f_c \in \mathbb{R}$, a hyperbox Ω_{box} is sought such that $\mu(\Omega_{box}) \rightarrow \max$ subject to $f(\mathbf{x}) \leq f_c$ for all $\mathbf{x} \in \Omega_{box}$.

Definition 2.2.2 (Good design / bad design). *A design $\mathbf{x} \in \Omega_{DS}$ which satisfies the constraint $f(\mathbf{x}) \leq f_c$ is called a good design. A design $\mathbf{x} \in \Omega_{DS}$ which violates the constraint $f(\mathbf{x}) \leq f_c$ is called a bad design.*

With these preparations at hand, we can state the following constrained, non-linear, and high-dimensional optimization problem:

$$\left. \begin{array}{l} \text{find } \mathbf{x}^{low}, \mathbf{x}^{up} \in \Omega_{DS} \text{ with } \mathbf{x}^{low} \leq \mathbf{x}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \text{ subject to } f(\mathbf{x}) \leq f_c \text{ for all } \mathbf{x} \in \Omega_{box}. \end{array} \right\} \quad (\text{P})$$

This optimization problem is a shape optimization problem which shall be solved without the use of gradients to be applicable to any engineering problem where the function $f(\mathbf{x})$ is not analytically given. The solution will be a hyperbox which results in fixed intervals for each input parameter. In practice, these intervals define requirements for the related components and will be used in the development process as design goals.

2.3 Known approaches

There are already approaches in the literature to solve optimization problems similar to (P). In the following sections, we overview on these approaches, which can be grouped into three main classes.

2.3.1 Cellular evolutionary strategy

Let $\Omega_{DS} := [x_{1,DS}^{low}, x_{1,DS}^{up}] \times [x_{2,DS}^{low}, x_{2,DS}^{up}] \times \cdots \times [x_{d,DS}^{low}, x_{d,DS}^{up}] \subseteq \mathbb{R}^d$ be the design space. In [66], the problem

$$\prod_{i=1}^d |x_i^* - m_i^*| \rightarrow \max_{\mathbf{x}^*, \mathbf{m}^* \in G} \text{ subject to } \mathbf{x} \in G \text{ for all } \mathbf{x} \in \Omega_{box} \quad (\text{P1})$$

is considered with $G := \{\mathbf{x} \in \Omega_{DS} : g(\mathbf{x}) \geq 0\}$. The set Ω_{box} is defined by

$$\Omega_{box} := [x_1^{low}, x_1^{up}] \times [x_2^{low}, x_2^{up}] \times \cdots \times [x_d^{low}, x_d^{up}]$$

with $x_i^{low} := m_i^* - |x_i^* - m_i^*|$ and $x_i^{up} := m_i^* + |x_i^* - m_i^*|$ for all $i = 1, 2, \dots, d$. By setting $g(\mathbf{x}) = f_c - f(\mathbf{x})$, this problem is equivalent to the optimization problem (P). It is solved by using an approach which combines cellular evolutionary strategies and interval arithmetic.

Definitions

Cellular evolutionary strategy combines the evolutionary strategy technique $ES(\mu, \lambda)$ with concepts from cellular automata [66]. The *evolutionary strategy* is one type of evolutionary algorithm where the candidates are represented by real-valued vectors. In an *evolutionary algorithm*, a population of individuals (designs) evolves iteratively towards better solutions by a selection process of the parents, by a recombination of the parents, by a mutation of individuals and by a substitution strategy. The evolutionary algorithm starts with a randomly generated population of individuals. Then, these individuals are evaluated by a fitness measure which can be, for example, an output value which has to be minimized. Some parents are selected based on their fitness. The parent selection is typically probabilistic. This means, individuals with a good fitness have a higher chance to become parents than individuals with a lower fitness. Then, a recombination of the parents is applied to obtain an offspring [66]. For example, if the genes of the parents are

$$5|1|3|4|6 \quad \text{and} \quad 2|3|1|6|5,$$

the genetic information is exchanged up from the third position, and the following offspring is obtained

$$5|1|3|6|5 \quad \text{and} \quad 2|3|1|4|6.$$

The resulting offspring is mutated, for example, by changing randomly one gene of the individual

$$5|1|3|6|5 \mapsto 5|1|3|4|5,$$

and, then, the offspring is evaluated. Based on their fitness, individuals are selected for the next generation. This process is iterated until an individual with a sufficient fitness is found or a predefined number of evaluations is reached (see [19, 44]). Algorithm 1 gives the pseudo-code of an evolutionary algorithm in accordance with [19].

```

begin
  Initialize a population with random candidate solutions;
  Evaluate each candidate;
  repeat
    1. Select parents;
    2. Recombine pairs of parents;
    3. Mutate the resulting offspring;
    4. Evaluate new candidates;
    5. Select individuals for the next generation;
  until Termination condition is satisfied;
end

```

Algorithm 1: Pseudo-code of an evolutionary algorithm.

The technique $ES(\mu, \lambda)$ represents the canonical version of the evolutionary strategy and is called comma-selection with $\mu < \lambda$. Here, μ is the number of parents and λ denotes the number of the offspring. The selection of the parents from the set of either the offspring is deterministically, i.e., the fitness of the individuals is ranked, and the μ best individuals are chosen (see [44]). The pseudo-code of this algorithm is given in the Algorithm 2.

Cellular automaton is a discrete model and consists of a regular grid of cells. Each cell has a neighborhood consisting of a set of cells.

In the cellular evolutionary strategy, the concepts of neighborhood, known from cellular automata, are used for the selection of the parents. Each individual is located randomly in a cell of an one-dimensional array. The parents are selected from the cells in the neighborhood of the individual which is to be updated. This is in contrast to the general evolutionary strategy where parents are selected from the whole population [51, 84].

In *interval arithmetic*, interval numbers are used. These interval numbers replace real numbers and are an ordered pair of real numbers representing the lower and upper bound of a parameter range, see [31, 54, 56]. Hence, an interval number is defined as $X := [a, b] = \{x \in \mathbb{R} : a \leq x \leq b\}$ with $a, b \in \mathbb{R}$. In interval arithmetic, a function f whose input is an interval number X produces an interval number $Y = [c, d] = \text{conv}\{f(x) : x \in X\}$ with $c, d \in \mathbb{R}$. Interval arithmetic is here applied to evaluate the generated hyperbox.

```

begin
  Initialize a population with random candidate solutions;
  Evaluate each candidate;
  repeat
    1. Select  $\lambda$  parents;
    2. Recombine pairs of parents;
    3. Mutate the resulting offspring;
    4. Evaluate new candidates;
    5. Select  $\mu$  individuals for the next generation;
  until Termination condition is satisfied;
end

```

Algorithm 2: Pseudo-code of the evolutionary strategy ES(μ, λ).

Algorithm

In order to solve the Problem (P1), the following procedure is presented in [66]. First, the variables \mathbf{x} and \mathbf{m} are chosen randomly. If one of the two variables is a bad design, then new designs are generated until both variables \mathbf{x} and \mathbf{m} are good designs. If a design is a good design, the design is called feasible. Then, a symmetric hyperbox Ω_{box} is generated by using \mathbf{m} as its center. Interval arithmetic is applied to check the feasibility of the hyperbox. The hyperbox is feasible if it holds

$$f(\mathbf{x}) \leq f_c \quad \text{for all } \mathbf{x} \in \Omega_{box}.$$

If the hyperbox is feasible, its volume is calculated. If the hyperbox is not feasible, new design points \mathbf{x} and \mathbf{m} are generated. Algorithm 3 is the corresponding algorithm as proposed in [66]. Here, the fitness function is defined as

$$\text{Fitness}(\mathbf{x}, \mathbf{m}) = \prod_{i=1}^d |x_i - m_i|.$$

The drawback of this algorithm is that the objective function $f(\mathbf{x})$ has to be known explicitly in order to apply interval arithmetic. Thus it cannot be treated as a black box. Unfortunately, in most practical applications, the objective function is represented by a numerical simulation, producing a result $f(\mathbf{x})$ from input parameters \mathbf{x} , and is not known analytically. A black box is defined as follows (cf. [8]).

Definition 2.3.1 (Black box). *A black box is a mapping $\mathbf{x} \xrightarrow{\text{black box}} f(\mathbf{x})$ which returns to every value $\mathbf{x} \in \mathbb{R}^d$ a function value $f(\mathbf{x}) \in \mathbb{R}$.*

```

begin
   $t = 0$ ;
  feasible=false;
  while feasible=false do
    | Generate random points  $\mathbf{x}$  and  $\mathbf{m}$ ;
    | If  $\mathbf{x}$  and  $\mathbf{m}$  are feasible, then feasible=true;
  end
   $P(0) = \mathbf{x}$ ;
  while Termination condition is not satisfied do
    | for  $i = 0$  to  $\mu$  do
      | for  $j = 0$  to 7 do
        | Select a new cell;
        | Select parents randomly in the neighborhood;
        | Recombination;
        | Mutation;
        | Generate a symmetrical hyperbox using  $\mathbf{m}$  as center;
        | Check hyperbox feasibility using interval arithmetic;
        | if the hyperbox is feasible then
          | | Evaluate the volume hyperbox;
          | | Fitness(offspring)=volume;
        | else
          | | Fitness(offspring)=0;
        | end
        | Store new offspring in  $Y$ ;
      | end
      | Replace the selected cell with the best element from  $Y$ ;
    | end
    |  $t = t + 1$ ;
  end
end

```

Algorithm 3: Pseudo-code of the cellular evolutionary strategy combined with interval arithmetic.

Mathematically speaking, a black box is just a mapping $\mathbf{x} \mapsto f(\mathbf{x})$ where, however, no additional information of f is given.

2.3.2 Cluster analysis

Given a sample $\mathcal{P} = \{\mathbf{x}_j \in \Omega_{DS} : j = 1, 2, \dots, N\}$, the problem

$$\sup\{\mu(\Omega_{box}) : \Omega_{box} \cap \mathcal{N}_{bad} = \emptyset, \Omega_{box} \subseteq \Omega_{DS}\} \quad (\text{P2})$$

with

$$\mu(\Omega_{box}) = \prod_{i=1}^d (x_i^{up} - x_i^{low}) \quad \text{and} \quad \mathcal{N}_{bad} = \{\mathbf{x} \in \mathcal{P} : f(\mathbf{x}) > f_c\}$$

is similar to the optimization problem (P) with the difference that in the problem statement (P2) only discrete sets of designs are considered. In [6, 60], this discrete optimization problem is solved with the aid of cluster analysis and fuzzy set theory.

Definitions

Cluster analysis is a special type of learning machines. A *learning machine* is a data mining method to solve pattern recognition problems (see [77]). The goal of data mining is to extract as much knowledge as possible from a given set of data $\{\mathbf{x}_j, y_j\}$ with $\mathbf{x}_j \in \mathbb{R}^d$ and $y_j \in \mathbb{R}$, $j = 1, 2, \dots, N$. This includes fitting models to given data as well as determining patterns from data, see [20]. The definition of a clustering is given in Definition 2.3.2 which is in accordance with [6].

Definition 2.3.2 (Clustering). *A cluster $C_\ell \subseteq \mathcal{N}_{good}$ is a non-empty set of sample points, where \mathcal{N}_{good} is the set of good sample points, i.e.,*

$$\mathcal{N}_{good} = \{\mathbf{x} \in \mathcal{P} : f(\mathbf{x}) \leq f_c\}.$$

A clustering $\mathcal{C} = \{C_\ell : \ell = 1, 2, \dots, n_C\}$ of \mathcal{N}_{good} is a complete, but not necessarily disjoint, subdivision of \mathcal{N}_{good} into clusters, i.e., it holds

$$\mathcal{N}_{good} = \bigcup_{\ell=1}^{n_C} C_\ell$$

for some $1 \leq n_C \leq N$.

Cluster analysis subdivides a given set of objects into clusters. Let $d(\mathbf{x}_j, \mathbf{x}_{j'})$ be a distance function between the points \mathbf{x}_j and $\mathbf{x}_{j'}$. Given a fixed number of clusters, we intend to construct a clustering such that the degree of similarity between elements within each particular cluster C_ℓ is maximum, i.e.,

$$\sum_{\mathbf{x}_j, \mathbf{x}_{j'} \in C_\ell} d(\mathbf{x}_j, \mathbf{x}_{j'}) \rightarrow \min \quad \text{for all } C_\ell, \ell = 1, 2, \dots, n_C,$$

while the degree of similarity between elements from different clusters $C_\ell, C_{\ell'}$ is minimum, i.e.,

$$\sum_{\mathbf{x}_j \in C_\ell, \mathbf{x}_{j'} \in C_{\ell'}} d(\mathbf{x}_j, \mathbf{x}_{j'}) \rightarrow \max \quad \text{for all } C_\ell, C_{\ell'}, \ell, \ell' = 1, 2, \dots, n_C, \ell \neq \ell'.$$

Hence, the objects within the same cluster are similar to each other and are different from the objects in the other cluster. The larger the similarity within a cluster and the larger the difference between clusters, the better the clustering (see [37]).

In [85], a fuzzy set is defined as follows.

Definition 2.3.3 (Fuzzy set). *A fuzzy set $A \subseteq \Omega_{DS}$ is characterized by a membership function $\mu_A : \Omega_{DS} \rightarrow [0, 1]$ which associates with each point in Ω_{DS} a real number in the interval $[0, 1]$. The value of $\mu_A(\mathbf{x})$ at \mathbf{x} represents the grade of the membership of \mathbf{x} in A .*

It holds $\mathbf{x} \notin A$ if $\mu_A(\mathbf{x}) = 0$ and $\mathbf{x} \in A$ if $\mu_A(\mathbf{x}) = 1$. Moreover, it holds $\mu_A(\mathbf{x}) \in (0, 1)$ if it is not sure that $\mathbf{x} \in A$ or $\mathbf{x} \notin A$. The *fuzzy set theory* permits the gradual assessment of the membership of elements in a set. Contrary, in the classical set theory, an element either belongs or does not belong to a set. This corresponds to the indicator functions which are special cases of the membership functions of fuzzy sets since the membership function only takes the values 0 or 1. Therefore, fuzzy sets are a generalization of classical sets.

Algorithm

The method to solve the Problem (P2) consists of four main parts [60]. First, a sample is produced within the design space Ω_{DS} . Second, the generated sample points are subdivided into good and bad designs with the aid of the fuzzy set theory. The third part consists in classifying the good designs by applying a cluster analysis in order to detect non-connected input spaces containing only good designs. Given a number of clusters n_C , the results of the cluster analysis are point sets C_1, C_2, \dots, C_{n_C} . The hyperbox with the maximum volume containing only good designs is identified as follows in the fourth part. For each C_ℓ , a hyperbox is identified by determining two opposite vertices $\mathbf{x}_{C_\ell}^{\min}$ and $\mathbf{x}_{C_\ell}^{\max}$ on the basis of available points $\mathbf{x} \in C_\ell$, i.e.,

$$\mathbf{x}_{C_\ell}^{\min} = \begin{bmatrix} \min(x_{1,1}^{C_\ell}, x_{1,2}^{C_\ell}, \dots, x_{1,n_{C_\ell}}^{C_\ell}) \\ \min(x_{2,1}^{C_\ell}, x_{2,2}^{C_\ell}, \dots, x_{2,n_{C_\ell}}^{C_\ell}) \\ \vdots \\ \min(x_{d,1}^{C_\ell}, x_{d,2}^{C_\ell}, \dots, x_{d,n_{C_\ell}}^{C_\ell}) \end{bmatrix}, \quad \mathbf{x}_{C_\ell}^{\max} = \begin{bmatrix} \max(x_{1,1}^{C_\ell}, x_{1,2}^{C_\ell}, \dots, x_{1,n_{C_\ell}}^{C_\ell}) \\ \max(x_{2,1}^{C_\ell}, x_{2,2}^{C_\ell}, \dots, x_{2,n_{C_\ell}}^{C_\ell}) \\ \vdots \\ \max(x_{d,1}^{C_\ell}, x_{d,2}^{C_\ell}, \dots, x_{d,n_{C_\ell}}^{C_\ell}) \end{bmatrix}$$

with n_{C_ℓ} denoting the number of designs in the cluster C_ℓ . Then, the hyperbox $\Omega_{box}^{C_\ell}$ generated by $\mathbf{x}_{C_\ell}^{min}$ and $\mathbf{x}_{C_\ell}^{max}$ is shrunk to obtain a hyperbox for which it holds

$$\Omega_{box}^{C_\ell} \cap \mathcal{N}_{bad} = \emptyset.$$

For each cluster C_ℓ , we thus obtain a hyperbox which contains only good sample points. From all these hyperboxes, the hyperbox with maximum volume is selected as the resulting hyperbox. The optimality of this hyperbox depends on the chosen number of clusters and the way how the bad designs are removed. However, in [6, 60], it is not presented in detail how the hyperboxes are shrunk.

Unfortunately, the fuzzy set theory needs some additional information like the membership function of the parameters which is typically not available in the engineering design development. Furthermore, the design space is sampled only once. Consequently, the number of sample points has in the mean to be larger than the volume of the design space divided by the volume of the solution space to detect good regions. Hence, for high-dimensional problems with many relevant input parameters, a very large number of sample points is required to identify the solution space.

2.3.3 Support vector machines

In [22], the problem

$$\max_{\mathbf{x} \in \Omega_{DS}} \log \left(\prod_{i=1}^d x_i \right) \text{ subject to } \sum_{i=1}^d w_i x_i = b \quad (\text{P3})$$

is considered where $\sum_{i=1}^d w_i x_i = b$ describes the linear hyperplane which separates the design space $\Omega_{DS} = [0, 1]^d$ in a space with good designs and in a space with bad designs. In Figure 2.3, the linear hyperplane (blue line) is illustrated which separates the good designs (green circles) and the bad designs (red triangles). The good and bad designs belong to a given sample $\mathcal{P} = \{\mathbf{x}_j \in \Omega_{DS} : j = 1, 2, \dots, N\}$. The hyperplane $\sum_{i=1}^d w_i x_i = b$ is identified by using a support vector machine which is a special type of learning machine. For problems where the bad and good design points are linearly separable, the problem is similar to the optimization problem (P).

Remark 2.3.4. *If the good design points are elements of the set $\{\mathbf{x}_j \in \Omega_{DS} : \mathbf{w}^T \mathbf{x}_j > b\}$, the design space is transformed in a space Ω_{DS}^* in which it holds $\{\mathbf{x}_j \in \Omega_{DS}^* : \mathbf{w}^T \mathbf{x}_j \leq b\}$ for the good sample points.*

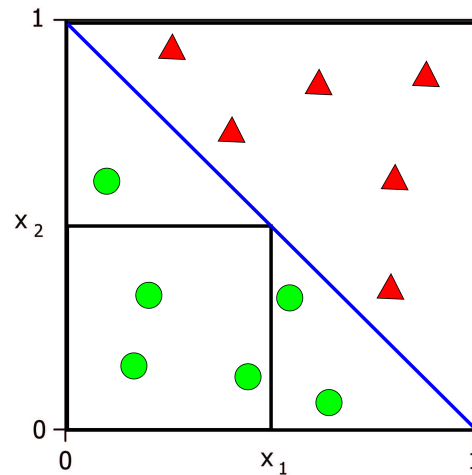


Figure 2.3: Linear separating hyperplane with the maximal hyperbox.

Definitions

Support vector machines are large margin classifiers because a set of data is subdivided in classes such that the distance (margin) of the boundary between the classes to the nearest training data point of any class is as large as possible. There are linear support vector machines, as described below, and non-linear support vector machines, which are presented in [12, 18, 52].

Linear support vector machines are introduced in [12] as follows. Assume that N training data $\{\mathbf{x}_j, y_j\}$ are given with $y_j \in \{-1, 1\}$ for all $j = 1, 2, \dots, N$. A hyperplane is identified which separates the data given as vectors in good data points with $y_j = 1$ and bad sample points $y_j = -1$. The hyperplane serves as the boundary between the two classes. For each point \mathbf{x} on the hyperplane, it holds

$$\mathbf{w}^T \mathbf{x} - b = 0$$

with \mathbf{w} being perpendicular to the hyperplane. The perpendicular distance from the hyperplane to the origin is given by

$$\frac{|b|}{\|\mathbf{w}\|}$$

with $\|\cdot\|$ being the Euclidean norm. In order to maximize the distance of the vectors which are as close as possible to the hyperplane, the following optimization problem has to be solved

$$\left. \begin{array}{l} \|\mathbf{w}\|^2 \rightarrow \min \\ \text{subject to } y_j(\mathbf{x}_j^T \mathbf{w} - b) - 1 \geq 0, \quad j = 1, 2, \dots, N \end{array} \right\} \quad (\text{Q})$$

Definition 2.3.5 (Support vectors). *If such a hyperplane exists, the training data $\{\mathbf{x}_{j'}, y_{j'}\}$ with $j' \in \{1, 2, \dots, N\}$ which satisfy*

$$y_{j'}(\mathbf{x}_{j'}^T \mathbf{w} - b) - 1 = 0$$

are called support vectors.

Support vectors are the training data which are necessary to describe the hyperplane as depicted in Figure 2.4. The associated optimization problem (Q) can be solved by means of Lagrange multipliers (see e.g. [43]).

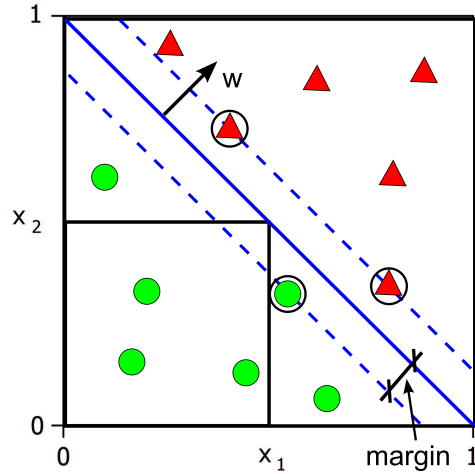


Figure 2.4: Linear separating hyperplane with the associated support vectors.

Algorithm

In [22], the Problem (P3) is solved by identifying the hyperplane which separates the classes in two groups. Then, the volume of a hyperbox is maximized within one of these two classes assuming that a transform onto the space $[0, 1]^d$ was done. This yields an optimization problem under inequality constraints which can be solved by means of Lagrange multipliers.

Unfortunately, hyperboxes are only computable for linearly separable data and not for general high-dimensional, non-linear and noisy problems. One could determine a separating

hyperplane for non-linearly separable data if linear support vectors are applied with an additional so-called slack variable which allows a few points to be misclassified, i.e., to be on the wrong side of the separating hyperplane. However, the resulting hyperbox could then include bad design points.

Chapter 3

Algorithm

In this chapter, we describe an algorithm which solves the constrained optimization problem (P). The algorithm has been introduced in [87] and identifies the maximum hyperbox within the solution space for arbitrary non-linear, high-dimensional and noisy problems. The method only requires function evaluations and, therefore, no access to the analytical expression of $f(\mathbf{x})$. Hence, the system $f(\mathbf{x})$ will be treated as a black box. This has the advantage that the function does not need to be analytically given, which is the case in the most engineering problems where $f(\mathbf{x})$ is evaluated by a numerical simulation. Thus, the proposed optimization method is non-intrusive.

The starting point of the algorithm is a design \mathbf{x} which fulfills the inequality $f(\mathbf{x}) \leq f_c$. It can be found by a classical optimization like differential evolution (see e.g. [73]). Then, an initial hyperbox is built around this admissible design and the algorithm's first phase is started. The first phase, called the *exploration phase*, is an iterative scheme which explores the landscape of the objective function. Finally, the second phase of the algorithm, called the *consolidation phase*, is performed. The consolidation phase includes an algorithm which shrinks the hyperbox such that it contains only good designs at a given probability level.

First, we present how the initial hyperbox is identified, second, the exploration phase and the consolidation phase are introduced in detail, and, finally, some extensions of the algorithm are given.

3.1 Identifying the initial hyperbox

The optimization algorithm differential evolution is used to identify a good design \mathbf{x} , which means, \mathbf{x} fulfills the inequality $f(\mathbf{x}) \leq f_c$. *Differential evolution* is a parallel direct search mode, and the following procedure is proposed in [73]. An initial population of designs (vectors) is chosen randomly and covers the entire parameter space. Then, the procedure which is given in Algorithm 4 is iterated until a maximum number of populations is reached or the cost function of a design in the population is smaller than a desired target value [73].

```

begin
  Initialize a population with random candidate solutions;
  while Termination condition is not satisfied do
    for each design in the population do
      Design = target design;
      Mutation. New parameter designs are generated by adding the weighted
      difference between two population designs to a third design;
      Crossover. The mutated design's parameters are then mixed with the
      parameters of another predetermined design, the target design, to yield
      the trial design;
      Selection. If the trial design yields a lower cost function value than the
      target design, the trial design replaces the target design in the following
      generation;
    end
  end
end

```

Algorithm 4: Pseudo-code of a differential evolution.

After identifying a good design, an initial hyperbox is built around this design, and the exploration phase is started.

3.2 Exploration phase

The exploration phase consists of four basic steps which are outlined in the flowchart in Figure 3.1 at the top.

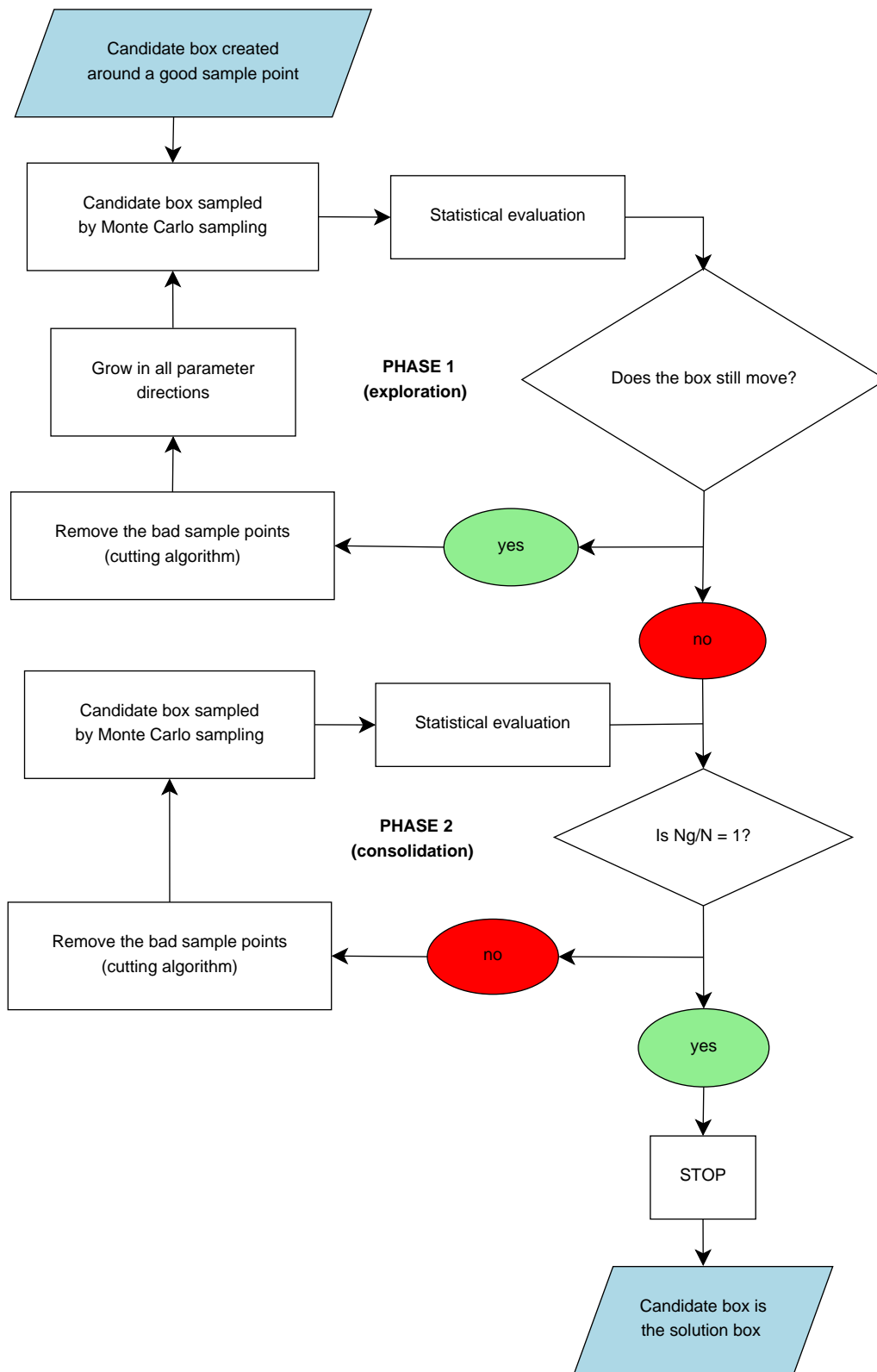


Figure 3.1: The flowchart of the algorithm to identify the maximum hyperbox.

3.2.1 Sampling methods

In the first step, i.e. the hyperbox evaluation, a population of designs is created by using a design of experiments technique such as Monte Carlo sampling or Latin hypercube sampling in the candidate hyperbox

$$\Omega_{cand} := [x_{1,cand}^{low}, x_{1,cand}^{up}] \times [x_{2,cand}^{low}, x_{2,cand}^{up}] \times \cdots \times [x_{d,cand}^{low}, x_{d,cand}^{up}].$$

Given a random variable $X(\omega)$ which is uniformly distributed, a Monte Carlo sample of length N is a set of N independent realizations $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N$ of the random variable $X(\omega)$ (cf. [50, 80]). Thus, a Monte Carlo sampling is a stochastic sampling method where independent deterministic models are chosen based on a uniform probability distribution (cf. [69]).

Contrary to a Monte Carlo sampling, the Latin hypercube sampling is a more deterministic sampling method for the uniformly distributed random variable $X(\omega)$. A Latin hypercube sampling with N sample points is obtained by the following rules (cf. [68]): The range of each parameter is divided into N intervals which have the same width. The number of the sample points is denoted by N . Hence, each interval has the same probability occurrence $1/N$. Then, a value is randomly taken from each interval. The N values for parameter 1 are randomly combined with the N values from Parameter 2. These pairs are then randomly combined with the values of the third parameter and so on until N d -tuple are obtained with d being the number of parameters.

3.2.2 Statistical evaluation

Stochastic sampling methods are employed to scan the space. The generated population $\{\mathbf{x}_j\}$ is divided in good sample points which fulfill $f(\mathbf{x}_j) \leq f_c$ and bad sample points for which it holds $f(\mathbf{x}_j) > f_c$. The hyperbox Ω_{cand} is then evaluated. The fraction of good sample points is defined as follows.

Definition 3.2.1 (Fraction of good sample points). *The ratio $\widehat{a} = N_g/N$ of the number N_g of good sample points and the total number N of sample points is called the fraction of good sample points.*

The fraction of good sample points in Ω_{cand} is computed. Then, the 95%-confidence interval is calculated as described in Subsection 3.3.2 in order to evaluate the candidate hyperbox. Moreover, the value of the fraction of good sample points is necessary to determine the growth rate in the fourth step of the exploration phase. The growth rate is introduced in Subsection 3.2.4.

3.2.3 Cutting algorithm

In the third step, the hyperbox is modified by removing all bad sample points. This is done by an algorithm which identifies a hyperbox which includes only good sample points.

Remark 3.2.2. *The algorithm will be called cutting algorithm as it removes the bad space by relocating the boundaries. In this sense, it cuts off the bad space.*

Optimal cutting algorithm

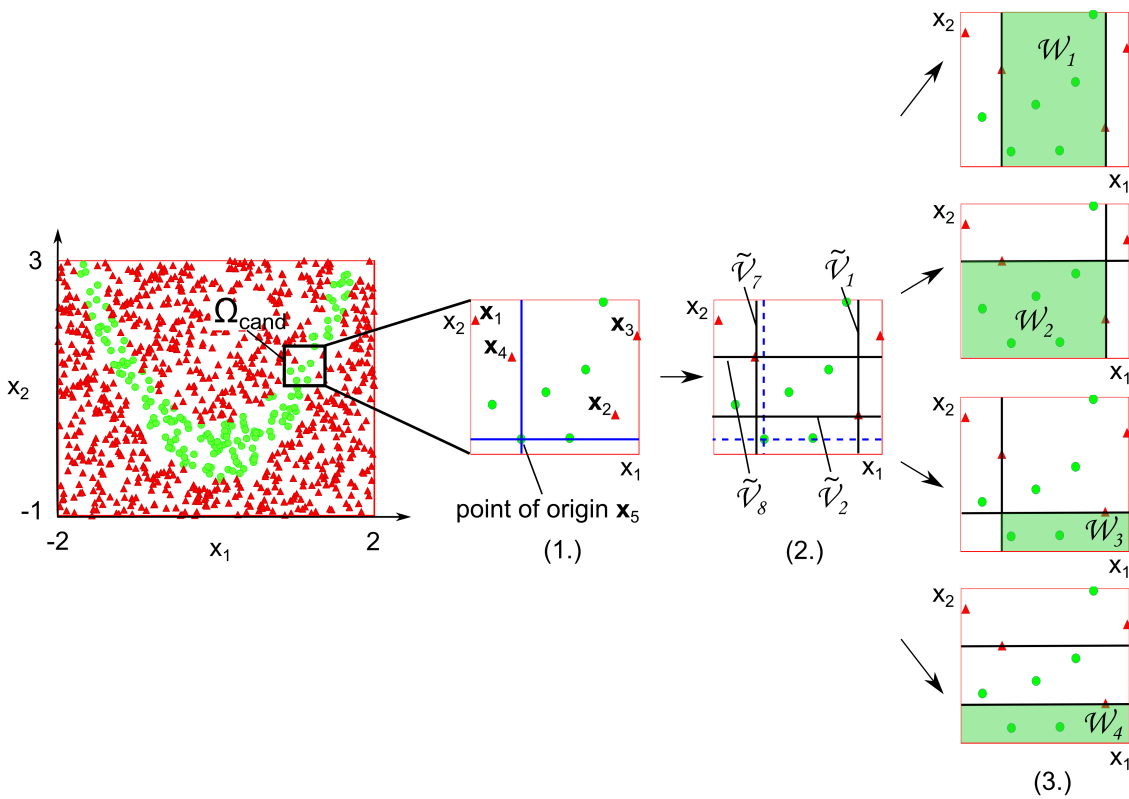


Figure 3.2: Optimal cutting algorithm to select the hyperbox with the largest volume in Ω_{cand} .

The input for the optimal cutting algorithm is a candidate hyperbox Ω_{cand} which contains N sample points. The largest hyperbox which includes only good sample points is determined according to the following rule which is applied to each good sample point.

A good sample point is chosen as the point of origin. The bad sample points which are located in the same corner with respect to the point of origin are assigned to the same

cluster C_ℓ , see the sketch in Figure 3.2 which is indicated by (1.). Here, we find

$$C_1 = \{\mathbf{x}_1, \mathbf{x}_4\} \quad \text{and} \quad C_2 = \{\mathbf{x}_2, \mathbf{x}_3\}.$$

Then, for each cluster C_ℓ , possible combinations \mathcal{V}_m are identified which contain all the possible combinations of the bad sample points with an assigned parameter direction. Note that in a combination, a parameter direction and a bad sample point, respectively, exist only once. By considering, for example, the cluster C_2 , we will obtain

$$\mathcal{V}_1 = \{2 \ 1\}, \quad \mathcal{V}_2 = \{2 \ 2\}, \quad \mathcal{V}_3 = \{3 \ 1\}, \quad \mathcal{V}_4 = \{3 \ 2\}, \quad \mathcal{V}_5 = \{2 \ 1, 3 \ 2\}, \quad \mathcal{V}_6 = \{2 \ 2, 3 \ 1\}.$$

Each element of a combination \mathcal{V}_m consists of a sample index and an assigned dimension. For each combination \mathcal{V}_m , we check if every element from \mathcal{V}_m , which is larger than the point of origin in the associated dimension, is maximal in the associated dimension with respect to all the other elements from \mathcal{V}_m . We check if every element from \mathcal{V}_m , which is smaller than the point of origin in the associated dimension, is minimal in the associated dimension with respect to all the other elements from \mathcal{V}_m . Moreover, we check if all bad sample points within the considered cluster C_ℓ are removed by removing all the elements from \mathcal{V}_m . All combinations $\tilde{\mathcal{V}}_m$ are selected from \mathcal{V}_m which fulfill these conditions. For example, for the cluster C_2 , only the combinations

$$\tilde{\mathcal{V}}_1 = \{2 \ 1\} \quad \text{and} \quad \tilde{\mathcal{V}}_2 = \{2 \ 2\}$$

remain for the next step. Cluster C_1 contains

$$\tilde{\mathcal{V}}_7 = \{4 \ 1\} \quad \text{and} \quad \tilde{\mathcal{V}}_8 = \{4 \ 2\}.$$

See the sketch in Figure 3.2 which is indicated by (2.) for an illustration.

Then, one combination $\tilde{\mathcal{V}}_m$ is picked from each C_ℓ , and all the possible combinations \mathcal{W}_n are built. This means, we obtain

$$\mathcal{W}_1 = \{\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_7\}, \quad \mathcal{W}_2 = \{\tilde{\mathcal{V}}_1, \tilde{\mathcal{V}}_8\}, \quad \mathcal{W}_3 = \{\tilde{\mathcal{V}}_2, \tilde{\mathcal{V}}_7\}, \quad \mathcal{W}_4 = \{\tilde{\mathcal{V}}_2, \tilde{\mathcal{V}}_8\}.$$

By removing the bad sample points from \mathcal{W}_n in their assigned directions, the hyperboxes depicted in Figure 3.2 (3.) are obtained. Then, for each \mathcal{W}_n , the volume of the resulting

hyperbox is calculated, and the hyperbox with maximum volume is selected.

```

Data: a hyperbox  $\Omega_{cand}$  and a set  $\mathcal{S} = \{\mathbf{x}_j \in \Omega_{cand}\}$  of sample points
Result: hyperbox  $\subseteq \Omega_{cand}$  which contains only good sample points
forall the good sample points  $\{\mathbf{x}^{good} \in \mathcal{S} : f(\mathbf{x}^{good}) \leq f_c\}$  do
  forall the bad sample points  $\{\mathbf{x}^{bad} \in \mathcal{S} : f(\mathbf{x}^{bad}) > f_c\}$  do
    assign all the  $\mathbf{x}^{bad}$  which are located in the same corner with respect to  $\mathbf{x}^{good}$ 
    to the same cluster  $C_\ell$ 
  end
forall the clusters  $C_\ell$  do
  for  $r = 1, 2, \dots, \min(d, |C_\ell|)$  do
    forall the combinations  $\mathcal{T} \subseteq 2^{C_\ell}$  with  $|\mathcal{T}| = r$  do
      forall the permutations  $\mathcal{P}$  of  $\mathcal{U} \subseteq 2^{\{1,2,\dots,d\}}$  with  $|\mathcal{U}| = r$  do
        assign  $e_1 = \mathcal{T}_1\mathcal{P}_1, \dots, e_r = \mathcal{T}_r\mathcal{P}_r$  to the combination  $\mathcal{V}_m$ ;
      end
    end
  end
forall the combinations  $\mathcal{V}_m$  do
  forall the elements  $e_s \in \mathcal{V}_m$  do
    if  $e_s$  is larger than  $x_{\mathcal{P}_s}^{good}$  in the associated dimension  $\mathcal{P}_s$  then
      (1) check if  $e_s$ ,  $s = 1, \dots, r$ , is maximal in the associated
      dimension  $\mathcal{P}_s$  with respect to all the other elements  $\in \mathcal{V}_m$ ;
    else
      (1) check if  $e_s$ ,  $s = 1, \dots, r$ , is minimal in the associated
      dimension  $\mathcal{P}_s$  with respect to all the other elements  $\in \mathcal{V}_m$ ;
    end
  end
  (2) check if all  $\mathbf{x}^{bad} \in C_\ell$  are removed by removing all  $e_s \in \mathcal{V}_m$ ;
  if (1) and (2) are fulfilled then  $\tilde{\mathcal{V}}_m = \mathcal{V}_m$ ;
end
end
  pick from each  $C_\ell$  one  $\tilde{\mathcal{V}}_m$  and built all the possible combinations  $\mathcal{W}_n$ ;
forall the combinations  $\mathcal{W}_n$  do
  remove all the  $\mathbf{x}^{bad} \in \mathcal{W}_n$ ;
  remember the hyperbox with maximum volume;
end
remember the hyperbox with maximum volume;
end

```

Algorithm 5: Pseudo-code of the optimal cutting algorithm.

This procedure is repeated for each good sample point in order to obtain a hyperbox for each good sample point. From these hyperboxes, the hyperbox with maximum volume is chosen.

The pseudo-code of this algorithm is shown in Algorithm 5. We denote by $2^{\mathcal{M}}$ the power set of \mathcal{M} which is the set which consists of all subsets \mathcal{N} of \mathcal{M} , i.e., $2^{\mathcal{M}} := \{\mathcal{N} : \mathcal{N} \subseteq \mathcal{M}\}$. Moreover, two sample points $\tilde{\mathbf{x}}$ and $\hat{\mathbf{x}}$ are in the same corner with respect to the point of origin \mathbf{x}^{good} if, for all $i = 1, 2, \dots, d$, either $(\tilde{x}_i > x_i^{good} \text{ and } \hat{x}_i > x_i^{good})$ or $(\tilde{x}_i \leq x_i^{good} \text{ and } \hat{x}_i \leq x_i^{good})$.

A good sample point serves as the point of origin. The point of origin defines uniquely the clusters. Then, the maximal possible corners are identified within each cluster, see Figure 3.3 for an illustration in $d = 2$. Here, every cluster includes two possible corners of the resulting hyperbox. By building all the possible combinations, picking from each cluster one admissible corner, the volume of each hyperbox is calculated, see Figure 3.3. The optimal hyperbox is identified by choosing the hyperbox with maximum volume from the obtained hyperboxes.

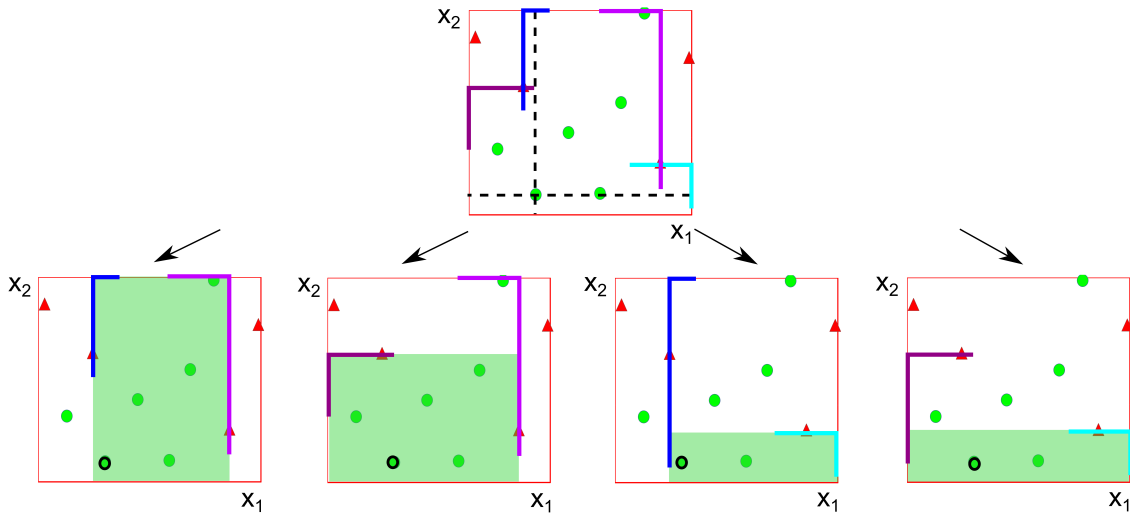


Figure 3.3: The possible corners of the resulting hyperbox and the possible hyperboxes.

However, the computational complexity of the Algorithm 5 is in general exponentially expensive because the probability that more than one sample point is located in the same cluster is very small in high dimensions.

If, for example, the point of origin is located in the center of the hyperbox, the probability

of success p for a sample point to lie within a cluster is

$$p = \left(\frac{1}{2}\right)^d.$$

Therefore, the probability that more than one point is located in the same cluster $P(\{\tilde{N} > 1\} \in C_\ell)$ is

$$P(\{\tilde{N} > 1\} \in C_\ell) = 1 - Np(1-p)^{N-1} - (1-p)^N \quad \text{with } p = \left(\frac{1}{2}\right)^d.$$

For $N = 100$ and $d = 10$, we obtain

$$P(\{\tilde{N} > 1\} \in C_\ell) \approx 4.43 \cdot 10^{-3},$$

while for $N = 100$ and $d = 20$, we already obtain $4.5 \cdot 10^{-9}$.

Consequently, the probability that at most one sample point is located in the same cluster is very large. Thus, there are N^d possibilities to combine from each cluster one possible corner with each other. This means, N^d operations are necessary to calculate the maximum volume. For $N = 100$ and $d = 10$, we obtain already $N^d = 10^{100}$ operations. Consequently, the computational costs are very expensive especially in high dimensions. Therefore, we present a fast cutting algorithm which is very cheap, but not optimal. However, the convergence to optimal solutions is reasonable well as shown in Chapter 4.

Fast cutting algorithm

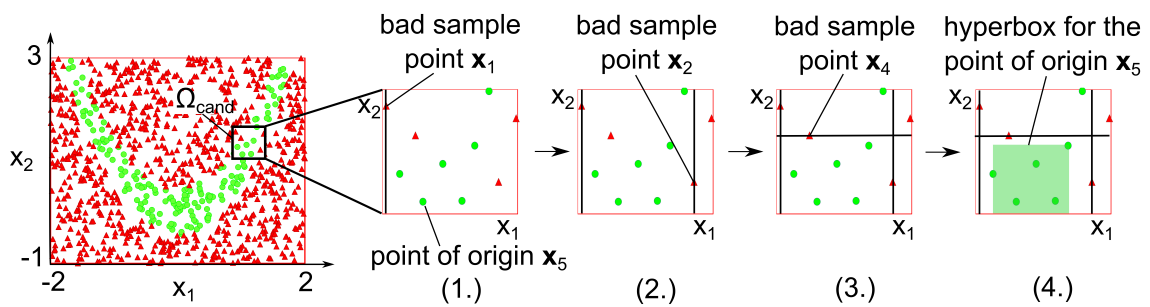


Figure 3.4: Cutting algorithm to select the hyperbox with the most good sample points in Ω_{cand} .

A candidate hyperbox Ω_{cand} which contains N sample points is the input for the fast cutting algorithm, i.e., the set $\mathcal{S} = \{\mathbf{x}_j \in \Omega_{cand} : f(\mathbf{x}_1) \geq \dots \geq f(\mathbf{x}_N)\}$ of sample points is given.

The following procedure is repeated for each good sample point $\{\mathbf{x}^{good} \in \mathcal{S} : f(\mathbf{x}^{good}) \leq f_c\}$ to determine the largest hyperbox which includes only good sample points.

A good sample point is used as the point of origin, this means, it will always be included, as visualized in the sketch of Figure 3.4 which is indicated by (1.).

Then, the bad sample point with the highest objective value (which we assume to be $\mathbf{x}_1 \in \mathcal{S}$) is removed by relocating the boundaries such that the fewest number of good sample points is lost, see the sketch of Figure 3.4 which is indicated by (1.) for an illustration. If this dimension is not unique, i.e., if there is more than one dimension which is associated with the loss of the same fewest number of good sample points, then the one is chosen where the most bad sample points are removed. If this dimension is not unique, the dimension is randomly selected from the dimensions with the loss of the same largest number of bad sample points.

The next bad sample point with the remaining highest value (say $\mathbf{x}_2 \in \mathcal{S}$) is removed in such a way that again the smallest number of good sample points are lost as shown in the sketch of Figure 3.4 which is indicated by (2.).

This procedure is repeated until there are no more bad sample points. In Figure 3.4, this is illustrated in the sketch which is indicated by (3.).

When all bad sample points have been removed, the hyperbox is shrunk in all dimensions where a bad sample point was removed to the outermost remaining good sample point. This step is displayed in the sketch of Figure 3.4 which is indicated by (4.).

After calculating the number of sample points which are contained in the resulting hyperbox, the procedure is repeated for the remaining good sample points.

Consequently, for each good sample point, there will be a hyperbox which contains only good sample points. In Figure 3.5, these hyperboxes are depicted. From all these hyperboxes, the one with the most sample points will be selected as the new candidate hyperbox.

Remark 3.2.3. *If this hyperbox is not unique, i.e., if there is more than one hyperbox which contains the same largest number of good sample points, the new candidate hyperbox is randomly selected from the hyperboxes with the same largest number of good sample points.*

The pseudo-code of the fast cutting algorithm is found in Algorithm 6.

One readily infers by checking the loops that the computational complexity of Algorithm 6 is $\mathcal{O}(N^3d)$ where N is the total number of sample points and d is the number of dimensions.

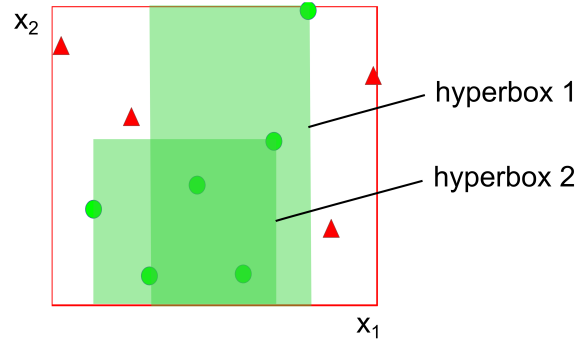


Figure 3.5: The two resulting hyperboxes from which the hyperbox with most good sample points is chosen in Ω_{cand} .

Optimal versus fast cutting algorithm

We want to compare the results of the optimal cutting algorithm with the results of the fast cutting algorithm for one iteration step. Here, the design space is given by

$$\Omega_{DS} = [0, 0.75]^d,$$

and the good space is defined by

$$\left\{ \mathbf{x} \in \Omega_{DS} : \sum_{i=1}^d x_i \leq \frac{d}{2} \right\}.$$

100 points are sampled by a Monte Carlo method in the design space and the respective cutting algorithm is applied. This experiment is repeated 1000 times for each dimension under consideration. The resulting mean of the volume $\mu(\Omega_{box}^{sol})_{avg}$ and the standard deviation of the volume are plotted in Figure 3.6 for $d = 2, 4, 6, 8$ and 10 dimensions.

We observe in Figure 3.6(a) that the volume of the resulting hyperboxes obtained by the optimal cutting algorithm is larger in the mean than the volume of the resulting hyperboxes obtained by the fast cutting algorithm for all dimensions under consideration. However, the distance between the resulting volume obtained by the optimal cutting algorithm and the one obtained by the fast cutting algorithm decreases when the dimension increases. Consequently, the hyperboxes of the fast cutting algorithm and the optimal algorithm differ only little, especially in high dimensions.

In Figure 3.6(b), the standard deviations $\sigma(\mu(\Omega_{box}^{sol}))$ of the resulting volumes are depicted. The standard deviation obtained by the optimal cutting algorithm is smaller than the

```

Data: a hyperbox  $\Omega_{cand}$  and a set  $\mathcal{S} = \{\mathbf{x}_j \in \Omega_{cand} : f(\mathbf{x}_1) \geq \dots \geq f(\mathbf{x}_N)\}$  of sample
points
Result: hyperbox  $\subseteq \Omega_{cand}$  which contains only good sample points
forall the good sample points  $\{\mathbf{x}^{good} \in \mathcal{S} : f(\mathbf{x}^{good}) \leq f_c\}$  do
  forall the bad sample points  $\{\mathbf{x}^{bad} \in \mathcal{S} : f(\mathbf{x}^{bad}) > f_c\}$  do
    for dimension  $i = 1, 2, \dots, d$  do
      if  $x_i^{bad} < x_i^{good}$  then
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \geq x_i \geq x_i^{low}$ ;
      else
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \leq x_i \leq x_i^{up}$ ;
      end
    end
    choose the dimensions  $i^+$  where the fewest good sample points are removed;
    choose the dimensions  $i^{++}$  where the most bad sample points are removed
    from the selected dimensions  $i^+$ ;
    choose randomly the dimension  $i^*$  from the selected dimensions  $i^{++}$ ;
    if  $x_{i^*}^{bad} < x_{i^*}^{good}$  then cut to  $x_{i^*}^{low}$ ;
    else cut to  $x_{i^*}^{up}$ ;
  end
  forall the dimensions  $i$  where a bad sample point is removed do
    if  $x_i^{bad} < x_i^{good}$  then
      |  $x_i^{low} := \min_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
    else
      |  $x_i^{up} := \max_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
    end
  end
  remember the hyperbox with most good sample points;
end

```

Algorithm 6: Pseudo-code of the fast cutting algorithm.

standard deviation obtained by the fast cutting algorithm for all dimensions under consideration. However, the difference between the optimal and the fast cutting algorithm decreases when the number of dimensions increases. Therefore, the fast cutting algorithm is a good alternative to the optimal cutting algorithm, especially in high dimensions.

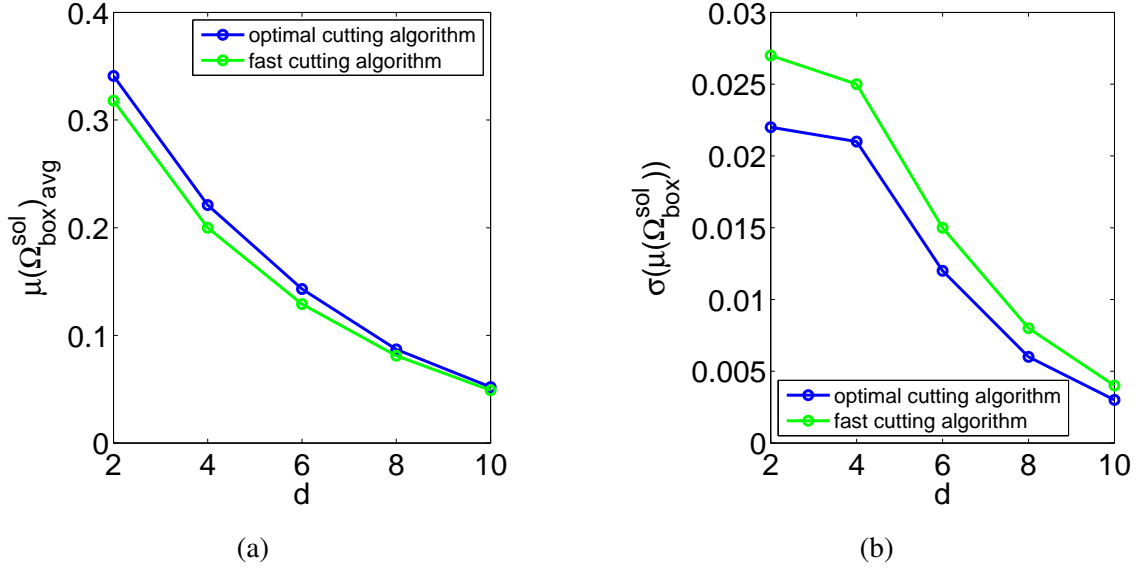


Figure 3.6: Optimal versus fast cutting algorithm: (a) Mean of the volume of 1000 runs. (b) Standard deviation of the volume of 1000 runs.

3.2.4 Growing

In the fourth step, the hyperbox is modified by growing in all parameter directions to enable the hyperbox to evolve towards beneficial directions with increasing hyperbox size in connection with the third step. This is done by a modification of the parameter boundaries. The boundaries in the $(k+1)$ -st iteration step are calculated from the boundaries of the k -th iteration by

$$[x_i^{\text{low}}]^{(k+1)} := [x_i^{\text{low}}]^{(k)} - \beta^{(k)}([x_i^{\text{up}}]^{(k)} - [x_i^{\text{low}}]^{(k)}) \quad \text{for all } i = 1, 2, \dots, d$$

for the lower boundary and by

$$[x_i^{\text{up}}]^{(k+1)} := [x_i^{\text{up}}]^{(k)} + \beta^{(k)}([x_i^{\text{up}}]^{(k)} - [x_i^{\text{low}}]^{(k)}) \quad \text{for all } i = 1, 2, \dots, d$$

for the upper boundary.

If $\beta^{(k)}$ would be constant over the iterations, this means $\beta := \beta^{(k)}$ for all iterations k , and the number of sample points is constant, β has to be chosen according to the equation (3.2) to obtain with a certain probability at least one good sample point in the new candidate hyperbox.

We need the following definition, cf. [11].

Definition 3.2.4 (Conditional probability). *The conditional probability of the event A , given that B has already occurred, is defined as*

$$P(A|B) := \frac{P(A \cap B)}{P(B)} \quad \text{if } P(B) > 0. \quad (3.1)$$

Theorem 3.2.5. *Let the good and bad space be uniformly distributed in $\Omega_{cut}^{(k)}$ where $\Omega_{cut}^{(k)}$ denotes the space of the hyperbox after applying the cutting algorithm in the k -th iteration step. Let a_k be the true fraction of good space in $\Omega_{cut}^{(k)}$. The conditional probability to obtain at least one good sample point in the new candidate hyperbox $\Omega_{cand}^{(k+1)}$ given a_k is at least*

$$1 - \left(1 - \frac{a_k}{(1 + 2\beta^{(k)})^d}\right)^N.$$

Proof. Define the event to obtain at least one good sample point in the $(k + 1)$ -st iteration step by

$$A := (N_g^{(k+1)} \geq 1).$$

The probability to hit the good space is denoted by p . In the worst case, the set $\Omega_{cand}^{(k+1)} \setminus \mu(\Omega_{cut}^{(k)})$ contains only bad space. Then, the probability of A given a_k is

$$P(A|a_k) = 1 - (1 - p)^N \geq 1 - \left(1 - \frac{a_k \mu(\Omega_{cut}^{(k)})}{\mu(\Omega_{cand}^{(k+1)})}\right)^N.$$

After applying the cutting algorithm in the k -th iteration step, the volume of the resulting hyperbox is

$$\mu(\Omega_{cut}^{(k)}) = \prod_{i=1}^d \ell_{i,cut}^{(k)}.$$

The volume of the new candidate hyperbox is

$$\mu(\Omega_{cand}^{(k+1)}) = \prod_{i=1}^d (\ell_{i,cut}^{(k)} + 2\beta^{(k)} \ell_{i,cut}^{(k)}).$$

Hence, we obtain

$$\frac{a_k \mu(\Omega_{cut}^{(k)})}{\mu(\Omega_{cand}^{(k+1)})} = \frac{a_k \prod_{i=1}^d \ell_{i,cut}^{(k)}}{\prod_{i=1}^d (\ell_{i,cut}^{(k)} + 2\beta^{(k)} \ell_{i,cut}^{(k)})} = \frac{a_k \prod_{i=1}^d \ell_{i,cut}^{(k)}}{(1 + 2\beta^{(k)})^d \prod_{i=1}^d \ell_{i,cut}^{(k)}} = \frac{a_k}{(1 + 2\beta^{(k)})^d}$$

which implies

$$P(A|a_k) \geq 1 - \left(1 - \frac{a_k}{(1 + 2\beta^{(k)})^d}\right)^N.$$

□

This theorem yields the following result under the assumption that the fraction a_k of good space is known. The growth factor $\beta^{(k)}$ has to be chosen such that the inequality

$$P(A|a_k) \geq 1 - \left(1 - \frac{a_k}{(1 + 2\beta^{(k)})^d}\right)^N \geq q$$

is satisfied in order to ensure that we have with probability $q \geq 0$ at least one good sample point in the new candidate hyperbox.

However, in practice, we have only a certain confidence interval $a_k^{low} < a_k < a_k^{up}$ (say, to the confidence level r) for the fraction of good space at hand. From $P(A|a_k) \geq P(A|a_k^{low})$, we conclude that $\beta^{(k)}$ has to be chosen according to

$$0 \leq \beta^{(k)} \leq \frac{1}{2} \left(\left(\frac{a_k^{low}}{1 - (1 - q/r)^{1/N}} \right)^{1/d} - 1 \right) \quad (3.2)$$

in order to ensure that there will be with probability q/r at least one good sample point in the next iteration.

To fulfill the inequality (3.2) over the whole iteration process, i.e., $\beta := \beta^{(k)}$ for all iteration steps k , either β has to be chosen very small or N has to be chosen very large. In the first case, the candidate hyperbox would move very slowly, which means that many iteration steps in the exploration phase may be necessary. To reduce the number of iteration steps and, therefore, to accelerate the algorithm, the growth rate $\beta^{(k)}$ has to change over the number of iterations to ensure that the resulting candidate hyperbox will contain good sample points and the speed of the hyperbox is satisfactory. Therefore, the growth rate $\beta^{(k+1)}$ is calculated according to

$$\beta^{(k+1)} := \frac{\widehat{a}_k}{a_{target}} \beta^{(k)}$$

where k is the iteration index, \widehat{a}_k is the fraction of good sample points in the iteration step k , and a_{target} is the desired fraction of good sample points.

The growth rate $\beta^{(k+1)}$ is determined such that the fraction of good sample points levels off at the desired fraction of good space during the exploration phase and to ensure that the resulting candidate hyperbox will contain sufficiently many good sample points.

The growing process is illustrated in Figure 3.7 for the Rosenbrock function. The Rosenbrock function is defined as

$$f(x_1, x_2) = (1 - x_1)^2 + 100(x_2 - x_1^2)^2. \quad (3.3)$$

Here, the design space is given by $\Omega_{DS} = [-2, 2] \times [-2, 3]$, see Figure 3.8(a). The critical value is chosen as $f_c = 20$, see Figure 3.8(b). In Figure 3.8(c), the good and bad parts

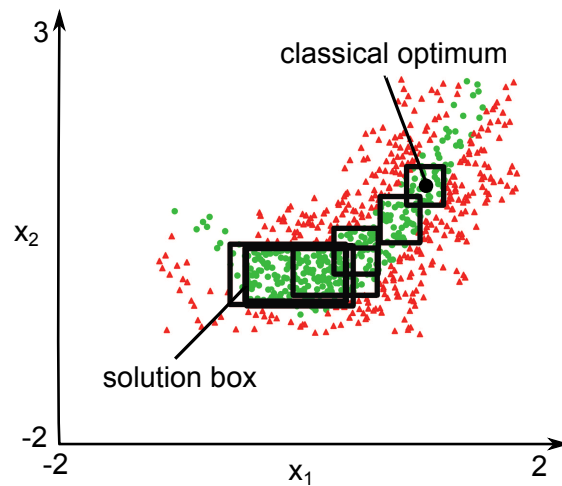


Figure 3.7: Iteration process starting in the classical optimum to move towards a beneficial direction with increasing hyperbox size.

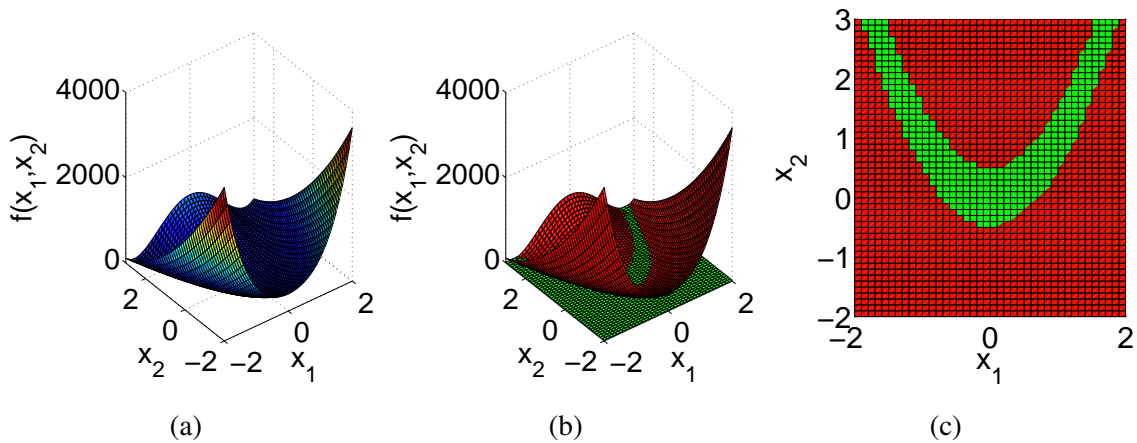


Figure 3.8: (a) The Rosenbrock function. (b) The Rosenbrock function with the critical value. (c) The good and bad space of the input space of the constrained Rosenbrock function.

of the design space of the Rosenbrock function constrained by $f_c = 20$ are illustrated. In Figure 3.7, it can be observed that the hyperbox moves away from the classical optimum towards a beneficial direction with increasing box size. If the hyperbox does not move and $\mu(\Omega_{box})$ does not significantly change anymore, the algorithm switches to the consolidation phase.

3.3 Consolidation phase

The consolidation phase is an iterative scheme where each iteration consists of three basic steps as seen in the flowchart in Figure 3.1.

3.3.1 Sampling methods

In the first step, a sample of designs is created by using Monte Carlo sampling or Latin hypercube sampling in the candidate hyperbox, cf. Subsection 3.2.1.

3.3.2 Statistical evaluation

Then, the candidate hyperbox is evaluated in the second step. This can be done by Bayes' theorem, cf. [47].

Theorem 3.3.1 (Bayes' theorem). *It holds*

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \quad (3.4)$$

where $P(A|B)$ and $P(B|A)$, respectively, are the conditional probabilities of the events A and B , respectively.

Proof. The conditional probability of the event A is given per definition by

$$P(A|B) = \frac{P(A \cap B)}{P(B)} \Rightarrow P(A \cap B) = P(A|B)P(B). \quad (3.5)$$

The conditional probability of the event B is given by

$$P(B|A) = \frac{P(A \cap B)}{P(A)} \Rightarrow P(A \cap B) = P(B|A)P(A). \quad (3.6)$$

From equation (3.5) and equation (3.6), it follows

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)},$$

and equation (3.4) is shown. □

The lower and upper boundary of a Bayesian confidence interval $[a_{low}, a_{up}]$ with a certain confidence level is calculated, using the following theorem according to [45].

Theorem 3.3.2. Let $N \in \mathbb{N}$ be the total number of sample points and $N_g \leq N$ the number of good sample points in the hyperbox $\Omega_{cand} = [x_{1,cand}^{low}, x_{1,cand}^{up}] \times [x_{2,cand}^{low}, x_{2,cand}^{up}] \times \cdots \times [x_{d,cand}^{low}, x_{d,cand}^{up}]$. Moreover, let a denote the true fraction of the good space in the hyperbox Ω_{cand} . The prior distribution of a is assumed to be uniform. Then, the confidence level that the probability of the fraction of good sample points (probability of success) lies within a given Bayesian confidence interval is

$$P(a^{low} < a < a^{up} | N_g(N)) = \frac{\int_{a^{low}}^{a^{up}} t^{N_g} (1-t)^{N-N_g} dt}{\int_0^1 s^{N_g} (1-s)^{N-N_g} ds}. \quad (3.7)$$

Here, a^{low} is the lower boundary of the confidence interval and a^{up} is the upper boundary of the confidence interval.

Proof. The probability of getting N_g good sample points from N sample points is $P(a | N_g(N))$. This probability is given for $P(N_g(N)) \neq 0$ by

$$P(a | N_g(N)) = \frac{P(a \cap N_g(N))}{P(N_g(N))}. \quad (3.8)$$

Applying Bayes' theorem, we get

$$P(a | N_g(N)) = \frac{P(N_g(N) | a) P(a)}{P(N_g(N))} \quad (3.9)$$

with $P(a)$ being the prior, which is the initial degree of belief in a , and $P(a | N_g(N))$ being the posterior probability, which is the degree of belief having accounted for $N_g(N)$. Moreover, $P(N_g(N) | a) / P(N_g(N))$ represents the probability of $N_g(N)$, cf. [4]. With

$$P(N_g(N)) = \int_0^1 P(N_g(N) | s) P(s) ds,$$

it follows

$$P(a | N_g(N)) = \frac{P(N_g(N) | a) P(a)}{\int_0^1 P(N_g(N) | s) P(s) ds}.$$

Since the prior distribution of a is assumed to be uniform, it holds $P(a) = 1$ and we obtain

$$P(a | N_g(N)) = \frac{P(N_g(N) | a)}{\int_0^1 P(N_g(N) | s) ds}.$$

$P(N_g(N) | a)$ is a binomial distribution since we have either good or bad sample points. Hence, we conclude

$$P(a | N_g(N)) = \frac{\binom{N}{N_g} a^{N_g} (1-a)^{N-N_g}}{\int_0^1 \binom{N}{N_g} s^{N_g} (1-s)^{N-N_g} ds} = \frac{a^{N_g} (1-a)^{N-N_g}}{\int_0^1 s^{N_g} (1-s)^{N-N_g} ds}. \quad (3.10)$$

The probability of the true fraction of the good space a in the hyperbox is thus given by the β -distribution, cf. [41]. The desired confidence level for $a \in [a^{low}, a^{up}]$ is finally

$$P(a^{low} < a < a^{up} | N_g(N)) = \int_{a^{low}}^{a^{up}} P(a | N_g(N)) da = \frac{\int_{a^{low}}^{a^{up}} t^{N_g} (1-t)^{N-N_g} dt}{\int_0^1 s^{N_g} (1-s)^{N-N_g} ds},$$

which completes the proof. □

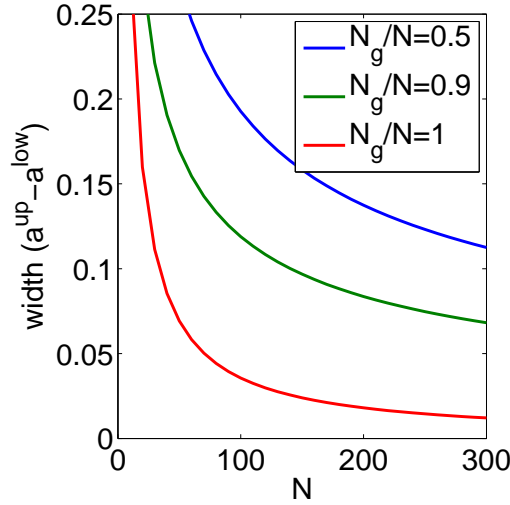


Figure 3.9: The width of the 95%-confidence interval around the probability of success.

For a uniform sampling of the input parameters, the probability of success corresponds to the good fraction of the input space volume. In Figure 3.9, the width of the Bayesian confidence interval around the probability of success is shown over the number of sample points N for different values of N_g/N . As it can be observed, the width of the confidence interval decreases when the ratio N_g/N increases. When there are 100 sample points and $N_g/N = 1$, which means that there are only good sample points, the true fraction of the good space is with 95% probability between 0.97 and 1, see [45]. Therefore, the width of the 95%-confidence interval is 0.03. The remaining 3% probability of failure is acceptable for two reasons. First, the layout of, e.g., a vehicle is likely to change over the course of the development process. Therefore, there is no need for a high accuracy. Second, the final design will always be verified by a detailed simulation anyway. This width of the 95%-confidence interval is assumed to be sufficiently small, especially in an early development phase where the knowledge of the final state of a design is limited. Therefore, 100 sample points are used in general for the evaluation of the candidate hyperbox.

3.3.3 Cutting algorithm

When there are bad sample points, the algorithm continues with the third step where the cutting algorithm introduced in Section 3.2.3 is applied to remove the bad sample points. When $N_g/N = 1$, the algorithm will stop and the last candidate hyperbox is chosen to be the final solution hyperbox.

3.4 Extensions of the algorithm

3.4.1 Measures for the hyperbox

The cutting algorithm is extended to obtain a hyperbox which is maximal regarding to different measures of the solution hyperbox. Different measures will yield to different sizes of the intervals of the resulting hyperbox. Within the cutting algorithm, a hyperbox is built around each good sample point. From these hyperboxes, the next candidate hyperbox is selected depending on the preferred measure of the hyperbox, see Algorithm 7.

Classically, the hyperbox with most good sample points is chosen in order to obtain a hyperbox with maximum volume. Consequently, a hyperbox which is as large as possible is obtained if the standard mode (Mode 1) is selected. This measure is referred to as $\mu_1(\Omega_{box})$.

The intervals of some dimensions may be very small while intervals of other dimensions may be very large. A small interval of a parameter allows only a little variability of the parameter. However, in practical applications, variability of the input parameters is often present due to uncertainty. Therefore, rather than a large volume, a hyperbox is desired where the smallest interval is as large as possible. Consequently, we introduce a new measure of the hyperbox. This measure consists of the smallest weighted width. Under the assumption that larger parameter values yield larger variabilities, the interval width is normalized by the mean of the appropriate interval. Consequently, the variability is taken into account relative to the parameter under consideration. The minimal weighted width is the minimum of the width of an interval normalized by the center of the appropriate interval. It is calculated by

$$\mu_2(\Omega_{box}) := \min_i \frac{x_i^{up} - x_i^{low}}{(x_i^{up} + x_i^{low})/2} \quad (3.11)$$

with $x_i^{up} + x_i^{low} \neq 0$, $i = 1, 2, \dots, d$. By maximizing this measure choosing Mode 2, a

hyperbox with maximum smallest interval width is obtained.

```

Data: a hyperbox  $\Omega_{cand}$  and a set  $\mathcal{S} = \{\mathbf{x}_j \in \Omega_{cand} : f(\mathbf{x}_1) \geq \dots \geq f(\mathbf{x}_N)\}$  of sample
  points
Result: hyperbox  $\subseteq \Omega_{cand}$  which contains only good sample points
forall the good sample points  $\{\mathbf{x}^{good} \in \mathcal{S} : f(\mathbf{x}^{good}) \leq f_c\}$  do
  forall the bad sample points  $\{\mathbf{x}^{bad} \in \mathcal{S} : f(\mathbf{x}^{bad}) > f_c\}$  do
    for dimension  $i = 1, 2, \dots, d$  do
      if  $x_i^{bad} < x_i^{good}$  then
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \geq x_i \geq x_i^{low}$ ;
      else
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \leq x_i \leq x_i^{up}$ ;
      end
    end
    choose the dimensions  $i^+$  where the fewest good sample points are removed;
    choose the dimensions  $i^{++}$  where the most bad sample points are removed
    from the selected dimensions  $i^+$ ;
    choose randomly the dimension  $i^*$  from the selected dimensions  $i^{++}$ ;
    if  $x_{i^*}^{bad} < x_{i^*}^{good}$  then cut to  $x_{i^*}^{low}$ ;
    else cut to  $x_{i^*}^{up}$ ;
  end
forall the dimensions  $i$  where a bad sample point is removed do
  if  $x_i^{bad} < x_i^{good}$  then
    |  $x_i^{low} := \min_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
  else
    |  $x_i^{up} := \max_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
  end
end
switch mode do
  case 1: remember the hyperbox with the largest measure  $\mu_1(\Omega_{box})$ ;
  case 2: remember the hyperbox with the largest measure  $\mu_2(\Omega_{box})$ ;
  case 3: remember the hyperbox with the largest measure  $\mu_3(\Omega_{box})$ ;
  case 4: remember the hyperbox with the largest measure  $\mu_4(\Omega_{box})$ ;
  case 5: remember the hyperbox with the largest measure  $\mu_5(\Omega_{box})$ ;
endsw
end

```

Algorithm 7: Pseudo-code of the cutting algorithm with extensions.

In Mode 3, the maximum of the weighted function

$$\mu_3(\Omega_{box}) := \lambda N_g + (1 - \lambda) \min_i \frac{x_i^{up} - x_i^{low}}{(x_i^{up} + x_i^{low})/2} \quad (3.12)$$

with $\lambda \in [0, 1]$ and $x_i^{up} + x_i^{low} \neq 0$, $i = 1, 2, \dots, d$, returns a hyperbox with most good sample points if $\lambda = 1$ and a hyperbox with the largest minimal weighted width if $\lambda = 0$. As usual, N_g denotes the number of good sample points within the hyperbox. Therefore, a compromise between large volume and large minimal weighted width is found.

The following modes are especially important for practical applications where the support points of a curve describing the functional relationship between two quantities are the input parameters.

Definition 3.4.1 (Corridor). *Let $I_1 = [x_1^{low}, x_1^{up}]$, $I_2 = [x_2^{low}, x_2^{up}]$, \dots , $I_d = [x_d^{low}, x_d^{up}]$ be the permissible intervals for the input parameters x_1, x_2, \dots, x_d and let the input parameters be the support points of a curve describing the functional relationship between two quantities. We generate one curve c^{low} by linearly interpolating between the lower boundaries $x_1^{low}, x_2^{low}, \dots, x_d^{low}$ and another curve c^{up} by linearly interpolating between the upper boundaries $x_1^{up}, x_2^{up}, \dots, x_d^{up}$. Then, the space between both curves is called a corridor.*

Remark 3.4.2. *If each curve is attached to a component, the support points of one component may generate a group of input parameters.*

Choosing the Mode 4, the hyperbox with the largest minimal coupled weighted width is the next candidate hyperbox. The minimal coupled weighted width is calculated by

$$\mu_4(\Omega_{box}) := \min_{coupl} \frac{\min_{i \in coupl} (x_i^{up} - x_i^{low})}{\max_{i \in coupl} ((x_i^{up} + x_i^{low})/2)} \quad (3.13)$$

where $\max_{i \in coupl} ((x_i^{up} + x_i^{low})/2) \neq 0$ and *coupl* denotes a group of input parameters. The smallest interval width of the parameters of a group is normalized by the largest mean of the intervals of the group. By using the same normalization for all interval widths within a group, the interval widths are equally weighted within a group in contrast to the measure μ_2 . Consequently, the variability is taken into account relative to the parameter with the largest center within the group under consideration.

In Mode 5, the hyperbox with the maximum of the function

$$\mu_5(\Omega_{box}) := \min_{coupl} \frac{\min_{x \in c_{coupl}^{low}, y \in c_{coupl}^{up}} \|y - x\|_2}{\max_{i \in coupl} (x_i^{up})} \quad (3.14)$$

with $\max_{i \in coupl} (x_i^{up}) \neq 0$ is selected. The enumerator of (3.14) measures the width of the corridor as illustrated in Figure 3.10(a). The widths of the intervals are identical in

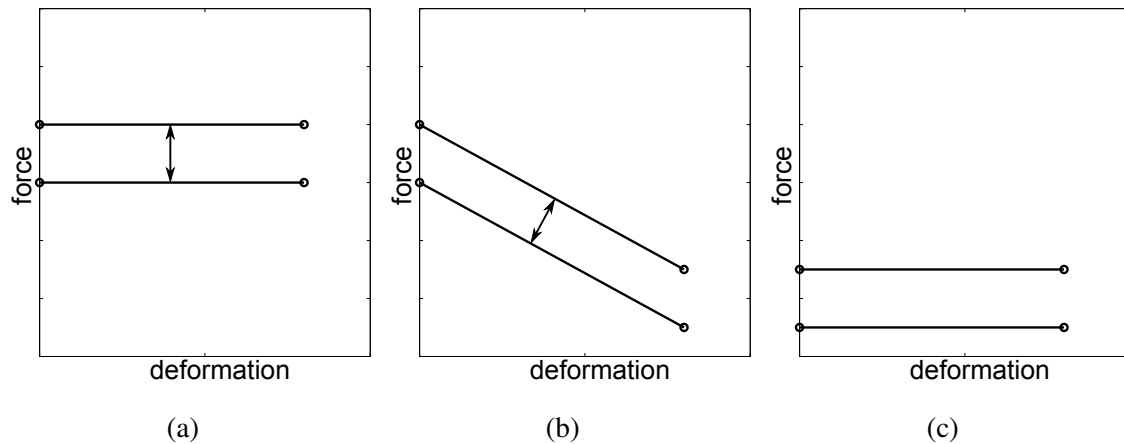


Figure 3.10: Force-deformation corridors: the width of the corridor is in (a) larger than in (b) while the widths of the intervals are identical. The corridor in (a) is smaller than the corridor in (c) due to a normalization of the width by the maximum upper boundary.

Figure 3.10(a) and 3.10(b) while the width of the corridor is larger in Figure 3.10(a) than in Figure 3.10(b). For example, in a front crash design problem where the force-deformation curve of each group has to lie within its associated force-deformation corridor, it is easier to lie within the corridor shown in Figure 3.10(a) than to lie within the corridor shown in Figure 3.10(b). Therefore, the minimum of the widths of the corridors are to be maximized in contrast to the measure μ_4 where the minimum of the widths of the intervals are to be maximized. Moreover, the widths are normalized by the maximum upper boundary for each group. By using the same normalization for all interval widths within a group, the interval widths are equally weighted within a group. A variability relative to the largest value within a group is taken into account. Therefore, the corridor in Figure 3.10(a) is smaller than the corridor in Figure 3.10(c).

3.4.2 Sensitivity of the solution hyperbox

We want to obtain some information about the *sensitivity* of the lower and upper boundaries of the solution hyperbox. The sensitivity of the lower / upper boundary of the i -th interval indicates how many bad sample points are obtained if the i -th interval is extended to the lower / upper boundary, see Figure 3.11. The hyperbox is extended by adding a certain fraction of the width of the i -th interval to its upper boundary in order to obtain the sensitivity of this upper boundary. If we want to obtain the sensitivity of the lower boundary of the i -th interval, we subtract a certain fraction of the width of the i -th interval

from its lower boundary in order to extend the hyperbox.

Definition 3.4.3 (Probed space). *The extended space is called the probed space.*

In order to rate the sensitivity, a sample is done in the probed space. If the sample contains many bad sample points, extending the boundary worsens extremely the quality of the hyperbox. In this sense, the hyperbox is sensitive with respect to the considered boundary.

Definition 3.4.4 (Very sensitive / a bit sensitive). *The boundary is called very sensitive if many bad sample points are gained by extending the hyperbox. If a few bad sample points are obtained, the boundary is called a bit sensitive.*

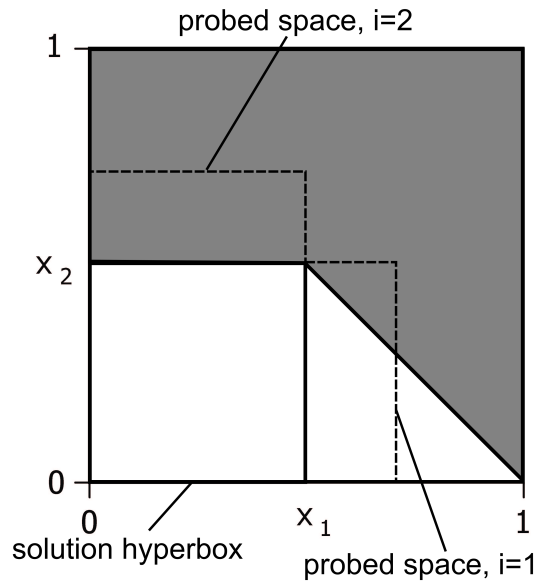


Figure 3.11: The solution hyperbox is extended in the parameter direction i . The extended space is called the *probed space*.

The following values indicate how sensitive a boundary is: The fraction of good space of the probed space in the dimension i is calculated by

$$[a^*]_i^{low} = \left[\frac{N_g}{N} \right]_{i, \Omega_{box}^{prob}}^{low}$$

for the lower boundary of the i -th interval, and by

$$[a^*]_i^{up} = \left[\frac{N_g}{N} \right]_{i, \Omega_{box}^{prob}}^{up}$$

for the upper boundary of the i -th interval.

Next, we calculate the fraction of good space of the hyperbox which is extended in the parameter direction i , i.e., the fraction of good space of the solution hyperbox together with the probed space. For this fraction of good space, we obtain in due consideration of the area ratio between the solution hyperbox and the probed space

$$\begin{aligned} [a^{**}]_i^{low} &= \frac{\left[\frac{N_g}{N}\right]_{\Omega_{box}^{sol}} + \left[\frac{N_g}{N}\right]_{i,\Omega_{box}^{prob}}^{low} (x_{i,sol}^{low} - x_{i,prob}^{low}) / (x_{i,sol}^{up} - x_{i,sol}^{low})}{(x_{i,sol}^{low} - x_{i,prob}^{low}) / (x_{i,sol}^{up} - x_{i,sol}^{low})} \\ &= \left[\frac{N_g}{N}\right]_{\Omega_{box}^{sol}} \frac{x_{i,sol}^{up} - x_{i,sol}^{low}}{x_{i,sol}^{low} - x_{i,prob}^{low}} + \left[\frac{N_g}{N}\right]_{i,\Omega_{box}^{prob}}^{low} \end{aligned}$$

for the lower boundary, and by

$$\begin{aligned} [a^{**}]_i^{up} &= \frac{\left[\frac{N_g}{N}\right]_{\Omega_{box}^{sol}} + \left[\frac{N_g}{N}\right]_{i,\Omega_{box}^{prob}}^{up} (x_{i,prob}^{up} - x_{i,sol}^{up}) / (x_{i,sol}^{up} - x_{i,sol}^{low})}{(x_{i,prob}^{up} - x_{i,sol}^{up}) / (x_{i,sol}^{up} - x_{i,sol}^{low})} \\ &= \left[\frac{N_g}{N}\right]_{\Omega_{box}^{sol}} \frac{x_{i,sol}^{up} - x_{i,sol}^{low}}{x_{i,prob}^{up} - x_{i,sol}^{up}} + \left[\frac{N_g}{N}\right]_{i,\Omega_{box}^{prob}}^{up} \end{aligned}$$

for the upper boundary.

If $[a^*]_i$ or $[a^{**}]_i$ is small, the boundary is very sensitive. The boundary is only a bit sensitive if $[a^*]_i$ or $[a^{**}]_i$ is large.

Chapter 4

Results of the algorithm

In this chapter, the following four benchmark problems are considered to study the convergence to optimal solutions.

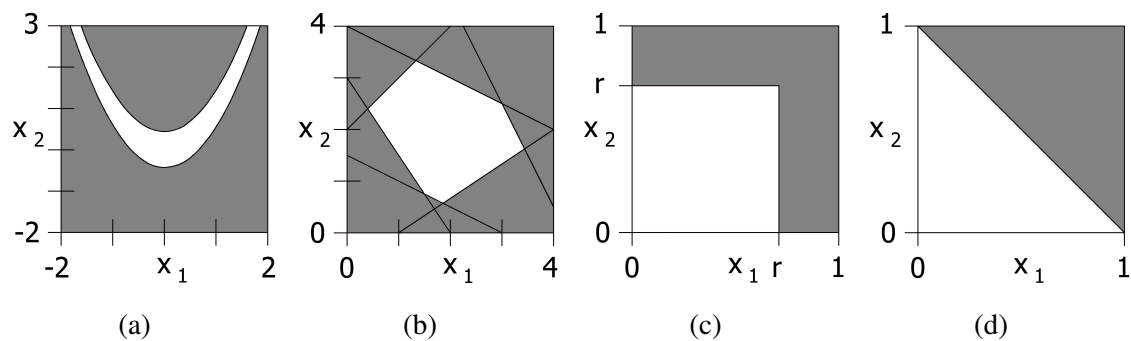


Figure 4.1: Problems considered: (a) Problem 1 (Rosenbrock), (b) Problem 2 (convex polytope), (c) Problem 3 (hyperbox) and (d) Problem 4 (tilted hyperplane).

- The solution space is given by restricting the two-dimensional Rosenbrock function by a constant value in Problem 1, see Figure 4.1(a).
- In Problem 2, the solution space is generated by a convex polytope as depicted in Figure 4.1(b).
- A hyperbox which defines the good space of the design space is inscribed in another hyperbox which is the design space (see Figure 4.1(c)). The ratio of the volume of the good space and the volume of the design space is 0.5. This is considered as Problem 3.

- In Problem 4, the solution space is given by a tilted hyperplane which divides a d -dimensional hyperbox $I_1 \times I_2 \times \cdots \times I_d \subseteq \mathbb{R}^d$ in two equal volumes, cf. Figure 4.1(d).

In Figure 4.1, the grey area indicates the bad space, and the white area shows the good space.

4.1 Problem 1. Restricted Rosenbrock function as boundary

The nonlinear optimization problem of maximizing a rectangle under the inequality constraint that the Rosenbrock function is smaller than a given constant is considered to compare the numerical results of the proposed algorithm with the optimal solution. The related good and bad space of this problem under consideration is visualized in Figure 4.1(a).

4.1.1 Analytical solution

Let $f(\mathbf{x})$ be the two-dimensional Rosenbrock function (3.3) and $\Omega_{DS} = [-2, 2] \times [-2, 3]$. We aim at finding the largest hyperbox

$$\Omega_{box} := [x_1^{low}, x_1^{up}] \times [x_2^{low}, x_2^{up}] \subseteq \Omega_{DS}$$

such that $f(\mathbf{x}) \leq f_c = 20$ for all $\mathbf{x} \in \Omega_{box}$. To that end, we consider the function

$$g(\mathbf{z}, \alpha, \beta) := \left(1 - \left(x_1^{low} + \alpha(x_1^{up} - x_1^{low})\right)\right)^2 + 100 \left(x_2^{low} + \beta(x_2^{up} - x_2^{low}) - \left(x_1^{low} + \alpha(x_1^{up} - x_1^{low})\right)\right)^2$$

where $\mathbf{z} := (\mathbf{x}^{low}, \mathbf{x}^{up}) = (x_1^{low}, x_2^{low}, x_1^{up}, x_2^{up})$. Then, any \mathbf{z} in the set

$$K := \{(\mathbf{x}^{low}, \mathbf{x}^{up}) \in \Omega_{DS} \times \Omega_{DS} : g(\mathbf{z}, \alpha, \beta) \leq 20 \text{ for all } \alpha, \beta \in [0, 1] \text{ and } \mathbf{x}^{low} \leq \mathbf{x}^{up}\}$$

defines obviously a hyperbox Ω_{box} such that $f(\mathbf{x}) \leq 20$ for all $\mathbf{x} \in \Omega_{box}$. Consequently, by defining the quadratic objective function

$$\mu(\Omega_{box}) := (x_1^{up} - x_1^{low})(x_2^{up} - x_2^{low}) = \frac{1}{2} \mathbf{z}^T \mathbf{D} \mathbf{z} \quad \text{with } \mathbf{D} = \begin{bmatrix} 0 & 2 & 0 & 0 \\ 0 & 0 & -2 & 0 \\ 0 & 0 & 0 & 2 \\ -2 & 0 & 0 & 0 \end{bmatrix},$$

we can formulate our problem as constrained optimization problem

$$\mu(\Omega_{box}) \rightarrow \max \text{ subject to } \mathbf{z} \in K. \quad (4.1)$$

Since the set K is convex, this optimization problem admits a maximum $\mu(\Omega_{box})$ due to the theorem of Weierstraß (cf. [65]).

Theorem 4.1.1 (Theorem of Weierstraß). *Let K be a non-empty and compact subset of \mathbb{R}^d and let $f : K \rightarrow \mathbb{R}^d$ be a continuous function on K . Then, there exist the global minimum and the global maximum of f .*

The analytical solution of (4.1) is found by means of Lagrange multipliers (see Appendix A) and has the values tabulated in the second column of Table 4.1 entitled $x_{i,opt}$. The values are rounded to three digits. A visualization of this maximum is found in Figure 4.2.

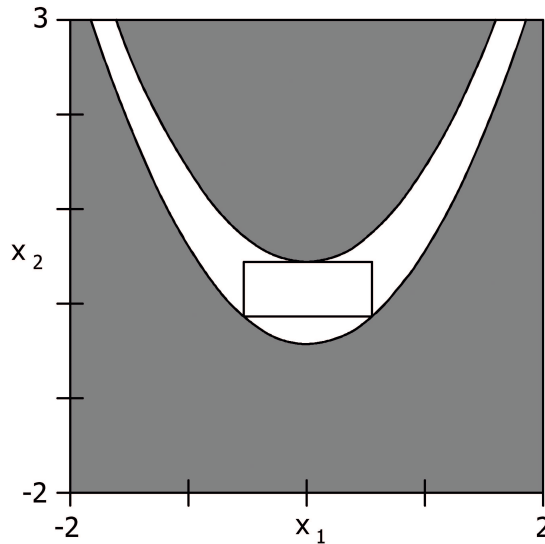


Figure 4.2: Problem 1. The hyperbox of maximum volume within the solution space.

4.1.2 Numerical solution

The constrained optimization problem (4.1) is equivalent to the constrained optimization problem (P) if $f(\mathbf{x})$ is the Rosenbrock function (3.3) and $f_c = 20$. We shall compare the numerical results produced by the optimization algorithm with the analytical solution. The

following process is executed to obtain the numerical results:

The algorithm is run 100 times, where the iterative process is started
 with a good design which is randomly chosen.
 200 iterations of the exploration phase
 and 100 iterations of the consolidation phase are run.

} (PR)

The distribution of the solution hyperboxes found by the algorithm is depicted in the histograms in Figure 4.3(a) for the parameters x_1^{low} and x_1^{up} and in Figure 4.3(b) for the parameters x_2^{low} and x_2^{up} .

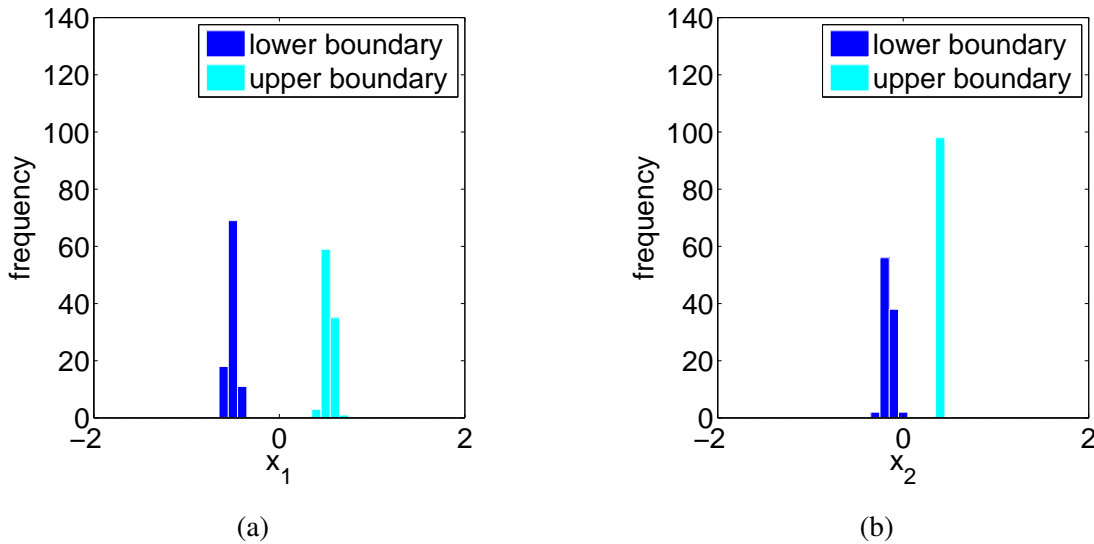


Figure 4.3: Problem 1. Distribution of the hyperboxes found by the algorithm for (a) coordinate x_1 and (b) coordinate x_2 for 100 simulations.

The columns of the Table 4.1 entitled $x_{i,avg}$ contain the mean of the coordinates of the final hyperboxes of the 100 simulations, the columns entitled $\sigma(x_i)$ are the related standard deviations, the columns entitled $\varepsilon(x_i)$ contain the absolute errors $|x_{i,avg} - x_{i,opt}|$, and the columns entitled “error in %” contain the relative errors $|x_{i,avg} - x_{i,opt}|/x_{i,opt}$ in %. The values are rounded to three digits. These results confirm that the proposed algorithm approximates the analytical solution.

	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1^{low}	-0.525	-0.510	0.0458	0.0150	2.86
x_2^{low}	-0.145	-0.161	0.0487	0.0160	11.0
x_1^{up}	0.547	0.533	0.0455	0.0140	2.56
x_2^{up}	0.436	0.432	0.00321	0.00400	0.917

Table 4.1: Problem 1. Analytical solution and related numerical results for 100 simulations.

4.2 Problem 2. A convex polytope as boundary

We consider the nonlinear optimization problem of maximizing a hyperbox within a convex polytope. In Figure 4.1(b), a solution space which is generated by a convex polytope is illustrated in two dimensions.

4.2.1 Analytical solution

Let $\Omega_{DS} := [0, L_1] \times [0, L_2] \times \cdots \times [0, L_d] \subseteq \mathbb{R}^d$ be the design space and

$$\Omega_{box} := [x_1^{low}, x_1^{up}] \times [x_2^{low}, x_2^{up}] \times \cdots \times [x_d^{low}, x_d^{up}] \subseteq \Omega_{DS}.$$

We will write $\mathbf{z} = (\mathbf{x}^{low}, \mathbf{x}^{up}) = (x_1^{low}, x_2^{low}, \dots, x_d^{low}, x_1^{up}, x_2^{up}, \dots, x_d^{up})$.

The inequality

$$\mathbf{g}(\mathbf{z}) := \mathbf{A}\mathbf{z} - \mathbf{b} \leq \mathbf{0} \quad \text{with } \mathbf{0} = [0, 0, \dots, 0]^T, \mathbf{A} \in \mathbb{R}^{n \times 2d}, \mathbf{b} \in \mathbb{R}^n$$

describes the cross section of $n \in \mathbb{N}$ half-spaces, called a convex polytope (see [27]).

The set

$$K := \{(\mathbf{x}^{low}, \mathbf{x}^{up}) \in \Omega_{DS} \times \Omega_{DS} : \mathbf{g}(\mathbf{z}) \leq \mathbf{0} \text{ and } \mathbf{x}^{low} \leq \mathbf{x}^{up}\}$$

is a compact subset of \mathbb{R}^d , because K is closed and bounded. Moreover, the constraint $\mathbf{g}(\mathbf{z})$ is affine and

$$\mu(\Omega_{box}) := \prod_{i=1}^d (x_i^{up} - x_i^{low}) \quad \text{with } x_i^{low} \leq x_i^{up} \text{ for all } i = 1, 2, \dots, d$$

is a convex function. Therefore, the optimization problem

$$\mu(\Omega_{box}) \rightarrow \max \quad \text{subject to } \mathbf{z} \in K \tag{4.2}$$

under affine inequality constraints admits a maximum $\mu(\Omega_{box})$.

The maximum of (4.2) can be identified by means of Lagrange multipliers.

The Lagrangian which belongs to the optimization problem under consideration is

$$L(\mathbf{z}, \boldsymbol{\lambda}) = \prod_{i=1}^d (x_i^{up} - x_i^{low}) + \boldsymbol{\lambda}^T (\tilde{\mathbf{A}}\mathbf{z} - \tilde{\mathbf{b}}) \quad \text{with} \quad \tilde{\mathbf{A}} = \begin{bmatrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{b}} = \begin{bmatrix} \mathbf{b} \\ \mathbf{c} \\ \mathbf{0} \end{bmatrix}.$$

Here, the matrices \mathbf{B} and \mathbf{C} are given by

$$\mathbf{B} = \begin{bmatrix} \mathbf{0} & \mathbf{I} \end{bmatrix} \in \mathbb{R}^{d \times 2d}, \quad \mathbf{C} = \begin{bmatrix} -\mathbf{I} & \mathbf{0} \\ \mathbf{I} & -\mathbf{I} \end{bmatrix} \in \mathbb{R}^{2d \times 2d}$$

where \mathbf{I} denotes the $(d \times d)$ -identity matrix. Moreover, it holds

$$\mathbf{c} = [L_1, L_2, \dots, L_d]^T.$$

Due to the KKT-Theorem (cf. Section A), there exists a unique vector of Lagrange multipliers $\boldsymbol{\lambda}^* \in \mathbb{R}^{3d+n}$, such that the following KKT-conditions are fulfilled:

$$\begin{aligned} \nabla_{\mathbf{z}} L(\mathbf{z}^*, \boldsymbol{\lambda}^*) &= \left[\frac{\partial L}{\partial x_1^{low}}, \frac{\partial L}{\partial x_2^{low}}, \dots, \frac{\partial L}{\partial x_d^{low}}, \frac{\partial L}{\partial x_1^{up}}, \frac{\partial L}{\partial x_2^{up}}, \dots, \frac{\partial L}{\partial x_d^{up}} \right] = \mathbf{0} \\ \tilde{\mathbf{A}}\mathbf{z}^* &\leq \tilde{\mathbf{b}} \quad \text{component-by-component,} \\ \lambda_s^* &\geq 0 \quad \text{for } s = 1, 2, \dots, 3d + n, \\ (\boldsymbol{\lambda}^*)^T (\tilde{\mathbf{A}}\mathbf{z}^* - \tilde{\mathbf{b}}) &= \mathbf{0} \quad \text{component-by-component.} \end{aligned}$$

The KKT-point will be found by solving these equations.

First example

The following two-dimensional example is considered to compare the numerical results of the algorithm with the optimal solution.

Let $\Omega_{DS} := [0, 4]^2$ be the design space and

$$\Omega_{box} := [x_1^{low}, x_1^{up}] \times [x_2^{low}, x_2^{up}] \subseteq \Omega_{DS}. \quad (4.3)$$

Each of the half-spaces

$$\begin{aligned} \mathbf{A}_1(\mathbf{z}) &= \begin{bmatrix} \frac{1}{8}x_1^{up} + \frac{1}{4}x_2^{up} - 1 \\ \frac{4}{17}x_1^{up} + \frac{2}{17}x_2^{up} - 1 \end{bmatrix}, & \mathbf{A}_3(\mathbf{z}) &= \begin{bmatrix} -\frac{1}{2}x_1^{low} - \frac{1}{3}x_2^{low} + 1 \\ -\frac{1}{3}x_1^{low} - \frac{2}{3}x_2^{low} + 1 \end{bmatrix}, \\ \mathbf{A}_2(\mathbf{z}) &= \begin{bmatrix} -\frac{1}{2}x_1^{low} + \frac{1}{2}x_2^{up} - 1 \end{bmatrix}, & \mathbf{A}_4(\mathbf{z}) &= \begin{bmatrix} x_1^{up} - \frac{3}{2}x_2^{low} - 1 \end{bmatrix} \end{aligned} \quad (4.4)$$

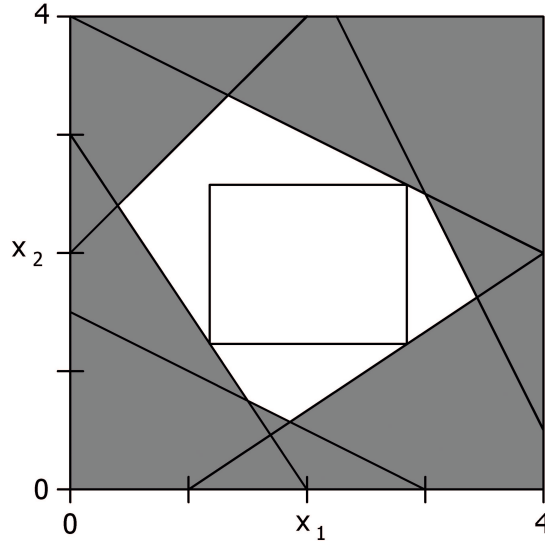


Figure 4.4: Problem 2 (first example). Convex polytope in $d = 2$ dimensions with the hyperbox of maximum volume.

constraints one of the corners of the hyperbox in the solution space. Therefore, the inequality

$$\mathbf{g}(\mathbf{z}) := \begin{bmatrix} \mathbf{A}_1(\mathbf{z}) \\ \mathbf{A}_2(\mathbf{z}) \\ \mathbf{A}_3(\mathbf{z}) \\ \mathbf{A}_4(\mathbf{z}) \end{bmatrix} = \mathbf{A}\mathbf{z} - \mathbf{b} \leq \mathbf{0} \quad \text{with } \mathbf{A} = \begin{bmatrix} 0 & 0 & \frac{1}{8} & \frac{1}{4} \\ 0 & 0 & \frac{4}{17} & \frac{2}{17} \\ -\frac{1}{2} & 0 & 0 & \frac{1}{2} \\ -\frac{1}{2} & -\frac{1}{3} & 0 & 0 \\ -\frac{1}{3} & -\frac{2}{3} & 0 & 0 \\ 0 & -\frac{3}{2} & 1 & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ -1 \\ 1 \end{bmatrix} \quad (4.5)$$

describes a convex polytope. For the specific values used in equation (4.4), we obtain the convex polytope which is depicted in Figure 4.4.

The maximum hyperbox within this convex polytope is identified as described above with $d = 2$. For

$$\begin{cases} \lambda_s^* > 0, & s \in \{1, 4, 6\}, \\ \lambda_s^* = 0, & \text{otherwise,} \end{cases}$$

the analytical solution is found and has the values tabulated in Table 4.2 in the column which is entitled $x_{i,opt}$. The values are rounded to three digits. A visualization of this maximum is depicted in Figure 4.4.

	analytical	numerical for $N = 50$				numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1^{low}	1.18	1.18	0.101	0	0	1.17	0.074	0.0100	0.847
x_2^{low}	1.23	1.24	0.138	0.0100	0.813	1.24	0.108	0.0100	0.813
x_1^{up}	2.85	2.69	0.169	0.160	5.61	2.73	0.140	0.120	4.21
x_2^{up}	2.58	2.65	0.092	0.0700	2.71	2.63	0.073	0.0500	1.94
average	–	–	0.125	0.0600	2.29	–	0.0986	0.0475	1.95

Table 4.2: Problem 2 (first example). Analytical solution and related numerical results for 100 simulations with $N = 50$ and $N = 100$.

Second example

The convergence to the optimal solution is studied by considering the following two-dimensional example.

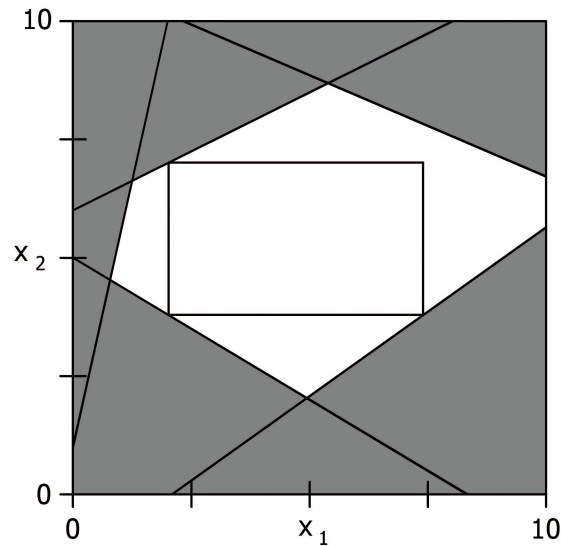


Figure 4.5: Problem 2 (second example). Convex polytope in $d = 2$ dimensions with the hyperbox of maximum volume.

Let $\Omega_{DS} := [0, 10]^2$ be the design space and Ω_{box} as in (4.3). The functions

$$\begin{aligned} \mathbf{A}_1(\mathbf{z}) &= \left[\frac{3}{77}x_1^{up} + \frac{1}{11}x_2^{up} - 1 \right], & \mathbf{A}_3(\mathbf{z}) &= \left[-\frac{3}{25}x_1^{low} - \frac{1}{5}x_2^{low} + 1 \right], \\ \mathbf{A}_2(\mathbf{z}) &= \left[-\frac{1}{12}x_1^{low} + \frac{1}{6}x_2^{up} - 1 \right], & \mathbf{A}_4(\mathbf{z}) &= \left[\frac{10}{21}x_1^{up} - \frac{2}{3}x_2^{low} - 1 \right] \end{aligned}$$

represent half-spaces, each of which constraints one of the corners of the hyperbox in the solution space. Therefore, the convex polytop is defined by

$$\mathbf{g}(\mathbf{z}) := \begin{bmatrix} \mathbf{A}_1(\mathbf{z}) \\ \mathbf{A}_2(\mathbf{z}) \\ \mathbf{A}_3(\mathbf{z}) \\ \mathbf{A}_4(\mathbf{z}) \end{bmatrix} = \mathbf{A}\mathbf{z} - \mathbf{b} \leq \mathbf{0} \quad \text{with } \mathbf{A} = \begin{bmatrix} 0 & 0 & \frac{3}{77} & \frac{1}{11} \\ -\frac{1}{12} & 0 & 0 & \frac{1}{6} \\ -\frac{9}{2} & 0 & 0 & 1 \\ -\frac{3}{25} & -\frac{1}{5} & 0 & 0 \\ 0 & -\frac{2}{3} & \frac{10}{21} & 0 \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} 1 \\ 1 \\ 1 \\ -1 \\ 1 \end{bmatrix}. \quad (4.6)$$

The convex polytope is illustrated in Figure 4.5.

By maximizing a hyperbox within this convex polytope, the analytical solution is found for

$$\begin{cases} \lambda_s^* > 0, & s \in \{2, 4, 5\}, \\ \lambda_s^* = 0, & \text{otherwise,} \end{cases}$$

as described above by means of Lagrange multipliers. The analytical solution has the values tabulated in the second column of Table 4.3, which are rounded to three digits. A visualization of this maximum is found in Figure 4.5.

	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1^{low}	2.02	2.38	0.403	0.360	17.8
x_2^{low}	3.79	3.97	0.245	0.180	4.75
x_1^{up}	7.40	7.68	0.335	0.280	3.78
x_2^{up}	7.01	7.21	0.205	0.200	2.85

Table 4.3: Problem 2 (second example). Analytical solution and related numerical results for 100 simulations with $N = 100$.

Third example

The following example is considered to compare the numerical results with the optimal solution in three dimensions.

Let $\Omega_{DS} := [0, 800]^3$ be the design space and

$$\Omega_{box} := [x_1^{low}, x_1^{up}] \times [x_2^{low}, x_2^{up}] \times [x_3^{low}, x_3^{up}] \subseteq \Omega_{DS}.$$

Each of the half-spaces

$$\begin{aligned} \mathbf{A}_1(\mathbf{z}) &= [x_3^{up} - 627.84], & \mathbf{A}_3(\mathbf{z}) &= [-x_1^{low} - x_2^{low} - x_3^{low} + 1210], \\ \mathbf{A}_2(\mathbf{z}) &= [x_1^{up} - x_2^{low}], & \mathbf{A}_4(\mathbf{z}) &= [x_2^{up} - x_3^{low}] \end{aligned}$$

constraints one of the corners of the hyperbox in the solution space. Thus, the inequality

$$\mathbf{g}(\mathbf{z}) := \begin{bmatrix} \mathbf{A}_1(\mathbf{z}) \\ \mathbf{A}_2(\mathbf{z}) \\ \mathbf{A}_3(\mathbf{z}) \\ \mathbf{A}_4(\mathbf{z}) \end{bmatrix} = \mathbf{A}\mathbf{z} - \mathbf{b} \leq \mathbf{0} \text{ with } \mathbf{A} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & -1 & 0 & 1 & 0 & 0 \\ -1 & -1 & -1 & 0 & 0 & 0 \\ 0 & 0 & -1 & 0 & 1 & 0 \end{bmatrix}, \mathbf{b} = \begin{bmatrix} 627.84 \\ 0 \\ -1210 \\ 0 \end{bmatrix} \quad (4.7)$$

describes a convex polytope.

The maximum hyperbox is found as described above for $d = 3$. The values of the analytical solution are tabulated in the second column of Table 4.4, and are rounded to three digits.

	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1^{low}	216	210	35.5	6.00	2.78
x_2^{low}	441	441	19.6	0	0
x_3^{low}	553	553	11.2	0	0
x_1^{up}	441	437	16.8	4.00	0.907
x_2^{up}	553	552	11.8	1.00	0.181
x_3^{up}	628	627	1.47	1.00	0.159

Table 4.4: Problem 2 (third example). Analytical solution and related numerical results for 100 simulations with $N = 100$.

4.2.2 Numerical solution

First example

If we express the constrained optimization problem (4.2) with the specific values (4.5) equivalently as

$$f(x_1, x_2) := \begin{cases} 0, & \text{if } \mathbf{g}(x_1, x_2, x_1, x_2) \leq \mathbf{0} \text{ component-by-component,} \\ 1, & \text{otherwise,} \end{cases}$$

and $f_c := 0.5$, the optimization problem (4.2) corresponds to the constrained optimization problem (P). Therefore, we can numerically solve the problem by using the algorithm presented in Chapter 3. The results of the numerical optimization obtained by executing the process (PR) for the convex polytope (4.5) are listed in Table 4.2 for $N = 50$ and $N = 100$ sample points per iteration.

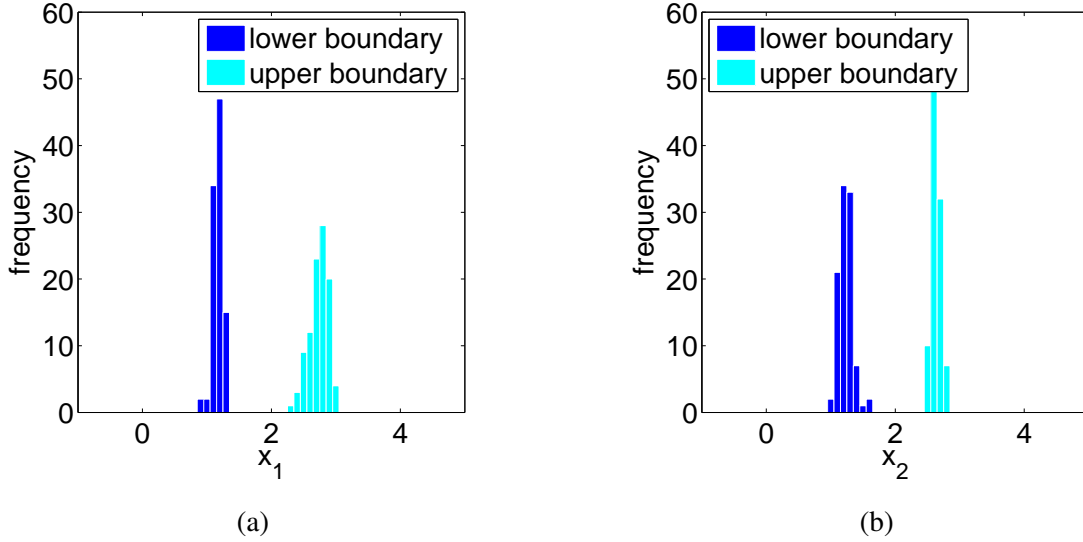


Figure 4.6: Problem 2 (first example). Distribution of the hyperboxes found by the algorithm for (a) coordinate x_1 and (b) coordinate x_2 for 100 simulations.

In Table 4.2, the mean $x_{i,avg}$ of the coordinates of the final hyperboxes of the 100 simulations, the related standard deviations $\sigma(x_i)$, the absolute errors $\varepsilon(x_i)$ and the relative errors $|x_{i,avg} - x_{i,opt}|/x_{i,opt}$ are tabulated. The values are rounded to three digits.

We observe in Table 4.2 that the standard deviation decreases if the number of sample points per iteration is doubled. The averaged error is 2.29% for $N = 50$ and 1.95% for $N = 100$. Therefore, the error between the analytical solution and the mean of the numerical solutions becomes smaller when the number of sample points per iteration is increased. In Section 5.5, it is shown that $N = 100$ sample points are a good choice to converge fast and to obtain a large volume in the consolidation phase of the algorithm. Therefore, $N = 100$ is chosen in the next test examples.

The distribution of the solution hyperboxes found by the algorithm is visualized for $N = 100$ sample points in Figure 4.6(a) for coordinate x_1 and in Figure 4.6(b) for coordinate x_2 . These plots illustrate that the proposed algorithm converges to the analytical solution in the sense that the average of the numerical solutions approximately agrees with the

analytical solutions.

Second example

If we set

$$f(x_1, x_2) := \begin{cases} 0, & \text{if } \mathbf{g}(x_1, x_2, x_1, x_2) \leq \mathbf{0} \text{ component-by-component,} \\ 1, & \text{otherwise,} \end{cases}$$

and $f_c := 0.5$, the constrained optimization problem (4.2) with the specific values (4.6) can be equivalently expressed as the constrained optimization problem (P). The results of the numerical optimization by performing the process (PR) are displayed in Table 4.3 for $N = 100$ sample points per iteration. The means $x_{i,avg}$ of the i -th coordinate of the final hyperboxes of the 100 simulations, the related standard deviations $\sigma(x_i)$, the absolute errors $\varepsilon(x_i) = |x_{i,avg} - x_{i,opt}|$, and the relative errors $|x_{i,avg} - x_{i,opt}|/x_{i,opt}$ in % are shown in Table 4.3. The values are rounded to three digits.

The distribution of the solution hyperboxes found by the algorithm is visualized via the histograms in the Figure 4.7(a) for the coordinate x_1 and in the Figure 4.7(b) for the coordinate x_2 . These plots and the values of the Table 4.3 validate again that the proposed algorithm converges to the analytical solution.

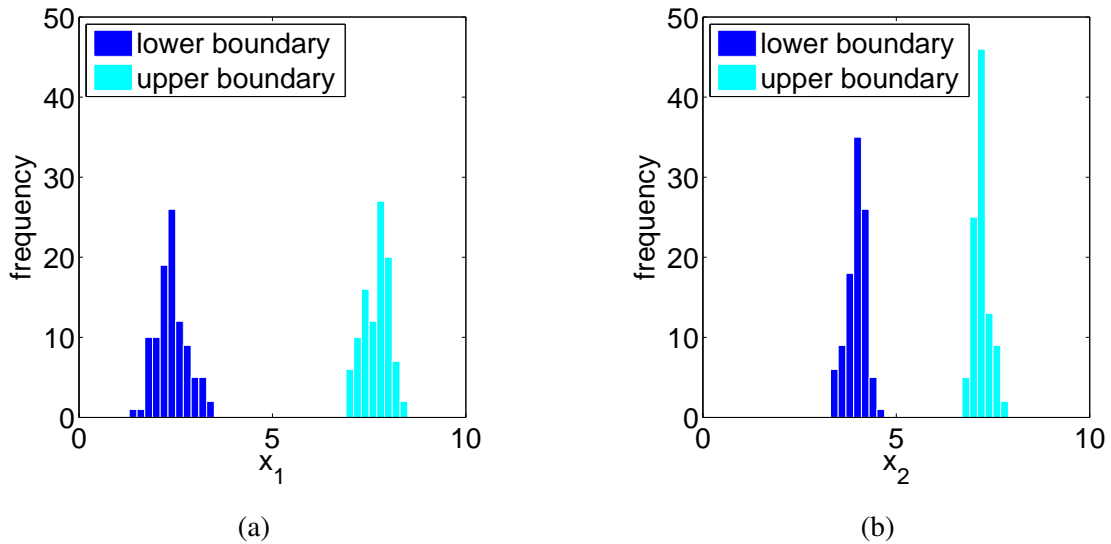


Figure 4.7: Problem 2 (second example). Distribution of the hyperboxes found by the algorithm for (a) coordinate x_1 and (b) coordinate x_2 for 100 simulations.

Third example

With

$$f(x_1, x_2, x_3) := \begin{cases} 0, & \text{if } \mathbf{g}(x_1, x_2, x_3, x_1, x_2, x_3) \leq \mathbf{0} \text{ component-by-component,} \\ 1, & \text{otherwise,} \end{cases}$$

and $f_c := 0.5$, the constrained optimization problem (4.2) with the specific values (4.7) can be equivalently expressed as the constrained optimization problem (P). Therefore, problem (4.2) with the specific values (4.7) can be solved numerically by using the algorithm presented in Chapter 3. The results of the numerical optimization which are obtained by executing the process (PR) for the convex polytope (4.7) are shown in Table 4.4 for $N = 100$ sample points per iteration. The mean of the coordinates of the final hyperboxes of the 100 simulations, the standard deviations, the absolute errors, and the relative errors are tabulated in Table 4.4. The values are rounded to three digits.

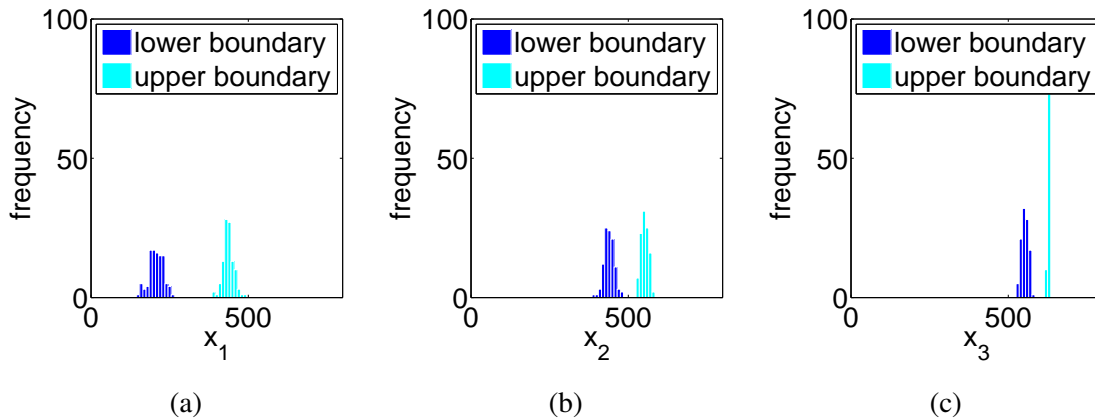


Figure 4.8: Problem 2 (third example). Distribution of the hyperboxes found by the algorithm for (a) the coordinate x_1 , (b) the coordinate x_2 and (c) the coordinate x_3 for 100 simulations.

The distribution of the solution hyperboxes found by the algorithm is visualized via histograms for $N = 100$ sample points in the Figure 4.8(a) for the coordinate x_1 , in the Figure 4.8(b) for the coordinate x_2 and in the Figure 4.8(c) for the coordinate x_3 . These plots and the values displayed in Table 4.4 show that the proposed algorithm converges to the analytical solution also in this three-dimensional example.

4.3 Problem 3. A hyperbox as boundary

The maximization of a hyperbox with another hyperbox as boundary is considered in d -dimensions for arbitrary $d > 1$. The convergence of the algorithm to the optimal solution is shown in high dimensions.

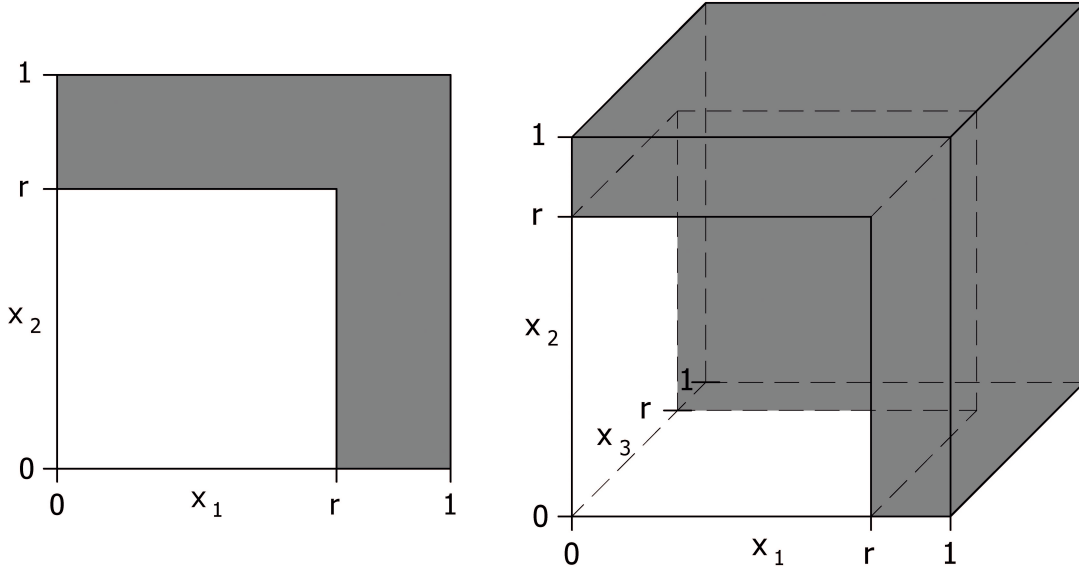


Figure 4.9: Problem 3. A hyperbox which defines the good space of the design space, inscribed in the unit d -cube.

4.3.1 Analytical solution

For $r \leq 1$, let $[0, r] \times [0, r] \times \cdots \times [0, r] \subseteq \mathbb{R}^d$ be a hyperbox which defines the good space of the design space. This hyperbox is inscribed in the d -dimensional unit cube

$$\Omega_{DS} := [0, 1] \times [0, 1] \times \cdots \times [0, 1] \subseteq \mathbb{R}^d \quad (4.8)$$

which serves as the design space. The constant $r = r(d)$ is chosen in such a way that the fraction of the good space is always 0.5, that is $r = \sqrt[d]{0.5}$ (see Figure 4.9 for a visualization in the case of $d = 2$ and $d = 3$ dimensions).

It holds that $x_i^{low} = 0$ for all $i = 1, 2, \dots, d$. Therefore, for $x_i := x_i^{up}$, $i = 1, 2, \dots, d$, we obtain the optimization problem

$$\mu(\Omega_{box}) = \prod_{i=1}^d x_i \rightarrow \max \quad (4.9)$$

under the affine inequality constraints

$$0 \leq x_i \leq r \quad \text{for all } i = 1, 2, \dots, d. \quad (4.10)$$

The analytical solution to this optimization problem is easily calculated by $x_i = x_{opt} := \sqrt[d]{0.5}$ for all $i = 1, 2, \dots, d$. The value x_{opt} tends to 1 as d tends to infinity which is seen by the values given in Table 4.5.

dimension	$d = 2$	$d = 3$	$d = 10$	$d = 20$	$d = 50$	$d = 100$
x_{opt}	0.707	0.794	0.933	0.966	0.986	0.993

Table 4.5: Problem 3. Optimal hyperbox for $d = 2, 3, 10, 20, 50, 100$ spatial dimensions.

Remark 4.3.1. *The inequality (4.10) describes the cross section of finitely many half-spaces. Thus, it describes a convex polytop.*

4.3.2 Numerical solution

The solution of the optimization problem (4.9) under the inequality constraint (4.10) is the same as the solution of problem (P) with

$$f(\mathbf{x}) := \begin{cases} 0, & \text{if } x_i \leq r \text{ for all } i = 1, 2, \dots, d, \\ 1, & \text{otherwise,} \end{cases}$$

and $f_c := 0.5$. This optimization problem is solved numerically for $d = 2, 3$ and $d = 10, 20, \dots, 100$ spatial dimensions. The algorithm is run with $N = 100$ sample points per iteration, and the process (PR) is executed.

The histograms in Figure 4.10 for $d = 2$ and in Figure 4.11 for $d = 3$ show the distribution of the resulting hyperboxes. In Table 4.6 for $d = 2$ and $d = 3$, respectively, the mean of the i -th coordinate $x_{i,avg}$ of the numerical solutions, the standard deviations $\sigma(x_i)$, the absolute errors $\varepsilon(x_i)$ and the relative errors in % are tabulated (rounded to three digits). The agreement between the numerical solutions of the algorithm and the analytical solutions is reasonably good.

To evaluate the results of the algorithm when the number of dimensions increases, the boundaries of the resulting hyperboxes $x_{i,\ell}$ are computed for every run ℓ . Then, the mean $x_{i,avg}$ of $x_{i,\ell}$ is calculated for every coordinate $i = 1, 2, \dots, d$. The mean x_{avg} of the coordinate means $x_{i,avg}$ is displayed in Figure 4.12(a) for $d = 2, 10, 20, \dots, 100$ dimensions. Note

d=2	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1	0.707	0.700	0.00728	0.00700	0.990
x_2	0.707	0.700	0.00748	0.00700	0.990
d=3	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1	0.794	0.784	0.00950	0.0100	1.26
x_2	0.794	0.784	0.00942	0.0100	1.26
x_3	0.794	0.784	0.0101	0.0100	1.26

Table 4.6: Problem 3. Analytical solution and related numerical results for 100 simulations for $d = 2$ and $d = 3$ dimensions.

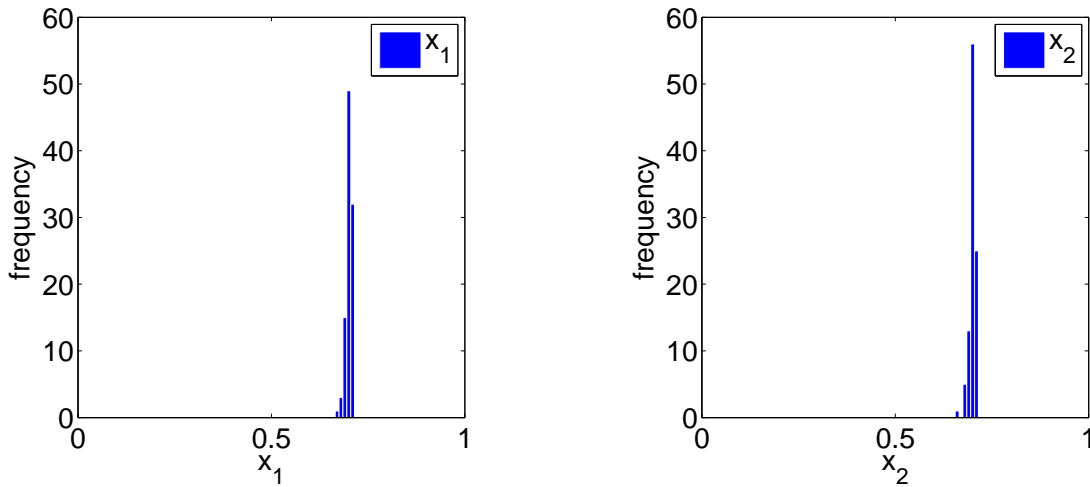


Figure 4.10: Problem 3. Distribution of 100 solution hyperboxes for $d = 2$ dimensions.

that the numerical solutions x_{avg} approximately agree with the analytical solutions, also in high dimensions. The relative error $|x_{avg} - x_{opt}|/x_{opt}$ is plotted in Figure 4.12(b). It is nearly independent of the number of dimensions, and it is smaller than 3%. Therefore, the algorithm approximates the analytical solution well, also in high dimensions.

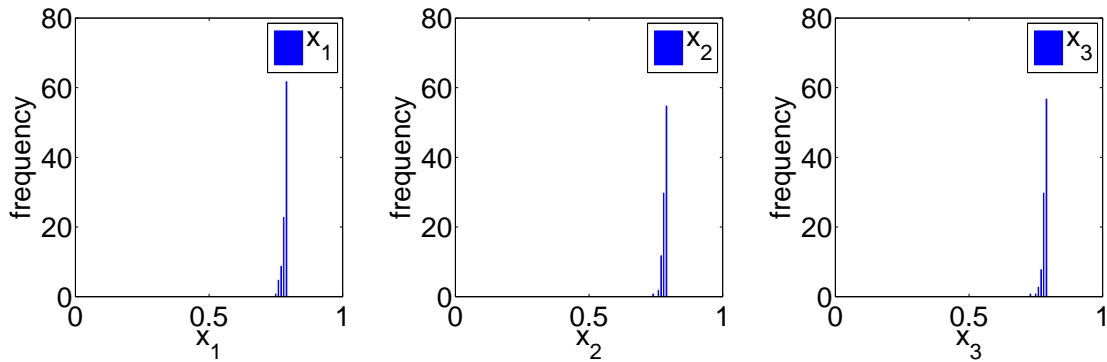


Figure 4.11: Problem 3. Distribution of 100 solution hyperboxes for $d = 3$ dimensions.

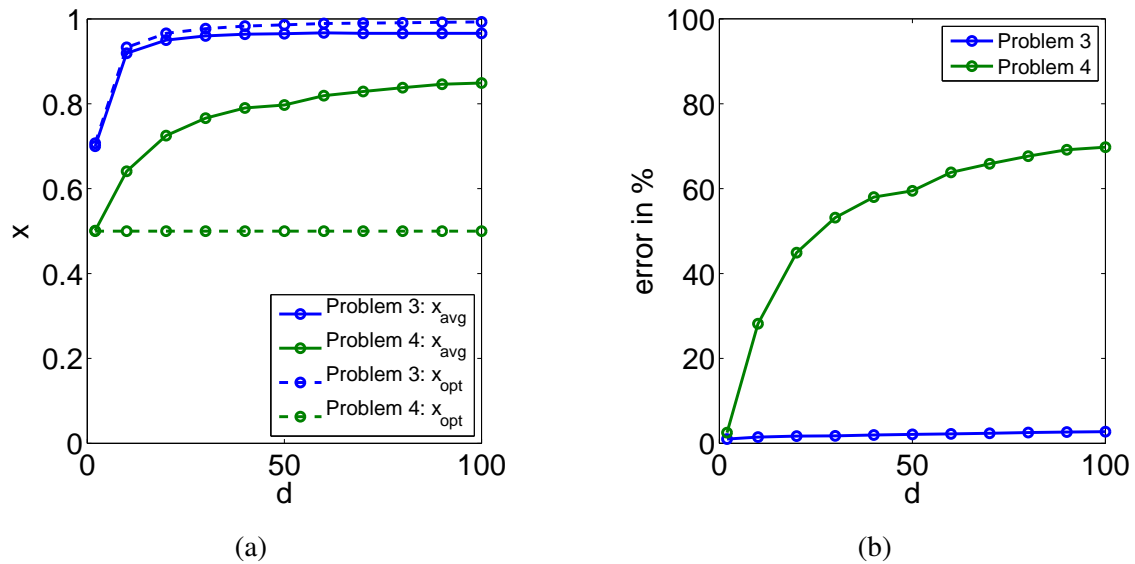


Figure 4.12: Numerical and analytical solutions for Problem 3 (hyperbox) and Problem 4 (tilted hyperplane). (a) Mean of the averaged simulations in comparison with the optimal solutions for $N = 100$ and $d = 2, 10, 20, \dots, 100$. (b) The relative error for $N = 100$ and $d = 2, 10, 20, \dots, 100$.

4.4 Problem 4. A tilted hyperplane as boundary

We consider now a d -dimensional problem with a tilted hyperplane as boundary of the good space. The analytical solution is computed and the algorithm's behavior for $d \rightarrow \infty$ is studied.

4.4.1 Analytical solution

Let us consider the d -dimensional unit cube (4.8) as design space Ω_{DS} . The boundary of the good space is a diagonal hyperplane which contains the point $[1, 1, \dots, 1]/2$ and has the normal vector $[1, 1, \dots, 1]/\sqrt{d}$. That is, the tilted hyperplane is described by the equation

$$g(\mathbf{x}) := \sum_{i=1}^d x_i - \frac{d}{2}$$

and intersects the design space in the middle. Note that the fraction of the good space is 50%, independently of d , see Figure 4.13(a) for $d = 2$ and Figure 4.13(b) for $d = 3$.

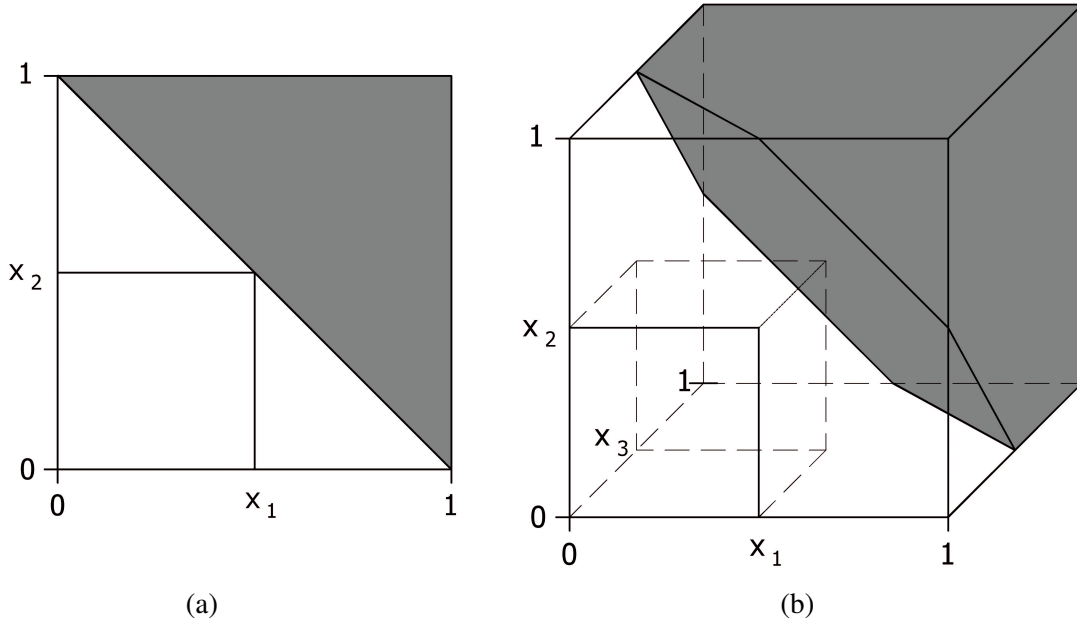


Figure 4.13: Problem 4. Tilted hyperplane in $d = 2$ and $d = 3$ dimensions with the maximum hyperbox.

The lower bounds are $x_i^{low} = 0$ for all $i = 1, 2, \dots, d$. Therefore, for $x_i := x_i^{up}$, $i = 1, 2, \dots, d$, we obtain the optimization problem

$$\mu(\Omega_{box}) = \prod_{i=1}^d x_i \rightarrow \max \text{ subject to } \mathbf{x} \in K. \quad (4.11)$$

Herein, the set

$$K := \{\mathbf{x} \in \Omega_{DS} : g(\mathbf{x}) \leq 0\}$$

is a compact subset of \mathbb{R}^d , because the set is closed and bounded. Thus, the optimization problem admits a solution on K , since the objective function $\mu(\Omega_{box})$ is convex and the constraint $g(\mathbf{x})$ is affine. The solution can be found with the help of Lagrangian multipliers, and is given by

$$\mathbf{x} = \left(\frac{1}{2}, \frac{1}{2}, \dots, \frac{1}{2} \right)$$

due to symmetry reasons. The associated hyperbox volume is

$$\mu(\Omega_{box}^{opt}) = \frac{1}{2^d}.$$

The solution is illustrated in Figure 4.13 for two and three spatial dimensions.

Remark 4.4.1. *The inequality $g(\mathbf{x}) \leq 0$ describes a half-space, and, therefore, it describes a convex polytop.*

4.4.2 Numerical solution

The constrained optimization problem (4.11) is equivalent to the constrained optimization problem (P) if we set $f(\mathbf{x}) = g(\mathbf{x})$ and $f_c = 0$. We shall compare the numerical results produced by the optimization algorithm with the analytical solutions. The algorithm is run with $N = 100$ sample points per iteration by executing the process (PR).

For low dimensions, $d = 2$ and $d = 3$, the distribution of the solution hyperboxes found by the algorithm is depicted in the histograms in Figure 4.14 for $d = 2$ dimensions and in Figure 4.15 for $d = 3$ dimensions, respectively. The mean of the i -th coordinate of the final hyperboxes $x_{i,avg}$, the associated standard deviation $\sigma(x_i)$, the absolute error $\varepsilon(x_i)$, and the relative error are tabulated in Table 4.7 for $d = 2$ and for $d = 3$ dimensions, respectively. The values are rounded to three digits. These results confirm that the proposed algorithm approximates the analytical solution in $d = 2$ and $d = 3$ dimensions.

Next, we compare the numerical results produced by the algorithm with the analytical solutions when the number of dimensions increases. We consider $d = 2, 10, 20, \dots, 100$ dimensions. As in the previous problem, $x_{i,avg}$ is calculated for each coordinate $i = 1, 2, \dots, d$. Then, the mean x_{avg} of the coordinate means versus the spatial dimension is plotted in Figure 4.12(a). In low dimensions, the results of the algorithm are in good agreement with the optimal solution. In high dimensions, i.e. $d \geq 10$, the numerical results strongly deviate from the analytical solutions. The relative error $|x_{avg} - x_{opt}|/x_{opt}$ is found in Figure 4.12(b). The error strongly increases when the number of dimensions increases. The volume of the solution hyperbox is larger than the optimal one in high dimensions.

d=2	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1	0.500	0.449	0.0598	0.0210	4.20
x_2	0.500	0.523	0.0597	0.0230	4.60
d=3	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1	0.500	0.515	0.0766	0.0150	3.00
x_2	0.500	0.508	0.0728	0.00800	1.60
x_3	0.500	0.519	0.0734	0.0190	3.80

Table 4.7: Problem 4. Analytical solution and related numerical results for 100 simulations for $d = 2$ and $d = 3$ dimensions.

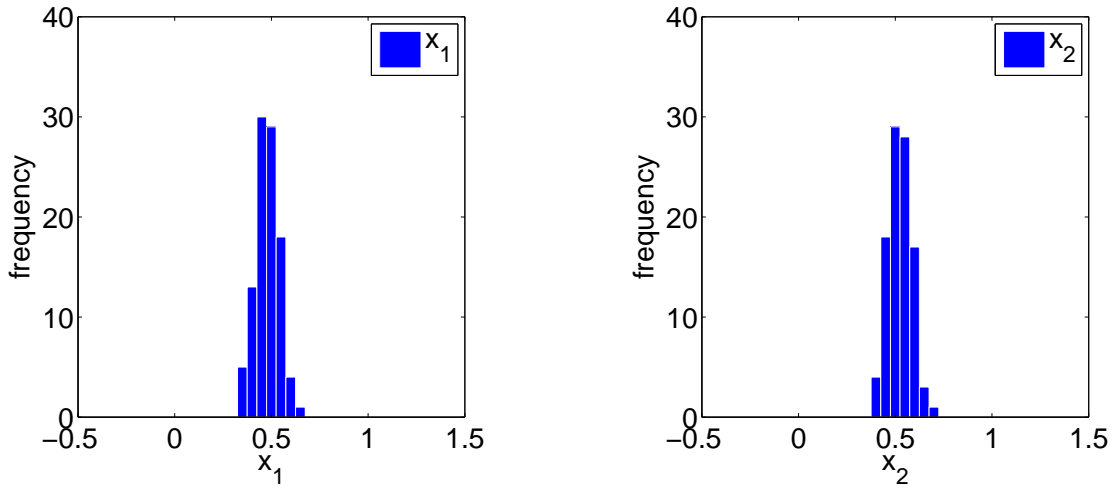


Figure 4.14: Problem 4. Distribution of 100 solution hyperboxes for $d = 2$ dimensions.

Although the numerical hyperbox boundaries deviate from the analytical hyperbox boundaries, the fraction of the bad space contained in the solution hyperboxes is small and of the order $1/N$. To identify this bad space, many sample points per iterations would be necessary. This effect reflects the curse of dimensionality (see [7]): If the interval widths increase a little with respect to the analytical solution hyperbox, the volume of the associated hyperbox increases so fast that the available data become sparse. For $N = 100$ sample points per iteration, the algorithm produces thus a hyperbox which is much larger than the analytical solution. The fraction of the good space is nevertheless close to 100%. We call

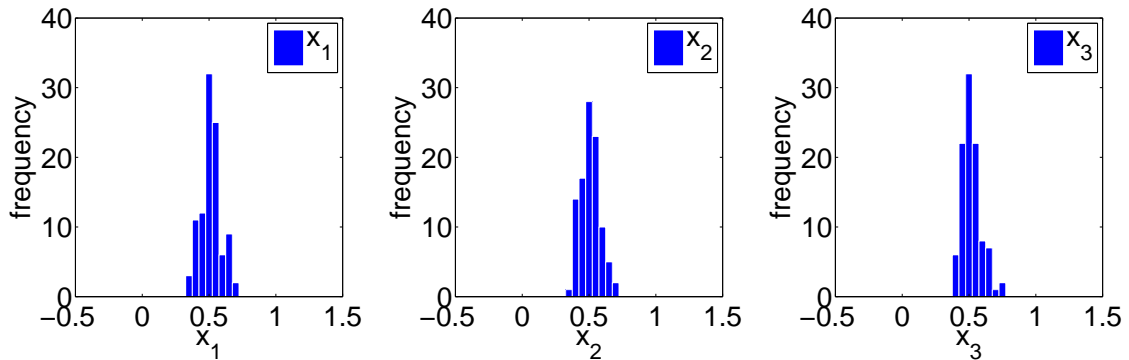


Figure 4.15: Problem 4. Distribution of 100 solution hyperboxes for $d = 3$ dimensions.

this effect *corner problem*.

In contrast to Problem 4, Problem 3 does not exhibit the curse of dimensionality. The effect depends on the shape of the boundary between good and bad space. If the interval widths increase a little with respect to the analytical solution hyperbox in Problem 3, a high fraction of bad volume is produced which is thus identified by the algorithm.

In summary, the algorithm ensures that the fraction a of good space is close to 1. However, the hyperbox boundaries may differ significantly from the analytical solution.

4.5 Corner problem

In this section, we investigate the corner problem in more detail. Therefore, we compare the numerical results of the algorithm with respect to its deviation from the optimal hyperbox in dependence on the type of the restricting boundary. The following problems will be considered.

- In Figure 4.16(a), the Problem 3 (hyperbox) which is described in Section 4.3 is shown.
- Figure 4.16(b) displays the Problem 4 (tilted hyperplane) which was introduced in Section 4.4.
- The good space is constrained by a quadrant of a circle around the d -dimensional vector $\mathbf{0}$ which is illustrated in Figure 4.16(c). This example is considered as Problem 5 (quadrant of a circle around $\mathbf{0}$).

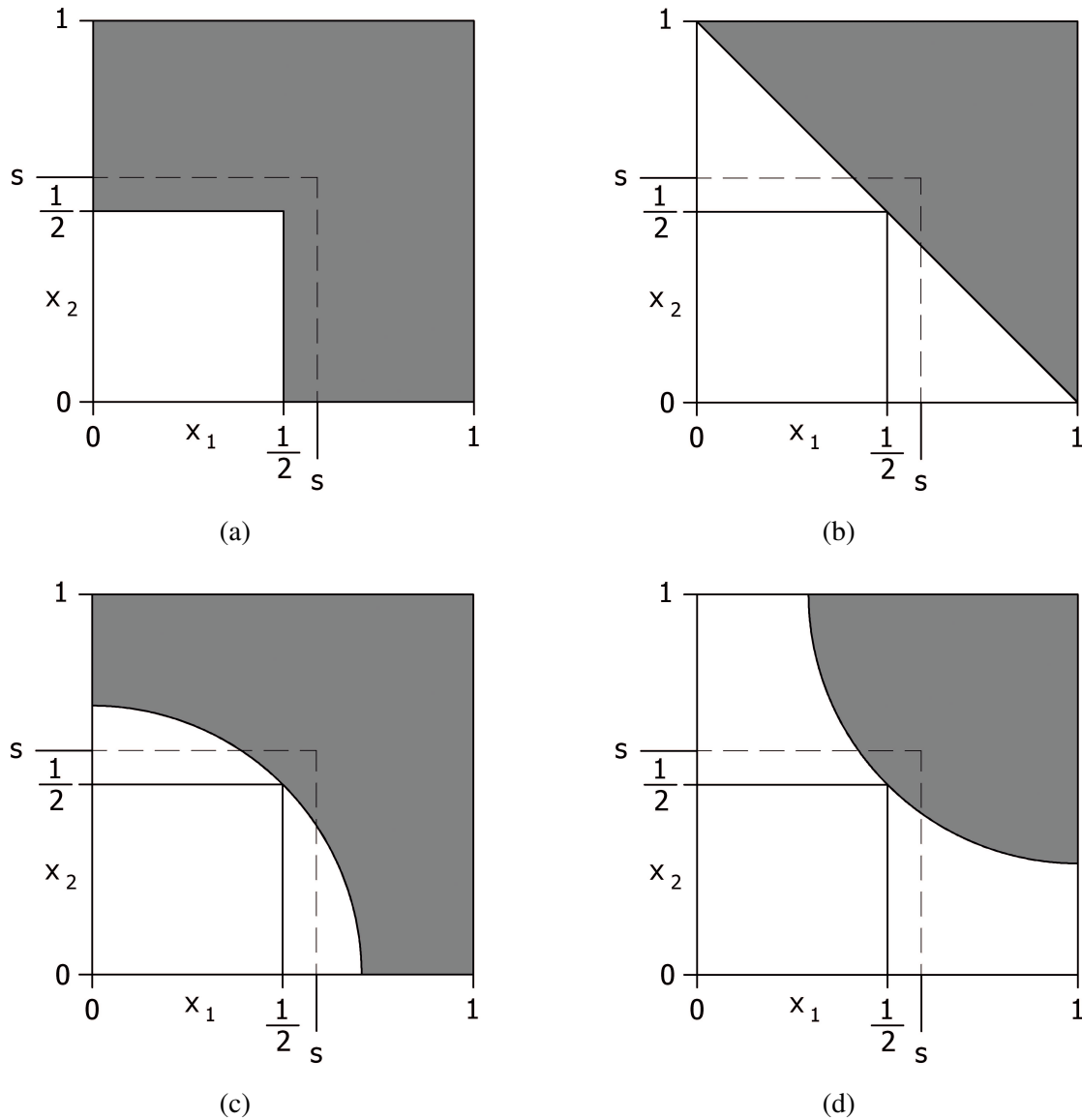


Figure 4.16: Problems considered: (a) Problem 3 (hyperbox), (b) Problem 4 (tilted hyperplane), (c) Problem 5 (quadrant of a circle around $\mathbf{0}$) and (d) Problem 6 (quadrant of a circle around $\mathbf{1}$).

- Problem 6 (quadrant of a circle around $\mathbf{1}$) is seen in Figure 4.16(d). The bad space V_{bad} is a quadrant of a circle around the d -dimensional vector $\mathbf{1}$, and the good space is given by $\Omega_{DS} \setminus V_{bad}$.

4.5.1 Numerical results

The algorithm is run for Problem 3, Problem 4, Problem 5 and Problem 6 with $N = 100$ sample points per iteration. The exploration phase is repeated 100 times, starting with the whole design space as initial guess. Then, the consolidation phase is run 100 times. Each problem is considered in $d = 2, 10, 20, \dots, 100$ dimensions.

The numerical results are shown in Figure 4.17. In this Figure, the ratio of the volume of the resulting hyperbox to the volume of the optimal hyperbox versus the number of dimensions is displayed. Each of the points corresponds to the average of 100 simulations.

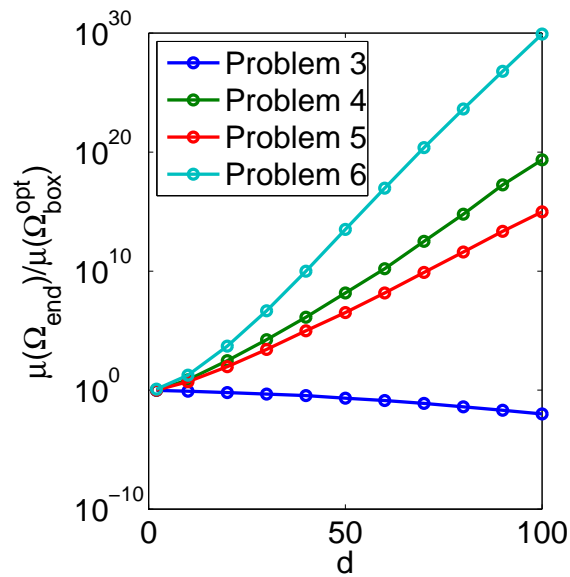


Figure 4.17: The volume of the resulting hyperbox divided by the volume of the optimal hyperbox versus the number of dimensions for the Problems 3–6.

Problem 3. It can be observed that the ratio $\mu(\Omega_{end})/\mu(\Omega_{box}^{opt})$ decreases when the dimensions increase. In addition, the diagram shows that the resulting hyperbox of the algorithm is smaller than the optimal hyperbox for all dimensions. Consequently, the corner problem does not show up for Problem 3.

Problems 4, 5 and 6. In contrast to Problem 3, the ratio $\mu(\Omega_{end})/\mu(\Omega_{box}^{opt})$ increases when the dimensions increase for Problem 4, and the volume of the resulting hyperbox of the algorithm is always larger than the volume of the optimal hyperbox. The same behavior is observed for the Problems 5 and 6. Therefore, the corner problem occurs in the Problems 4, 5 and 6. Although the volume of the numerically computed hyperbox is much larger than the solution hyperbox volume, the fraction of bad space contained in the numerical

solution hyperboxes is small. To identify this bad space, many sample points per iteration would be necessary. This reflects the curse of dimensionality as described in Section 4.4.

Comparing now the Problems 4, 5 and 6, the diagram in Figure 4.17 illustrates that the corner problem is stronger for Problem 6 than for Problem 4, because the deviation of $\mu(\Omega_{end})$ from $\mu(\Omega_{box}^{opt})$ is larger for Problem 6 than for Problem 4 for all dimensions considered, especially in high dimensions. Problem 4 illustrates a larger deviation of $\mu(\Omega_{end})$ from $\mu(\Omega_{box}^{opt})$ than Problem 5, and consequently, the corner problem is larger for Problem 4 than for Problem 5. Consequently, the corner problem softens in the order Problem 6, Problem 4, Problem 5.

4.5.2 Dependence on the boundary of the good space

We calculate the fraction a of good space in the hyperbox

$$\Omega_{box} := [0, s] \times [0, s] \subseteq \Omega_{DS}$$

for the Problems 3–6 in two dimensions. For all problems, the hyperbox volume $\mu(\Omega_{box})$ is s^2 and the hyperbox volume of the optimal hyperbox $\mu(\Omega_{box}^{opt})$ is $1/4$. Moreover, for Problem 3, we obtain for the fraction a of good space

$$a = \frac{1}{4s^2}.$$

For Problem 4, the fraction a of good space is

$$a = 1 - \frac{2(s - 1/2)^2}{s^2}.$$

For Problem 5, it holds

$$\begin{aligned} a &= 1 - \frac{\int_0^s \int_0^s \sqrt{1/2-s^2} \sqrt{1/2-x_1^2} 1 \, dx_2 dx_1}{s^2} \\ &= 1 - \frac{s^2 - \pi/8 - s \sqrt{1/2 - s^2} + 1/8 \left(\pi + 2s \sqrt{2} \sqrt{1 - 2s^2} + 2 \arcsin(\sqrt{1 - 2s^2}) \right)}{s^2} \\ &\quad + \frac{s \sqrt{2 - 4s^2} + \arcsin(\sqrt{2}s)}{4s^2} \end{aligned}$$

and for Problem 6, we obtain for a

$$\begin{aligned}
 a &= 1 - \frac{\int_{1-\sqrt{1/2-(s-1)^2}}^s \int_{1-\sqrt{1/2-(x_1-1)^2}}^s 1 \, dx_2 dx_1}{s^2} \\
 &= 1 - \frac{(s-1)(2s-2 + \sqrt{8s-2-4s^2})}{2s^2} \\
 &\quad - \frac{(s-1)\sqrt{8s-2-4s^2} + \arcsin(\sqrt{2}(s-1))}{4s^2} \\
 &\quad - \frac{\sqrt{2}|s-1|\sqrt{4s-1-2s^2} + \arcsin(\sqrt{4s-1-2s^2})}{4s^2}.
 \end{aligned}$$

Then, the first derivatives of the fraction of good space a with respect to $\mu(\Omega_{box})$ in $\mu(\Omega_{box}) = \mu(\Omega_{box}^{opt})$, this means

$$\left. \frac{da}{d\mu(\Omega_{box})} \right|_{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})},$$

and the associated second and third derivatives

$$\left. \frac{d^2a}{d\mu^2(\Omega_{box})} \right|_{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})} \quad \text{and} \quad \left. \frac{d^3a}{d\mu^3(\Omega_{box})} \right|_{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})}$$

are calculated and listed in Table 4.8.

	Problem 3	Problem 4	Problem 5	Problem 6
$\left. \frac{da}{d\mu(\Omega_{box})} \right _{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})}$	-4	0	0	0
$\left. \frac{d^2a}{d\mu^2(\Omega_{box})} \right _{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})}$	32	-16	-16	-16
$\left. \frac{d^3a}{d\mu^3(\Omega_{box})} \right _{\mu(\Omega_{box})=\mu(\Omega_{box}^{opt})}$	-384	288	256	320

Table 4.8: The first, second and third derivatives of the fraction of good space a with respect to $\mu(\Omega_{box})$ in $\mu(\Omega_{box}) = \mu(\Omega_{box}^{opt})$.

We observe in the Table 4.8 that the first derivative is -4 for Problem 3 and always 0 for Problems 4, 5 and 6. The second derivative is positive for Problem 3 while the second derivative is always -16 for Problems 4, 5 and 6. For Problem 3, the third derivative is negative and positive for Problems 4, 5 and 6. Moreover, the third derivative increases in the order Problem 5, Problem 4 and Problem 6.

The fraction a of good space versus the volume of the hyperbox is plotted in Figure 4.18 for the Problems 3–6. We observe that the fraction of good space decreases very fast when

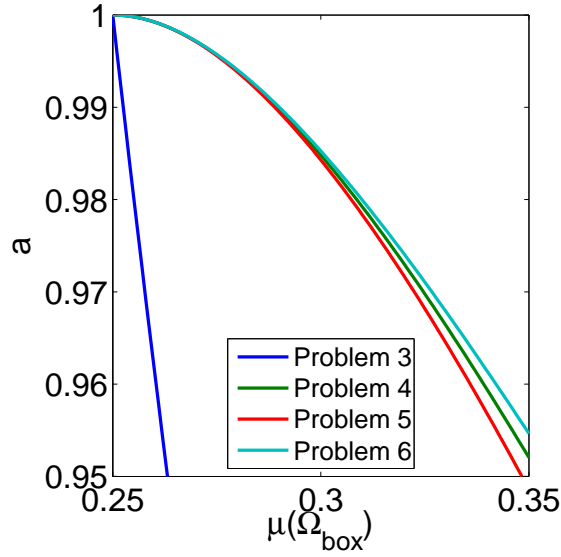


Figure 4.18: The fraction a of good space versus the volume of the hyperbox for the Problems 3–6.

the volume increases for Problem 3. Consequently, we do not have the corner problem. Figure 4.18 shows for Problem 6 that the fraction of good space decreases very slowly with increasing volume. Therefore, the corner problem is very large. For Problem 4, the fraction of good space decreases faster than for Problem 6 with increasing volume and the corner problem is smaller than for Problem 6. In Figure 4.18, the fraction of good space decreases faster for Problem 5 than for Problem 4 with increasing volume. Consequently, the corner problem is for Problem 4 larger than for Problem 5.

These considerations confirm the observations of the previous subsection, and let us conclude that we do not have the corner problem if the first derivative is not equal to 0, i.e., if

$$\left. \frac{da}{d\mu(\Omega_{\text{box}})} \right|_{\mu(\Omega_{\text{box}})=\mu(\Omega_{\text{box}}^{\text{opt}})} \neq 0.$$

If the first derivative is equal to 0, i.e., if

$$\left. \frac{da}{d\mu(\Omega_{\text{box}})} \right|_{\mu(\Omega_{\text{box}})=\mu(\Omega_{\text{box}}^{\text{opt}})} = 0,$$

and the second derivative is negative, i.e., if

$$\left. \frac{d^2a}{d\mu^2(\Omega_{\text{box}})} \right|_{\mu(\Omega_{\text{box}})=\mu(\Omega_{\text{box}}^{\text{opt}})} < 0,$$

then we will observe the corner problem. In addition, we can conclude that the corner problem softens if the second derivative decreases.

Moreover, the results of this and the previous subsection indicate that a concave boundary of the good space yields a larger corner problem, i.e. a larger deviation of the resulting hyperbox from the optimal hyperbox, than a convex boundary of the good space.

Chapter 5

Convergence behavior in the consolidation phase

The user of the algorithm seeks a hyperbox with a large measure $\mu(\Omega_{box})$. Furthermore, a large fraction a of the good design space is required. The fraction a is typically small during the exploration phase. The purpose of the consolidation phase is to increase this fraction to a desired level, possibly at the cost of a smaller hyperbox volume. This chapter studies how the number of dimensions and the number of sample points affect the quality of the resulting solution hyperbox and the speed of convergence in the consolidation phase.

The following three problems are considered to study the convergence behavior in the consolidation phase.

- A hyperbox which defines the good space of the design space is inscribed in another hyperbox which is the design space (see Figure 5.1(a)). The ratio of the volume of the good space and the volume of the design space is 0.5. This is considered as Problem 3.
- In Problem 4, the good space is constrained by a tilted hyperplane which intersects a d -dimensional hyperbox $I_1 \times I_2 \times \dots \times I_d \subseteq \mathbb{R}^d$ in the middle, cf. Figure 5.1(b).
- Problem 7 is a high-dimensional and non-linear engineering problem from crash analysis as described in the Section 2.1. The gray surface in Figure 5.1(c) shows the bad space of a two-dimensional cross section.

First, in Subsection 5.1, a convergence coefficient is introduced as a measure of the convergence speed. In Subsection 5.2, an analytical model is derived which describes the

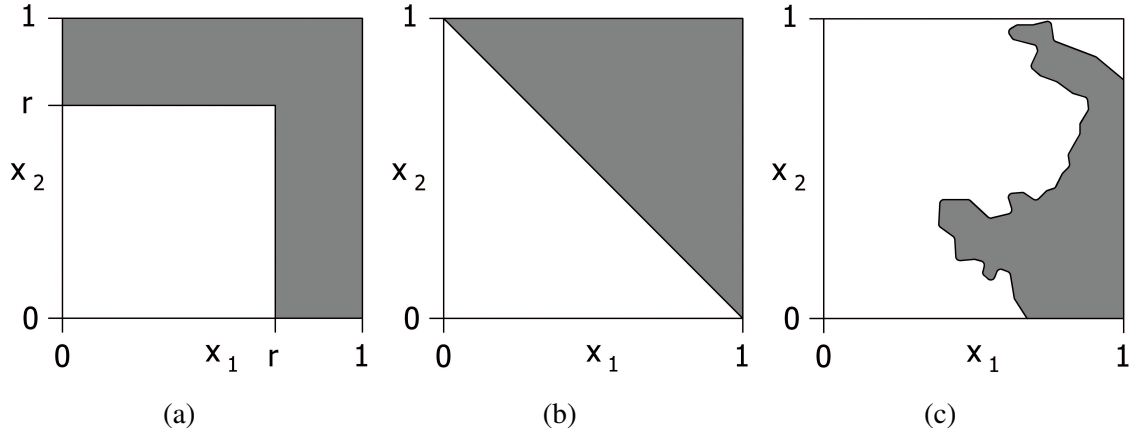


Figure 5.1: Problems considered: (a) Problem 3 (hyperbox), (b) Problem 4 (tilted hyperplane), and (c) Problem 7 (front crash).

behavior of the speed of convergence for Problem 3. Then, in Subsection 5.3, the Problems 3 (hyperbox), 4 (tilted hyperplane), and 7 (front crash) are considered to investigate the influence of dimensionality on the convergence behavior. Afterwards, the Subsection 5.4 is dedicated to demonstrate that the speed of convergence and the volume of the resulting solution hyperbox can be controlled by the choice of the sample size. In Subsection 5.5, it is shown that typically the speed of convergence and the size of the resulting solution hyperbox are in conflict with each other. The conflict is illustrated by appropriate Pareto frontiers for all problems. Depending on the preference for speed or volume size, the sample size may be chosen and the total number of required simulations may be estimated.

5.1 Convergence coefficient

To quantify the convergence speed, we introduce the *convergence coefficient* which describes the dependence of the fraction N_g/N of good sample points on the number of function evaluations. The fraction of good sample points versus the number of iterations is depicted for Problem 4 with $d = 100$ and $N = 100$ in Figure 5.2.

The convergence coefficient is defined as the ratio of c and N , where the coefficient c with $0 < c \leq \bar{c}$ is identified by a discrete least-squares fit in the iteration steps $0 \leq k < M$. Here, M is the number of iterations. That is, we seek the coefficients c and b such that

$$\sum_{k=0}^{M-1} (a_k - \tilde{a}_k)^2 \rightarrow \min$$

where a_k denotes the fraction of good sample points in the k -th iteration step, $\tilde{a}_k := 1 - b \exp(-ck)$ and $b := 1 - \tilde{a}_0$.

Figure 5.2 displays the fraction of good sample points and the fitted curve \tilde{a}_k . The agreement is acceptable and the convergence coefficient is used as a measure for the convergence speed of the algorithm.

Remark 5.1.1. *In the discrete least-squares fit, c would be infinite if $N_g/N = 1$ holds after only one iteration step. Therefore, the constant c has an upper limit \bar{c} . This limit \bar{c} is calculated by a continuous least-squares-fit of a curve $a_x = 0.5 + 0.5x$ with $a_0 = 0.5$ and $a_1 = 1$. This means that \bar{c} is the minimum of the problem*

$$\int_0^1 (a_x - \tilde{a}_x)^2 dx \rightarrow \min .$$

Consequently, we obtain $\bar{c} \approx 3.7$.

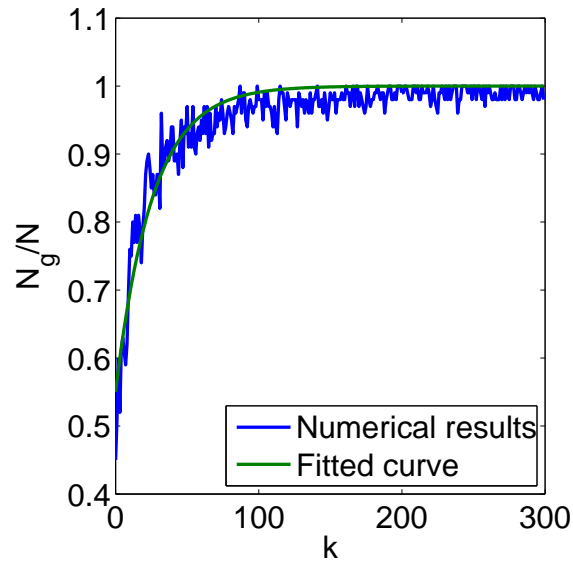


Figure 5.2: Problem 4. Convergence speed of the fraction of good sample points: Numerical results and fitted curve \tilde{a}_k for $d = 100$ and $N = 100$.

5.2 Analytical model

An analytical model is derived to describe the behavior of the convergence speed for Problem 3 in $d \in \mathbb{N}$ dimensions.

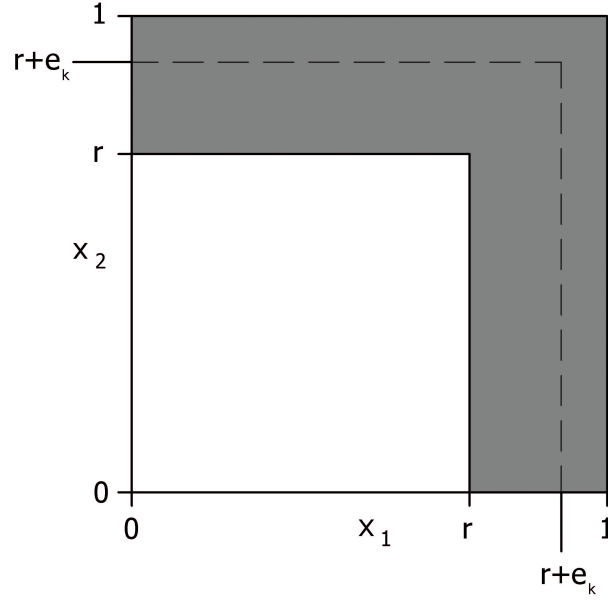


Figure 5.3: Problem 3. Candidate hyperbox in iteration step k for the analytical model.

As introduced in Subsection 4.3, the design space is $\Omega_{DS} := [0, 1]^d$ and the good space of the design space is the hyperbox $[0, r]^d$. In the k -th iteration of the cutting algorithm, we suppose that the candidate hyperbox is $[0, r + e_{k,1}] \times [0, r + e_{k,2}] \times \cdots \times [0, r + e_{k,d}]$. This hyperbox is sampled by a Monte Carlo method with the number N of sample points and the cutting algorithm is applied. In the model, the boundaries are relocated in all dimensions i where the dimension x_i of a bad sample point \mathbf{x} is larger than r , i.e. $x_i \geq r$. Therefore, for each dimension, it holds $e_k := e_{k,i}$ with $i = 1, \dots, d$ (see Figure 5.3 for an illustration in the case of two dimensions). The size e_{k+1} of the bad space in the $(k + 1)$ -st iteration depends only on the size e_k of the bad space in the k -th iteration. This means that the sequence $\{e_k\}$ constitutes a Markov chain, i.e., it holds

$$P(e_{k+1}|e_0, e_1, \dots, e_k) = P(e_{k+1}|e_k). \quad (5.1)$$

A Markov chain is defined according to [30].

Definition 5.2.1 (Markov chain). *Let the state space S be a finite or countable infinite set. Then, the sequence X_0, X_1, \dots builds a Markov chain if the following Markov property holds: For all $N \in \mathbb{N}_0$ and for all $x_0, x_1, \dots, x_N \in S$ with*

$$P(X_0 = x_0, X_1 = x_1, \dots, X_N = x_N) > 0$$

it holds that

$$P(X_{N+1} = x_{N+1}|X_0 = x_0, X_1 = x_1, \dots, X_N = x_N) = P(X_{N+1} = x_{N+1}|X_N = x_N). \quad (5.2)$$

Theorem 5.2.2. *It holds*

$$\mathbb{E}(e_{k+1}|e_k) = \frac{1}{(r + e_k)^N} \frac{1}{N + 1} \left\{ (r + e_k)^{N+1} - r^{N+1} \right\}. \quad (5.3)$$

Proof. Due to (5.1), we get with [14]

$$\mathbb{E}(e_{k+1}|e_k) = \int_{\mathbb{R}^N} e_{k+1}(x_1, x_2, \dots, x_N) p_1(x_1) p_2(x_2) \cdots p_N(x_N) d(x_1, x_2, \dots, x_N).$$

Herein, the probability density $p_j(x_j)$ for a uniform sampling is independent of the number j of the sample point x_j , i.e., $p(x) := p_1(x) = p_2(x) = \cdots = p_N(x)$, and given by

$$p(x) = \begin{cases} 1/(r + e_k), & \text{if } 0 \leq x \leq r + e_k, \\ 0, & \text{otherwise.} \end{cases}$$

Therefore, it follows that

$$\mathbb{E}(e_{k+1}|e_k) = \int_{[0, r+e_k]^N} \frac{e_{k+1}(x_1, \dots, x_N)}{(r + e_k)^N} d(x_1, x_2, \dots, x_N).$$

Note that $e_{k+1} = e_k$ if no bad sample point exists. Otherwise, e_{k+1} is reduced to the bad sample point which has the smallest distance to the boundary position of the good space. Therefore,

$$e_{k+1} = \begin{cases} e_k, & \text{if } 0 \leq \max\{x_j : j = 1, 2, \dots, N\} \leq r, \\ \min\{x_j - r : x_j > r\}, & \text{if } r < \max\{x_j : j = 1, 2, \dots, N\} \leq r + e_k, \end{cases} \quad (5.4)$$

and

$$\begin{aligned} \mathbb{E}(e_{k+1}|e_k) &= \frac{1}{(r + e_k)^N} \left\{ \int_{[0, r]^N} e_k d(x_1 x_2 \dots x_N) \right. \\ &\quad \left. + \int_{[0, r+e_k]^N \setminus [0, r]^N} \left(\min_j \{x_j - r : x_j > r\} \right) d(x_1, x_2, \dots, x_N) \right\} \\ &= \frac{1}{(r + e_k)^N} \left\{ r^N e_k + \int_{[0, r+e_k]^N \setminus [0, r]^N} \left(\min_j \{x_j - r : x_j > r\} \right) d(x_1, x_2, \dots, x_N) \right\}. \end{aligned}$$

The integral on the right hand side of this expression is

$$\begin{aligned} &\int_{[0, r+e_k]^N \setminus [0, r]^N} \left(\min_j \{x_j - r : x_j > r\} \right) d(x_1, x_2, \dots, x_N) \\ &= \sum_{\ell=1}^N \binom{N}{\ell} \int_{[0, r]^{N-\ell}} \int_{[r, r+e_k]^\ell} \min_{1 \leq j \leq \ell} \{x_j - r\} d(x_1, x_2, \dots, x_\ell) d(x_{\ell+1}, x_{\ell+2}, \dots, x_N) \\ &= \sum_{\ell=1}^N \binom{N}{\ell} r^{N-\ell} \int_{[0, e_k]^\ell} \min_{1 \leq j \leq \ell} \{x_j\} d(x_1, x_2, \dots, x_\ell). \end{aligned}$$

There are $\ell!$ different combinations of the tuple $(x_1, x_2, \dots, x_\ell)$. Thus, the domain $[0, e_k]^\ell$ of integration is divided into $\ell!$ identical subsets. Due to symmetry, we get

$$\begin{aligned} & \int_{[0, e_k]^\ell} \min_{1 \leq j \leq \ell} \{x_j\} d(x_1, x_2, \dots, x_\ell) \\ &= \ell! \int_{[0, e_k]^\ell} x_1 \mathcal{X}_{\{x_1 < x_2 < \dots < x_\ell\}} d(x_1, x_2, \dots, x_\ell) \\ &= \ell! \int_{[0, e_k]^{\ell-1}} \left(\int_0^{e_k} x_1 \mathcal{X}_{\{x_1 < x_2\}} dx_1 \right) \mathcal{X}_{\{x_2 < x_3 < \dots < x_\ell\}} d(x_2, x_3, \dots, x_\ell). \end{aligned}$$

With

$$\int_0^{e_k} x_m^m \mathcal{X}_{\{x_m < x_{m+1}\}} dx_m = \int_0^{x_{m+1}} x_m^m dx_m = \frac{1}{m} x_{m+1}^{m+1} \text{ for all } m = 1, 2, \dots, \ell - 1,$$

it follows that

$$\begin{aligned} \int_{[0, e_k]^\ell} \min_{1 \leq j \leq \ell} \{x_j\} d(x_1, x_2, \dots, x_\ell) &= \frac{1}{2} \ell! \int_{[0, e_k]^{\ell-1}} x_2^2 \mathcal{X}_{\{x_2 < x_3 < \dots < x_\ell\}} d(x_2, x_3, \dots, x_\ell) \\ &= \frac{1}{2} \cdot \frac{1}{3} \ell! \int_{[0, e_k]^{\ell-2}} x_3^3 \mathcal{X}_{\{x_3 < x_4 < \dots < x_\ell\}} d(x_3, x_4, \dots, x_\ell) \\ &\quad \vdots \\ &= \frac{1}{\ell!} \ell! \int_0^{e_k} x_\ell^\ell dx_\ell \\ &= \frac{e_k^{\ell+1}}{\ell + 1}. \end{aligned}$$

Inserting this expression into the original integral, yields

$$\begin{aligned} \mathbb{E}(e_{k+1}|e_k) &= \frac{1}{(r + e_k)^N} \left\{ r^N e_k + \sum_{\ell=1}^N \binom{N}{\ell} r^{N-\ell} \frac{e_k^{\ell+1}}{\ell + 1} \right\} \\ &= \frac{1}{(r + e_k)^N} \sum_{\ell=0}^N \binom{N}{\ell} r^{N-\ell} \frac{e_k^{\ell+1}}{\ell + 1} \\ &= \frac{1}{N + 1} \frac{1}{(r + e_k)^N} \sum_{\ell=0}^N \binom{N + 1}{\ell + 1} r^{N-\ell} e_k^{\ell+1}. \end{aligned}$$

By using the binomial theorem

$$(r + e_k)^{N+1} = \sum_{\ell=0}^{N+1} \binom{N + 1}{\ell} r^{N+1-\ell} e_k^\ell, \quad (5.5)$$

it can be concluded that the expectation value of the width of the bad space in the $(k + 1)$ -st iteration is given by

$$\mathbb{E}(e_{k+1}|e_k) = \frac{1}{N + 1} \frac{1}{(r + e_k)^N} \{(r + e_k)^{N+1} - r^{N+1}\}.$$

□

With the help of this theorem, we can approximate the expectation value of the fraction of good sample points a_{k+1} by

$$\mathbb{E}(a_{k+1}) = \mathbb{E} \left[\left(\frac{r}{r + e_{k+1}} \right)^d \right] \stackrel{(\star)}{=} \left[\mathbb{E} \left(\frac{r}{r + e_{k+1}} \right) \right]^d \stackrel{(\star\star)}{\approx} \left(\frac{r}{r + \mathbb{E}(e_{k+1}|e_k)} \right)^d. \quad (5.6)$$

Because the dimensions are assumed to be independent of each other, the identity (\star) holds in (5.6). The approximation $(\star\star)$ in (5.6) results from the following theorem.

Theorem 5.2.3. *Let $z(\omega) = \mathbb{E}(z) + y(\omega)$ be a random variable such that $|y(\omega)| \leq \widehat{y}$ almost surely. Then, for a smooth function $f : z \mapsto \mathbb{R}$, it holds*

$$\mathbb{E}(f(z)) = f(\mathbb{E}(z)) + \mathcal{O}(\widehat{y}^2). \quad (5.7)$$

Proof. We shall abbreviate $\bar{z} = \mathbb{E}(z)$. Due to $y(\omega) = z(\omega) - \bar{z}$, the random variable y is centered, i.e., $\mathbb{E}(y) = 0$, because it holds

$$\mathbb{E}(y) = \mathbb{E}(z - \bar{z}) = \mathbb{E}(z) - \bar{z} = \mathbb{E}(z) - \mathbb{E}(z) = 0.$$

The Taylor expansion of f in the neighborhood of \bar{z} reads as

$$f(z) = f(\bar{z}) + (z - \bar{z})f'(\bar{z}) + \mathcal{O}(|y|^2) = f(\bar{z}) + yf'(\bar{z}) + \mathcal{O}(\widehat{y}^2).$$

Here, only y is stochastic, the other expressions are deterministic. For the expectation value of $f(z)$, we obtain by inserting the Taylor expansion of f

$$\begin{aligned} \mathbb{E}(f(z)) &= \mathbb{E}(f(\bar{z}) + yf'(\bar{z}) + \mathcal{O}(\widehat{y}^2)) \\ &= \mathbb{E}(f(\bar{z})) + \mathbb{E}(yf'(\bar{z})) + \mathcal{O}(\widehat{y}^2). \end{aligned} \quad (5.8)$$

Because \bar{z} is deterministic, so also $f(\bar{z})$, it holds

$$\mathbb{E}(f(\bar{z})) = f(\bar{z}). \quad (5.9)$$

Moreover, $f'(\bar{z})$ is a deterministic scalar, and hence

$$\mathbb{E}(yf'(\bar{z})) = \mathbb{E}(y)f'(\bar{z})$$

which, in view of $\mathbb{E}(y) = 0$, yields

$$\mathbb{E}(yf'(\bar{z})) = 0. \quad (5.10)$$

Consequently, from equation (5.8), we obtain with (5.9) and (5.10)

$$\mathbb{E}(f(z)) = f(\bar{z}) + \mathcal{O}(\widehat{y}^2) = f(\mathbb{E}(z)) + \mathcal{O}(\widehat{y}^2).$$

□

Equation (5.7) implies that

$$\mathbb{E}(f(z)) \approx f(\mathbb{E}(z))$$

is an approximation of second order in \widehat{y} provided that the fluctuation of z is small.

In Figure 5.4, the results of the analytical formula (5.3) and the numerical results of the algorithm are depicted for increasing dimensions where e_k was chosen as $\mathbb{E}(e_{k+1}|e_k)$. For the numerical calculation, the constant r (which describes the boundary between the good and bad space in one dimension) is chosen in such a way that the fraction of the good space is 50% of the design space for all dimensions under consideration. The prediction of the convergence speed by the formula is in a good agreement with the result of the algorithm.

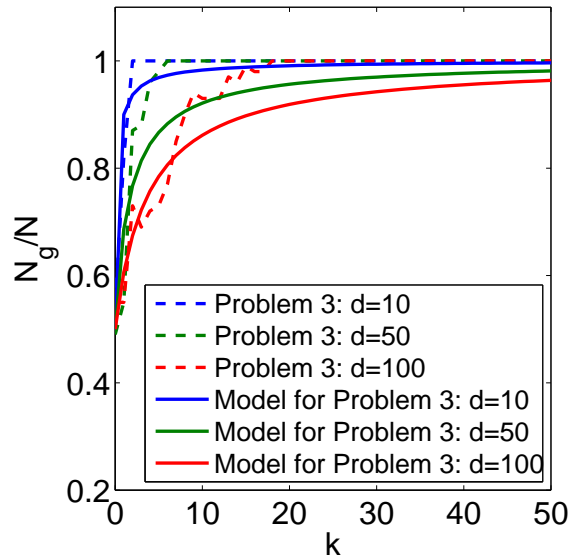


Figure 5.4: Problem 3. Fraction of good sample points in each iteration step for $N = 100$: Comparison of the results of the algorithm and the calculation by the analytical model.

Remark 5.2.4. *The model (5.6) has the following shortcomings: (i.) While the algorithm removes a bad sample point by relocating only one boundary, a bad sample point in the model is removed by relocating the boundary of every dimension i for which $r < x_i < r + e_{k,i}$. This effect is negligible for the following reason: the model should converge faster (more bad space is removed per bad sample point). However, Figure 5.4 exhibits the contrary behavior: the convergence of the analytical model is slower than the convergence of the algorithm. Therefore, this shortcoming is not dominating.*

(ii.) The results of the analytical formula (5.3) are recursively calculated, i.e., in every iteration step the expectation of e_k is taken to calculate the expectation of e_{k+1} , this means

$\mathbb{E}(e_{k+1}|\mathbb{E}(e_k|\mathbb{E}(e_{k-1}|...)))$). Therefore, the trajectory of the expectations of the values e_k is calculated. Note that this is not the average of the trajectories which is $\mathbb{E}(e_{k+1}|e_0)$.

(iii.) In the model, it is assumed that the boundary of the hyperbox will be relocated in every iteration step to the bad sample point with the smallest distance to the boundary position of the good space. In the algorithm, the boundary of the hyperbox will be relocated in every iteration step to the good sample point with the smallest distance to the boundary position of the good space. As a consequence, the numerical results for Problem 2 exhibit a higher convergence speed than the results of the analytical model.

5.3 Influence of the dimensionality

The influence of dimensionality on the convergence speed is illustrated in Figure 5.5 for Problems 3, 4 and 7. The fraction of good sample points versus the number of iterations is shown for changing dimensions and $N = 100$. The convergence coefficient is found in the legend of the plots. Note that for all problems the convergence speed decreases when the dimension increases. This behavior is also observed via the analytical formula (5.6), cf. Figure 5.4.

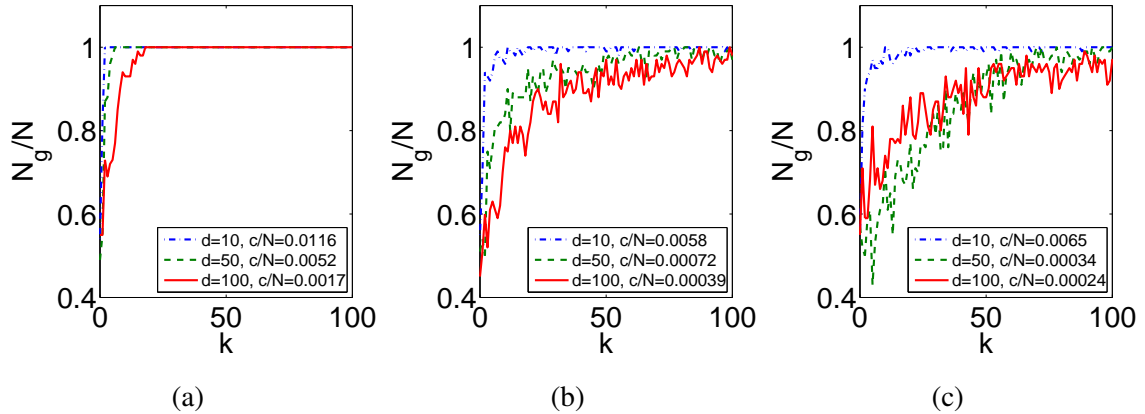


Figure 5.5: Fraction of good sample points versus the number of iterations for $N = 100$ for (a) Problem 3 (hyperbox), (b) Problem 4 (tilted hyperplane), and (c) Problem 7 (front crash).

The following mechanism describes the influence of dimensionality, which will be called *each bad sample point can be used for one dimension only*: If, for example, there are 100 sample points in a hyperbox and the fraction of good points is 80%, then 20 bad sample

points are used to remove bad space. In 10 dimensions, there will be enough sample points to remove some of the bad space in each dimension because we find two bad sample points for each dimension on average. The problem which arises in 100 dimensions is that the bad sample space can be removed in 20 dimensions at most because a bad sample point can be used to remove bad space in one dimension only. Bad space without bad sample points cannot be removed. Consequently, the optimal hyperbox will be overestimated in some dimensions. This is illustrated in Figure 5.7(a) for Problem 2 in $d = 2$ dimensions and one bad sample point. The grey area describes the bad space of the design space and the white area describes the good space. This fact explains that the algorithm converges slower in higher dimensions.

5.4 Influence of the number of sample points

The influence of the number of sample points per iteration on the convergence speed and the hyperbox volume is studied in this subsection for the high-dimensional Problems 3, 4 and 7. Every point in the diagrams in Figure 5.6 displays the mean of five calculations.

5.4.1 Number of sample points versus convergence speed

Problem 3. Figure 5.6(a) shows a maximum of the convergence coefficient for $d = 10$ and $d = 50$ dimensions. The reason for this maximum is that at this point the fraction N_g/N of good sample points reaches the value 1 in only one iteration step. Thus, $c \approx 3.7$ (see Subsection 5.1). For N larger than the peak location, $N_g/N = 1$ is also reached in only one iteration step, and c/N decreases. Left of the peak, the convergence coefficient increases with increasing number of sample points. This can be explained by the mechanism *each bad sample point can be used for one dimension only*, see Subsection 5.3. For each dimension, at least one bad sample point with $x_i \geq r$ has to exist to remove the bad space, see Figure 5.7(a). If the number of sample points increases, the number of dimensions where bad volume can be removed increases. Consequently, the convergence speed increases until there are sufficiently many bad sample points, i.e. in every dimension at least one.

Problem 4. Figure 5.6(b) shows that the convergence coefficient increases when the number of sample points per iteration decreases independently of the number of dimensions. If the number of sample points per iteration is small, the volume which is removed per bad sample point is large, also the bad volume. To explain the mechanism, consider a two-dimensional sample for Problem 3 in the design space $[0, 1] \times [0, 1]$. Recall that the optimal

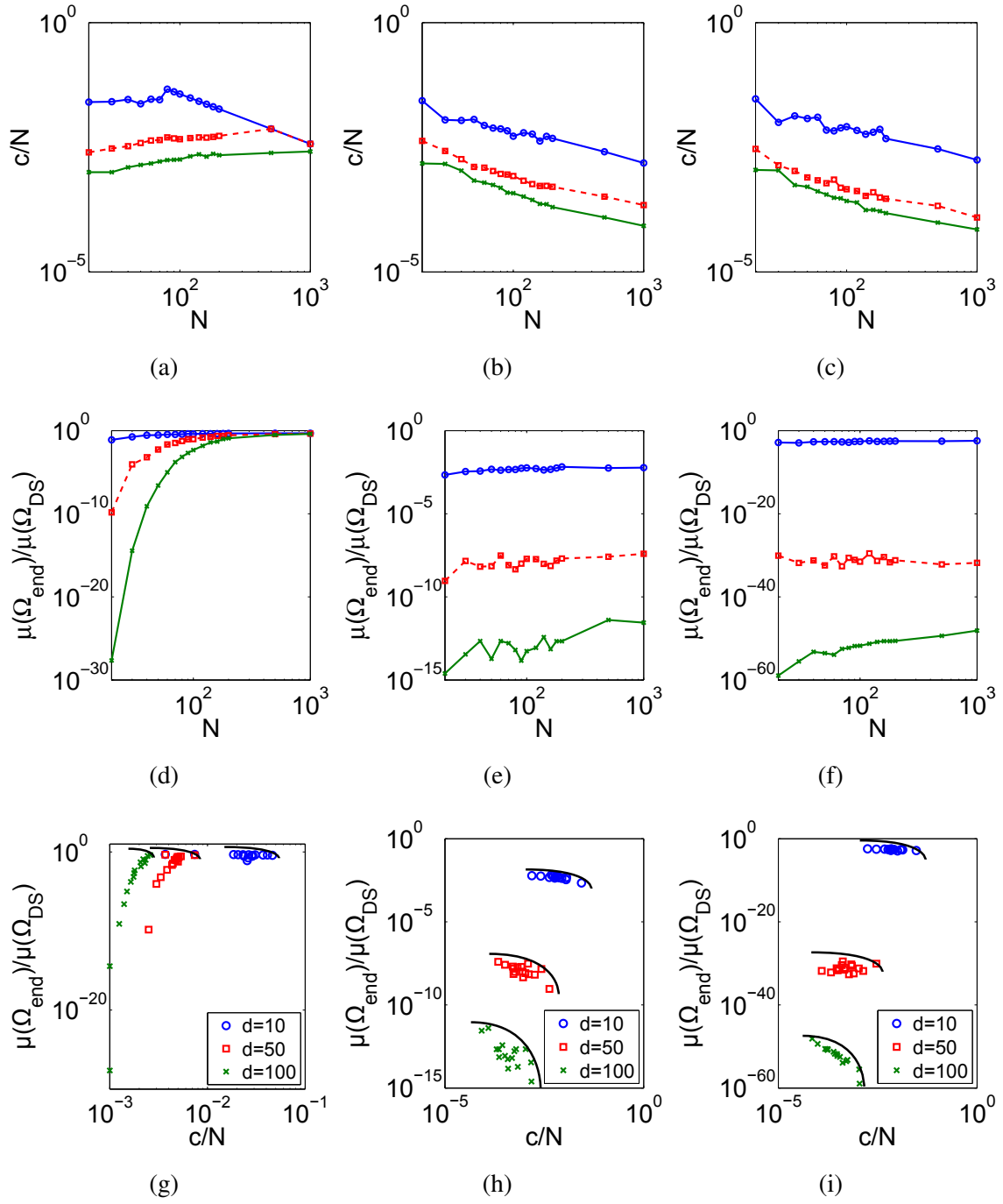


Figure 5.6: Convergence coefficient (first row), normalized volume (second row), Pareto frontier for the convergence coefficient and the normalized volume (third row) for Problem 2 (first column), Problem 3 (second column), and Problem 4 (third column).

solution is $x_1 = x_2 = 0.5$. Then, the distance between the optimal boundary location 0.5 and the sample point x_j where the boundary of the candidate hyperbox will be located is calculated in dimension $i = 1$, i.e.,

$$\varepsilon = x_j - 0.5.$$

In Figure 5.7(b), the measured distance ε is illustrated by four sample points. The distances for $L = 10\,000$ repetitions for each number N of sample points are averaged and plotted in Figure 5.8(b). The average of the measure ε is always larger than 0 and increases with an increasing number N of sample points. If $\varepsilon > 0$, the optimal hyperbox is overestimated, therefore, this phenomenon is called *overestimation due to sparse sampling*. The larger the overestimation the slower the convergence speed. Consequently, the convergence speed decreases with increasing N .

Problem 7. Changing now to Problem 4 (front crash), according to Figure 5.6(c), a behavior can be observed that is similar to the one observed for Problem 3. The convergence coefficient c/N decreases when the number N of sample points increases. This indicates similar shapes of the boundaries separating good from bad space.

5.4.2 Number of sample points versus volume

In addition to the convergence speed of the fraction of good sample points, described by the convergence coefficient c/N , the volume of the final hyperbox needs to be taken into account.

In Figure 5.6(d) can be observed that for Problem 3 the normalized volume increases a lot when the number of sample points per iteration increases. Whereas, the corresponding diagram for Problem 4 (see Figure 5.6(e)) shows that here the volume increases only a little when the number of sample points per iteration is increased. The same is observed for Problem 7, cf. Figure 5.6(f).

The increase of volume for larger N can be explained by *overestimation due to sparse sampling* which was explained in the previous Subsection 5.4.1, see Figure 5.7(b). This effect shows that with increasing number of sample points N , the average of the considered distance ε increases, see Figure 5.8(b). If the number of sample points is small, the volume which is removed per bad sample point is large and the resulting volume is small. Therefore, the volume of the final hyperbox increases with increasing number of sample points.

Another reason for the increasing volume is the *impossibility of boundary corrections*: To explain this mechanism, Problem 3 is considered in one dimension. One sample with N

sample points is made in the interval $[0, 1]$. The expectation of the distance ε_u between the optimal point $r = 0.5$ and the sample point x_j where the boundary will be after removing the bad sample points is obtained from the following theorem.

Theorem 5.4.1. *It holds*

$$\mathbb{E}(\varepsilon_u) = \int_{[0,1]^N \setminus [0,0.5]^N} \min\{0.5 - x_j : x_j < 0.5\} d(x_1, \dots, x_N) = \frac{1}{N+1} \left(1 - \frac{2}{2^{N+1}}\right). \quad (5.11)$$

Proof. The proof is quite similar to the proof of Theorem 5.2.2. Let $p_j(x_j)$ denote the probability density which belongs to the sample x_j . Since it is uniform and independent of the number j , it holds

$$p_j(x_j) = \begin{cases} 1/(r + e_k), & \text{if } 0 \leq x_j \leq r + e_k, \\ 0, & \text{otherwise,} \end{cases}$$

for all $j = 1, 2, \dots, N$. Therefore, it follows

$$\begin{aligned} \mathbb{E}(\varepsilon_u) &= \int_{\mathbb{R}^N} \varepsilon_u(x_1, x_2, \dots, x_N) p_1(x_1) p_2(x_2) \cdots p_N(x_N) d(x_1, x_2, \dots, x_N) \\ &= \int_{[0, r+e_k]^N} \frac{\varepsilon_u(x_1, \dots, x_N)}{(r + e_k)^N} d(x_1, x_2, \dots, x_N). \end{aligned}$$

We conclude that $\varepsilon_u = 0$ if no bad sample point exists. Otherwise, ε_u is the distance between the boundary position of the good space and the good sample point with the largest parameter value. We thus arrive at

$$\varepsilon_u = \begin{cases} 0, & \text{if } 0 \leq \max\{x_j : j = 1, 2, \dots, N\} \leq r, \\ \min\{r - x_j : x_j < r\}, & \text{if } r < \max\{x_j : j = 1, 2, \dots, N\} \leq r + e_k. \end{cases} \quad (5.12)$$

Hence, we find

$$\begin{aligned} \mathbb{E}(\varepsilon_u) &= \frac{1}{(r + e_k)^N} \int_{[0, r+e_k]^N \setminus [0, r]^N} \left(\min_j \{r - x_j : x_j < r\} \right) d(x_1, x_2, \dots, x_N) \\ &= \frac{1}{(r + e_k)^N} \sum_{\ell=1}^N \binom{N}{\ell} \int_{[0, r]^{N-\ell}} \min_{\ell+1 \leq j \leq N} \{r - x_j\} \left(\int_{[r, r+e_k]^\ell} d(x_1, \dots, x_\ell) \right) d(x_{\ell+1}, \dots, x_N) \\ &= \frac{1}{(r + e_k)^N} \sum_{\ell=1}^N \binom{N}{\ell} e_k^\ell \int_{[0, r]^{N-\ell}} \min_{\ell+1 \leq j \leq N} \{r - x_j\} d(x_{\ell+1}, x_{\ell+2}, \dots, x_N) \\ &= -\frac{1}{(r + e_k)^N} \sum_{\ell=1}^N \binom{N}{\ell} e_k^\ell \int_{[0, r]^{N-\ell}} \max_{\ell+1 \leq j \leq N} \{x_j - r\} d(x_{\ell+1}, x_{\ell+2}, \dots, x_N) \\ &= -\frac{1}{(r + e_k)^N} \sum_{\ell=1}^N \binom{N}{\ell} e_k^\ell \int_{[-r, 0]^{N-\ell}} \max_{\ell+1 \leq j \leq N} \{x_j\} d(x_{\ell+1}, x_{\ell+2}, \dots, x_N). \end{aligned}$$

There are $(N - \ell)!$ different combinations of the tuple $(x_{\ell+1}, x_{\ell+2}, \dots, x_N)$. Thus, the domain $[-r, 0]^{N-\ell}$ of integration is divided into $(N - \ell)!$ identical subsets. Symmetry implies

$$\begin{aligned}
& \int_{[-r,0]^{N-\ell}} \max_{\ell+1 \leq j \leq N} \{x_j\} d(x_{\ell+1}, x_{\ell+2}, \dots, x_N) \\
&= (N - \ell)! \int_{[-r,0]^{N-\ell}} \chi_{\{x_{\ell+1} < x_{\ell+2} < \dots < x_N\}} x_N d(x_{\ell+1}, x_{\ell+2}, \dots, x_N) \\
&= (N - \ell)! \int_{[-r,0]^{N-\ell-1}} \left(\int_{-r}^0 \chi_{\{x_{\ell+1} < x_{\ell+2}\}} dx_{\ell+1} \right) \chi_{\{x_{\ell+2} < x_{\ell+3} < \dots < x_N\}} x_N d(x_{\ell+2}, x_{\ell+3}, \dots, x_N) \\
&\quad \vdots \\
&= (N - \ell)! \int_{-r}^0 \frac{1}{(N - \ell - 1)!} (r + x_N)^{N-\ell-1} x_N dx_N \\
&= \frac{r^{1-\ell+N}}{\ell - N - 1}.
\end{aligned}$$

With this equation, we obtain

$$\begin{aligned}
\mathbb{E}(\varepsilon_u) &= \frac{1}{(r + e_k)^N} \left\{ - \sum_{\ell=1}^N \binom{N}{\ell} e_k^\ell \frac{r^{1-\ell+N}}{\ell - N - 1} \right\} \\
&= \frac{1}{(r + e_k)^N} \sum_{\ell=1}^N \binom{N+1}{\ell} \frac{1}{N+1} e_k^\ell r^{N+1-\ell} \\
&= \frac{1}{N+1} \frac{1}{(r + e_k)^N} \left\{ \sum_{\ell=0}^{N+1} \binom{N+1}{\ell} e_k^\ell r^{N+1-\ell} - r^{N+1} - e_k^{N+1} \right\}.
\end{aligned}$$

By using again the binomial theorem (5.5), the expectation value of ε_u in the $(k + 1)$ -st iteration is thus given by

$$\mathbb{E}(\varepsilon_u) = \frac{1}{N+1} \frac{1}{(r + e_k)^N} \{(r + e_k)^{N+1} - r^{N+1} - e_k^{N+1}\}.$$

With $e_k = 0.5$ and $r = 0.5$, the desired result follows. \square

Due to the rectangular boundaries of Problem 3, the optimal hyperbox will always be underestimated, see Figure 5.7(a), and ε_u is an underestimation measure. If the solution hyperbox boundary is larger than 0.5, then $\mathbb{E}(\varepsilon_u) = 0$. We observe in Figure 5.8(a) that with an increasing number N of sample points, the expectation $\mathbb{E}(\varepsilon_u)$ decreases. Therefore, the volume of the resulting hyperbox increases.

In Problem 3, the dependency of the final hyperbox volume on the number of sample points is greater than in Problem 4. Problem 3 has boundaries which are axis-parallel and, therefore, the effect of the impossibility of boundary corrections is much stronger in Problem 3 than in Problem 4.

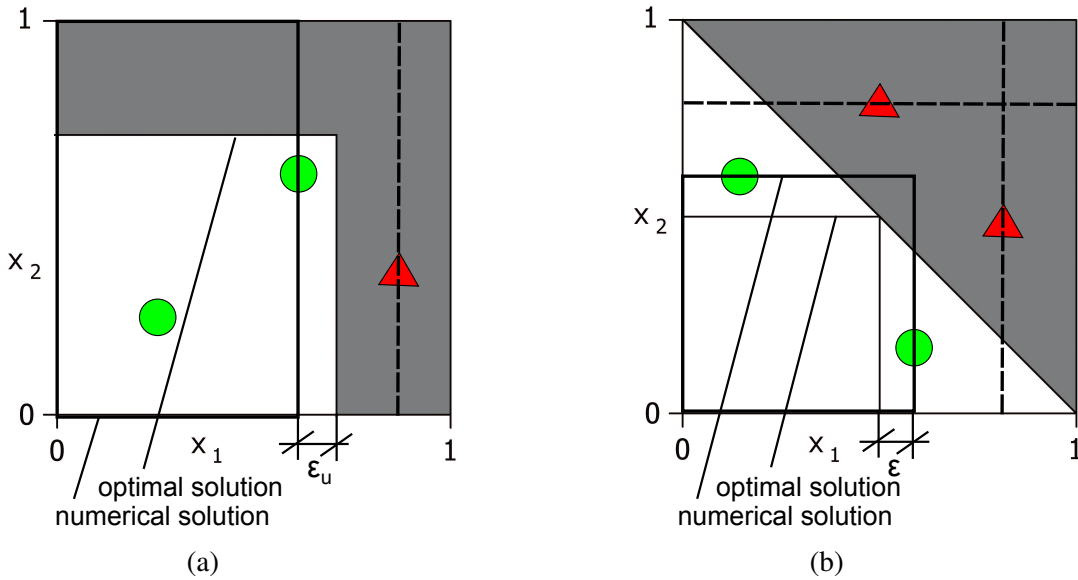


Figure 5.7: Mechanisms explaining over- and underestimation. (a) Problem 3. *Each bad sample point can be used for one dimension only and impossibility of boundary corrections.* (b) Problem 4. *Overestimation due to sparse sampling.*

5.5 Convergence speed versus hyperbox volume

Typically, there is a conflict between the fast convergence and the volume size of the resulting hyperbox. For Problems 4 and 7, the convergence coefficient decreases whereas the resulting hyperbox volume increases upon increasing N . This conflict can be visualized by a Pareto frontier, see Figure 5.6. For Problem 3, this conflict exists only for large N , that is, right of the peak in Figure 5.6(a). For small N however, both, convergence coefficient and volume size, increase, see Section 5.4.

A Pareto frontier can be defined in accordance with [62].

Definition 5.5.1 (Pareto frontier). *An element $x^* \in X$ is a Pareto optimal solution if the following holds: if there exists a solution $x' \in X$ such that $f_i(x') < f_i(x^*)$ for an i , then there exists a j with $f_j(x') > f_j(x^*)$. The set of all efficient solutions X_E is called a Pareto optimal set. If x^* is Pareto optimal, then $z^* = f(x^*)$ is called non-dominated point. The set of all non-dominated points is referred as Pareto frontier.*

If a large volume is desired, as many sample points as possible have to be chosen. This will slow down the algorithm. If fast convergence is desired, as few sample points as possible have to be chosen. The resulting hyperbox will be small. For computing the

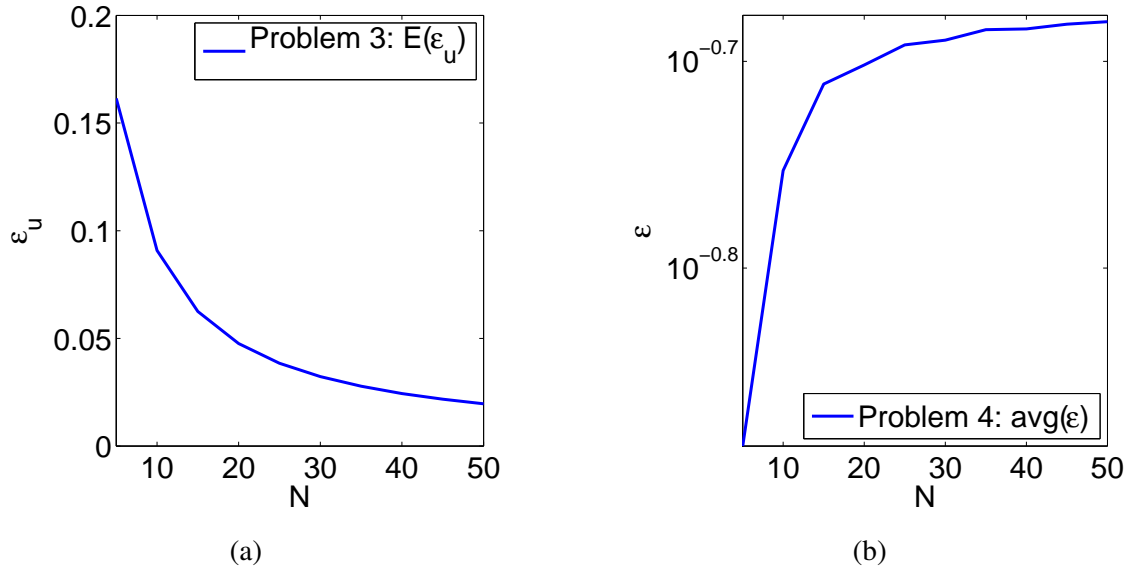


Figure 5.8: Mechanisms explaining over- and underestimation. (a) Expectation value of the underestimation measure $\mathbb{E}(\epsilon_u)$ for Problem 3. (b) Average $\text{avg}(\epsilon)$ of the measure ϵ for Problem 4.

desired speed of convergence	c/N	0.01	0.001	0.0001	0.00001
number of evaluations	NM	267	2661	26594	265928

Table 5.1: Convergence coefficient and the total number of function evaluations required for $\tilde{a} = 97\%$ for a fraction of the good space of 50% in the initial candidate hyperbox.

optimal number of sample points, the following procedure is proposed: Depending on the preference for speed or volume size, c/N is chosen using Figure 5.6. The total number of required evaluations for a particular choice of c/N is tabulated in Table 5.1 for an $a_0 = 50\%$, where a_0 is the fraction of the good space in the initial candidate hyperbox.

Chapter 6

Applications

In this chapter, different applications of the algorithm to optimization problems in the automobile industry are presented. These problems arise from front crash design, from a forming process, and from rear passenger safety. We introduce the underlying problem formulation and report on the results of the algorithm.

6.1 Front vehicle crash design

Results are presented for a vehicle front crash problem to demonstrate the applicability to high-dimensional and non-linear industrial problems. In Subsection 6.1.4, the front crash problem has 64 degrees of freedom. In Subsection 6.1.5, the front crash problem has 89 degrees of freedom.

6.1.1 Evaluation

The numerical optimization of the USNCAP front crash is considered which was already introduced in Section 2.1. The vehicle structure and the restraint systems are to be designed such that the loads on crash test dummy and the deformation of the passenger cell stay below critical threshold values. Design work is done on three levels, the vehicle level, the component level and the detail level – this is similar to target cascading [39].

Design goals on the vehicle level

The primary focus of structural development in an early design phase lies on satisfying the following design goal [21, 38, 88]: The maximum deceleration of the passenger cell

measured at the bottom of the B-pillar should not exceed the critical threshold value $a_{pulse,c}$, that is, $a_{pulse} \leq a_{pulse,c}$. Note that this criterion is only sufficient for the preliminary design. In final tests, the vehicle performance will be evaluated with respect to dummy loads.

Component properties on the component level

The structural behavior in a USNCAP-type front crash depends primarily on the distributed vehicle mass and the resistance force of structural elements against deformation, expressed as *force-deformation characteristics*

$$F = \hat{F}(u),$$

see [21, 38]. F is a longitudinal force exerted by the structural component under the relative longitudinal displacement $u = u_b - u_a$, with u_a and u_b being the x-displacements of the component boundaries, see Figure 6.1. For a fixed vehicle mass, the maximum deceleration is given by

$$a_{pulse} = f(\hat{F}_1(u_1), \hat{F}_2(u_2), \dots, \hat{F}_n(u_n)) \quad (6.1)$$

where $\hat{F}_k(u_k)$ denotes the force-deformation characteristic of the k -th out of n components.

Detail level

The force-deformation characteristics depend again on geometrical and material detail parameters p_j^k , that is

$$\hat{F}_k(u_k) = \bar{F}_k(u_k; p_1^k, p_2^k, \dots, p_m^k), \quad (6.2)$$

with j being a parameter index and m being the number of detail parameters. Detail parameters may be sheet metal thicknesses, profile geometries, yield strengths or hardening curves. They will determine the force-deformation characteristics of each structural member and, thus, also determine the overall structural behavior.

In classical vehicle design, detail parameters are varied, until the design goals on the vehicle level are satisfied. In a new design approach, the component level was introduced to enable the design of component properties such as force-deformation characteristics without specifying the underlying detail parameters [38]. This is useful, for example in an early design phase, when detail parameters are difficult to be specified or simply unknown. For a flexible and robust design, requirements on component properties are to be identified as permissible intervals [87]. In a subsequent development step, detail parameters are then specified such that all component requirements are satisfied.

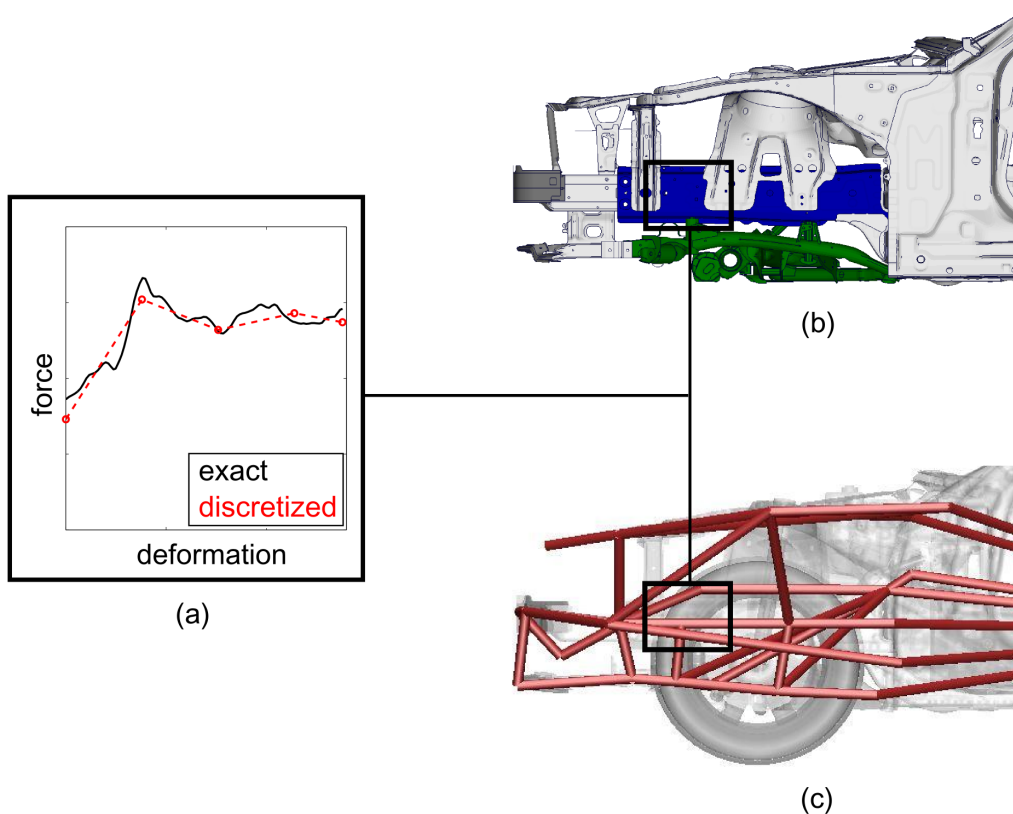


Figure 6.1: (a) Force-deformation characteristics of a component of the vehicle structure: exact and discretized. (b) Detail vehicle model. (c) Reduced model.

6.1.2 Crash simulation models

The front crash is modeled by differential equations. A numerical method to approximately solve these differential equations is the finite element method which is introduced in Subsection 6.1.3.

Detail finite element model

In a detailed finite element model, all detail parameters are specified. For crash simulations, this is typically a model of the entire vehicle. From this, all quantities on the vehicle level, such as the vehicle deceleration, and on the component level, such as force-deformation characteristics, can be computed. Force-deformation characteristics are derived from section forces $F(t)$ and deformations $u(t)$ for $\dot{u}(t) \geq 0$. A detail finite element model maps the detail level onto the vehicle level and the component level.

Reduced model

By contrast, the reduced model described in [21, 38] computes the vehicle behavior directly from force-deformation characteristics, as in expression (6.1). It maps the component level onto the vehicle level. The reduced model is depicted in Figure 6.1(c).

The structural components of the vehicle are represented through one-dimensional force elements. The nodes which connect the one-dimensional force elements of the finite element model have only one degree of freedom each, namely the translation in x -direction, i.e. in the direction of the movement of the vehicle. The reference force-deformation characteristics of the detail finite element model which is presented in Subsection 6.1.1 are mapped onto the force elements. In Figure 6.1(c), the one-dimensional force elements are illustrated by red lines. The discrete mass distribution is similar to the detail finite element model. The model accurately represents the reference model in forces and masses and, therefore, in the resulting deceleration, obtained through time-integration.

Force-deformation characteristics contain more information than necessary. The mechanical behavior of a structural member that is relevant for a USNCAP-type front crash can be sufficiently well approximated by 4–10 discrete force values at specified support points. Therefore, $\hat{F}_k(u_k)$ is discretized as shown in Figure 6.1(a). The maximum vehicle deceleration is then given by

$$a_{pulse} = f(F_1, F_2, \dots, F_d) \quad (6.3)$$

with F_i being the force values at specified support points of the force-displacement characteristics, and d being the total number of discrete force values of the force elements which are relevant for the crash design. To compute expression (6.3), we use the reduced crash model.

6.1.3 Finite element method

In order to numerically solve partial differential equations of elasto-plasto-problems, the finite element method is used. In the finite element method, the surface of the entire vehicle is divided into a certain finite number of one-, two- or three-dimensional elements. We consider first order finite elements. Beside special purpose elements (e.g. mass, springs, dashpots), the model is mainly composed with structural (in one dimension e.g. beams, trusses and in two dimensions e.g. shells, membranes) and continuum elements (e.g. solids). The elements are connected by nodes. At the nodes, density and stress are discretized as

nodal masses and nodal forces. Depending on the underlying theory of the elements, the nodes have three or six degrees of freedom – translation and rotation – and conjugated quantities – forces and moments as well as masses and rotary inertias. In dynamic applications, at the nodes, the inertia forces, the internal forces due to deformation and the external forces (e.g. gravity) equilibrate each other. The following is found in [16] and [79]. The time dependent equation of motion in the dynamic equilibrium is

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{F}_{int}(\mathbf{u}, \dot{\mathbf{u}}) - \mathbf{F}_{ext} = \mathbf{0}$$

$$\mathbf{M}\ddot{\mathbf{u}} + \mathbf{C}\dot{\mathbf{u}} + \mathbf{K}\mathbf{u} - \mathbf{F}_{ext} = \mathbf{0}$$

with \mathbf{M} being a diagonal matrix, \mathbf{C} being the damping matrix and \mathbf{K} being the stiffness matrix. The vectors \mathbf{u} , $\dot{\mathbf{u}}$ and $\ddot{\mathbf{u}}$ are the displacement, velocity and acceleration vectors, respectively. \mathbf{F}_{ext} denotes the external nodal forces, and \mathbf{F}_{int} denotes the internal resisting forces due to the deformation and damping of the material. The term $\mathbf{M}\ddot{\mathbf{u}}$ represents the inertial forces generated by the acceleration $\ddot{\mathbf{u}}$. The dynamic equation is integrated in time by starting from an initial condition at time zero

$$\mathbf{u}(0) = \mathbf{u}_0, \dot{\mathbf{u}}(0) = \dot{\mathbf{u}}_0.$$

The central difference method is used as an explicit time integration method. Time steps Δt_n are taken. For the central difference expressions of the velocity and the acceleration, it holds

$$\dot{\mathbf{u}}_n = \frac{\mathbf{u}_{n+1} - \mathbf{u}_{n-1}}{2\Delta t_n}$$

$$\ddot{\mathbf{u}}_n = \frac{\dot{\mathbf{u}}_{n+1/2} - \dot{\mathbf{u}}_{n-1/2}}{\Delta t_n} = \frac{\mathbf{u}_{n+1} - 2\mathbf{u}_n + \mathbf{u}_{n-1}}{(\Delta t_n)^2}.$$

By substituting these expressions into the dynamic equation, we obtain

$$\mathbf{M} \frac{\mathbf{u}_{n+1} - 2\mathbf{u}_n + \mathbf{u}_{n-1}}{(\Delta t_n)^2} + \mathbf{C} \frac{\mathbf{u}_{n+1} - \mathbf{u}_{n-1}}{2\Delta t_n} + \mathbf{K}\mathbf{u} - \mathbf{F}_{ext} = \mathbf{0}.$$

Because \mathbf{u}_n and \mathbf{u}_{n-1} are assumed to be known, we obtain

$$\left(\frac{\mathbf{M}}{(\Delta t_n)^2} + \frac{\mathbf{C}}{2\Delta t_n} \right) \mathbf{u}_{n+1} = \mathbf{F}_{ext} - \left(\mathbf{K} - \frac{2\mathbf{M}}{(\Delta t_n)^2} \right) \mathbf{u}_n - \left(\frac{\mathbf{M}}{(\Delta t_n)^2} - \frac{\mathbf{C}}{2\Delta t_n} \right) \mathbf{u}_{n-1}.$$

During the impact, the system of equations is solved at each discrete point in time. This method is called an explicit method because the solution \mathbf{u}_{n+1} is determined from the dynamic equation at the previous time steps which is in contrast to an implicit method where also the dynamic equation at the $(n+1)$ -st time step is used to calculate \mathbf{u}_{n+1} . Solvers which use an explicit time integration method are called explicit dynamic solvers.

6.1.4 Example 1

The optimization of a front vehicle crash design is considered by using the reduced model introduced in Subsection 6.1.2. To simulate this model, the explicit dynamic solver *Abaqus explicit* is applied.

Problem statement

The front car structure of the vehicle considered is modeled by sixteen components which are relevant for the crash design. The number of parameters (force levels) per component is four. In total, there are $d = 64$ parameters. The optimization problem under inequality constraints is given as follows:

$$\left. \begin{array}{l} \text{find } \mathbf{F}^{low}, \mathbf{F}^{up} \in \Omega_{DS} \text{ with } \mathbf{F}^{low} \leq \mathbf{F}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \text{ subject to } a_{pulse} = f(\mathbf{F}) \leq f_c \text{ for all } \mathbf{F} \in \Omega_{box}. \end{array} \right\} \quad (6.4)$$

Here, the function $f : \Omega_{DS} \rightarrow \mathbb{R}$ is a mapping which is provided by a numerical simulation of a front crash as described above.

Numerical results

To illustrate the function, two-dimensional cross sections of the design space are shown in Figure 6.2 where circles define good designs which satisfy the constraint $f(\mathbf{F}) \leq f_c$ and triangles define bad designs which violate this constraint. Here, the highly non-linear structure of the optimization problem (6.4) can be recognized.

The algorithm is applied to the optimization problem (6.4) with $N = 100$ sample points per iteration. The solution is depicted in Figure 6.3 via normalized intervals $[F_i^{low}, F_i^{up}]$ for each parameter F_i with $i = 1, 2, \dots, 64$.

In Figure 6.4, the calculated force-deformation intervals of the components of the front car structure are illustrated. A good and a bad design are also illustrated in Figure 6.4. The bad design lies outside the hyperbox. Note however that not all designs which are outside of the hyperbox have to be bad.

A sample with 100 randomly chosen force-deformation curves which are located inside of the intervals are depicted in Figure 6.5 for three of the 16 components of the front car structure. For all curves, the maximum crash pulse is subcritical. Hence, N_g/N is equal to 1 and, according to Theorem 3.3.2, the true fraction of the good space is between 0.97 and 1 with 95% probability.

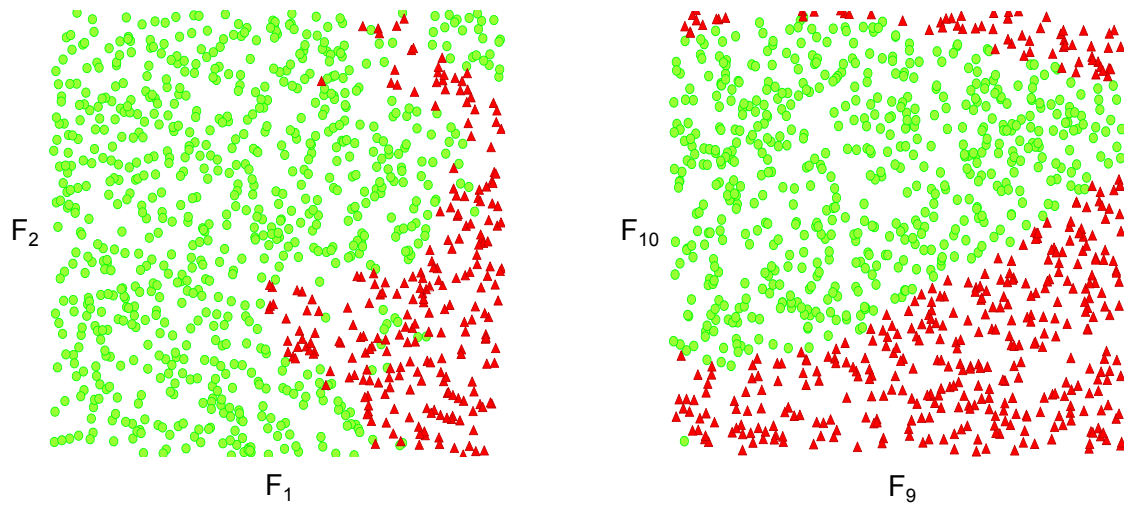


Figure 6.2: Problem 7. Two-dimensional cross sections of the design space. Shown are good (circles) and bad (triangles) design points.

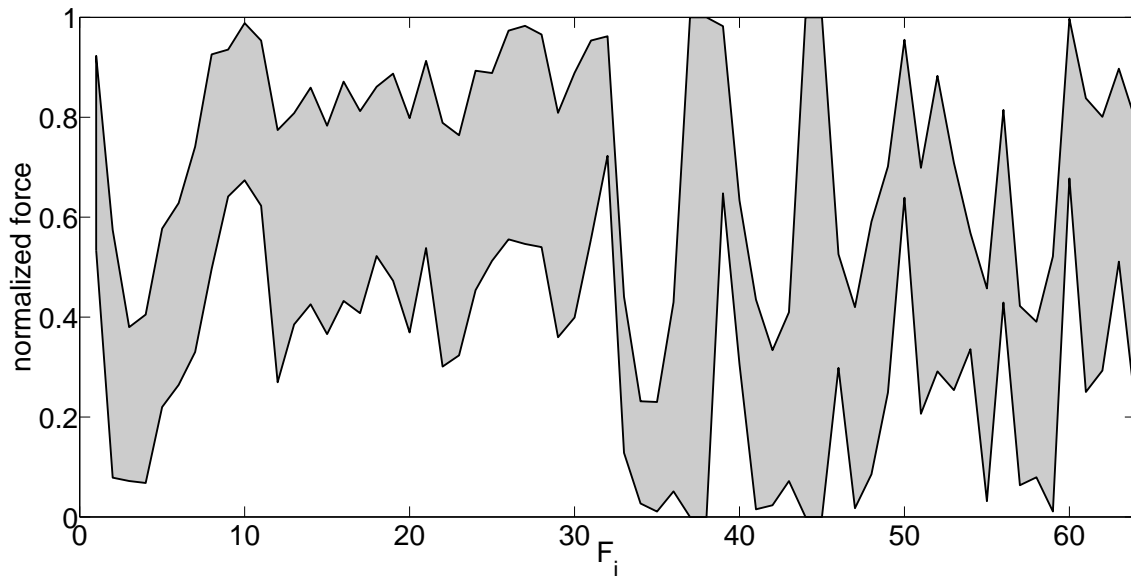


Figure 6.3: Problem 7. Normalized intervals for each parameter F_i .

The fraction of good sample points is depicted in Figure 6.6(a). During the exploration phase, the fraction of good sample points oscillates between 0.7 and 0.9. In the consolidation phase, this fraction tends towards 1. By reaching $N_g/N = 1$, the true fraction of the good space is between 0.97 and 1 with 95% probability.

In Figure 6.6(b), the normalized volume is depicted for different phases of the algorithm.

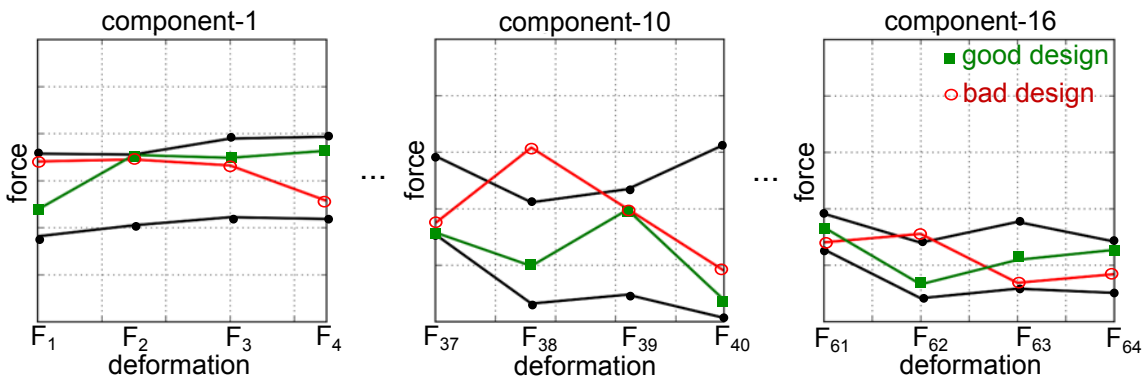


Figure 6.4: Problem 7. Force-deformation intervals for a vehicle structure. A bad and a good design are shown.

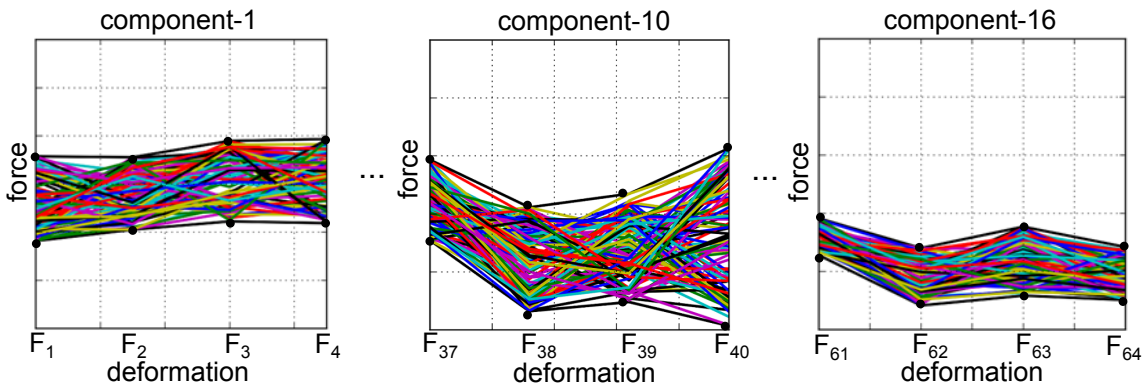


Figure 6.5: Problem 7. Force-deformation intervals in case of the front crash with $N = 100$ sample points.

The volume grows in the exploration phase (Phase 1) of the algorithm and decreases in the consolidation phase (Phase 2).

Figure 6.6(c) shows the normalized volume over the fraction of good sample points. In the exploration phase, the curve stagnates, and it converges in the consolidation phase. The algorithm converges to a hyperbox with a fraction of good space of 100% which is illustrated by normalized intervals in Figure 6.3. Hence, the algorithm is applicable to high-dimensional and non-linear engineering problems.

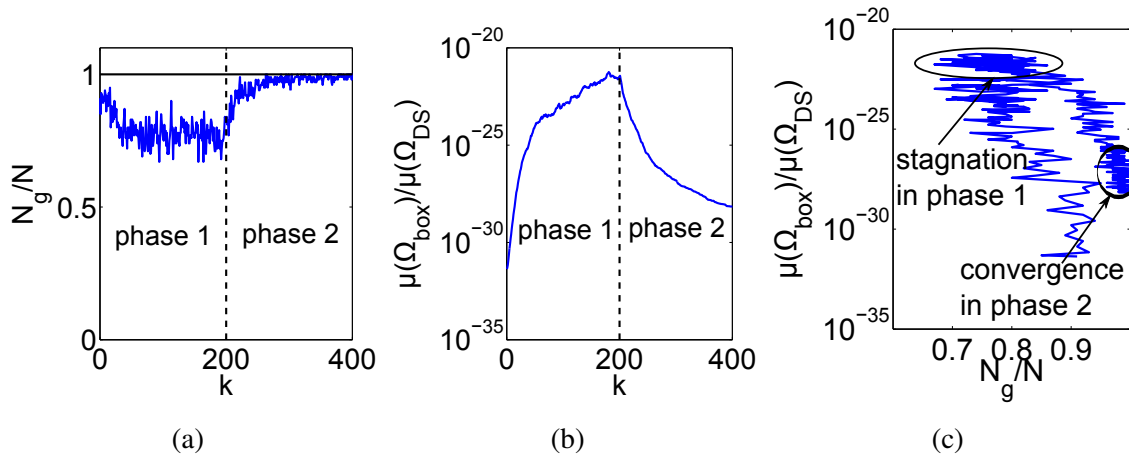


Figure 6.6: Problem 7. (a) The fraction of good sample points versus the number of iterations. (b) The hyperbox volume versus the number of iterations. (c) The hyperbox volume over the fraction of good sample points.

6.1.5 Example 2

To achieve economies of scale, as many parts as possible should fit in different kinds of cars. As a consequence, the main parts of the front structure have to be the same for different kinds of cars whereas each car has to fulfill its design goals. In the front crash, the maximum deceleration generated by the vehicle structure of each car has to fulfill the defined deceleration criterion. To calculate the maximum deceleration, reduced models are applied which are described in Subsection 6.1.2. To simulate these models, the explicit dynamic solver *Abaqus explicit* is used.

Problem statement

We consider eight vehicles which share some parts of the vehicle front structure as specified in Figure 6.7. Each column represents one car, and each row shows a part of the front structure. The black points indicate that a vehicle contains the corresponding part. In total, there are 15 parts. These parts consist of 1–3 components and each component is described by its force-deformation characteristics. In Table 6.1, an overview is given over the parts, the associated number of components and the number of the parameters which are the force levels of the force-deformation curves per part. If the number of parameters is summed up, the resulting total number of parameters is $d = 89$. The particular optimization problem

under inequality constraints is given as follows:

$$\left. \begin{array}{l} \text{find } \mathbf{F}^{low}, \mathbf{F}^{up} \in \Omega_{DS} \text{ with } \mathbf{F}^{low} \leq \mathbf{F}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \text{ subject to } f(\mathbf{F}) \leq f_c \text{ for all } \mathbf{F} \in \Omega_{box}. \end{array} \right\} \quad (6.5)$$

Here, it holds $f_c = 0$ and $f(\mathbf{F})$ is defined as

$$f(\mathbf{F}) := \max_{\ell} \left(\frac{a_{pulse}^{car_{\ell}} - f_c^{car_{\ell}}}{f_c^{car_{\ell}}} \right)$$

with

$$a_{pulse}^{car_{\ell}} = f(\mathbf{F}_{car_{\ell}}), \quad \ell = 1, 2, \dots, 8.$$

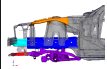
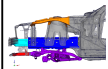
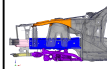
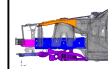

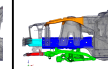


	car 1	car 2	car 3	car 4	car 5	car 6	car 7	car 8
								
carrying beam								
wheel house 1								●
wheel house 2	●	●	●	●	●	●	●	
front rail 1								●
front rail 2	●	●	●	●	●	●	●	
crush box 1								●
crush box 2	●	●				●	●	
crush box 3			●					
crush box 4				●	●			
front axle carrier 1								●
front axle carrier 2					●	●	●	
front axle carrier 3	●	●	●	●				
front axle carrier crush box 1								●
front axle carrier crush box 2	●	●				●	●	
front axle carrier crush box 3			●					
front axle carrier crush box 4				●	●			

Figure 6.7: Eight vehicles which share some parts of the vehicle front structure.

part name	number of components	number of nodes
carrying beam wheel house 1	2	7
carrying beam wheel house 2	2	7
front rail 1	2	7
front rail 2	2	7
crush box 1	1	4
crush box 2	1	4
crush box 3	1	4
crush box 4	1	4
front axle carrier 1	3	10
front axle carrier 2	3	9
front axle carrier 3	3	10
front axle carrier crush box 1	1	4
front axle carrier crush box 2	1	4
front axle carrier crush box 3	1	4
front axle carrier crush box 4	1	4

Table 6.1: The parts of the vehicle front structure with the corresponding number of components and the number of parameters per part.

Numerical results

In our numerical experiments, we shall compare different measures of choosing a hyperbox in the cutting algorithm. In particular, we will compare the results when using the hyperbox measures $\mu_1, \mu_2, \dots, \mu_5$ which are introduced in Section 3.4.1. The measures are selected by choosing Mode 1, Mode 2, \dots , Mode 5.

In Figures 6.8(a), 6.8(b) and 6.8(c), the results of the algorithm which is run with the standard mode (Mode 1) are illustrated. Then, the hyperbox with the most good sample points is chosen. Because the fraction of good sample points in a uniform sampling corresponds to the fraction of good space, a hyperbox with maximum volume will be obtained. In Figure 6.8(a), the fraction $\mu_1(\Omega_{box})$ of good sample points versus the number of iterations is illustrated. We observe that in the first phase, the fraction of good sample points oscillates between 0.6 and 0.8. In the second phase, the fraction of good sample points converges to 100%. Note that in the plots, the first and second phase of the algorithm are separated by a dashed line. Figure 6.8(b) shows the normalized volume versus the number of iterations. The volume increases in the exploration phase and converges in

the consolidation phase. In Figure 6.8(c), the normalized volume versus the fraction of good sample points is shown where a stagnation can be observed in the exploration phase. Whereas, convergence is observed in the consolidation phase.

Next, the algorithm is run with Mode 2 where the hyperbox with the largest minimal weighted width is selected. The results are displayed in Figures 6.8(d), 6.8(e) and 6.8(f). Maximizing the minimal weighted width results in a hyperbox where the smallest interval is as large as possible. In Figure 6.8(d), the fraction of good sample points versus the number of iterations is shown. The curve oscillates in the first phase and converges in the second phase. The measure $\mu_2(\Omega_{box})$ versus the number of iterations is plotted in Figure 6.8(e). This graph grows in the first phase and converges in the second phase. Figure 6.8(f) shows the measure $\mu_2(\Omega_{box})$ versus the fraction of good sample points. In particular, we observe stagnation in the exploration phase and convergence in the consolidation phase.

The maximum of the weighted function, this means an optimization in Mode 3, returns a hyperbox with the largest volume if $\lambda = 1$. If $\lambda = 0$, the hyperbox with the largest minimal weighted width is selected, see Section 3.4.1. Here, we choose $\lambda = 0.5$. The results of these calculations are depicted in Figure 6.8(g), 6.8(h) and 6.8(i). In Figure 6.8(g), the fraction of good sample points versus the number of iterations is illustrated. It oscillates in the first phase and converges in the second phase. The measure $\mu_3(\Omega_{box})$ versus the number of iterations is depicted in Figure 6.8(h). The graph grows in the first phase and converges in the second phase. The measure $\mu_3(\Omega_{box})$ versus the iterations is shown in Figure 6.8(i) where stagnation in the first phase and convergence in the second phase is observed.

Figures 6.9(a), 6.9(b) and 6.9(c) shows the results of the algorithm if it is executed in Mode 4. Now, the hyperbox with the largest minimal coupled weighted width is selected. Figure 6.9(a) shows that the fraction of good sample points oscillates in the exploration phase and converges in the consolidation phase. The measure $\mu_4(\Omega_{box})$ is growing in the first phase and converges in the second phase as seen in Figure 6.9(b). The stagnation in the exploration phase and the convergence in the consolidation phase is illustrated in Figure 6.9(c).

In Figures 6.9(d), 6.9(e) and 6.9(f), the results of the algorithm are plotted when Mode 5 is used. Then, the hyperbox with the maximum of the function $\mu_5(\Omega_{box})$ (cf. (3.14)) is selected. The fraction of good sample points versus the number of iterations is shown in Figure 6.9(d). It oscillates in the exploration phase and converges in the consolidation phase. Figure 6.9(e) illustrates that $\mu_5(\Omega_{box})$ grows in the first phase and converges in the second phase. The measure $\mu_5(\Omega_{box})$ versus the fraction of good sample points stagnates in the exploration phase and converges in the consolidation phase, see Figure 6.9(f).

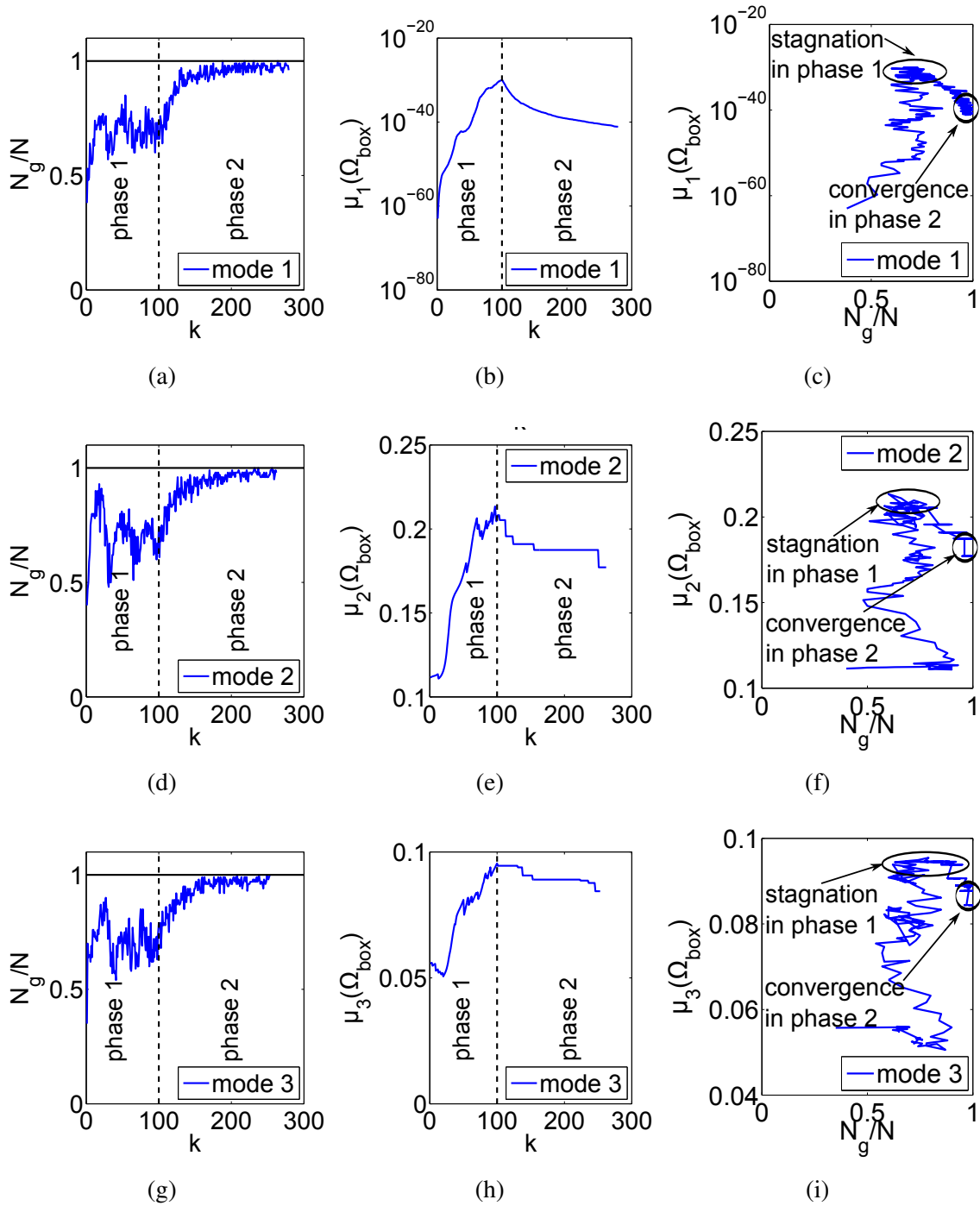


Figure 6.8: Fraction of good sample points versus iteration steps (a) Mode 1, (d) Mode 2 and (g) Mode 3. Hyperbox measure versus iteration steps (b) Mode 1, (e) Mode 2 and (h) Mode 3. Hyperbox measure versus fraction of good sample points (c) Mode 1, (f) Mode 2 and (i) Mode 3.

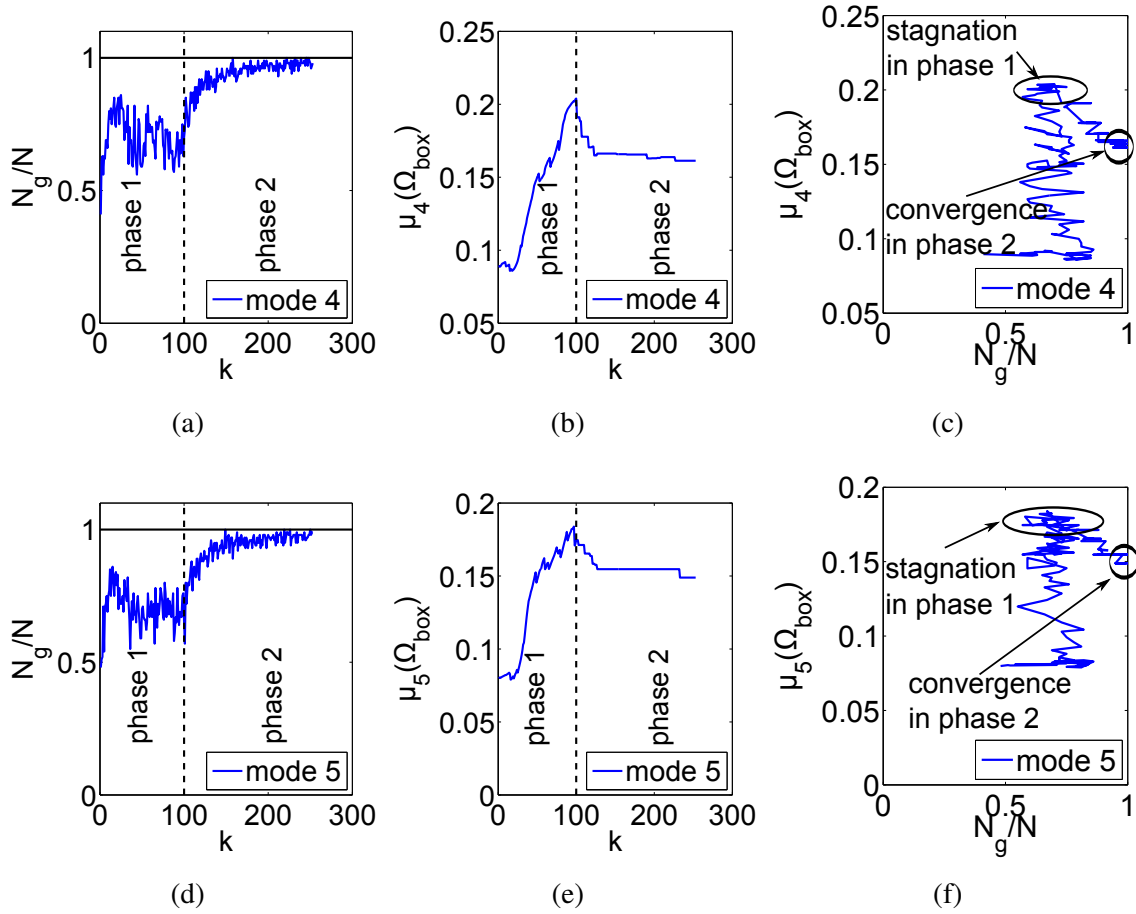


Figure 6.9: Fraction of good sample points versus iteration steps (a) Mode 4 and (d) Mode 5. Hyperbox measure versus iteration steps (b) Mode 4 and (e) Mode 5. Hyperbox measure versus fraction of good sample points (c) Mode 4 and (f) Mode 5.

For all the modes, essentially the same convergence behavior is observed. In the exploration phase, the fraction of good sample points oscillates between 0.6 and 0.8 while the selected hyperbox measure grows. In the consolidation phase, the fraction of good sample points and the selected measure converge.

We obtain the hyperbox with the largest volume by choosing Mode 1. If Mode 2 is chosen, the resulting hyperbox is a hyperbox with maximum smallest interval width at the expense of the hyperbox volume. In Mode 3, a compromise between the hyperbox measures μ_1 and μ_2 is found. Mode 4 and Mode 5 consider that the input parameters are support points of a curve. In Mode 4, the result is a hyperbox with maximum smallest interval width. Each interval width is normalized by the largest mean of the intervals within the component it

belongs to. In contrast to Mode 4, a hyperbox with maximum smallest corridor width is obtained by choosing Mode 5. The widths are normalized by the maximum upper boundary of each component. Therefore, by choosing Mode 5, large gradients of the lower and upper boundaries, respectively, are avoided within a component.

If the convergence coefficient is determined for the different modes, the values tabulated in Table 6.2 are obtained. The convergence coefficient is between 0.0002 and 0.0003. The order of magnitude of these values corresponds to the values which we observed in Chapter 5. There, Figure 5.6(c) shows that, for $N = 100$ sample points per iteration and $d = 100$ dimensions, the convergence coefficient is 0.0003. In the same figure, we observe that the convergence coefficient decreases when the dimension increases. In the present example, the number of dimensions is 89 and the convergence coefficient is also between 0.0002 and 0.0003. Consequently, the values are in good agreement with these from Chapter 5.

	Mode 1	Mode 2	Mode 3	Mode 4	Mode 5
c/N	0.00027	0.00026	0.00024	0.00022	0.00024
\bar{a}_0	0.72	0.73	0.77	0.80	0.78

Table 6.2: Convergence coefficients for the different modes.

Next, in accordance with Section 3.4.2, the sensitivity of the boundaries of the solution hyperbox are determined for the example under consideration. The fraction of good space of the probed space in the dimension i is calculated by

$$[a^*]_i^{low} = \left[\frac{N_g}{N} \right]_{i, \Omega_{box}^{prob}}^{low} \quad \text{and} \quad [a^*]_i^{up} = \left[\frac{N_g}{N} \right]_{i, \Omega_{box}^{prob}}^{up}$$

for the lower and upper boundary, respectively. The results of these computations are signified in Figure 6.10 by differently colored points. If $[a^*]_i$ is larger than 0.95, the point is green. This means that only a few bad sample points are gained by enlarging the hyperbox in the dimension i . The point is yellow if $[a^*]_i$ is between 0.7 and 0.95. If $[a^*]_i$ is smaller than 0.7, the point is red. For example, the lower boundary of the component 14 at the first support point (associated with F_{49}) is only a bit sensitive because the point is green. The upper boundary, however, is very sensitive because the point is red.

The optimization problem (6.5) under inequality constraints with $d = 89$ disintegrates in two subproblems because these subproblems are independent of each other. This can be observed in Figure 6.7. The car in the last column – car 8 – does not share any part

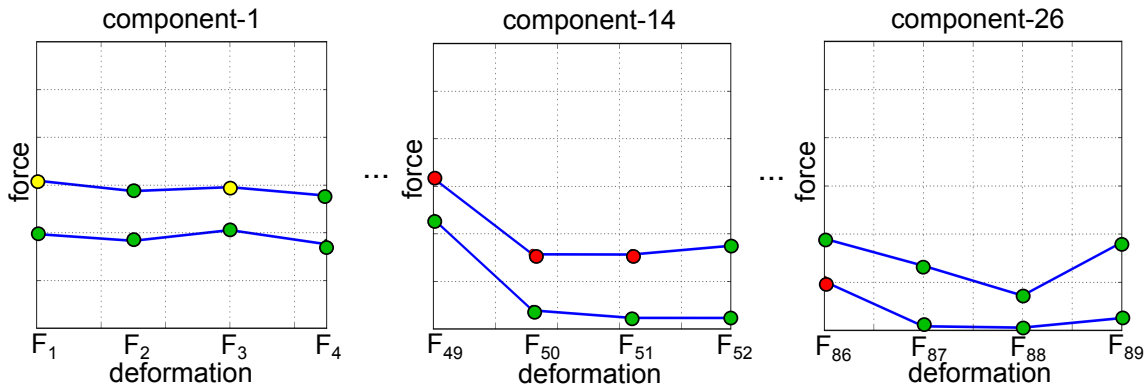


Figure 6.10: Sensitivities of the boundaries of the solution hyperbox indicated by colored points.

with another car. Therefore, it can be considered independently of the other cars, and the problem can be subdivided in two subproblems. The first subproblem (Subproblem A) has $d = 57$ dimensions (see Table 6.3), and the second subproblem (Subproblem B) has $d = 32$ dimensions (see Table 6.4).

By solving Subproblem A independently of Subproblem B, we obtain the red and green corridors shown in Figure 6.11. The convergence coefficient of Subproblem A and Subproblem B are tabulated in Table 6.5.

part name	number of components	number of nodes
carrying beam wheel house 2	2	7
front rail 2	2	7
crush box 2	1	4
crush box 3	1	4
crush box 4	1	4
front axle carrier 2	3	9
front axle carrier 3	3	10
front axle carrier crush box 2	1	4
front axle carrier crush box 3	1	4
front axle carrier crush box 4	1	4

Table 6.3: The parts of the vehicle front structure with the corresponding number of components and the number of nodes per part for the Subproblem A.

If we solve the optimization problem under inequality constraints (6.5) with $d = 89$,

part name	number of components	number of nodes
carrying beam wheel house 1	2	7
front rail 1	2	7
crush box 1	1	4
front axle carrier 1	3	10
front axle carrier crush box 1	1	4

Table 6.4: The parts of the vehicle front structure with the corresponding number of components and the number of nodes per part for the Subproblem B.

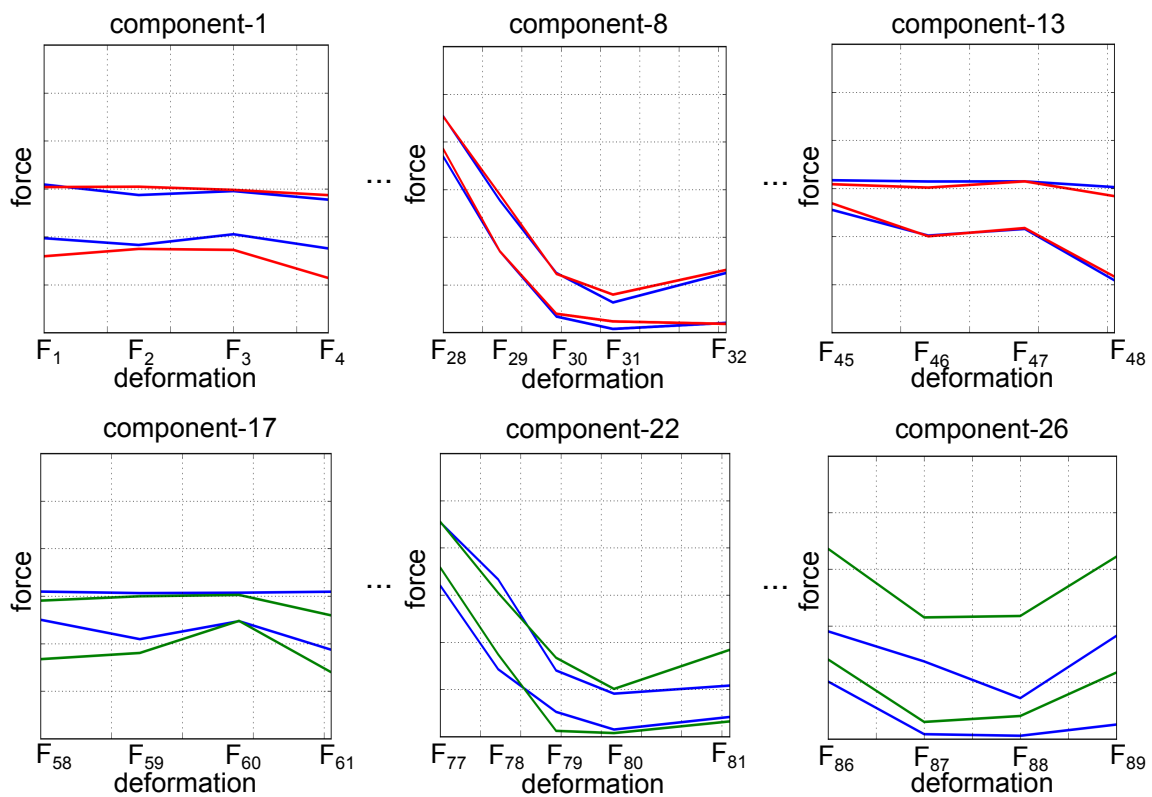


Figure 6.11: The solution corridors of the optimization problem with $d = 89$ dimensions, and the solution corridors of the optimization problem subdivided in two subproblems because these two subproblems are independent of each other.

the blue corridors shown in Figure 6.11 are obtained. The corresponding convergence coefficient is found in Table 6.5. Obviously, the convergence coefficient decreases when the dimension increases. This behavior confirms again the observations from Chapter 5. Moreover, the order of magnitude of the different dimensions corresponds to the values

	Full problem	Subproblem A	Subproblem B
c/N	0.00027	0.00042	0.00084
\tilde{a}_0	0.72	0.77	0.68

Table 6.5: The convergence coefficient of the problem with $d = 89$, and the convergence coefficients of the independent problems with $d = 57$ and $d = 32$, respectively.

obtained in Chapter 5. Finally, Figure 6.11 illustrates that the results for the two independent subproblems and the results for the full problem agree reasonably well.

6.2 Forming process

In Subsection 6.2.1, the determination of a hyperbox for a forming process is motivated. The evaluation of a stamped part is shown in Subsection 6.2.2. In Subsection 6.2.3, it is explained how to obtain a response surface model of the forming process. Finally, a specific example problem is presented in Subsection 6.2.4.

6.2.1 Motivation

The sheet metal forming for car body parts is influenced by variations of the process and the material. Variations of the input parameters can result in significant variations of the part's quality. This can lead to additional costs to revise the parts or even to higher production costs due to rejected parts, see [26, 81, 82]. Therefore, a robust forming process is required. We simulate the forming process to assess its robustness. Scattering input parameters of forming simulations are, for example, the bead forces and the tool binder force.

The aim is to choose, for example, the bead forces and the tool binder force such that problems of producibility like cracks and insufficient hardening are avoided. Because the bead forces and the tool binder force underlie variations in the full-scale forming process, intervals for the force levels are considered to assess robustness of the forming process and to improve the product characteristics.

6.2.2 Evaluation

The forming simulation is performed by the finite element method with explicit time integration as introduced in Section 6.1.3. The feasibility of a stamped part is evaluated by

the risk of cracks and sufficient hardening. The *risk of cracks* is evaluated by a cracking value as defined in [26]. The cracking value is defined as the major strain of the strain state under consideration normalized by the forming limit curve. A forming limit curve (FLC) is depicted in Figure 6.12. It is obtained by measurements of the particular material. In the same figure, a safety margin which is 80% of the FLC is shown. The space below the white line indicates the undefined region of the diagram. The region of wrinkles is constrained from above by the line $\phi_1 = -\phi_2$. Here, the wrinkle tendency is located below the line $\phi_1 = -2\phi_2$. Severe thinning exists for $\phi_3 > 0.3$. Finally, an inadequate stretch is given for $\phi_3 < 0.02$. Note that $\phi_1 + \phi_2 = -\phi_3$. In the green area, all elements of an ideal stamped part are located.

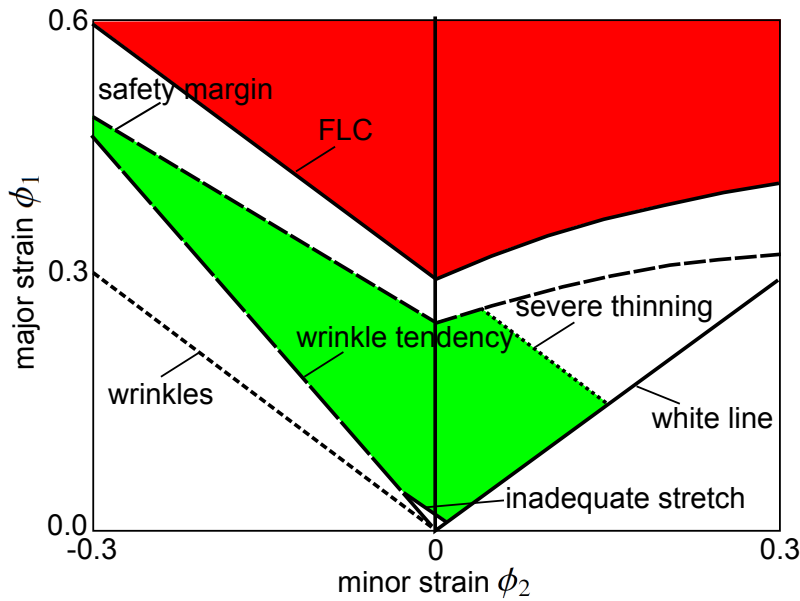


Figure 6.12: Forming limit diagram.

If the FLC is described as a function $FLC(\phi_2)$, the cracking value is given by normalizing the present strain by the FLC

$$cracking_{\ell} = \frac{\phi_1}{FLC(\phi_2)}$$

for every element ℓ of the stamped part.

To ensure a *sufficient hardening*, a fraction of hardening is defined in accordance with [81]. Namely, the fraction of hardening of a stamped part is determined by dividing the number of elements where the thickness reduction (thinning) is greater than 2% by the total number of elements

$$hardening_{frac} = \frac{\# \text{ elements } > 2\% \text{ thinning}}{\# \text{ elements}}.$$

6.2.3 Response surface model

Instead of using a stamping simulation for the evaluation of a stamped part, we will use a response surface model. A response surface model is an approximation of the simulation model. The problem to obtain a response surface model can be formulated as follows (see [5, 64]):

The design parameters $\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_N \in \mathbb{R}^d$ and the function values $y_1, y_2, \dots, y_N \in \mathbb{R}$ are given. We seek a function $s : \mathbb{R}^d \rightarrow \mathbb{R}$ such that

$$s(\mathbf{x}_j) = y_j \quad \text{for all } j = 1, 2, \dots, N. \quad (6.6)$$

Definition 6.2.1 (Radial basis function). *A function $\gamma : \mathbb{R}^d \rightarrow \mathbb{R}$ is called radial basis function (RBF), if its value depends only on the distance to the origin, so that*

$$\gamma(x) = \gamma(\|\mathbf{x}\|_2).$$

The Table 6.6 contains some specific radial basis functions $\gamma(r)$.

Name of RBF	$\gamma(r), r \geq 0$	Smoothness
multiquadric	$\sqrt{1 + (\varepsilon r)^2}$	infinitely smooth
inverse multiquadric	$\frac{1}{\sqrt{1 + (\varepsilon r)^2}}$	infinitely smooth
inverse quadric	$\frac{1}{1 + (\varepsilon r)^2}$	infinitely smooth
generalized multiquadric	$(1 + (\varepsilon r)^2)^\beta$	infinitely smooth
gaussian	$e^{-(\varepsilon r)^2}$	infinitely smooth
thin plate spline	$r^2 \log(r)$	piecewise smooth
linear	r	piecewise smooth
cubic	r^3	piecewise smooth
monomial	r^{2k-1}	piecewise smooth

Table 6.6: Examples of radial basis functions [64].

To solve the interpolation problem (6.6), we make the ansatz

$$s(\mathbf{x}) := \sum_{j=1}^N \lambda_j \gamma(\|\mathbf{x} - \mathbf{x}_j\|_2).$$

Here, a radial basis function is centered in each design parameter \mathbf{x}_j . The unknown coefficients $\lambda_j \in \mathbb{R}$ are now determined by

$$s(\mathbf{x}_k) = y_k \Leftrightarrow \sum_{j=1}^N \lambda_j \gamma(\|\mathbf{x}_k - \mathbf{x}_j\|_2) = y_k \quad \text{for all } k = 1, 2, \dots, N.$$

This can be expressed as a linear system of equations $\Gamma\lambda = y$:

$$\begin{bmatrix} \gamma(\|\mathbf{x}_1 - \mathbf{x}_1\|_2) & \gamma(\|\mathbf{x}_1 - \mathbf{x}_2\|_2) & \dots & \gamma(\|\mathbf{x}_1 - \mathbf{x}_N\|_2) \\ \gamma(\|\mathbf{x}_2 - \mathbf{x}_1\|_2) & \gamma(\|\mathbf{x}_2 - \mathbf{x}_2\|_2) & \dots & \gamma(\|\mathbf{x}_2 - \mathbf{x}_N\|_2) \\ \vdots & \vdots & \vdots & \vdots \\ \gamma(\|\mathbf{x}_N - \mathbf{x}_1\|_2) & \gamma(\|\mathbf{x}_N - \mathbf{x}_2\|_2) & \dots & \gamma(\|\mathbf{x}_N - \mathbf{x}_N\|_2) \end{bmatrix} \begin{bmatrix} \lambda_1 \\ \lambda_2 \\ \vdots \\ \lambda_N \end{bmatrix} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}.$$

The matrix Γ is symmetric. To ensure uniqueness of the solution, Γ has to be nonsingular.

Definition 6.2.2. *The function $\gamma : [0, \infty) \rightarrow \infty$ is completely monotonic on $(0, \infty)$, if γ is arbitrarily often differentiable and if $(-1)^\ell \gamma^{(\ell)}(r) \geq 0$ for all $r \in (0, \infty)$ and $\ell \geq 0$.*

In [53], it is proven that the matrix Γ is positive definite and therefore invertible if $\mathbf{x}_j \neq \mathbf{x}_k$ for all $j \neq k$ and if γ is completely monotonic on $(0, \infty)$.

6.2.4 Example

The optimization of a forming process of a wind draw is considered. The forming simulation uses the finite element method as introduced in Section 6.1.3. Here, the explicit dynamic solver *LS-DYNA* is applied.

Problem statement

The input parameters are ten bead forces and one tool binder force. They are described by the vector $\mathbf{F} = (F_1, F_2, \dots, F_{11})$. The ten beads of the wind draw are shown in Figure 6.13. A first constraint is that the maximum cracking value $\max_\ell \text{cracking}_{\ell, \mathbf{F}}$ is smaller than a critical value f_{c1} for a stamped part simulated with the forces \mathbf{F} . Another constraint consists of the fraction of hardening $\text{hardening}_{\text{frac}, \mathbf{F}}$ which has to be larger than a critical value f_{c2} for a stamped part simulated with the forces \mathbf{F} .

To identify a hyperbox for the forces \mathbf{F} , the following constrained optimization problem will be solved by using the presented algorithm:

$$\left. \begin{array}{l} \text{find } \mathbf{F}^{\text{low}}, \mathbf{F}^{\text{up}} \in \Omega_{DS} \text{ with } \mathbf{F}^{\text{low}} \leq \mathbf{F}^{\text{up}} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{\text{box}}) \rightarrow \max \\ \text{subject to } \max \left(\frac{\max_\ell \text{cracking}_{\ell, \mathbf{F}} - f_{c1}}{f_{c1}}, \frac{f_{c2} - \text{hardening}_{\text{frac}, \mathbf{F}}}{f_{c2}} \right) \leq 0 \text{ for all } \mathbf{F} \in \Omega_{\text{box}}. \end{array} \right\} \quad (6.7)$$

Every stamped part in the hyperbox has no cracks and provides a sufficient hardening. Moreover, the design space Ω_{DS} is of dimension 11, this means, $\mathbf{F}^{\text{low}} = (F_1^{\text{low}}, F_2^{\text{low}}, \dots, F_{11}^{\text{low}})$ and $\mathbf{F}^{\text{up}} = (F_1^{\text{up}}, F_2^{\text{up}}, \dots, F_{11}^{\text{up}})$. The maximum cracking value and the fraction of hardening are determined after performing a forming simulation with the forces \mathbf{F} .

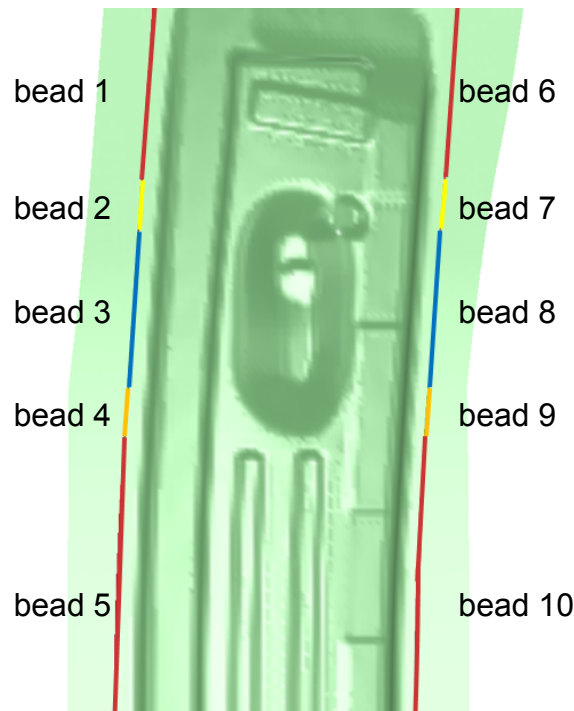


Figure 6.13: A stamping simulation of a wind draw with ten beads.

Numerical results

In Figure 6.14, different stamping simulations of wind draws and the corresponding elements in the forming limit curve are shown. Figure 6.14(a) shows a stamping simulation of a wind draw with cracks. In the form limit diagram, some elements are in the region of cracks. Consequently, the chosen bead forces and the tool binder force are not appropriate. A stamping simulation of a wind draw without cracks but with insufficient hardening is depicted in Figure 6.14(b). The gray surface indicates that most of the elements have less than 2% thinning. A stamping simulation of a wind draw without cracks and with sufficient hardening is illustrated in Figure 6.14(c).

Because a forming simulation of the wind draw needs about half an hour to be calculated, two response surface models are constructed as described in Section 6.2.3. To generate these models, 1000 forming simulations are executed. On the basis of these simulations, a response surface model for the maximum cracking value and a response surface model for the fraction of hardening are generated by a cubic radial basis function interpolant. The coefficient of determination indicates how much the variance of one variable is determined by the variance of another variable.

Definition 6.2.3 (Coefficient of determination). *Let X and Y be two random variables.*

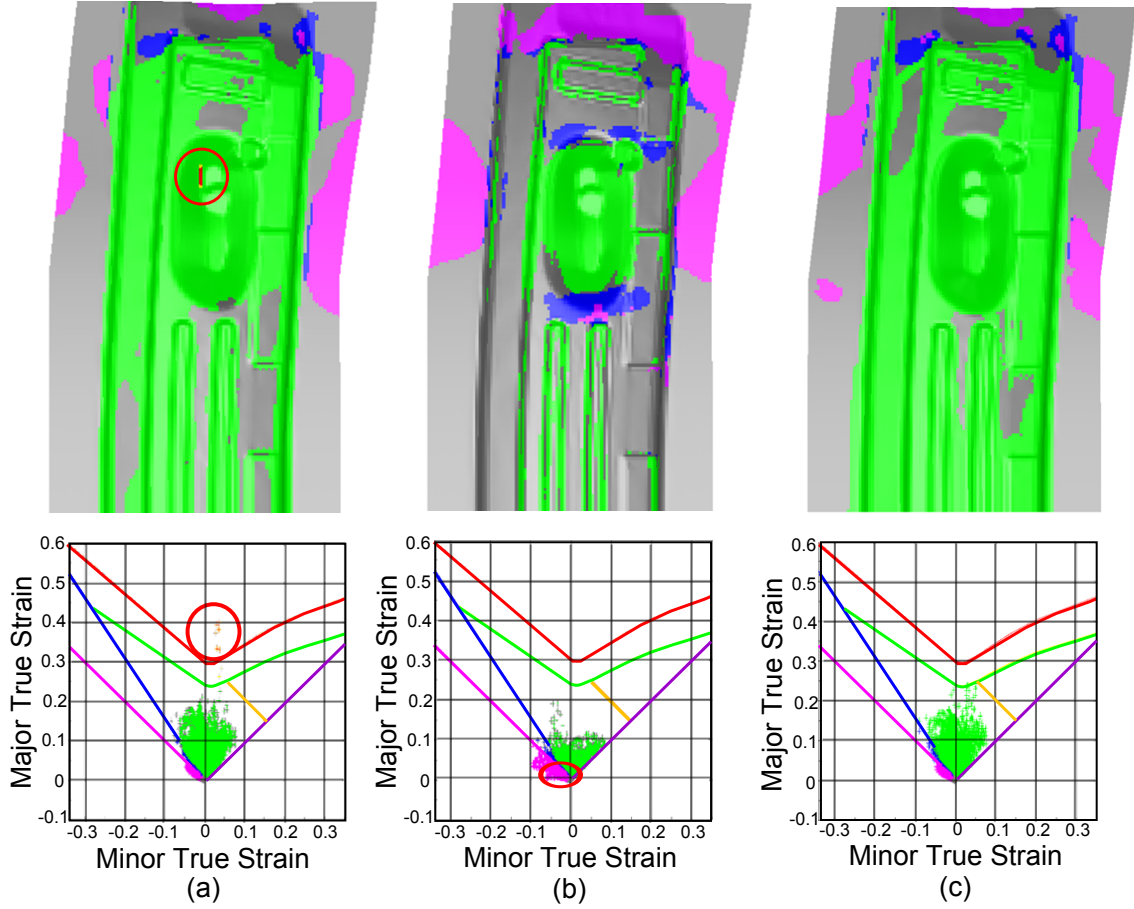


Figure 6.14: (a) A stamping simulation of a wind draw with cracks and the form limit diagram with the elements of the stamped part. (b) A stamping simulation of a wind draw with insufficient hardening and the form limit diagram with the elements of the stamped part. (c) A stamping simulation of a wind draw without cracks and with sufficient hardening and the form limit diagram with the elements of the stamped part.

Then, the coefficient of determination is defined by

$$R_{press}^2 = \frac{Cov(X, Y)}{Var(X)Var(Y)} = \frac{\mathbb{E}((X - \mathbb{E}(X))(Y - \mathbb{E}(Y)))}{\mathbb{E}((X - \mathbb{E}(X))^2)\mathbb{E}((Y - \mathbb{E}(Y))^2)}$$

where $Cov(X, Y)$ denotes the covariance of X and Y and $Var(X)$ and $Var(Y)$ denote the variance of X and Y , respectively.

Let $\{y_j\}$ be a given sample. We solve the integral of the coefficient of determination by a Monte Carlo method and approximate the coefficient of determination R_{press}^2 by

$$\tilde{R}_{press}^2 = 1 - \frac{\sum_{j=1}^{\bar{N}} (y_j - \hat{y}_j)^2}{\sum_{j=1}^{\bar{N}} (y_j - \bar{y})^2} = 1 - \frac{\text{sum of squares of residuals}}{\text{total sum of squares}} \quad (6.8)$$

with \tilde{N} being the number of retests, \hat{y}_j being the predicted values, and $\bar{y} = \frac{1}{\tilde{N}} \sum_{j=1}^{\tilde{N}} y_j$ being the mean of the y_j .

By executing 100 forming simulations which gives the sample $\{y_j\}$, the coefficient of determination of the response surface model for the fraction of hardening is approximated and has the value $\tilde{R}_{press}^2 = 0.97$. The coefficient of determination of the response surface model for the cracking value is approximated and has the value $\tilde{R}_{press}^2 = 0.83$. The approximated coefficients of determination obtained here thus are reasonable well.

With these preparations at hand, intervals for the ten bead forces and the tool binder force are calculated. In doing so, the values $\max_i cracking_{i,F}$ and $hardening_{frac,F}$ are calculated by using the response surface models. The intervals shown in Figure 6.15 are the solution of the optimization problem (6.7) computed by the presented algorithm using the standard mode (Mode 1).

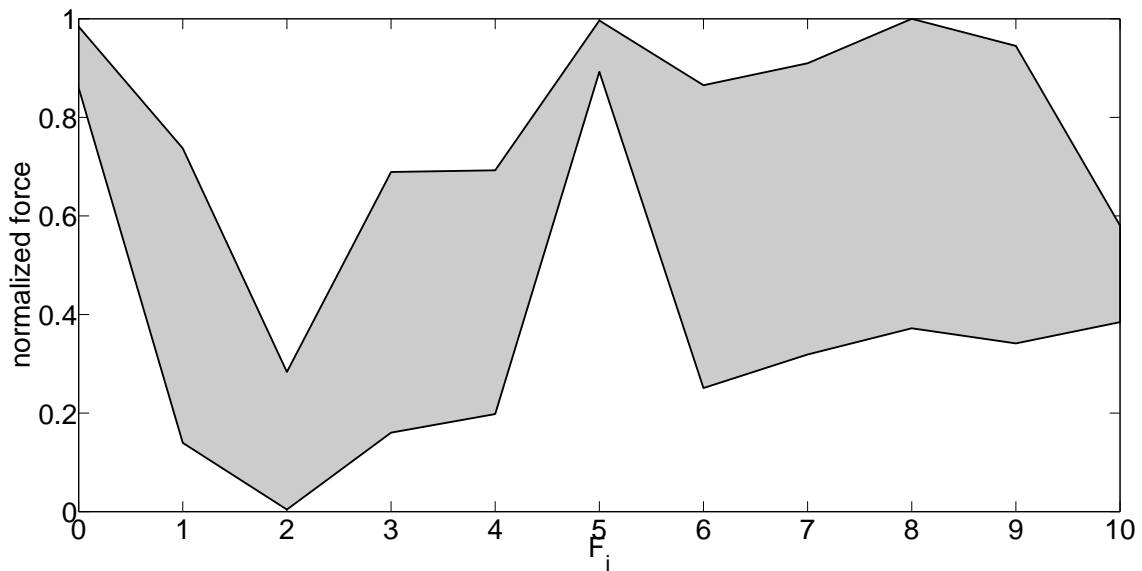


Figure 6.15: Forming process: Normalized intervals for each parameter F_i .

Within the intervals, every combination of forces produces with 95% probability a stamped wind draw with sufficient hardening and without cracks under the assumption that the response surface models predict the same bad space as the simulation model. The bad space consists of stamped parts where $\max_{\ell} cracking_{\ell,F} > f_{c_1}$ and $hardening_{frac,F} < f_{c_2}$. Thus, our assumption means that the bad space obtained from the simulated forming process coincides with the bad space obtained from the response surface models. The results of the iteration process are visualized in Figure 6.16. Figure 6.16(a) illustrates the fraction of

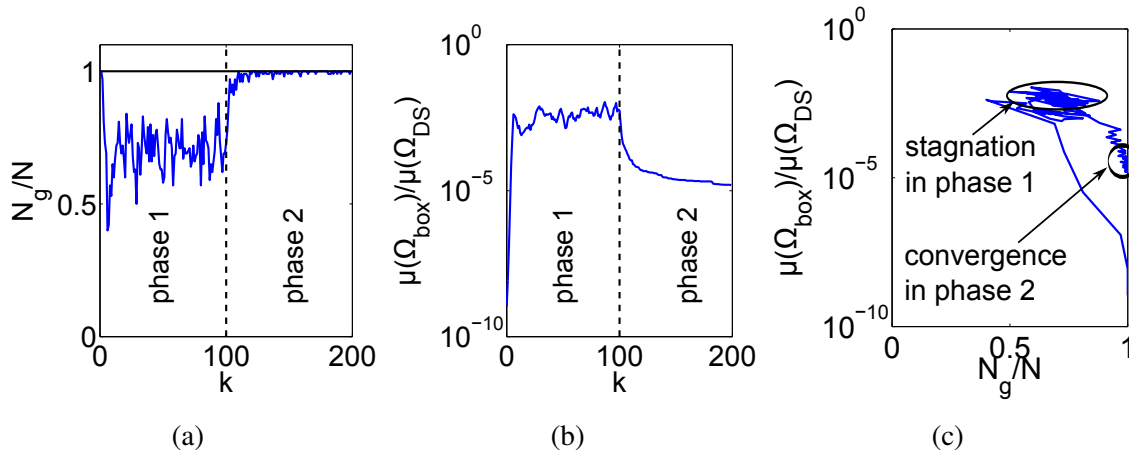


Figure 6.16: Forming process: (a) The fraction of good sample points versus the number of iterations. (b) The hyperbox volume versus the number of iterations. (c) The hyperbox volume versus the fraction of good sample points.

good sample points versus the number of iterations. In the exploration phase, the curve oscillates and, in the consolidation phase, convergence is observed. We obtain for the convergence coefficient $c/N = 0.003$ with $\tilde{a}_0 = 0.74$. Here, the number of dimensions is $d = 11$. Comparing the convergence coefficient with that in Chapter 5 for $d = 10$, we conclude that the order of magnitude agrees. The normalized hyperbox volume versus the number of iterations is shown in Figure 6.16(b). The volume increases in the first phase and converges in the second phase. In Figure 6.16(c), the hyperbox volume versus the fraction of good sample points is depicted. We observe stagnation in the first phase and convergence in the second phase.

6.3 Rear passenger safety

In the Subsection 6.3.1, the motivation for an optimization problem for the rear passenger safety is given. The evaluation of submarining is considered in Subsection 6.3.2. Finally, a specific problem is presented in Subsection 6.3.3.

6.3.1 Motivation

A main task of the rear passenger safety is to avoid submarining in a front crash. For an illustration of submarining, see Figure 6.17. Submarining occurs if the lap belt slips off the

iliac crest and cuts into the soft abdomen region. It can result in serious internal injuries. The key parameters to avoid submarining are the position of the belt anchor and the belt lock of the seat belt. The belt anchor and the belt lock are illustrated in Figure 6.18.

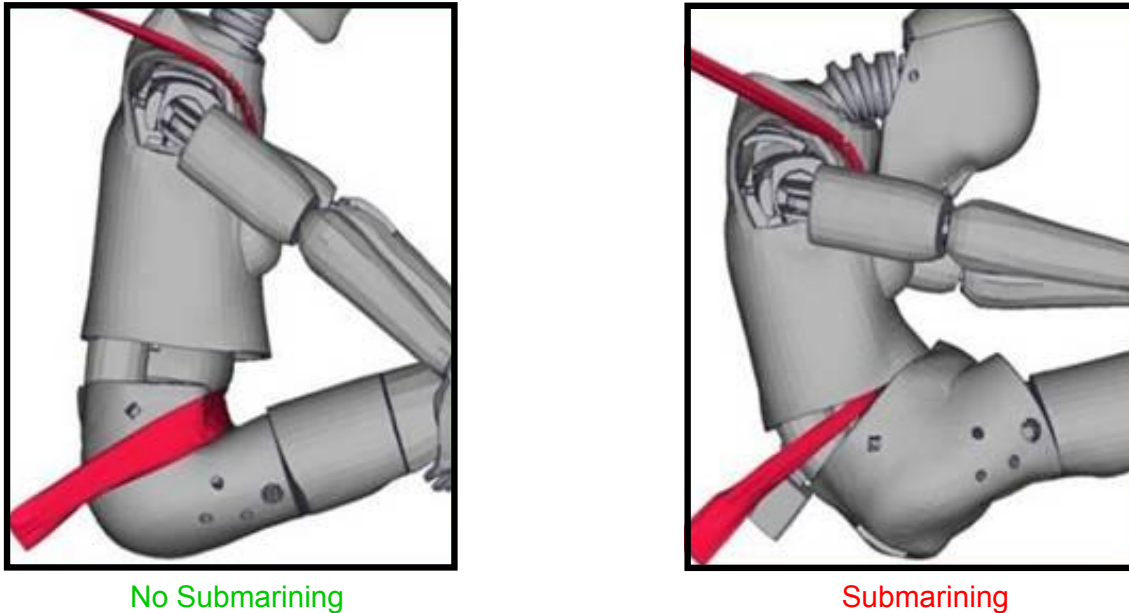


Figure 6.17: Submarining and no submarining.

In the traditional car development process, one design team is responsible for the rear passenger safety while another design team has to consider the package. In the automobile industry, a package means that all different components in a car fit together. To avoid many coordination meetings, the design team which is responsible for the rear passenger safety may define solution spaces where submarining does not occur. To decouple the parameters from each other, a hyperbox is thus sought for the relevant parameters.

6.3.2 Evaluation

To evaluate submarining, the front crash is simulated as presented in Section 6.1. Then, the distance of the vertebral column to the abdominal wall in the midplane of the dummy is measured as seen in Figure 6.19.

If the minimal distance goes below a critical value during the simulation, submarining occurs. In Figure 6.20, the distance versus the crash time is illustrated. The critical value is indicated by a dashed line. The red line shows a simulation where submarining arises.

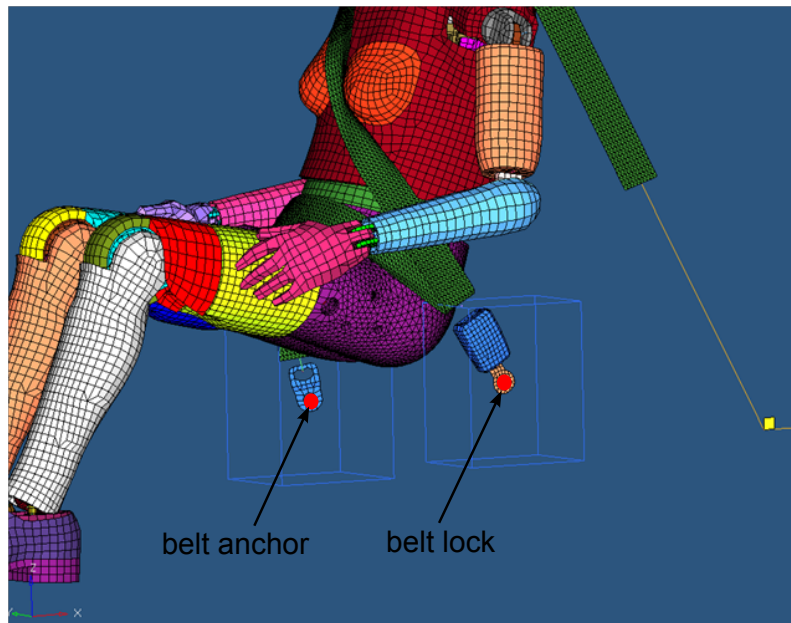


Figure 6.18: The belt anchor and the belt lock of the seat belt.

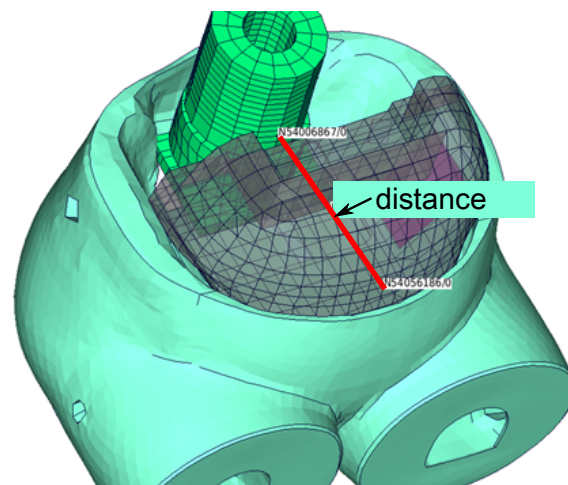


Figure 6.19: The distance of the vertebral column to the abdominal wall in the midplane of the dummy.

The green line corresponds to a simulation without submarining. The critical value is empirically chosen.

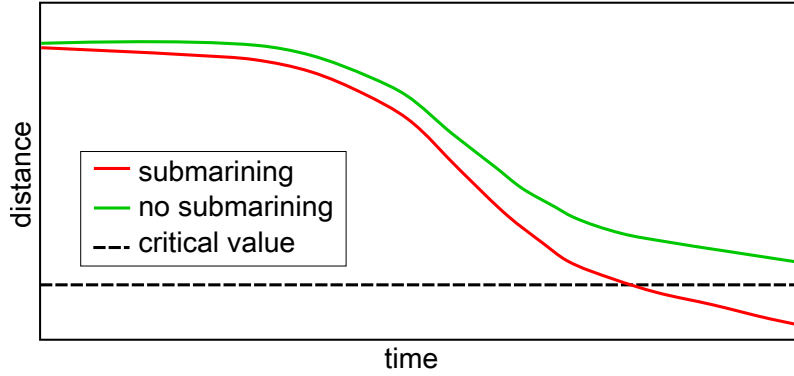


Figure 6.20: Submarining occurs if the minimal distance becomes smaller than a critical value in the simulation.

6.3.3 Example

A simulation of a front crash with a fifth female dummy is considered. A fifth female dummy represents the smallest segment of the adult population. The simulation is performed by using a finite element method as introduced in Section 6.1.3. Here, we use the explicit dynamic solver *Abaqus explicit*.

Problem statement

The input parameters are the position of the belt anchor and the belt lock of the seat belt. Consequently, there are six input parameters. They are described by the vector $\mathbf{b} = (a_x, a_y, a_z, \ell_x, \ell_y, \ell_z)$. The parameters a_x, a_y and a_z are the x -, y - and z -positions of the belt anchor and the parameters ℓ_x, ℓ_y and ℓ_z are the x -, y - and z -positions of the belt lock. The constraint is that the minimal distance $\min_t d(t)$ is above a critical value during the simulation which means

$$\min_t d(t) \geq d_c,$$

cf. Figure 6.20.

In all, we arrive at the following optimization problem for the positions \mathbf{b} :

$$\left. \begin{array}{l} \text{find } \mathbf{b}^{low}, \mathbf{b}^{up} \in \Omega_{DS} \text{ with } \mathbf{b}^{low} \leq \mathbf{b}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \\ \text{subject to } \min_t d(t) \geq d_c \text{ for all } \mathbf{b} \in \Omega_{box}. \end{array} \right\} \quad (6.9)$$

Every position of the belt anchor and the belt lock of the seat belt within the resulting hyperbox avoids the submarining. The design space Ω_{DS} is of dimension six, this means, $\mathbf{b}^{low} = (b_1^{low}, b_2^{low}, \dots, b_6^{low})$ and $\mathbf{b}^{up} = (b_1^{up}, b_2^{up}, \dots, b_6^{up})$.

Numerical results

Because a simulation needs about five hours to be calculated, a response surface model is constructed to obtain a faster calculation of the output value. To generate this model, a Monte Carlo sampling with 200 simulations is executed in the design space, see Figure 6.21. On the basis of these simulations, a response surface model for the minimal distance is generated by using a cubic radial basis function interpolant in accordance with Section 6.2.3. The solution is a response surface model for the minimal distance. A retest is executed by a Monte Carlo sampling with 30 simulations to evaluate the predictive quality of the response surface model, see Figure 6.22. For our particular response surface model, the approximated coefficient of determination is $\tilde{R}_{press}^2 = 0.87$. The approximated coefficient of determination is given in equation (6.8).

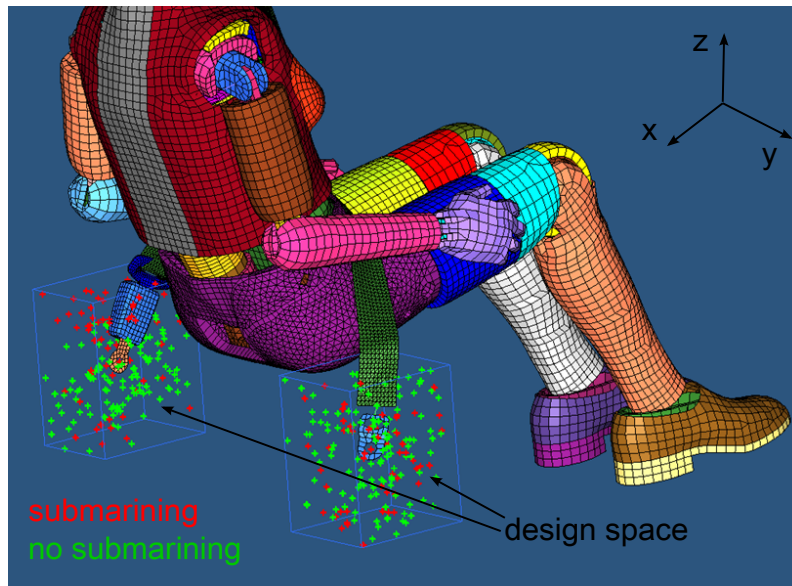


Figure 6.21: 200 sample points within the design space.

We apply the algorithm (Mode 1) to the optimization problem (6.9) by using the produced response surface model to calculate the output value $\min_t d(t)$. The solution of the optimization problem (6.9) are intervals for the six input parameter. The solution hyperboxes are shown in Figure 6.24. Within the hyperboxes, every position is with 95% probability a

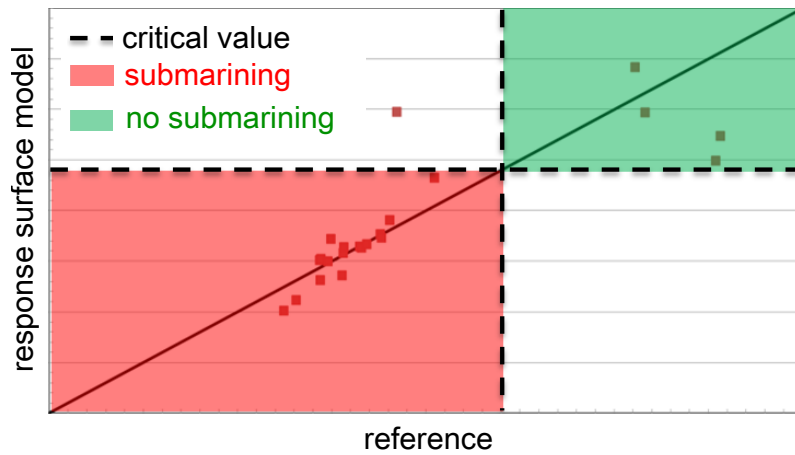


Figure 6.22: Retest with 30 sample points to evaluate the predictive quality of the response surface model.

good design under the assumption that the response surface models predict the same bad space as the simulation.

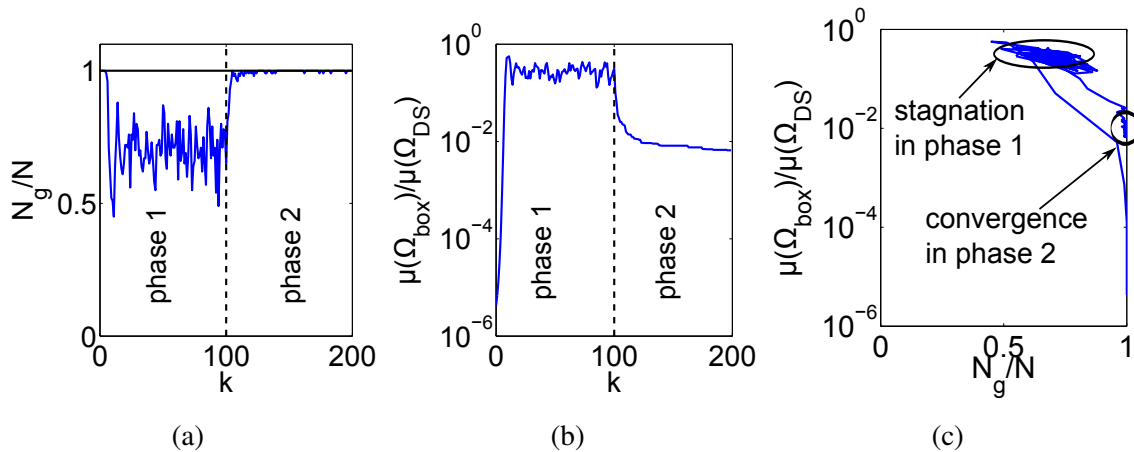


Figure 6.23: (a) The fraction of good sample points versus the number of iterations. (b) The hyperbox volume versus the number of iterations. (c) The hyperbox volume over the fraction of good sample points.

The behavior of the algorithm during the iteration is found in Figure 6.23. In Figure 6.23(a), the fraction of good sample points versus the number of iterations is depicted. The convergence coefficient is $c/N = 0.005$ with $\tilde{a}_0 = 0.65$. Figure 6.23(b) shows the normalized hyperbox volume versus the number of iterations. The normalized hyperbox volume versus the fraction of good sample points is illustrated in Figure 6.23(c). The curve stagnates

in the first phase while convergence is seen in the second phase. We observe that the behavior of the algorithm is quite similar to the other problems under consideration. The only difference consists in a larger convergence coefficient, i.e. a faster convergence in the consolidation phase, due to less dimensions.

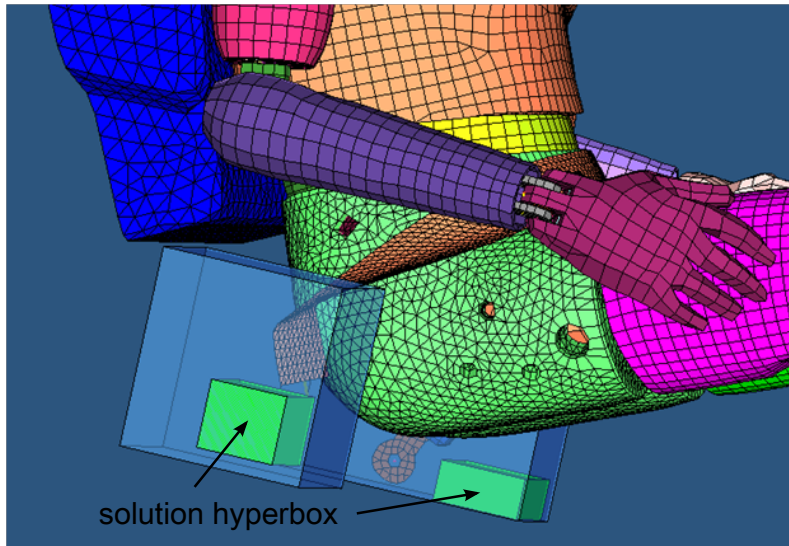


Figure 6.24: Solution hyperbox.

Chapter 7

Identifying key parameters

In this chapter, a procedure is presented to identify key parameters in high-dimensional and non-linear systems which are subject to uncertainty. We aim at improving a design which fails the design goals by changing the key parameters such that they lie within the desired intervals.

Section 7.1 motivates the need for a method to identify key parameters in order to turn a bad design into a good design with comparatively little effort. In Section 7.2, a simple example problem is considered. The mathematical problem statement for the hyperbox optimization with constraints is given in Section 7.3. Section 7.4 presents the extensions of the algorithm which was introduced in Chapter 3 to identify the key parameters in order to improve a design with little effort. The applicability of the algorithm is validated by two-dimensional example problems in Section 7.5. In Section 7.6, the proposed method is applied to a non-linear and high-dimensional engineering crash problem.

7.1 Motivation

Designs that fail to meet their design goals may be improved by appropriately changing relevant design parameters. When design parameters are subject to uncertainty, this can be very difficult. The deviation between desired and realized parameter settings may lead to catastrophic design failure, in particular when the design problem is non-linear and the system response abruptly changes under parameter variation. Uncertainty is present when parameters or component properties cannot be controlled exactly. As an example, the force-deformation characteristic of a structural member is difficult to adjust by detail parameters like the metal sheet thickness. This chapter is concerned with, first, identifying

the key parameters that can be used to improve a design with least effort, and, second, providing information on how these key parameters need to be modified in order to turn a bad into a good design in the presence of uncertainty.

Classical approaches to identify relevant parameters are sensitivity analysis, classical optimization and robust design optimization. *Sensitivity analysis* quantifies the importance of input parameters for the variability of the output [3, 66, 67, 74]. Local sensitivity analysis investigates the local influence of each input parameter on the output. This kind of analysis is well suited for problems that can be well approximated by linear functions. Local sensitivity measures are obtained by computing partial derivatives of the output function with respect to the input parameters. Global sensitivity analysis takes the entire design space into account to apportion the variability of the output parameter to the variability in each input parameter. There are several measures used in global sensitivity analysis: A *regression coefficient* quantifies the slope of a linear approximation, *Pearson correlation coefficient* measures to what degree an input parameter determines the output in a linear relationship. The *Spearman correlation coefficient* quantifies the monotony in the relationship between one input parameter and the output. *Sobol indices* are particularly tailored for multidimensional functions, see [33, 40, 58, 70, 78]. The first order Sobol index is a measure for the direct effect of an input parameter on the output. The higher order indices quantify the influence of the interactions between the input parameters. The fraction of the output variation that is related to each input parameter is measured by the total order index, see [58, 67, 70, 78].

Every sensitivity measure measures the importance of an input parameter in one particular sense. As all the information on how input and output parameters are related is reduced to one measure, other information is lost. Therefore, in the sensitivity measures mentioned before, no information is included on how the parameters have to be changed in order to obtain a particular result.

Contrary to sensitivity measures, *classical optimization* provides this information by seeking an optimum in the design space. Unfortunately, however, classical optimization does not take uncertainty into account. Optimal designs may be non-robust and quite sensitive to parameter variabilities. Due to the underlying uncertainty, realizing an optimum in a practical design may be impossible.

Both, sensitivity analysis and classical optimization are not concerned with uncertainty and therefore of limited use for the purpose of this chapter. *Robust design optimization* does take uncertainty into account by seeking a design point in a particular neighborhood with little output variation or sufficient performance (see [10]). The size of that neighborhood is

specified in advance and represents parameter variability associated with an measured or assumed underlying uncertainty. Robust design optimization prescribes the permissible variability and cannot optimize the tolerance to variations.

The approach presented in Chapter 3 is similar to robust design optimization in that it also computes a permissible region rather than one design point. The solution space, however, is constructed to be as *large as possible* to make it easier to reach the target. Robust design optimization does not seek a large solution space, it rather looks for a neighborhood with good output performance and fixed size. For a robust design optimization problem with a performance threshold value, this implies that, either, there is no solution if the neighborhood was chosen too large, or, there is a solution with an associated neighborhood which may not be as large as possible. Maximizing the solution space, however, provides a target space which is as large as possible and therefore easier to reach.

In this chapter, the method which was presented in Chapter 3 is extended with a focus on reducing the effort to turn a bad design in a good design. A large solution space is sought that already includes as many parameters from the bad design as possible by formulating appropriate constraints. Parameters without constraints may lie outside of the solution space and will have to be changed to fulfill the design goal.

7.2 A simple example problem

A simple example problem from crash analysis is considered as presented in [87]. The load case is similar to a USNCAP front crash, where the vehicle hits a rigid barrier at a speed of $v_0 = 56$ km/h with full overlap, see Figure 7.1.

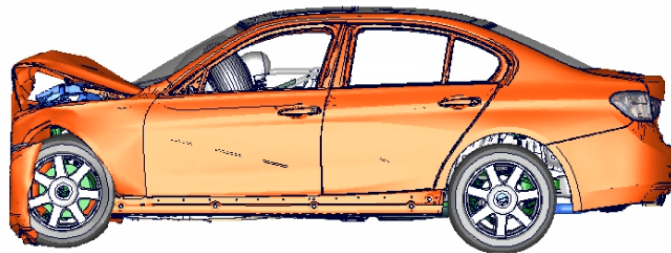


Figure 7.1: USNCAP front crash.

A model of the vehicle structure is used that consists of two structural components, see Figure 7.2. The structural components 1 and 2 are the only deformable parts and have the deformation measures u_1 and u_2 , respectively. The rest of the vehicle model is rigid. The

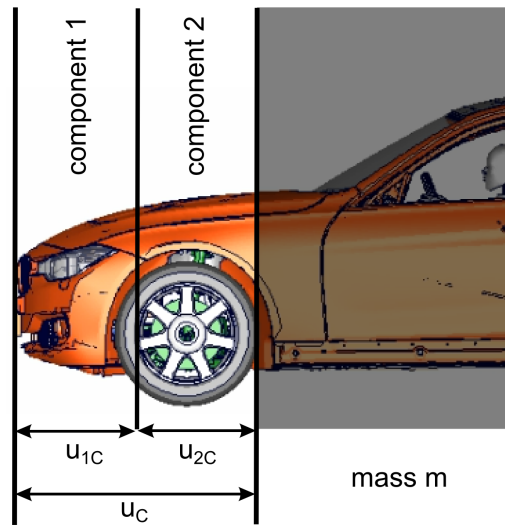


Figure 7.2: Vehicle structure of the example problem consisting of two deformable components.

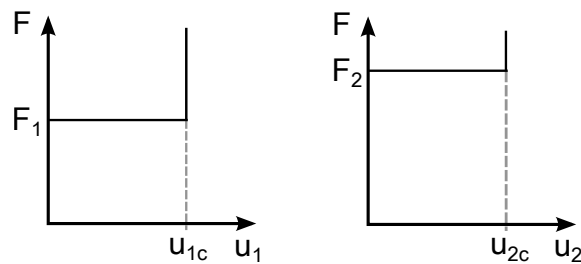


Figure 7.3: Force-deformation characteristics of the structural components 1 and 2.

deformable components have no mass, all mass is located on the rigid part. The forces necessary to deform components 1 and 2 are F_1 and F_2 , respectively, see Figure 7.3. F_1 and F_2 are assumed to be constant while deforming. If the maximum deformations u_{1c} and u_{2c} are reached, the forces may become arbitrarily large in order to avoid further deformation. The crash performance is measured by the acceleration of the passenger cell and the order of structural deformation, that is, whether component 1 or component 2 deform first. The design goals for the reduced example problem are:

- The maximum deceleration should not exceed the critical threshold value $a_{pulse,c}$, that is, $a_{pulse} \leq a_{pulse,c}$.
- Component 1 should deform before component 2 deforms.

This translates to the requirements on F_1 and F_2 that $F_1 \leq F_2$, that the deformation force

of component 2 is $F_2 \leq ma_{pulse,c}$ and that the entire kinetic energy is absorbed, that is $\frac{1}{2}mv_0^2 \leq F_1u_{1c} + F_2u_{2c}$.

With the performance function

$$f(F_1, F_2) = \begin{cases} 1, & \text{if } \frac{1}{2}mv_0^2 > F_1u_{1c} + F_2u_{2c} \\ 1, & \text{if } F_1 > F_2 \\ \frac{F_2/m - a_{pulse,c}}{a_{pulse,c}}, & \text{otherwise,} \end{cases} \quad (7.1)$$

the design goal is met, when

$$f(F_1, F_2) \leq 0. \quad (7.2)$$

The solution space defined by expression (7.2) is shown for $m = 2000$ kg, $a_{pulse,c} = 32$ g, $v_0 = 15.6$ m/s and $u_{1c} = u_{2c} = 0.3$ m in Figure 7.4(a).

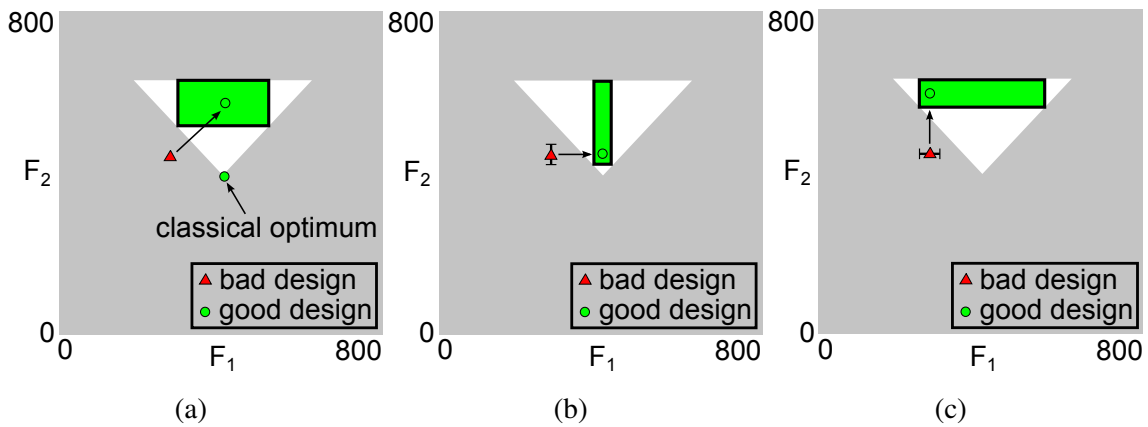


Figure 7.4: Changes necessary to meet the design goal: (a) F_1 and F_2 , (b) only F_1 , (c) only F_2 .

Now consider a design with $F_1 = 275$ kN and $F_2 = 450$ kN. It violates (7.2). In order to identify what parameter may be changed and by how much in order to improve the design with least effort, three scenarios are compared:

(a) A classical solution hyperbox with maximum volume is shown in Figure 7.4(a). Both components 1 and 2 have to be modified in order to meet the design goal.

(b) A solution hyperbox that includes $F_2 = 450$ kN is shown in 7.4(b). In order to meet the design goal, only component 1, that is, F_1 , will have to be changed. Note that F_2 is included in the solution hyperbox with a safety margin of ± 25 kN. This is necessary, since F_2 cannot be controlled exactly.

(c) Finally, a solution hyperbox that includes $F_1 = 275$ kN is shown in Figure 7.4(c). In order to meet the design goal, only component 2, that is, F_2 will have to be changed. The same safety margin as in scenario (b) is provided.

The solution hyperboxes of scenarios (b) and (c) are smaller than the one of scenario (a). In this sense, designs from these hyperboxes are less robust and more difficult to realize. However, scenario (a) requires redesigning two components, while scenarios (b) and (c) only require redesigning one component. A designer, knowing that component 1 is easier to redesign than component 2, would therefore prefer scenario (b). The deformation force F_1 would be the *key parameter* to meet the design goal.

This procedure can be generalized by seeking solution hyperboxes under the constraint that certain parameters are included with a specified safety margin. The associated mathematical problem statement is provided in the following section.

7.3 General problem statement

The optimization problem to maximize the size of the solution space as introduced in Chapter 2 reads as

$$\left. \begin{array}{l} \text{find } \mathbf{x}^{low}, \mathbf{x}^{up} \in \Omega_{DS} \text{ with } \mathbf{x}^{low} \leq \mathbf{x}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \text{ subject to } f(\mathbf{x}) \leq f_c \text{ for all } \mathbf{x} \in \Omega_{box}. \end{array} \right\} \quad (\text{P})$$

As motivated in the previous section, the classical problem statement is now enriched by constraints to ensure that the parameter values are included in the resulting solution hyperbox. More specifically, we shall consider the optimization problem

$$\left. \begin{array}{l} \text{find } \mathbf{x}^{low}, \mathbf{x}^{up} \in \Omega_{ds} \text{ with } \mathbf{x}^{low} \leq \mathbf{x}^{up} \text{ component-by-component} \\ \text{such that } \mu(\Omega_{box}) \rightarrow \max \text{ subject to } f(\mathbf{x}) \leq f_c \text{ for all } \mathbf{x} \in \Omega_{box} \\ \text{and } x_k^{low} \leq x_{c,k}^{low}, x_\ell^{up} \geq x_{c,\ell}^{up} \end{array} \right\} \quad (\text{P}^*)$$

with $x_{c,k}^{low}$ and $x_{c,\ell}^{up}$ being the constraint for the lower and upper boundary of the solution hyperbox, respectively.

7.4 Computing solution spaces with constraints

Algorithm 6 presented in Chapter 3 is extended to account for constraints and solve problem (P^{*}). The extension is done by modifying the cutting algorithm, where candidate

hyperboxes without bad designs are computed in three nested loops. In the original algorithm, the largest hyperbox is chosen as the new candidate hyperbox for the next iteration step. In the extended algorithm with constraints, an error measure is introduced that quantifies to what degree constraints are violated. The *constraint violation error* ε^2 is defined as

$$\varepsilon^2 = \sum_k \omega_k \varepsilon_k^2 + \sum_\ell \omega_\ell \varepsilon_\ell^2, \quad (7.3)$$

with k and ℓ being the indices of the lower and upper constrained parameter boundaries and with

$$\varepsilon_k = \begin{cases} 0, & \text{if } x_k^{low} \leq x_{c,k}^{low}, \\ x_k^{low} - x_{c,k}^{low}, & \text{otherwise} \end{cases} \quad (7.4)$$

and

$$\varepsilon_\ell = \begin{cases} 0, & \text{if } x_\ell^{up} \geq x_{c,\ell}^{up}, \\ x_\ell^{up} - x_{c,\ell}^{up}, & \text{otherwise.} \end{cases} \quad (7.5)$$

ω_k and ω_ℓ are weights of the constraints. If there are several hyperboxes satisfying all constraints, that is, $\varepsilon^2 = 0$, the hyperbox with the largest number of good sample points is chosen. The pseudo-code of the extended cutting algorithm is found in Algorithm 8 where the extension is colored in blue, cf. Algorithm 6.

7.5 Analytical examples

Two-dimensional problems are considered to show the functionality of the modified algorithm.

7.5.1 Problem 1: Rosenbrock function

In Section 4.1, we considered the non-linear optimization problem of maximizing the size of a rectangle in case of the Rosenbrock function. In addition to the problem given there, we impose the constraint that the variable x_1^{low} is smaller than a prescribed constant.

Analytical solution

If the constraint

$$x_1^{low} \leq -0.9 \quad (7.6)$$

```

Data: a hyperbox  $\Omega_{cand}$  and a set  $\mathcal{S} = \{\mathbf{x}_j \in \Omega_{cand} : f(\mathbf{x}_1) \geq \dots \geq f(\mathbf{x}_N)\}$  of sample
points
Result: hyperbox  $\subseteq \Omega_{cand}$  which contains only good sample points
forall the good sample points  $\{\mathbf{x}^{good} \in \mathcal{S} : f(\mathbf{x}^{good}) \leq f_c\}$  do
  forall the bad sample points  $\{\mathbf{x}^{bad} \in \mathcal{S} : f(\mathbf{x}^{bad}) > f_c\}$  do
    for dimension  $i = 1, 2, \dots, d$  do
      if  $x_i^{bad} < x_i^{good}$  then
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \geq x_i \geq x_i^{low}$ ;
      else
        | count the good sample points  $\mathbf{x}$  with  $x_i^{bad} \leq x_i \leq x_i^{up}$ ;
      end
    end
    choose the dimensions  $i^+$  where the fewest good sample points are removed;
    choose the dimensions  $i^{++}$  where the most bad sample points are removed
    from the selected dimensions  $i^+$ ;
    choose randomly the dimension  $i^*$  from the selected dimensions  $i^{++}$ ;
    if  $x_{i^*}^{bad} < x_{i^*}^{good}$  then cut to  $x_{i^*}^{low}$ ;
    else cut to  $x_{i^*}^{up}$ ;
  end
forall the dimensions  $i$  where a bad sample point is removed do
  if  $x_i^{bad} < x_i^{good}$  then
    |  $x_i^{low} := \min_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
  else
    |  $x_i^{up} := \max_j x_{i,j}$  for all remaining good sample points  $\mathbf{x}_j$ ;
  end
end
remember the hyperbox with smallest  $\varepsilon^2$ ;
if  $\varepsilon^2 = 0$  then
  | remember the hyperbox with most good sample points;
end
end

```

Algorithm 8: Pseudo-code of the extended cutting algorithm.

is added to the optimization problem (4.1), the analytical solution admits the values tabulated in the second column of Table 7.1. A visualization of this maximum is found in

Figure 7.5.

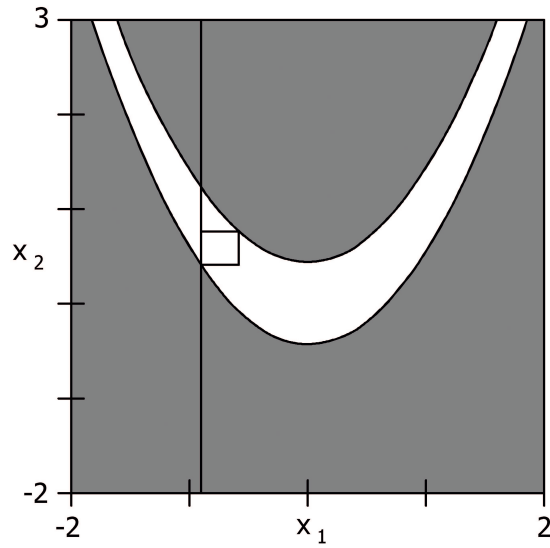


Figure 7.5: Problem 1 (additional constraints). The hyperbox of maximum volume with an additional constraint.

	analytical	numerical for $N = 100$			
	$x_{i,opt}$	$x_{i,avg}$	$\sigma(x_i)$	$\varepsilon(x_i)$	error in %
x_1^{low}	-0.900	-0.903	0.0164	0.00300	0.333
x_2^{low}	0.405	0.410	0.0307	0.00500	1.23
x_1^{up}	-0.586	-0.563	0.0966	0.0230	3.92
x_2^{up}	0.761	0.749	0.0880	0.0120	1.58

Table 7.1: Problem 1 (additional constraints). Analytical solution and related numerical results for 100 simulations.

Numerical solution

The optimization problem (4.1) with the additional constraint (7.6) is solved numerically by using Algorithm 8. The results of the numerical optimization obtained by executing the process (PR) are shown in Table 7.1 for $N = 100$ sample points per iteration. The mean $x_{i,avg}$ of the coordinates of the final hyperboxes of the 100 simulations, the related standard deviations $\sigma(x_i)$, the absolute errors $\varepsilon(x_i) = |x_{i,avg} - x_{i,opt}|$, and the relative errors $|x_{i,avg} - x_{i,opt}|/x_{i,opt}$ are tabulated in Table 7.1.

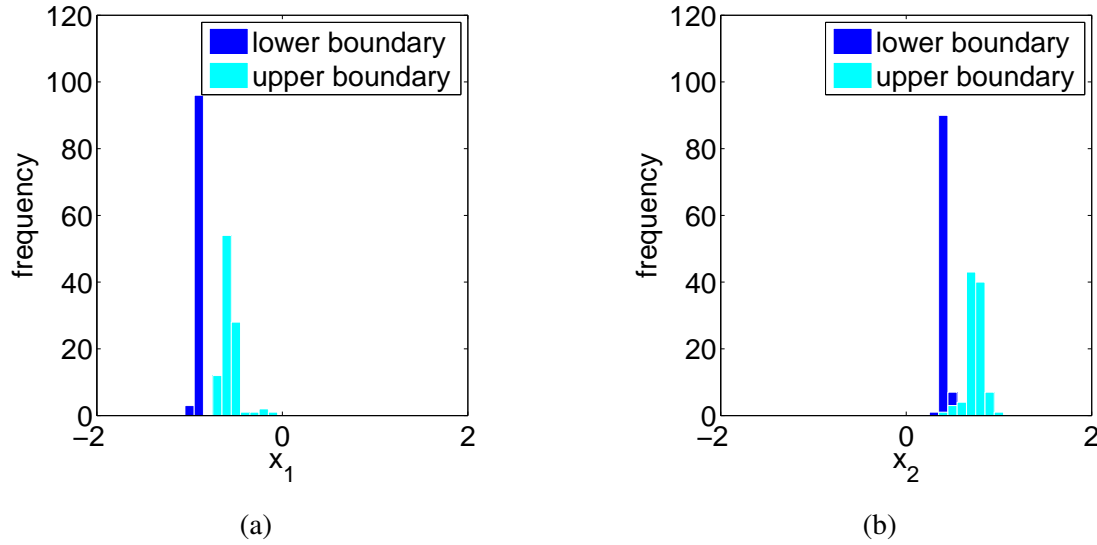


Figure 7.6: Distribution of the hyperboxes found by the algorithm for (a) coordinate x_1 and (b) coordinate x_2 for 100 simulations.

The distribution of the solution hyperboxes found by the algorithm is visualized via histograms in the Figure 7.6(a) for the coordinate x_1 and in the Figure 7.6(b) for the coordinate x_2 . These plots show that the proposed algorithm converges to the analytical solution in the sense that the average of the numerical solutions approximately agree with the analytical solutions and fulfills the additional constraint (7.6).

7.5.2 Numerical results of the simple example problem

The extended algorithm is applied to the simple example problem from Section 7.2. All three scenarios are computed with $N = 100$ designs per Monte Carlo sample. The first candidate hyperbox includes the classical optimum at the lower tip of the solution space triangle. The exploration phase and the consolidation phase are run for 20 and 10 iteration steps, respectively. This procedure describes a simulation.

Figure 7.7 shows how the extended algorithm drives the evolution of the candidate hyperbox for scenario (b). The constraints are chosen such that only F_1 has to be modified, that is, the constraints are $F_2^{low} \leq F_{c,2}^{low} = 425$ kN and $F_2^{up} \geq F_{c,2}^{up} = 475$ kN. The first row illustrates the exploration phase: the algorithm enlarges the volume of the candidate hyperbox by extending the hyperbox boundaries. It cannot extend the hyperbox boundaries in the direction of F_1 , as this would violate the constraints. In the consolidation phase, shown in

the second row, bad designs are removed, until the final solution hyperbox is obtained that satisfies the constraints.

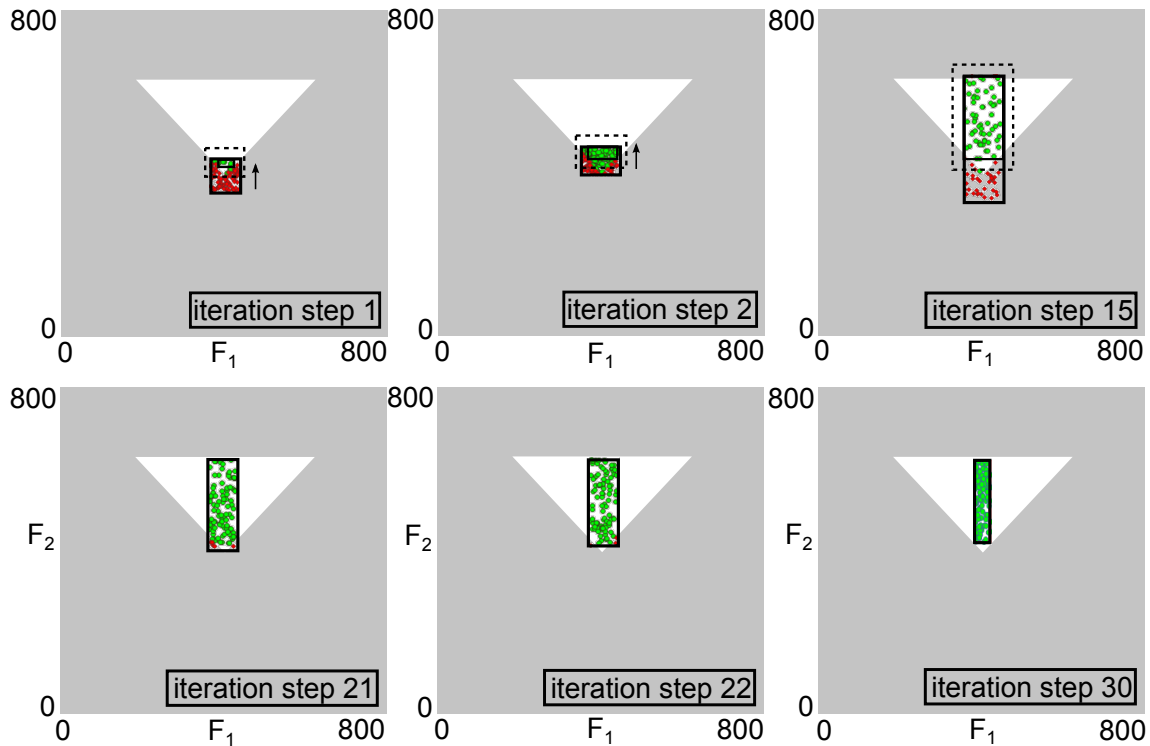


Figure 7.7: Evolution of the candidate hyperbox in the exploration phase (top row) and consolidation phase (bottom row) for constraints of scenario (b) ensuring that only F_1 needs to be changed, that is, $F_2^{low} \leq F_{c,2}^{low} = 425$ kN and $F_2^{up} \geq F_{c,2}^{up} = 475$ kN.

The numerical results of 100 simulations for scenario (a) are given in Table 7.2. The results of an arbitrarily chosen simulation is depicted in Figure 7.8(a). By $F_{i,avg}$, the average of 100 simulations for each interval boundary of each input parameter is indicated. The standard deviation is denoted by $\sigma(F_i)$. The absolute $\varepsilon(F_i)$ is calculated by $|F_{i,avg} - F_{i,opt}|$ and the relative error is calculated by $\varepsilon(F_i)/F_{i,opt}$. In Figure 7.9, the histograms of the results of 100 simulations are shown.

The values of the resulting hyperboxes of 100 simulations for scenario (b) are found in Table 7.3. An arbitrarily chosen simulation is seen in Figure 7.8(b). The histograms of the numerical results of 100 simulations are shown for F_1 and F_2 in Figure 7.10.

Finally, in Table 7.4, the numerical results of 100 simulations are tabulated for scenario (c). The solution hyperbox of an arbitrarily chosen simulation is shown in Figure 7.8(c). The histograms of the numerical results of 100 simulations are depicted for the coordinate F_1

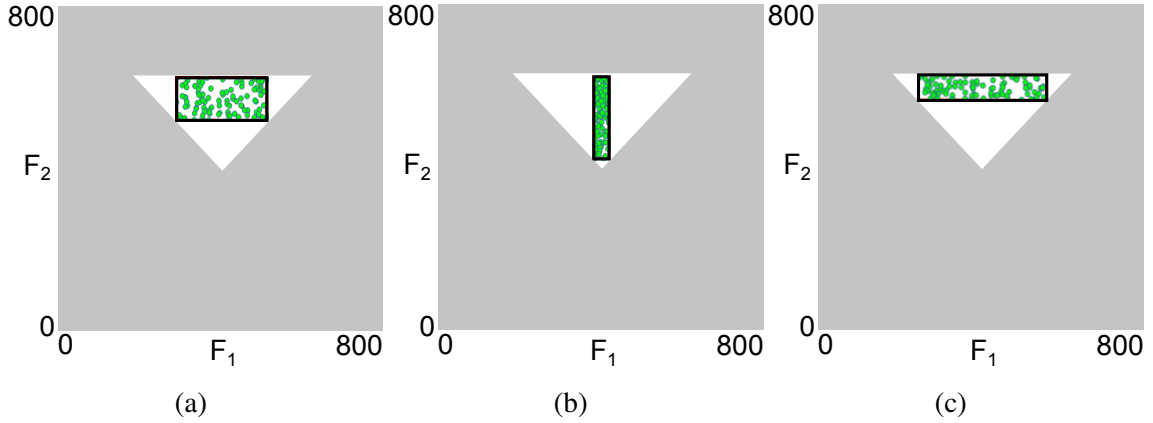


Figure 7.8: Computed solution hyperboxes (a) without constraints, (b) constraints ensuring that only F_1 needs to be changed ($F_2^{low} \leq F_{c,2}^{low} = 425$ kN, $F_2^{up} \geq F_{c,2}^{up} = 475$ kN), and (c) constraints ensuring that only F_2 needs to be changed ($F_1^{low} \leq F_{c,1}^{low} = 250$ kN and $F_1^{up} \geq F_{c,1}^{up} = 300$ kN).

	analytical	numerical for $N = 100$			
	$F_{i,opt}$	$F_{i,avg}$	$\sigma(F_i)$	$\varepsilon(F_i)$	error in %
F_1^{low}	291	291	14.0	0	0
F_2^{low}	516	516	13.6	0	0
F_1^{up}	516	515	13.5	1.00	0.194
F_2^{up}	628	626	1.46	2.00	0.318

Table 7.2: Simple example problem (scenario (a): F_1 and F_2 to be changed). Analytical solution and numerical results.

	analytical	numerical for $N = 100$			
	$F_{i,opt}$	$F_{i,avg}$	$\sigma(F_i)$	$\varepsilon(F_i)$	error in %
F_1^{low}	382	380	3.21	2.00	0.524
F_2^{low}	425	428	3.30	3.00	0.706
F_1^{up}	425	427	3.17	2.00	0.471
F_2^{up}	628	625	2.73	3.00	0.478

Table 7.3: Simple example problem (scenario (b): only F_1 to be changed ($F_2^{low} \leq F_{2,ref}^{low} = 425$ kN and $F_2^{up} \geq F_{2,ref}^{up} = 475$ kN)). Analytical solution and numerical results.

and the coordinate F_2 in Figure 7.11.

	analytical	numerical for $N = 100$			
	$F_{i,opt}$	$F_{i,avg}$	$\sigma(F_i)$	$\varepsilon(F_i)$	error in %
F_1^{low}	250	249	6.03	1.00	0.400
F_2^{low}	557	558	5.28	1.00	0.180
F_1^{up}	557	557	5.82	0	0
F_2^{up}	628	627	0.717	1.00	0.159

Table 7.4: Simple example problem (scenario (c): only F_2 to be changed ($F_1^{low} \leq F_{1,ref}^{low} = 250$ kN and $F_1^{up} \geq F_{1,ref}^{up} = 300$ kN)). Analytical solution and numerical results.

The numerical approximation and the analytical solution coincide within an error of less than 1% for all scenarios under consideration.

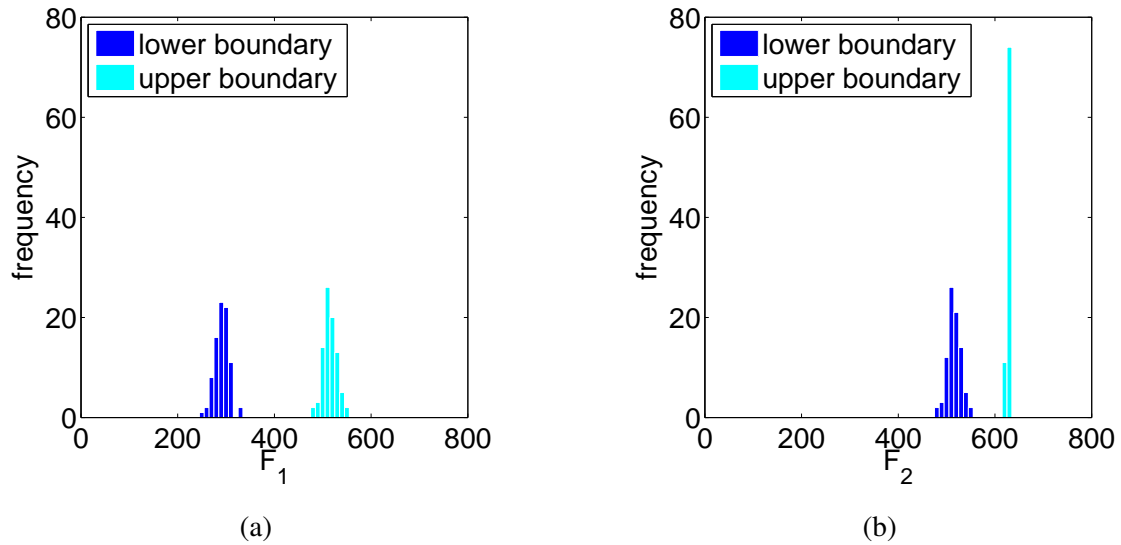


Figure 7.9: Simple example problem (scenario (a): F_1 and F_2 to be changed). Distribution of the hyperboxes found by the algorithm for (a) coordinate F_1 and (b) coordinate F_2 .

7.6 High-dimensional crash problem

A USNCAP front crash is considered. For the details of the modelling and the numerical simulation, we refer to Section 6.1.4.

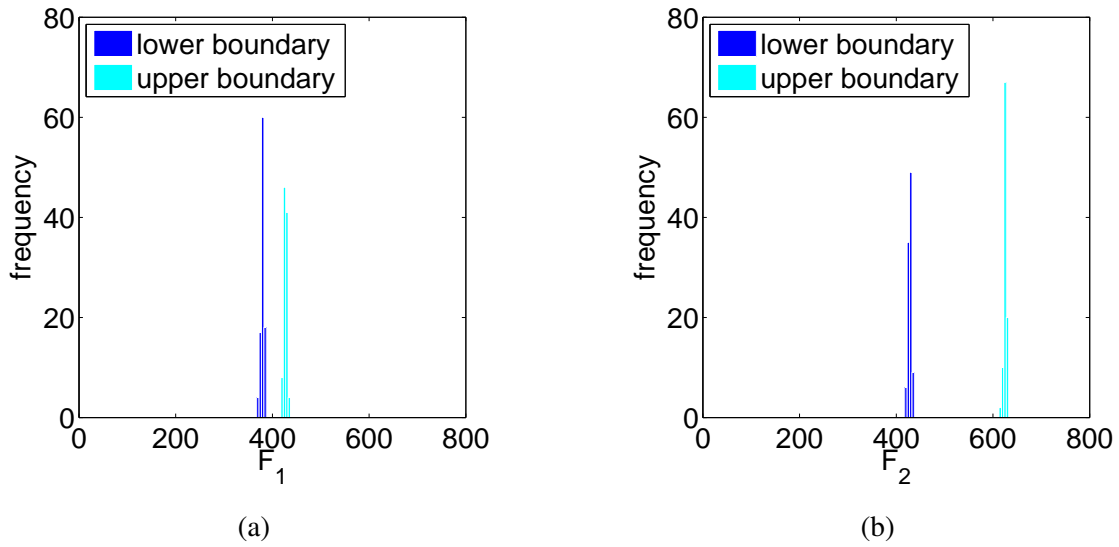


Figure 7.10: Simple example problem (scenario (b): only F_1 to be changed). Distribution of the hyperboxes found by the algorithm for (a) coordinate F_1 and (b) coordinate F_2 .

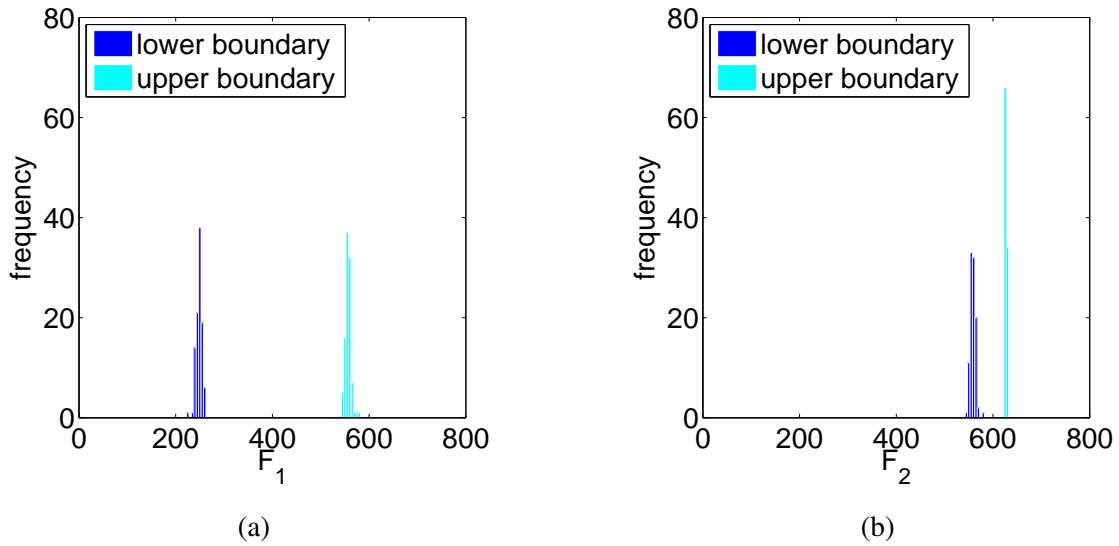


Figure 7.11: Simple example problem (scenario (c): only F_2 to be changed). Distribution of the hyperboxes found by the algorithm for (a) coordinate F_1 and (b) coordinate F_2 .

7.6.1 Why vehicle crash design is difficult

Improving bad designs is difficult because of non-linearity. All mappings between the detail, component and vehicle level are typically highly non-linear. Non-linearity between the component and the vehicle level can be observed in Figure 7.12. For a bad design with maximum deceleration $a_{pulse} > a_{pulse,c}$, the force-deformation characteristics of the crash hyperbox and the front rail are modified. When making only the crash box stronger, a_{pulse} becomes worse. Strengthening the front rail, improves a_{pulse} , however it remains supercritical. Combining the modification with worsening effect with the modification with insufficient effect, produces a good design with $a_{pulse} \leq a_{pulse,c}$. The influence of the design parameter *force-deformation characteristic of the crash box* is changed by modifying the other parameter *force-deformation characteristic of the front rail*. This effect will be called *parameter interaction*.

The parameter interaction between the force-deformation characteristics of the crash box and the front rail can be explained physically: while deforming, structural members exert a decelerating force on the vehicle that reduces the kinetic energy. When all structural members completed their deformation, the remaining kinetic energy is absorbed in an abrupt collision of the passenger cell with the engine block that is already in contact with the barrier wall. This final collision is associated with a deceleration signal, that increases with increasing remaining kinetic energy. One may assume that increasing the force of the crash box should increase the initial force that decelerates the vehicle, and therefore reduce the remaining kinetic energy and the maximum deceleration.

However, this is not the case for variant (2) in Figure 7.12: the front rail behind the crash is not strong enough to support the load of the crash box, resulting in a premature collapse of the front rail. This leads to an even lower force to decelerate the vehicle in the beginning, and, consequently, to a higher maximum deceleration. In variant (4), the front rail is sufficiently strong, the influence of the parameter *force-deformation characteristic of the crash box* is reversed, and the system exhibits the desired overall behavior.

In addition to parameter interaction, other non-linear phenomena are present in crash design, such as abrupt changes of vehicle responses or non-monotonous dependencies on design variables. Non-linearities make it difficult to assess the influence of each parameter, and therefore obstruct the identification of the key parameters and their setting necessary to turn a bad into a good design.

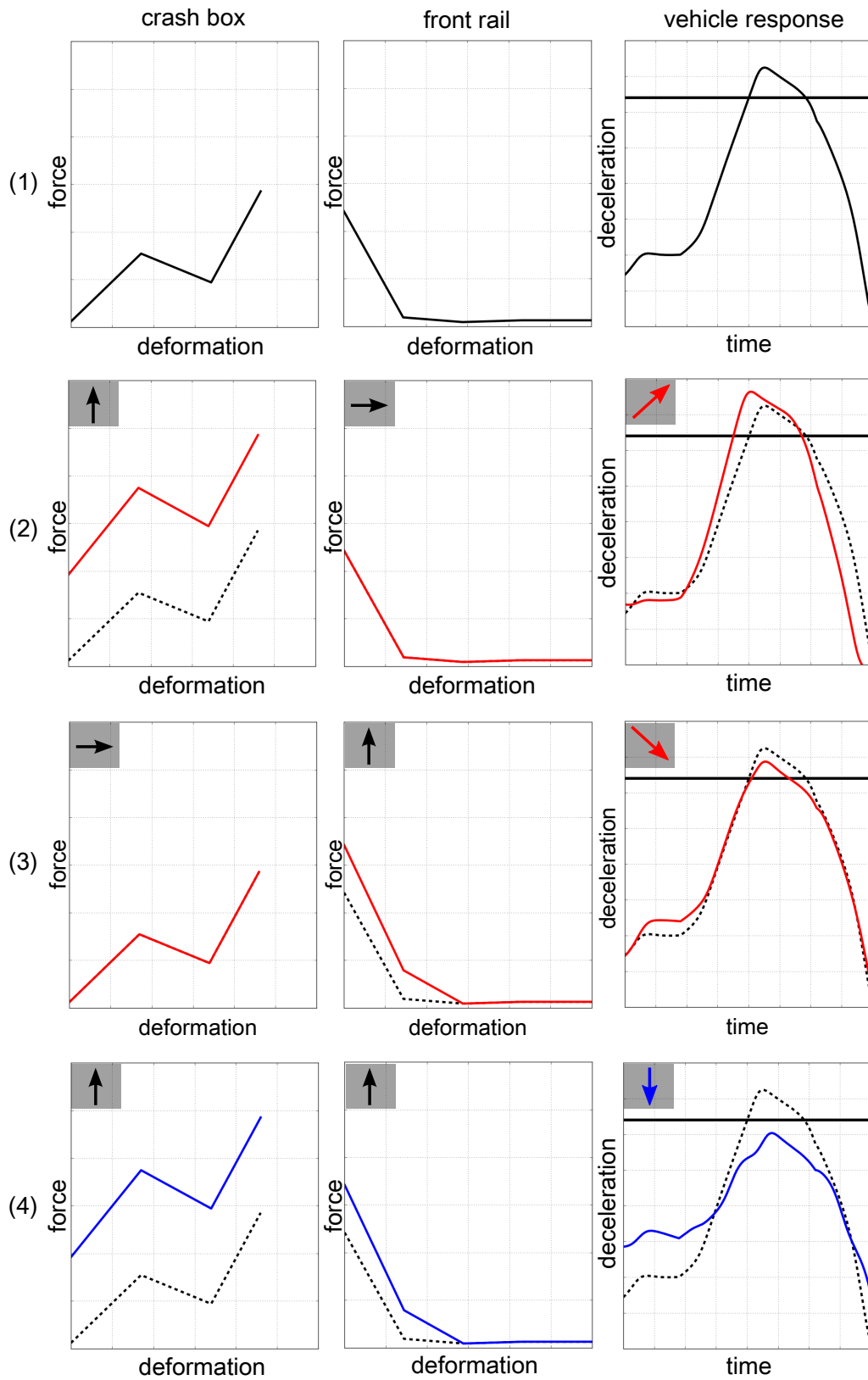


Figure 7.12: Force-deformation characteristics and their non-linear influence on the maximum deceleration. (1) Original design. (2)&(3) 2 modifications yielding bad designs each. (4) Combined modification yielding a good design.

7.6.2 Application in crash design

Identification of a key component

A vehicle structure as shown in Figure 6.1(b) is considered. It consists of nine structural members with force-deformation characteristics as shown in Figures 7.13(a). The force-deformation characteristics are measured in a detail finite element model. The performance in the USNCAP front crash is insufficient, because $a_{pulse} = a_{pulse,1} > a_{pulse,c}$. In order to identify the relevant components and the necessary modifications, solution spaces for the force-deformation characteristics are computed.

Remark 7.6.1. *In the present example, a bad vehicle design is given and all detail parameters are known. Rather than varying detail parameters according to the classical design approach, however, the relevant component to be modified is identified by computing appropriate solution spaces on the component level with the aid of the reduced model.*

The solution space for a force-deformation characteristic is represented by an upper and a lower boundary line in a force-deformation diagram, see the solid bold lines in Figure 7.13(a). The region bounded by these two lines is called a corridor. Corridors are approximated by linear interpolation between two support points.

A reduced crash model provides the mapping (6.3) for a total of $d = 55$ parameters F_i . For each component k , a solution space Ω_k is computed under the constraint, that all force-deformation characteristics are included, except the one of component k . This procedure is similar to the one that was applied in Section 7.2. Unfortunately, no corridor is obtained that strictly satisfied the constraints. Ω_5 however, see Figure 7.13, violates the constraints only to a negligible degree for components 4, 6 and 9.

Remark 7.6.2. *For component 4, the force-deformation characteristic lies outside the corridor for very large deformations where the force measurement is assumed to be inaccurate. For components 6 and 9, the force-deformation characteristic lies outside the corridor for deformation intervals that are much smaller than the total deformation.*

Noting that the force-deformation characteristic of component 5 lies below the associated corridor, it can be concluded that

- component 5 is a key component, that is, its force-deformation characteristic is a key parameter, and
- it needs to be reinforced to lie within its corridor and to turn the bad into a good design.

Design improvement

A straightforward reinforcement, for example by increasing the sheet metal thickness, is unfortunately not a good design measure: the force-deformation characteristic of component 5 is already at the upper limit for deformations close to 0. By analyzing the deformation of component 5 during the crash, however, an appropriate design measure can be identified. The detailed finite element simulation shows that at the deformation u^* the deformation force drops below the corridor. This happens exactly when the profile of component 5 collapses by forming a distinct fold as shown in Figure 7.13(b).

The fold forms at a location that does not deform before the profile collapses. Therefore, a local reinforcement of this location has no effect on the force-deformation characteristic for $u < u^*$, and the force-deformation characteristic does not cross the upper boundary line for deformations close to 0. It does nevertheless increase the deformation force at $u \approx u^*$, as intended.

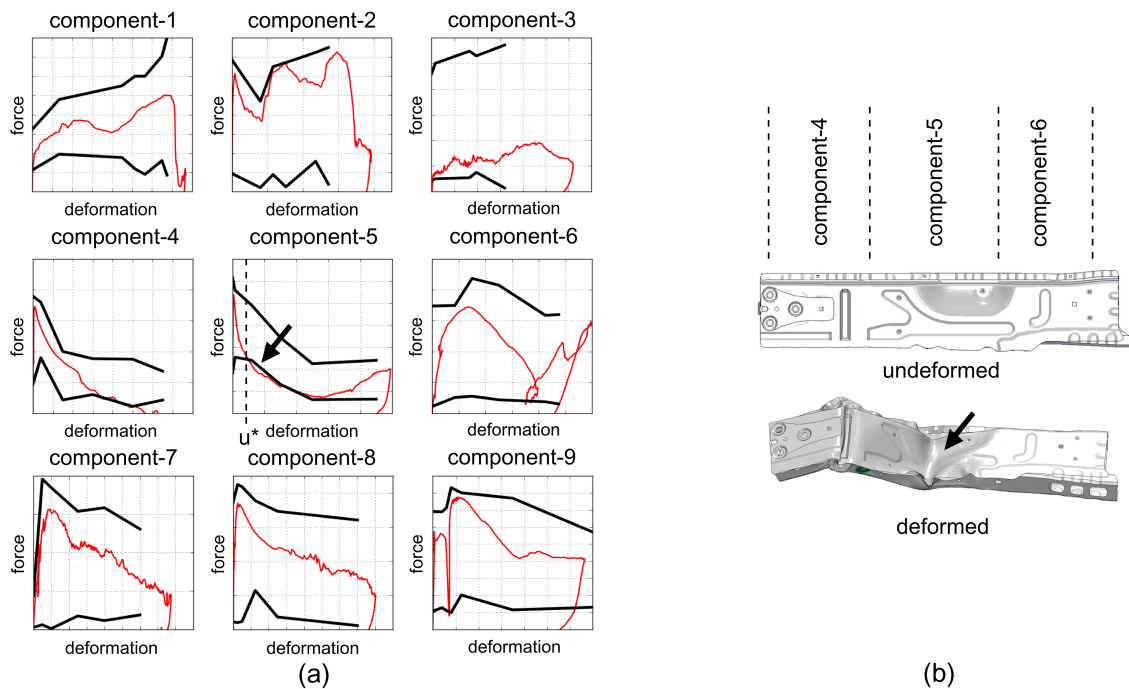


Figure 7.13: (a) Force-deformation characteristics of design 1 with $a_{pulse} > a_{pulse,c}$ and corridors Ω_5 . (b) The front rail undeformed and deformed.

The corridor provides a target region for the required force-deformation characteristic. A target region rather than a target point is necessary, as the component properties cannot be controlled exactly, that is, the component properties are *uncertain*. Wide corridors are necessary for successful design work under uncertainty.

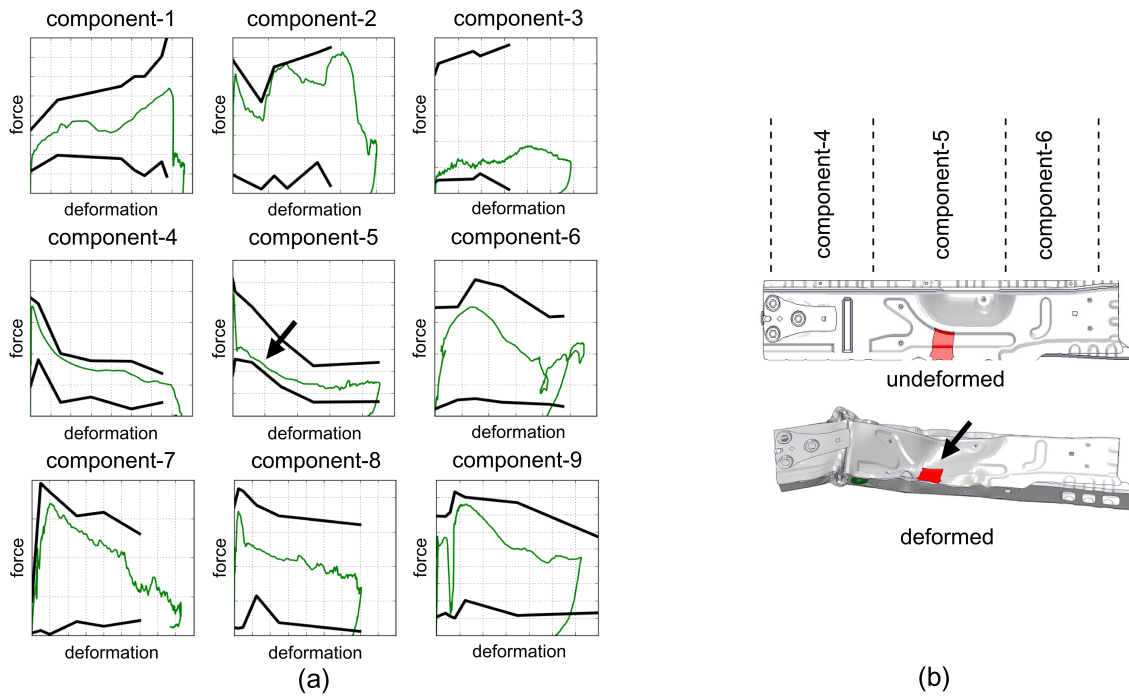


Figure 7.14: (a) Force-deformation characteristics of design 2 with $a_{pulse} < a_{pulse,c}$ and corridors Ω_5 . (b) The reinforced front rail undeformed and deformed.

The reinforcement is realized by increasing the sheet thickness locally. The resulting deformation and the force-deformation characteristics are shown in Figure 7.14. All force-deformation characteristics lie within their corridors, and the maximum deceleration dropped below the critical value, see Figure 7.15.

Remark 7.6.3. The forces measured in components 5, 6, 8 and 9 cross the corridor lines only in case of unloading, that is, when $\dot{u} < 0$.

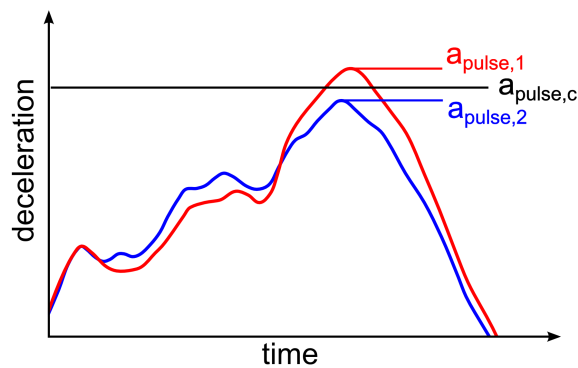


Figure 7.15: Deceleration of the good and bad design.

Note that the force-deformation characteristic of component 4 also changed, although this was not intended. The local reinforcement in component 5 has a stiffening effect on component 4, because they both share parts of the same structural member. This may be regarded as *uncertainty*. As the corridor for component 4 is wide enough, there is enough tolerance for the unintended variation: the force-deformation characteristic still lies within the corridors, and the design remains a good design.

The physical explanation for the improvement is similar to the one in Section 7.6.1: The reinforcement of the front rail, that is active at the deformation level u^* , decelerates the vehicle more in the beginning of the crash. The remaining kinetic energy and, thus, the maximum deceleration become smaller. If component 5 had been reinforced such that the force is increased for deformation levels close to 0 (which would result in a force-deformation characteristic outside the corridor), a different component may collapse, causing the initial deceleration to decrease and the maximum deceleration to increase.

Note that the local thickness at the reinforcement was identified as relevant detail parameter by the physical information that was extracted from the force-deformation characteristics and the associated target corridors. Identifying relevant detail parameters by variation instead could be prohibitively expensive, because the local thicknesses of many possible locations would have to be considered.

Using corridors for force-deformation characteristics as design goals helped identifying a key component, a key parameter and an appropriate design measure. In the example considered here, the design measure is small and modifies accurately the mechanical behavior of the non-linearly interacting structural members.

Remark 7.6.4. *To evaluate the corridors which were calculated on the basis of the reduced crash model with respect to the detail finite element model, the members of the primary load paths, this means the front rail, the crash box and the front axle carrier, were varied in strength.*

We made 20 variations by varying the strength of the front rail around the nominal value with $\pm 0.35\text{mm}$. Another 20 variations were obtained by varying the strength of the crash box around the nominal value with $\pm 0.7\text{mm}$. Finally, we made 20 variations of the strength of the front axle carrier by varying the nominal value with $\pm 0.7\text{mm}$. In total, 61 variations were done. When evaluating whether a curve lies within the corridor, elastic loading and unloading is disregarded.

We obtain 27 bad designs which are displayed by the red curves in Figure 7.16 and 34 good designs which are shown by the blue curves in the Figure 7.16. We observe in this

figure that 19 designs lie within the corridors. All of these are good designs. Therefore, we conclude that the corridors are valid with respect to the detail finite element models.

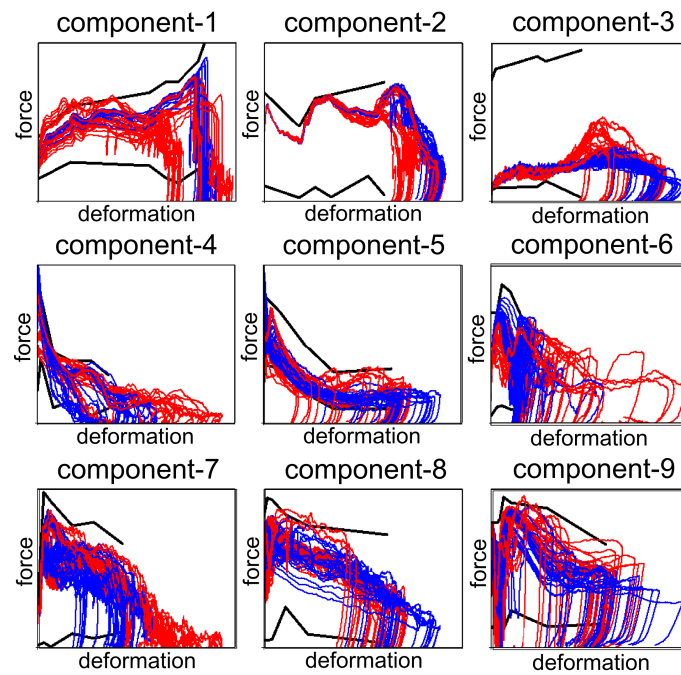


Figure 7.16: 61 variations: 27 bad designs (red curves) and 34 good designs (blue curves). 19 designs lie within the corridors. All of these are good designs.

Chapter 8

Conclusion

In the present thesis, we analyzed a new method to identify a hyperbox with maximum volume in the design space such that all designs inside this hyperbox are subcritical. The method can be applied to any high-dimensional, non-linear and noisy system. For a design to be good, the choice of a parameter value within its assigned interval does not depend on the values of the other parameters as long as they are within their respective intervals. In this sense, the parameters are decoupled from each other.

Robustness can be measured by the size of the resulting intervals. Moreover, intervals help to identify relevant parameters to improve a non-robust or critical solution. They may be combined with intervals of other disciplines – their cross sections are global solution spaces.

Several benchmark problems were constructed to study the convergence of the algorithm. The convergence in the mean to the optimal solution was shown in low dimensions and for problems with rectangular boundaries in high dimensions. In Problem 4 (tilted hyperplane), the volume of the hyperbox found by the algorithm is very large compared to the optimal hyperbox in high dimensions. However, the bad volume contained in the computed hyperboxes is small. If the widths of the intervals of a hyperbox are slightly larger than the widths of the intervals of the analytical solution hyperbox, the volume of the considered hyperbox is considerably larger than the volume of the analytical solution hyperbox in high dimensions, that is, the available data become sparse. This effect reflects the curse of dimensionality. Nevertheless, this observation does not affect the practical applicability since the fraction of the good space is still close to 100%. This was demonstrated by an engineering model for a front crash. Moreover, the dependency of the corner problem from the boundary of the good space was investigated.

The convergence behavior of the consolidation phase of the algorithm was analyzed. It

turned out that the convergence speed decreases when the number of dimensions increases. The convergence coefficient was introduced to measure the convergence speed. The algorithm was identified as a Markov chain. For a problem where the good space is a hyperbox contained in the design space, an analytical model was derived which describes the algorithm's convergence behavior

Moreover, the conflict between the resulting volume and the convergence speed was studied for several high-dimensional optimization problems. The volume of the solution hyperbox increases when the number of sample points per iteration is increased. However, the convergence slows down which is reflected by the decreasing convergence coefficient, i.e., more iterations are needed to converge. This conflict corresponds to a Pareto frontier. Mechanisms explaining this behavior were identified as *overestimation due to sparse sampling*, *impossibility of boundary corrections* and *each bad sample point can be used for one dimension only*. For Problem 4 (tilted hyperplane) and Problem 7 (front crash), the same convergence behavior was observed. This indicates that, in Problem 4, the boundary of the good space has a similar shape as in Problem 7.

The presented method was applied to different engineering problems. The first problem was a front crash design problem where corridors for the force-deformation characteristics of the car's components were calculated such that each curve within the corridors leads to a subcritical deceleration maximum during the front crash. Then, we applied the algorithm to a problem from a forming process. For the forming process, intervals were determined for the bead forces and the tool binder force with the aid of a response surface model. Within the intervals, it holds that the stamped part provides a sufficient hardening and does not contain cracks. Finally, we presented a problem from the rear passenger safety where intervals were determined for the positions of the belt anchor and the belt lock of the seat belt. The goal was to avoid submarining which occurs if the lap belt slips off the iliac crest and cuts into the soft abdomen region. All these results show the applicability and the feasibility of the optimization algorithm to real life engineering problems.

The hyperbox can also be used to identify which parameters and how much they have to be changed in order to reach the design goal. The method was extended such that a hyperbox with maximum volume was identified which includes all parameter values of a bad design except for a few parameter values. These few parameters were called key parameters because they may be changed with little effort in order to lie within their intervals – so-called target regions. Often, the effort is small if there are only a few key parameters. This methodology was demonstrated with a simple example problem from crash analysis with two input parameters. Different two-dimensional benchmark problems were considered

to validate the accuracy of the algorithm. The applicability to large engineering problems was validated by considering a front crash design problem. Starting from a bad design, the corresponding target regions were calculated and the key parameters were identified. By an appropriate modification, the design was changed to reach the design goal.

Appendix A

Theory of the optimization under constraints

In this appendix, we review briefly the mathematical theory of constrained optimization problems. To this end, let us consider the following optimization problem under constraints, see [23]:

$$\left. \begin{array}{l} \text{find } \mathbf{x} \in \mathbb{R}^d \\ \text{such that } f(\mathbf{x}) \rightarrow \min \\ \text{subject to } g_s(\mathbf{x}) \leq 0, \quad s = 1, 2, \dots, m, \text{ and } h_t(\mathbf{x}) = 0, \quad t = 1, 2, \dots, p. \end{array} \right\} \quad (\text{Q})$$

The following theory is given in [9, 23, 35, 43].

A.1 Definitions

The definition of a tangential cone is given as follows.

Definition A.1.1 (Tangential cone). *Let $X \subseteq \mathbb{R}^d$ be a non-empty set. Then, a vector $\mathbf{d} \in \mathbb{R}^d$ is called tangential to X in $\mathbf{x} \in X$, if sequences $\{\mathbf{x}^k\} \subseteq X$ and $\{t_k\} \subseteq \mathbb{R}$ with $t_k \downarrow 0$ exist such that*

$$\mathbf{x}^k \rightarrow \mathbf{x} \quad \text{and} \quad \frac{\mathbf{x}^k - \mathbf{x}}{t_k} \rightarrow \mathbf{d}$$

for $k \rightarrow \infty$. The set of all these directions is called tangential cone of X in $\mathbf{x} \in X$ and is denoted by $\mathcal{T}_X(\mathbf{x})$, i.e.,

$$\mathcal{T}_X(\mathbf{x}) = \left\{ \mathbf{d} \in \mathbb{R}^d : \exists \{\mathbf{x}^k\} \subseteq X \exists t_k \downarrow 0 \text{ with } \mathbf{x}^k \rightarrow \mathbf{x} \text{ and } \frac{\mathbf{x}^k - \mathbf{x}}{t_k} \rightarrow \mathbf{d} \right\}.$$

Next, we define a set of admissible points and a local minimum.

Definition A.1.2 (Set of admissible points). *The set of admissible points X of the optimization problem (Q) is defined by*

$$X := \{\mathbf{x} \in \mathbb{R}^d : g_s(\mathbf{x}) \leq 0, s = 1, 2, \dots, m, \text{ and } h_t(\mathbf{x}) = 0, t = 1, 2, \dots, p\}.$$

Definition A.1.3 (Local minimum). *The point $\mathbf{x}^* \in X$ is called a local minimum of the optimization problem (Q) if it holds*

$$f(\mathbf{x}^*) \leq f(\mathbf{x}) \quad \text{for all } \mathbf{x} \in X \cap \mathcal{U}$$

with $\mathcal{U} \subseteq \mathbb{R}^d$ being a neighborhood of \mathbf{x}^* .

Now, we introduce the definition of a linearized tangential cone.

Definition A.1.4 (Linearized tangential cone). *Let $\mathbf{x} \in X$ be an admissible point of the optimization problem (Q). Then, the set*

$$\mathcal{T}_{lin}(\mathbf{x}) = \{\mathbf{d} \in \mathbb{R}^d : \nabla g_s(\mathbf{x}^*)^T \mathbf{d} \leq 0, s \in I(\mathbf{x}), \nabla h_t(\mathbf{x}^*)^T \mathbf{d} = 0, t = 1, 2, \dots, p\}$$

is the linearized tangential cone of X in \mathbf{x} . The set

$$I(\mathbf{x}) := \{s \in \{1, 2, \dots, m\} : g_s(\mathbf{x}) = 0\}$$

is the set of active inequality constraints in \mathbf{x} .

The Lagrange function and the Lagrange multipliers are defined as follows.

Definition A.1.5 (Lagrange function / Lagrange multipliers). *The Lagrange function of the optimization problem under constraints (Q) is defined by*

$$L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := f(\mathbf{x}) + \sum_{s=1}^m \lambda_s g_s(\mathbf{x}) + \sum_{t=1}^p \mu_t h_t(\mathbf{x}).$$

The parameters $\boldsymbol{\lambda} = [\lambda_s]$, $s = 1, 2, \dots, m$, and $\boldsymbol{\mu} = [\mu_t]$, $t = 1, 2, \dots, p$, are called Lagrange multipliers.

To define the Karush-Kuhn-Tucker condition and the Karush-Kuhn-Tucker point, we have to assume that f , g_s , $s = 1, 2, \dots, m$, and h_t , $t = 1, 2, \dots, p$, are continuously differentiable.

Definition A.1.6 (Karush-Kuhn-Tucker condition / Karush-Kuhn-Tucker point). *Consider the optimization problem (Q).*

1. The conditions

$$\begin{aligned}
\nabla_{\mathbf{x}}L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) &= \mathbf{0} \\
h_t(\mathbf{x}) &= 0, \quad t = 1, 2, \dots, p \\
\lambda_s &\geq 0, \quad s = 1, 2, \dots, m \\
g_s(\mathbf{x}) &\leq 0, \quad s = 1, 2, \dots, m \\
\lambda_s g_s(\mathbf{x}) &= 0, \quad s = 1, 2, \dots, m
\end{aligned}$$

are called the Karush-Kuhn-Tucker conditions (KKT-conditions) of the optimization problem (Q) with

$$\nabla_{\mathbf{x}}L(\mathbf{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := \nabla f(\mathbf{x}) + \sum_{s=1}^m \lambda_s \nabla g_s(\mathbf{x}) + \sum_{t=1}^p \mu_t \nabla h_t(\mathbf{x})$$

being the gradient of the Lagrange function L with respect to \mathbf{x} .

2. Each vector $(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*)$ which fulfills the KKT-conditions is called Karush-Kuhn-Tucker point (KKT-point) of the optimization problem (Q).

Definition A.1.7 (Abadie constraint qualification (Abadie CQ)). An admissible point \mathbf{x} of the optimization problem (Q) fulfills the Abadie constraint qualification (Abadie CQ) if it holds $\mathcal{T}_{\mathcal{X}}(\mathbf{x}) = \mathcal{T}_{\text{lin}}(\mathbf{x})$.

Definition A.1.8 (Mangasarian-Fromovitz constraint qualification (MFCQ)). Let the point \mathbf{x} be an admissible point of the optimization problem (Q) and the set $I(\mathbf{x}) = \{s \in \{0, 1, \dots, m\} : g_s(\mathbf{x}) = 0\}$ be the set of active inequality constraints. The vector \mathbf{x} fulfills the Mangasarian-Fromovitz constraint qualification (MFCQ) if the following conditions are fulfilled:

(a) The gradients

$$\nabla h_t(\mathbf{x}), \quad t = 1, 2, \dots, p$$

are linear independent.

(b) A vector $\mathbf{d} \in \mathbb{R}^d$ exists with

$$\nabla g_s(\mathbf{x})^T \mathbf{d}, \quad s \in I(\mathbf{x}), \quad \text{and} \quad \nabla h_t(\mathbf{x})^T \mathbf{d} = 0, \quad t = 1, 2, \dots, p.$$

The linear independence constraint qualification is defined as follows.

Definition A.1.9 (Linear independence constraint qualification (LICQ)). Let $\mathbf{x} \in \mathbb{R}^d$ be an admissible point of the optimization problem (Q) and $I(\mathbf{x}) = \{s \in \{0, 1, \dots, m\} : g_s(\mathbf{x}) = 0\}$ the corresponding set of the active inequality constraints. Then, \mathbf{x} fulfills the linear independence constraint qualification (LICQ) if the gradients

$$\nabla g_s(\mathbf{x}), \quad s \in I(\mathbf{x}), \quad \text{and} \quad \nabla h_t(\mathbf{x}), \quad t = 1, 2, \dots, p,$$

are linear independent. This means that all elements of the set of all gradients

$$\mathcal{G} = \{\nabla g_s(\mathbf{x}) : s \in I(\mathbf{x})\} \cup \{\nabla h_t(\mathbf{x}) : t = 1, 2, \dots, p\}$$

are linearly independent of each other.

Finally, we define a convex set and a convex function.

Definition A.1.10 (Convex). A set $\mathcal{M} \subseteq \mathbb{R}^d$ is called convex if it holds

$$\alpha x + (1 - \alpha)y \in \mathcal{M}$$

for all $x, y \in \mathcal{M}$ and $\alpha \in [0, 1]$. A function $f : \mathcal{M} \rightarrow \mathbb{R}$ is called convex if \mathcal{M} is not empty and convex and if it holds

$$f(\alpha x + (1 - \alpha)y) \leq \alpha f(x) + (1 - \alpha)f(y)$$

for all $x, y \in \mathcal{M}$ and $\alpha \in [0, 1]$.

A.2 Theorems

With these preparations at hand, the following theorems can be formulated.

Theorem A.2.1 (KKT-conditions under Abadie CQ). Let $\mathbf{x}^* \in \mathbb{R}^d$ be a local minimum of the optimization problem (Q) which fulfills the Abadie CQ. Then, Lagrange multipliers λ^* and μ^* exist such that the triple $(\mathbf{x}^*, \lambda^*, \mu^*)$ is a KKT-point of (Q).

The proof of the Theorem A.2.1 and also of the following theorem can be found in [23].

Theorem A.2.2 (KKT-conditions under MFCQ). Let $\mathbf{x}^* \in \mathbb{R}^d$ be a local minimum of the optimization problem (Q) which fulfills the MFCQ-condition. Then, Lagrange multipliers λ^* and μ^* exist such that the triple $(\mathbf{x}^*, \lambda^*, \mu^*)$ is a KKT-point of (Q).

Theorem A.2.3 (KKT-conditions under LICQ). Let $\mathbf{x}^* \in \mathbb{R}^d$ be a local minimum of the optimization problem (Q) which fulfills the LICQ-condition. Then, there are unique vectors of Lagrange multipliers λ^* and μ^* such that the triple $(\mathbf{x}^*, \lambda^*, \mu^*)$ is a KKT-point of (Q).

For the sake of completeness, we present the proof of this theorem as it can be found in [23].

Proof. First, we show that the MFCQ-condition results from the LICQ-condition. Therefore, let the LICQ-condition be fulfilled in \mathbf{x}^* . Then, the part (a) of the Definition A.1.8 is obviously fulfilled. In order to prove the existence of a vector $\mathbf{d} \in \mathbb{R}^d$ with the properties which are given in the part (b) of the Definition A.1.8, we denote by $I(\mathbf{x}^*)$ the set of active inequality constraints and by m^* the number of elements in $I(\mathbf{x}^*)$.

Let $\mathbf{A} \in \mathbb{R}^{d \times d}$ be a matrix which is built as follows: The first m^* row vectors are the gradients $\nabla g_s(\mathbf{x}^*)^T$, $s \in I(\mathbf{x}^*)$, the following p row vectors are the gradients $\nabla h_t(\mathbf{x}^*)^T$ with $t = 1, 2, \dots, p$, and the remaining rows are filled up such that the matrix \mathbf{A} is regular. This is always possible due to the LICQ-condition.

Then, we define a vector $\mathbf{b} \in \mathbb{R}^d$ as follows: The first m^* entries of \mathbf{b} are all equal to -1 , the following p components of \mathbf{b} are all equal to 0, and the remaining components of \mathbf{b} are arbitrarily chosen.

Because the matrix \mathbf{A} is regular, the linear system of equations $\mathbf{A}\mathbf{d} = \mathbf{b}$ has a unique solution $\mathbf{d} \in \mathbb{R}^d$. The definitions of \mathbf{A} and \mathbf{b} imply that the vector \mathbf{d} fulfills all the properties of the part (b) of the Definition A.1.8. Therefore, we proved that the LICQ-condition implies the validity of the MFCQ-condition.

Consequently, we can apply the Theorem A.2.2: Lagrange multipliers λ^* and μ^* exist such that the triple $(\mathbf{x}^*, \lambda^*, \mu^*)$ is a KKT-point of (Q).

The equation $\lambda_s^* = 0$ for $s \notin I(\mathbf{x}^*)$ follows from the KKT-condition. Thus, the uniqueness of the Lagrange multipliers λ_s^* for $s \in I(\mathbf{x}^*)$ and μ_t^* for $t = 1, 2, \dots, p$ results from the LICQ-condition and $\nabla_{\mathbf{x}}L(\mathbf{x}^*, \lambda^*, \mu^*) = \mathbf{0}$. \square

The KKT-conditions are necessary conditions. If the functions f and g_s , $s = 1, 2, \dots, m$, are convex, and the functions h_t , $t = 1, 2, \dots, p$, are linear, then the KKT-conditions are also sufficient conditions for the optimality of \mathbf{x}^* .

Define

$$I_0(\mathbf{x}^*) := \{s \in I(\mathbf{x}^*) : \lambda_s^* = 0\} \quad \text{and} \quad I_>(\mathbf{x}^*) := \{s \in I(\mathbf{x}^*) : \lambda_s^* > 0\}.$$

If we assume that f , g_s , $s = 1, 2, \dots, m$, and h_t , $t = 1, 2, \dots, p$, are twice continuously differentiable, the following sufficient condition of second order can be formulated.

Theorem A.2.4 (Sufficient condition of second order). *Let there exist a vector of Lagrange multipliers λ^* and a vector of Lagrange multipliers μ^* for an admissible point $\mathbf{x}^* \in X$ such that the KKT-conditions are fulfilled. Moreover, let be*

$$\mathbf{d}^T \nabla_{\mathbf{xx}}^2 L(\mathbf{x}^*, \lambda^*, \mu^*) \mathbf{d} > 0 \quad \text{for all } \mathbf{d} \in \mathcal{T}(\mathbf{x}^*, \lambda^*, \mu^*) \setminus \{\mathbf{0}\}$$

with

$$\mathcal{T}(\mathbf{x}^*, \boldsymbol{\lambda}^*, \boldsymbol{\mu}^*) = \{\mathbf{d} \in \mathbb{R}^d : \nabla g_s(\mathbf{x}^*)^T \mathbf{d} = 0, s \in I_>(\mathbf{x}^*), \nabla g_s(\mathbf{x}^*)^T \mathbf{d} \leq 0, s \in I_0(\mathbf{x}^*), \\ \nabla h_t(\mathbf{x}^*)^T \mathbf{d} = 0, t = 1, 2, \dots, p\}.$$

Then, \mathbf{x}^* is the solution of the optimization problem (Q).

A proof of this theorem is again found in [23].

Appendix B

Extension of the analytical model

We consider the analytical model which was introduced in Section 5.2. This analytical model describes the behavior of the convergence speed in the consolidation phase for Problem 3 in $d \in \mathbb{N}$ dimensions. Recall that we approximated the expectation value of the fraction of good sample points a_{k+1} by

$$\mathbb{E}(a_{k+1}) \approx \left(\frac{r}{r + \mathbb{E}(e_{k+1}|e_k)} \right)^d \quad (\text{B.1})$$

with

$$\mathbb{E}(e_{k+1}|e_k) = \frac{1}{N+1} \frac{1}{(r+e_k)^N} \{(r+e_k)^{N+1} - r^{N+1}\},$$

cf. Theorem 5.2.2.

We want to extend the model described in equation (B.1) such that it is applicable to the exploration phase, too. In the exploration phase, first, the cutting algorithm is applied. Then, the candidate hyperbox is modified by growing in all parameter directions to enable the hyperbox to evolve towards beneficial directions with increasing box size. This is in contrast to the consolidation phase where the candidate hyperbox is only modified by the cutting algorithm.

Repeating the arguments of Section 5.2, we conclude that the expectation value of the fraction a_{k+1} of good sample points in consideration of the exploration phase and the consolidation phase is given as follows:

$$\mathbb{E}(a_{k+1}) \approx \begin{cases} \left(\frac{r}{r + \mathbb{E}(e_{k+1}|e_k)} \right)^d, & \text{in the consolidation phase,} \\ \left(\frac{r}{\min(\gamma + r + \mathbb{E}(e_{k+1}|e_k), 1)} \right)^d, & \text{in the exploration phase,} \end{cases} \quad (\text{B.2})$$

with

$$\mathbb{E}(e_{k+1}|e_k) = \frac{1}{N+1} \frac{1}{(r+e_k)^N} \{(r+e_k)^{N+1} - r^{N+1}\}$$

and $\gamma > 0$ being a constant. Hence, the expectation value of the volume of the candidate hyperbox $\mu(\Omega_{box}^{(k+1)})$ in the $(k+1)$ -st iteration step is calculated by

$$\mathbb{E}(\mu(\Omega_{box}^{(k+1)})) \approx \begin{cases} \left(r + \mathbb{E}(e_{k+1}|e_k)\right)^d, & \text{in the consolidation phase,} \\ \left(\min(\gamma + r + \mathbb{E}(e_{k+1}|e_k), 1)\right)^d, & \text{in the exploration phase.} \end{cases} \quad (\text{B.3})$$

The fraction a_{k+1} of good sample points and the volume $\mu(\Omega_{box}^{(k+1)})$ of the candidate hyperbox can be determined in the $(k+1)$ -st iteration step by the equation (B.2) and the equation (B.3), respectively. Successively, in Figure B.1, the volume $\mu(\Omega_{box})$ versus the fraction a of good sample points calculated analytically are shown and compared with the numerical results. For the diagrams, the exploration phase was repeated 100 times and the consolidation phase was iterated 100 times. The rows correspond to different numbers of sample points per iteration step, the columns correspond to different dimensions. The results of the algorithm are plotted by blue dashed lines and the results of the analytical model are plotted by green lines. In all diagrams except for $N = 10$, a stagnation in the exploration phase and the convergence in the consolidation phase are observed.

For $d = 2$, $N = 100$ and $N = 200$, the results of the analytical model and the numerical results agree reasonable well. For $N = 1000$, the agreement is very good. The same behavior is observed for $d = 3$. For $d = 10$, the agreement is reasonable well for $N = 200$, and the agreement is very good for $N = 1000$. For $d = 50$ and $d = 100$, the results of the analytical model and the numerical results agree only well for $N = 1000$.

We conclude that, on the one hand, the agreement of the iteration process of the analytical model and the numerical results increases if the number of sample points increases. On the other hand, the agreement decreases if the number of dimensions increases.

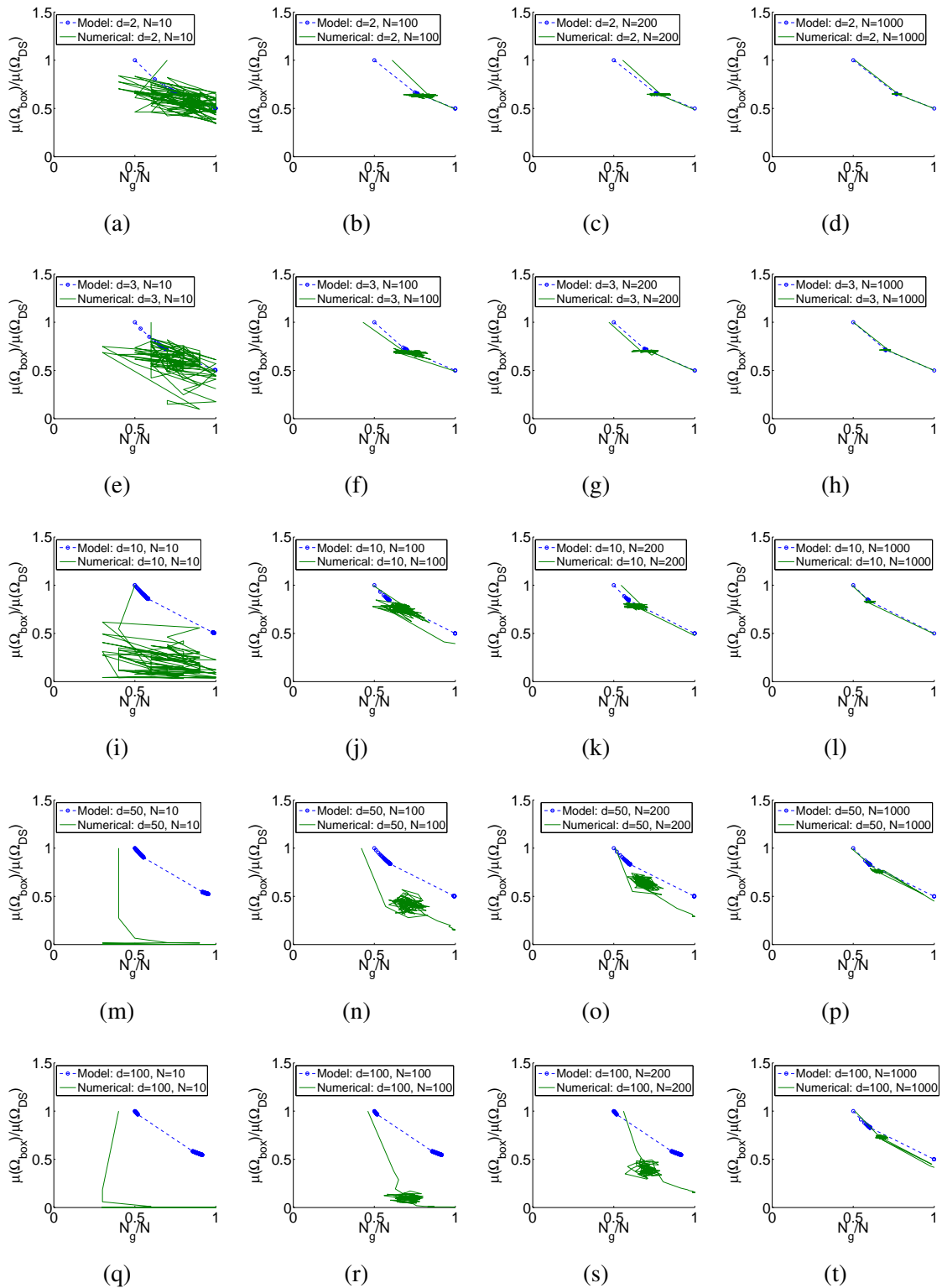


Figure B.1: Problem 3. $d = 2$ and $\gamma = 0.1$: (a) $N = 10$ (b) $N = 100$ (c) $N = 200$ (d) $N = 1000$. $d = 3$ and $\gamma = 0.1$: (e) $N = 10$ (f) $N = 100$ (g) $N = 200$ (h) $N = 1000$. $d = 10$ and $\gamma = 0.05$: (i) $N = 10$ (j) $N = 100$ (k) $N = 200$ (l) $N = 1000$. $d = 50$ and $\gamma = 0.01$: (m) $N = 10$ (n) $N = 100$ (o) $N = 200$ (p) $N = 1000$. $d = 100$ and $\gamma = 0.005$: (q) $N = 10$ (r) $N = 100$ (s) $N = 200$ (t) $N = 1000$.

Bibliography

- [1] N.M. Alexandrov, M.Y. Hussaini: *Multidisciplinary design optimization: state of the art*. Society for Industrial and Applied Mathematics, Philadelphia, 1997.
- [2] M. Allen, M. Rauli, K. Maute, D.M. Frangopol: *Reliability-based analysis and design optimization of electrostatically actuated MEMS*. *Computers and Structures*, 82(13–14):1007–1020 (2004).
- [3] J.C. Ascough II, T.R. Green, L. Ma, L.R. Ahjua: *Key criteria and selection of sensitivity analysis methods applied to natural resource models*. In *Proceedings of the International Congress on Modeling and Simulation (Modsim 2005)*, Modeling and Simulation Society of Australia and New Zealand, Melbourne, Australia, pp. 2463–2469, 2005.
- [4] T. Bayes, R. Price: *An essay towards solving a problem in the doctrine of chance. By the late rev. Mr. Bayes, communicated by Mr. Price, in a letter to John Canton, M.A. and F.R.S.* *Philosophical Transactions of the Royal Society of London*, 53(0):370–418 (1763).
- [5] B.J.C. Baxter: *The interpolation theory of radial basis functions*. PhD Thesis, Cambridge University, Cambridge, 1992.
- [6] M. Beer, M. Liebscher: *Designing robust structures. A nonlinear simulation based approach*. *Computers and Structures*, 86(10):1102–1122 (2008).
- [7] R.E. Bellman: *Adaptive control processes. A guide tour*. Princeton University Press, Princeton, 1961.
- [8] E.A. Bender: *An introduction to mathematical modeling*. Dover Publications, New York, 2000.
- [9] D.P. Bertsekas: *Constrained optimization and lagrange multiplier methods*. Athena Scientific, Belmont, Massachusetts, 1996.

- [10] H.-G. Beyer, B. Sendhoff: *Robust optimization. A comprehensive survey*. Computer Methods in Applied Mechanics and Engineering, 196(33–34):3190–3218 (2007).
- [11] L. Breiman: *Probability*. Society for Industrial and Applied Mathematics, Massachusetts, 1992.
- [12] C.J.C. Burges: *A tutorial on support vector machines for pattern recognition*. Data mining and Knowledge Discovery, 2(2):121–167 (1998).
- [13] C. Campbell, N. Cristianini, A. Smola: *Query learning with large margin classifiers*. In Proceedings of the Seventeenth International Conference of Machine Learning (ICML 2000), Morgan Kaufmann, San Francisco, USA, pp. 111–118, 2000.
- [14] G. Casella, R.L. Berger: *Statistical inference*. Duxbury Press, Pacific Grove, 2002.
- [15] G. Cauwenberghs, T. Poggio: *Incremental and decremental support vector machine learning*. In Proceedings of the Fourteenth Conference on Advances in Neural Information Processing Systems (NIPS 2001), Massachusetts Institute of Technology Press, Cambridge, MA, Vol. 13, pp. 409–415, 2001.
- [16] O.J. Centeno G.: *Finite element modeling of rubber bushing for crash simulation*. Master Thesis, Lund University, Lund, Sweden, 2009.
- [17] J. Cheng, M.J. Druzdzel: *Latin hypercube sampling in Bayesian networks*. In Proceedings of the Thirteenth International Florida Artificial Intelligence Research Society Conference (FLAIRS 2000), The Association for the Advancement of Artificial Intelligence Press, Menlo Park, CA, pp. 287–292, 2000.
- [18] N. Cristianini, J. Shawe-Taylor: *An introduction to support vector machines*. Cambridge University Press, Cambridge, 2000.
- [19] A.E. Eiben, J.E. Smith: *Introduction to evolutionary computing*. Natural Computing Series, Springer-Verlag, Berlin–Heidelberg, 2003.
- [20] U. Fayyad, G. Platetsky-Shapiro, P. Smyth: *From data mining to knowledge discovery in databases*. Artificial Intelligence Magazine, 17(3):37–54 (1996).

- [21] J. Fender, F. Duddeck, M. Zimmermann: *On the calibration of simplified vehicle crash models*. Accepted in Structural and Multidisciplinary Optimization, 2013.
- [22] G. Fung, S. Sandilya, R.B. Rao: *Rule extraction from linear support vector machines*. In Proceedings of the Eleventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD 2005), Association for Computing Machinery Press, New York, pp. 32–40, 2005.
- [23] C. Geiger, C. Kanzow: *Theorie und Numerik restringierter Optimierungsaufgaben*. Springer-Verlag, Berlin–Heidelberg, 2002.
- [24] L. Graff, H. Harbrecht, M. Zimmermann: *On the computation of solution spaces in high dimensions*. Preprint SPP1253-138, DFG Priority Program 1253, Universität Erlangen-Nürnberg, Germany, 2012.
- [25] L. Graff, F. Fender, H. Harbrecht, M. Zimmermann: *Key parameters in high-dimensional systems with uncertainty*. Preprint 2013-12, Mathematisches Institut, Universität Basel, Switzerland, 2013.
- [26] K. Grossenbacher: *Virtuelle Planung der Prozessrobustheit in der Blechumformung*. PhD Thesis, ETH Zürich, Switzerland, 2008.
- [27] B. Gruenbaum: *Convex polytopes*. Second edition, Springer-Verlag, New York, 2003.
- [28] S. Gunawan, P.Y. Papalambros: *A Bayesian approach to reliability-based optimization with incomplete information*. Journal of Mechanical Design, 128(4):909–919 (2006).
- [29] I. Hacking: *The emergence of probability*. Cambridge University Press, Cambridge, 1975.
- [30] O. Haeggstroem: *Finite Markov chains and algorithmic applications*. London Mathematical Society, Cambridge University Press, Cambridge, 2002.
- [31] E. Hansen: *Global optimization using interval analysis*. Marcel Dekker, New York, 1992.
- [32] E.M.T. Hendrix, C.J. Mecking, T.H.B. Hendriks: *Finding robust solutions for product design problems*. European Journal of Operational Research, 92(1):28–36 (1996).

- [33] B. Hohage, A. Förderer, G. Geißler, H. Müllerschön: *Global sensitivity analysis in industrial application with LS-OPT*. 9th LS-DYNA Forum, Bamberg, 2010.
- [34] M. Huang: *Vehicle crash mechanics*. Chemical Rubber Company Press, Boca Raton, Florida, 2002.
- [35] F. Jarre, J. Stoer: *Optimierung*. Springer-Verlag, Berlin–Heidelberg, 2004.
- [36] M. Kalsi, K. Hacker, K. Lewis: *A comprehensive robust design approach for decision trade-offs in complex systems design*. Journal of Mechanical Design, 123(1):1–10 (2001).
- [37] L. Kaufman, P. Rousseeuw: *Finding groups in data: an introduction to cluster analysis*. Wiley, New York, 1990.
- [38] H. Kerstan, W. Bartelheimer: *Innovative Prozesse und Methoden in der Funktionsauslegung. Auslegung für den Frontcrash*. Fahrzeugsicherheit, VDI-Berichte, 2078:185–196 (2009).
- [39] H.M. Kim, D.G. Rideout, P.Y. Papalambros, J.L. Stein: *Analytical Target Cascading in Automotive Vehicle Design*. Journal of Mechanical Design, 125:474–480 (2003).
- [40] D.M. King, B.J.C. Perera: *Sensitivity analysis for evaluating importance of variables used in an urban water supply planning model*. In Proceedings of the International Congress on Modeling and Simulation (Modsim 2007), Modeling and Simulation Society of Australia and New Zealand, Melbourne, Australia, pp. 2768–2774, 2007.
- [41] K.R. Koch: *Einführung in die Bayes-Statistik*. Springer-Verlag, Berlin–Heidelberg, 2000.
- [42] J.J. Korte, R.P. Weston, T.A. Zang: *Multidisciplinary optimization methods for preliminary design*. AGARD, Future aerospace technology in the service of the Alliance, Paris, Vol. 3, pp. 1–10, 1997.
- [43] H.W. Kuhn, A.W. Tucker: *Nonlinear programming*. In Proceedings of the second Berkeley Symposium, University of California Press, Berkeley, California, pp. 481–492, 1951.

- [44] F. Kursawe: *Evolution strategies – simple models of natural process?* Revue Internationale de Systemique, 7(5):627–642 (1993).
- [45] M. Lehar, M. Zimmermann: *An inexpensive estimate of failure probability for high-dimensional systems with uncertainty.* Structural Safety, 36–37:32–38 (2012).
- [46] D. Lim, Y.-S. Ong, Y. Jin, B. Sendhoff, B.S. Lee: *An adaptive inverse multi-objective robust evolutionary design optimization.* Genetic Programming and Evolvable Machines, 7(4):383–404 (2007).
- [47] R. Lupton: *Statistics in theory and practice.* Princeton University Press, Princeton, 1993.
- [48] J. Marczyk: *Stochastic multidisciplinary improvement: beyond optimization.* American Institute of Aeronautics and Astronautics, AIAA 2000–4929, 2000.
- [49] M. Martin: *On-line support vector machines for function approximation.* Technical report LSI-02-11-R, Software Department, Universitat Politecnica de Catalunya, Spain, 2002.
- [50] M.D. McKay, R.J. Beckman, W.J. Conover: *A comparison of three methods for selecting values of input variables in the analysis of output from a computer code.* Technometrics, 21(2):239–245 (1979).
- [51] M. Medina, N. Carrasquero, J. Moreno: *Estrategias evolutivas celulares para la optimizacion de funciones.* Sexto Congresso Iberoamericano de Inteligencia Artificial (IBERAMIA 1998), Lisboa, Portugal, 1998.
- [52] J. Mercer: *Functions of positive and negative type and their connection with the theory of integral equations.* Philosophical Transactions of the Royal Society A, 209:415–446 (1909).
- [53] C.A. Micchelli: *Interpolation of scattered data: distance matrices and conditionally positive definite functions.* Constructive approximation, 2:11–22 (1986).
- [54] R. Moore: *Methods and applications of interval analysis.* Studies in Applied Mathematics, Philadelphia, 1979.
- [55] Z.P. Mourelatos, J. Liang: *A Methodology for trading-off performance and robustness under uncertainty.* Journal of Mechanical Design, 128(4):1195–1204 (2006).

- [56] A. Neumaier: *Interval methods for systems of equations*. Cambridge University Press, Cambridge, 1990.
- [57] W.L. Oberkampf: *Uncertainty quantification using evidence theory*. Stanford University, Stanford, 2005.
- [58] J.M. Pangilinan, G.K. Janssens, A. Caris: *Sensitivity analysis of a genetic algorithm for a competitive facility location problem*. Innovations and Advanced Techniques in Systems, Computing Sciences and Software Engineering, Springer-Verlag, Bridgeport, USA, pp. 266–271, 2008.
- [59] S. Pannier, W. Graf: *Sensitivity measures for fuzzy numbers based on artificial neural networks*. In Proceedings of the International Conference on Applications of Statistics and Probability in Civil Engineering (ICASP), Taylor & Francis, London, pp. 139–140, 2011.
- [60] S. Pannier, W. Graf, M. Kaliske, K. Grossenbacher, M. Ganser, A. Lipp, M. Liebscher, H. Müllerschön: *Affecting reliability of deep drawing processes in early design stages*. In Proceedings of the 8th Association for Structural and Multidisciplinary Optimization in the UK (ASMO UK)/ISSMO Conference Engineering Design Optimization, London, pp. 57–58, 2010.
- [61] G.-J. Park, T.-H. Lee, K.H. Lee, K.-H. Hwang: *Robust design. An overview*. American Institute of Aeronautics and Astronautics Journal, 44(1):181–191 (2006).
- [62] S.N. Parragh, K.F. Doerner, X. Gandibleux, R.F. Hartl: *Approaching the pareto-frontier of the multi-objective dial-a-ride problem*. International Workshop on Vehicle Routing and Transportation, Georgia, 2007.
- [63] M.S. Phadke: *Quality engineering using robust design*. Prentice Hall, Englewood Cliffs, New Jersey, 1989.
- [64] C. Piret: *Analytical and numerical advances in radial basis functions*. PhD Thesis, Faculty of the Graduate School of the University of Colorado, USA, 2007.
- [65] R. Reinhardt, A. Hoffmann, T. Gerlach: *Nichtlineare Optimierung*. Springer Spektrum, Springer-Verlag, Berlin–Heidelberg, 2013.

- [66] C.M. Rocco, J.A. Moreno, N. Carrasquero: *Robust design using a hybrid-cellular-evolutionary and interval-arithmetic approach: a reliability application*. Reliability Engineering and System Safety, 79:149–159 (2003).
- [67] A. Saltelli, K. Chan, E.M. Scott: *Sensitivity analysis*. Wiley, New York, 2000.
- [68] A. Sieber: *Parameterstudien und Unsicherheitsanalysen mit dem Einzugsgebietsmodell TAC*. Diploma Thesis, University of Freiburg i. Br., Germany, 2003.
- [69] K. Siebertz, D. van Bebbber, T. Hochkirchen: *Statistische Versuchsplanung: Design of Experiments (DoE)*. Springer-Verlag, Berlin–Heidelberg, 2010.
- [70] I.M. Sobol': *Sensitivity estimates for nonlinear mathematical models*. Mathematical Modeling and Computational Experiment, 1(4):407–414 (1993).
- [71] I.M. Sobol', S.S. Kucherenko: *Global sensitivity indices for nonlinear mathematical models. Review*. Wilmott Magazine, 2:56–61 (2005).
- [72] R. Stocki: *A method to improve reliability using optimal Latin hypercube sampling*. Polish Academy of Sciences, Poland, 2006.
- [73] R. Storn, K. Price: *Differential evolution. A simple and efficient heuristic for global optimization over continuous spaces*. Journal of Global Optimization, 11:341–359 (1997).
- [74] B. Sudret: *Global sensitivity analysis using polynomial chaos expansions*. Reliability Engineering and System Safety, 93(7):964–979 (2007).
- [75] L.P. Swiler, A.A. Giunta: *Aleatory and epistemic uncertainty quantification for engineering applications*. Sandia Technical Report, SAND2007–2670C, 2007.
- [76] G. Taguchi, E. Elsayed, T. Hsiang: *Quality engineering in production systems*. McGraw-Hill, New York, 1989.
- [77] V.N. Vapnik: *The nature of statistical learning theory*. Second edition, Springer-Verlag, New York, 1999.
- [78] S. Wagner: *Global sensitivity analysis of predictor models in software engineering*. In Proceedings of the Third International Workshop on Predictor Models in Software Engineering (PROMISE 2007), IEEE Computer Society Washington, DC, USA, 2007.

- [79] C. Westerberg: *Finite element simulation of crash testing of self-piercing rivet joints, peel specimen*. Master Thesis, Lund University, Sweden, 2002.
- [80] S.S. Wilks: *Mathematical statistics*. J. Wiley and Sons, New York–London, 1962.
- [81] J. Will: *Robust design optimization in forming process simulation*. ANSYS Conference & 25th CADFEM Meeting of the Users 2007, November 21–23, Congress Center Dresden, Germany, 2007.
- [82] J. Will, K. Grossenbacher: *Using robustness and sensitivity evaluation for setting up a reliable basement for robust design optimization*. Forming Technology Forum 2007, Application of Stochastic and Optimization Methods 14th–15th March, IVP, ETH Zürich, Switzerland, 2007.
- [83] W. Witteman: *Adaptive frontal structure design to achieve optimal deceleration pulses*. In Proceedings of the 19th International Technical Conference on the Enhanced Safety of Vehicles, United States, Washington DC, 2005.
- [84] S. Wolfram: *Cellular automata as models of complexity*. Nature, 311:419–424 (1984).
- [85] L.A. Zadeh: *Fuzzy sets*. Information and Control, 8:338–353 (1965).
- [86] Y.G. Zhao, T. Ono: *A general procedure for first / second-order reliability method (FORM / SORM)*. Structural Safety, 21(2):95–112 (1999).
- [87] M. Zimmermann, J.E. von Hoessle: *Computing solution spaces for robust design*. International Journal for Numerical Methods in Engineering, 94(3):290–307 (2013).
- [88] M. Zimmermann: *Vehicle front crash design accounting for uncertainties*. In Proceedings of the FISITA 2012 World Automotive Congress, Lecture Notes in Electrical Engineering, Springer-Verlag, Berlin-Heidelberg, Vol. 197, pp. 83–89, 2013.