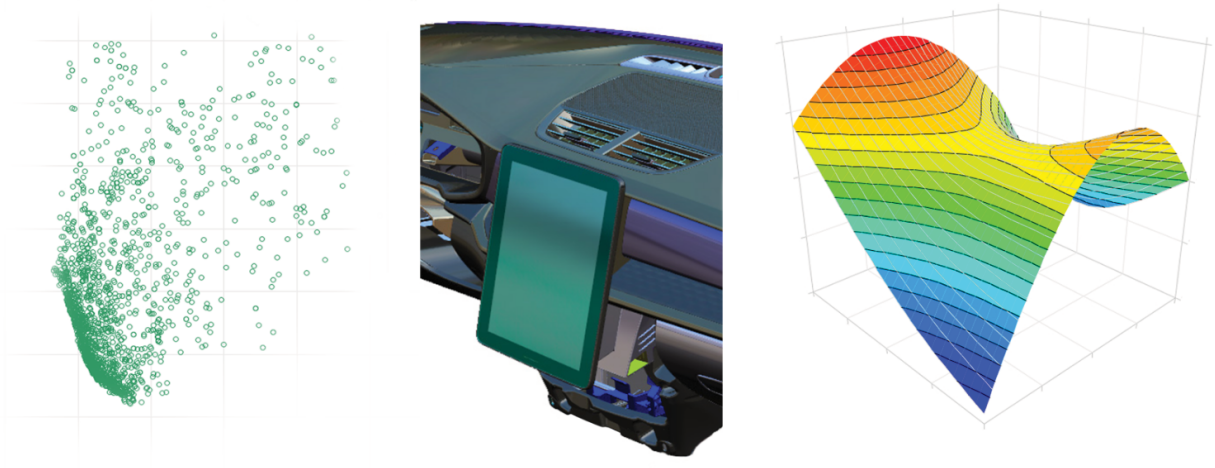
CHALMERS
UNIVERSITY OF TECHNOLOGY



Development of a Multidisciplinary Design Optimization Process for Automotive Components

Metamodel-Based Optimization of a Center Stack Display Bracket

Master's Thesis in Product Development

PHILIP BLOM
FELIX SÖDERLIND ENGLUND

Department of Industrial and Material Science
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2021

Master's thesis 2021

Development of a Multidisciplinary Design Optimization Process for Automotive Components

Metamodel-Based Optimization of a Center Stack Display Bracket

PHILIP BLOM
FELIX SÖDERLIND ENGLUND

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Industrial and Material Science
Division of Product Development
Chalmers University of Technology
Gothenburg, Sweden 2021

Development of a Multidisciplinary Design Optimization Process for Automotive Components
Metamodel-Based Optimization of a Center Stack Display Bracket
PHILIP BLOM
FELIX SÖDERLIND ENGLUND

Industrial supervisor: Halil Salifov, Volvo Car Corporation
Supervisor: Kanishk Bhadani, Department of Industrial and Material Science
Examiner: Gauti Asbjörnsson, Department of Industrial and Material Science

Master's Thesis 2021
Department of Industrial and Material Science
Division of Product Development
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Collage of (from the left) MOGA-II scatter plot, Center Stack Display CAD model, and metamodel surface.

Typeset in LaTeX
Printed by Chalmers Reproservice
Gothenburg, Sweden 2021

Abstract

The design of automotive components requires balancing requirements from multiple engineering disciplines to arrive at a satisfactory design in increasingly short development cycles. Multidisciplinary Design Optimization (MDO) can facilitate learning and discovery of high-performing designs with conflicting requirements in a time-efficient manner.

This thesis presents an MDO process for interior structural components subject to crash and noise, vibration, and harshness (NVH) requirements. Internal needs at Volvo Car Corporation for such a method are identified using semi-structured interviews. Relying on previous findings in the field of metamodel-based MDO, a sizing optimization workflow is developed in the software modeFRONTIER using a simple geometry subject to a head impact load case and a modal requirement. The presented process involves variable selection by parameterization and statistical variable screening, sampling using an experimental Uniform Latin Hypercube design, metamodeling with multiple metamodel algorithms, and subsequent global, nongradient optimization with the genetic algorithm MOGA-II.

The MDO process is applied to a Center Stack Display bracket design problem subject to three head impact load cases and an NVH-related modal requirement. Significant performance increases for critical objectives are shown compared to a pre-optimized design; however, the optimization does not reach all performance targets, thus prompting a discussion about process limitations, including the need for global information and implications of geometry lock-in. Organizational, societal, ethical, and ecological aspects related to MDO in the automotive industry are briefly considered. Finally, recommendations regarding future work are given, including the implementation of finite element mesh morphing to allow for more complex design variables and method improvements for variable screening.

Keywords: Multidisciplinary Design Optimization, Variable Screening, Metamodel-Based Optimization, Design of Experiments, Latin Hypercube Sampling

Acknowledgements

This work could not have been possible without the support of several people who kindly offered their time during the duration of our thesis. First of all, we would like to thank our supervisor, Kanishk Bhadani, for his continuous engagement and valuable guidance throughout the entire duration of the thesis. We would also like to thank our thesis examiner Gauti Asbjörnsson. His mentoring and academic input kept us on the right track during crucial stages of the project.

We want to express our gratitude to Halil Salifov, our Volvo Car Corporation supervisor, for his assistance, continuous support, and for providing the proper prerequisites for us to succeed. Moreover, we would like to thank Stefan Pauli at VCC for his kindness in spending many hours teaching and aiding us with the CAE software stack.

Furthermore, we would like to thank all the CAE experts, needs assessment interviewees, and other engineers at Volvo Car Corporation for their input and engagement in our work. We want to thank Harald Hasselblad for his nuanced reflections, which helped form our views of the organizational considerations concerning the implementation of Multidisciplinary Design Optimization in the automotive industry. We also thank Alexander Govik for his guidance in modeFRONTIER during the initial stages of the work.

Lastly, we would like to express our gratitude to our family and friends for their support and encouragement during the duration of our thesis.

Philip Blom & Felix Söderlind Englund, Gothenburg, June 2021

Table of Contents

Notation

AAO	All-at-once
ASO	Asymmetric subspace optimization
ATC	Analytical target cascading
BLISS	Bilevel integrated system synthesis
CAD	Computer-aided design
CAE	Computer-aided engineering
CFD	Computational fluid dynamics
CO	Collaborative optimization
CSD	Center stack display
CSDB	Center stack display bracket
CSSO	Concurrent subspace optimization
DG	Development geometry
DoE	Design of experiments
DV	Design variable
ECO	Enhanced collaborative optimization
EPD	Exact penalty decomposition
FEM	Finite element method
GA	Genetic algorithm
IDF	Individual discipline feasible
IP	Instrument panel
IPD	Inexact penalty decomposition
LHS	Latin hypercube sampling
LP	Linear programming
MAE	Mean absolute error
MAUD	Modular analysis and unified derivatives
MDA	Multidisciplinary analysis
MDF	Multidisciplinary feasible
MDO	Multidisciplinary design optimization
MDOIS	Multidisciplinary design optimization of independent subspaces
MOGA	Multi-objective genetic algorithm
MOO	Multi-objective optimization
MRE	Mean relative error
NLP	Nonlinear programming
NVH	Noise, vibration, and harshness
P-B	Plackett-Burman
QSD	Quasi-separable decomposition
RBF	Radial basis function
SAND	Simultaneous analysis and design
SOO	Single-objective optimization
SSANOVA	Smoothing spline analysis of variance
ULH	Uniform latin hypercube
VCC	Volvo Car Corporation
XDSM	Extended design structure matrix

1
Introduction

The Master's Thesis is carried out at Volvo Car Corporation (VCC) with guidance from the Department of Industrial and Material Science, Chalmers University of Technology. This chapter covers a brief background to contextualize the work. Furthermore, the chapter presents the purpose and aim, scope, limitations, and research questions. The final section in the chapter describes the structure of the thesis report in outline.

1.1 Background

A brief company background is presented in addition to an introduction to optimization in product development. Subsequently, multidisciplinary design optimization and the verification design problem are introduced.

1.1.1 Volvo Car Corporation

Volvo Car Corporation (VCC) is an automobile manufacturer founded in 1927 in Gothenburg, Sweden. Initially owned by AB Svenska Kullagerfabriken (SKF) as a subsidiary company, it is under the ownership of Zhejiang Geely Holding as of 2010. VCC is a company with a global footprint employing around 40,000 full-time employees as of December 2020. Its head office with product development and supporting functions is located in Gothenburg, Sweden. Manufacturing and assembly are likewise located in Sweden but also in Belgium, Malaysia, India, China, and the US [1]. Focusing on premium-segment car models of sedans, versatile estates, and SUVs, Volvo Cars has in 2020 sold 661,713 cars and reports annual revenues of 262,833 MSEK [2].

1.1.2 Optimization in the Product Development Process

As software capabilities are continuously improving, companies can bridge the gap between physical and virtual testing, enabling computer-aided engineering (CAE) methods and tools to drive the product development process. Ulrich and Eppinger [3] describe the traditional product development process as depicted in Figure 1.1. The figure shows the phase of interest for this thesis.

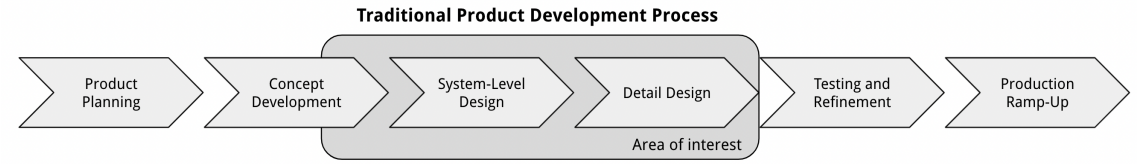


Figure 1.1: A traditional product development process [3] with the phases of interest marked.

Compared to the traditional product development approach of having engineers develop hardware for testing, a CAE-driven product development process allows for the integration of, e.g., structural optimization procedures such as size-, shape-, and topology optimization. However, depending on when in the product development process CAE is implemented, the prerequisites may differ. While there typically are fewer restrictions on the design space early on in the product development process, simplified load cases are often required, imposing limitations on the accuracy of optimization results. On the other hand, CAE in the later stages of the product development process allows for optimization using a more established design space and well-defined load cases. However, with a detailed component, the design space gets narrower, complicating the CAE's implementation to yield significant improvement. In this work, the use of size optimization is investigated to fine-tune a component in the later stages of the product development process.

1.1.3 Multidisciplinary Design Optimization

Global competition, regulatory pressures, and ambitious visions by companies contribute to increased demands on automotive engineering teams to create complex structural systems, often with conflicting requirements. The development effort of automotive components typically involves several disciplines, e.g., noise, vibration, and harshness (NVH), crashworthiness, durability, and solidity. Multidisciplinary Design Optimization (MDO) aims to coordinate these disciplines effectively to arrive at a design that considers the interactions between disciplines, preferably by exploiting synergism [4], to arrive at a significantly improved, if not optimal, design. Although MDO simultaneously considers several disciplines with some degree of automation, MDO may be best viewed as a tool that helps engineers explore a given design space. It should be interpreted as such, as opposed to an automated process that arrives at a complete design without human intervention [5], [6]. Not all knowledge can be made explicit and included in a set of models, nor is the interpretation of the results without ambiguity, which is why interactions between engineers and the MDO process as a tool are essential.

MDO was initially developed in the literature on structural optimization during the early 1970s. After seminal work utilizing Nonlinear Programming (NLP) [7], MDO concepts were applied in aerospace applications with mass reduction being the primary focus [8]. More recently, MDO has seen use in the automotive sector, which differs in engineering focus to aerospace, the former focusing primarily on crashworthiness and the latter focusing on fatigue with considerable elastic behav-

ior (e.g., in the wings of an airplane) [6], [9]. In general, methods cannot be readily transferred from one industry to another or from one type of engineering problem to another because the particular disciplines and their interrelationships require different MDO approaches. Therefore, a study is needed before choosing or synthesizing a method for a particular application type.

1.1.4 Center Stack Display Bracket Requirement Conflict

During the development of an MDO process, a simplified geometry is used. However, a real design problem with conflicting requirements is used to verify the approach. The developed process is verified using the Center Stack Display Bracket (CSDB), which is an interior component of the car belonging to the Instrument Panel (IP), see Figure 1.2.

All car parts depicted are from a commercially available car model.

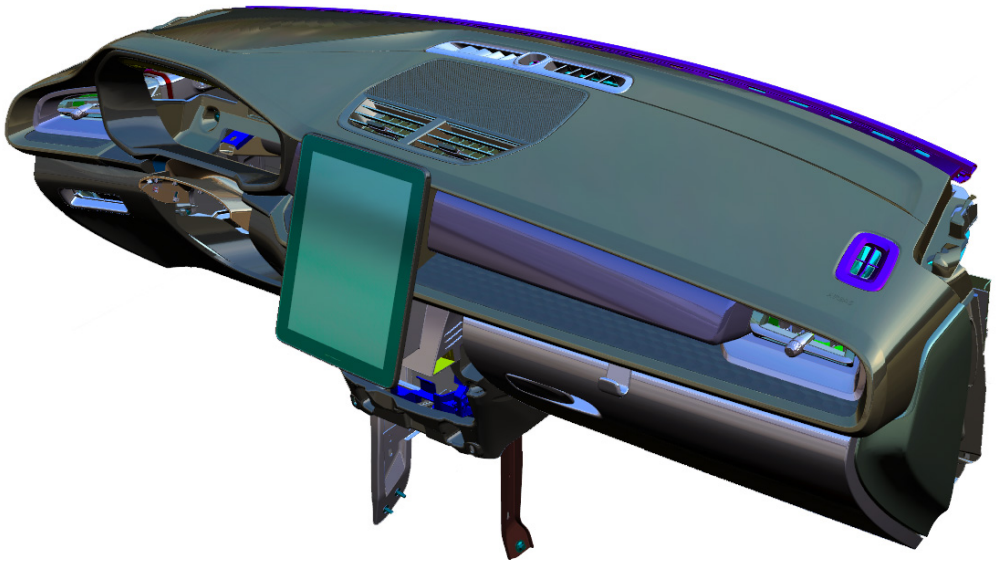


Figure 1.2: CAD model of a recent VCC car model. Note the CSD in the center.

To implement a Center Stack Display (CSD), as shown in Figure 1.3, multiple disciplines have to be taken into account to guarantee safety and comfort for the customer. Legal requirements and internal targets exist, dictating required performance for head impact safety during a crash. Also, internal targets from NVH aim to increase the frequency of vibration at which resonance occurs. Furthermore, the two disciplines impose potentially contradictory requirements on the component, e.g., decreased structural stiffness generally improves head impact outcomes while worsening modal outcomes. To assure that these conflicting requirements are met and that an appropriate balance is found, geometric features of the bracket are optimized, taking multiple disciplines into account simultaneously in the optimization process.

The CSDB is located between the CSD and the subframe, see Figure 1.3.

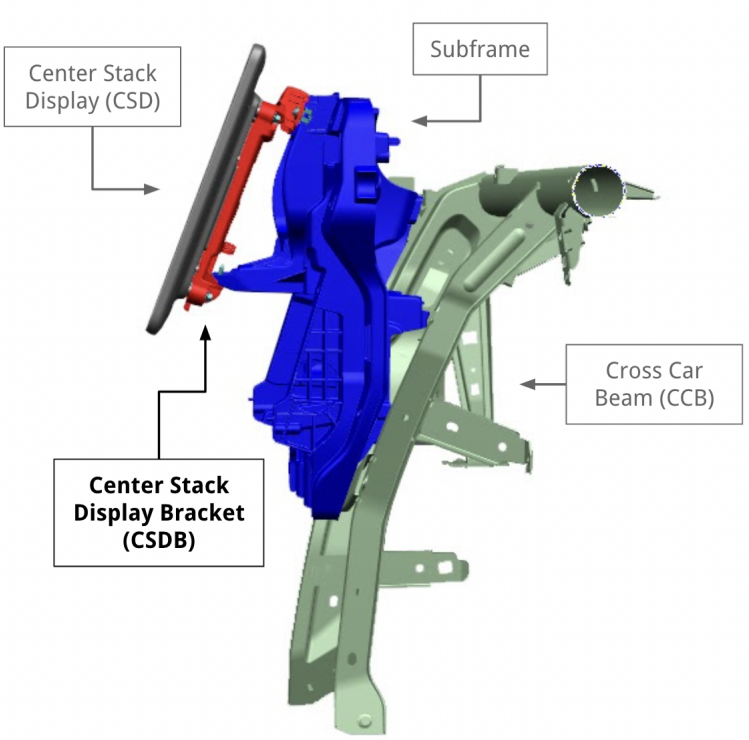


Figure 1.3: Side view of a CSDB and its associated components from a recent VCC model. Note that this is not the exact geometry used in this work.

Current design work relies on communication among discipline experts who test and optimize components within their respective discipline, and by collaborating and iterating the process until a final design is reached. This is a challenging and potentially time-inefficient process. The thesis work investigates and develops a multidisciplinary optimization process to be used by the CAE team of the Interior Room Integration department at VCC to alleviate this situation.

1.2 Purpose and Aim

The purpose of the thesis is to explore and develop an MDO methodology to be used for solving design problems exhibiting conflicting requirements using commercial software. It aims to establish a practical process to be used in VCC's development of interior, structural components. The considered disciplines in this study are NVH and crashworthiness — specifically, head impact accelerations and eigenfrequency at the first mode. A CSDB is used as a case study to verify that the suggested MDO process can be used for similar problems in the future.

1.3 Limitations

The delimitations and limitations restricting the scope of the thesis are as follows:

- The disciplines under consideration are NVH and crashworthiness, however, the MDO process is to be developed to allow for other disciplines as well.
- Software used are ANSA for pre-processing, MSC Nastran for modal analysis, LS-DYNA for crash analysis, META for post-processing, and modeFRONTIER for MDO workflow creation.
- Programming languages used are Bash and Python.
- The choice of material for the component under optimization is treated as given and therefore material selection is not considered.
- The interfaces between the CSDB and other components are fixed.
- Cost models are not included.
- The optimization procedure is limited to optimize the geometry of a pre-existing, mature concept.
- Optimization is restricted to sizing optimization, but the MDO process is designed to allow for other optimization types as well.
- The work aims to integrate MDO into a practical workflow. Therefore, mathematical features of particular MDO methods and optimization algorithms will not be investigated in any greater detail.
- CAD interpretation of optimization results is not performed.
- The work is carried out by two students during 20 weeks.

1.4 Research Questions

The project intends to investigate the following questions:

1. What needs exist for an MDO process within the application area, i.e., optimization of interior structural components subject to crash and NVH requirements?
2. How can an MDO process be developed and implemented in selected software?
3. How can the developed MDO process be verified?
4. What are the organizational and sustainability implications of employing MDO in automotive engineering?

1.5 Thesis Outline

In Chapter 2, theory relevant to the work is presented, e.g., in areas of statistics, optimization, Design of Experiments, metamodeling, and MDO. Chapter 3 describes the methodology of the thesis work. Subsequently, Chapter 4 reports on empirical findings from a pre-study and demonstrates the development of an MDO process using a simplified geometry. To verify that the MDO process developed in Chapter 4 can be applied to a real design problem, Chapter 5 presents a multidisciplinary optimization of the CSDB and evaluates two problem formulation variations as well as software implementation in MDO platform modeFRONTIER by ESTECO SpA. Penultimately, the discussion in Chapter 6 places particular emphasis on the fourth research question, see Section 1.4, as well as organizational implications and limitations of the thesis and proposed process. Recommendations for future work are also given. Finally, Chapter 7 concludes significant findings of the work.

2
Theory

This chapter introduces the reader to relevant theoretical concepts that are used in this thesis. Topics include: basics of modal and crash analysis, an introduction to design optimization, statistical theory, Design of Experiments, metamodels, and finally, multidisciplinary design optimization.

2.1 Modal and Crash Analysis

The theoretical basis of the engineering disciplines used in the problem introduced in Section 1.1.4 is briefly presented below. Both analyses utilize Finite Element (FE) models with which theory the readers are assumed to be familiar.

2.1.1 Modal Analysis

Modal analysis is defined by Fu and He [10] as *"the process of determining the inherent dynamic characteristics of a system in forms of natural frequencies, damping factors, and mode shapes, and using them to formulate a mathematical model for its dynamic behavior"*. Natural frequencies, also called eigenfrequencies, are frequencies of vibration at which the system's amplitude asymptotically increases to infinity. Each natural frequency has an associated mode shape that describes how the system deforms, although the amplitude at a particular time step is unknown. Fundamentally, natural frequencies and modes are determined by a system's physical properties [10]. Any physical system or structure can be modeled as a set of springs, masses, and dampers. If a system is modeled as an undamped spring-mass system, its natural frequency f can be described as:

$$f = \frac{1}{2\pi}\sqrt{\frac{k}{m}} \text{ [Hz]} \tag{2.1}$$

where k is the spring constant, i.e., its stiffness, and m is the mass. Increasing a system's stiffness increases its natural frequency while increasing its mass decreases its natural frequency. The stiffness of a system in a particular load case depends on its material properties (i.e., its Young's modulus) and its geometry. The eigenfrequency of the first mode is often of interest in modal analysis. The significance of this property in the automotive industry is that it is a key performance metric in NVH.

2.1.2 Crash Analysis

Crash analysis, or crashworthiness, is defined by Alkbir et al. [11] as *"the capability of a vehicle to protect its occupants and passengers from serious injury and harm or death in case of accidents or sudden impacts of a specified magnitude"*. During crash events, automotive components can undergo significant plastic deformations, absorbing large amounts of energy [12]. Complex interactions between parts and numerical instabilities can lead to simulated vehicle crashes exhibiting nonlinear and noisy responses, e.g., for peak acceleration [13].

Two crash metrics of interest in automotive crash analysis are the maximum head acceleration within a 3-millisecond interval, called clip3ms or clip3m, according to regulation [14], as well as the maximum head acceleration, see Figure 2.1.

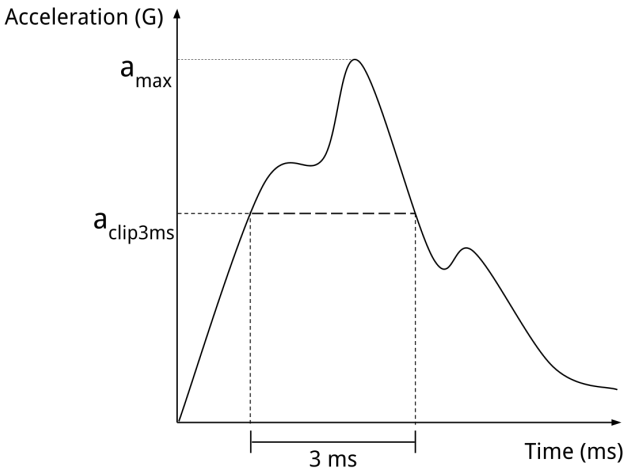


Figure 2.1: Schematic representation of acceleration curve under head impact. Maximum acceleration a_{max} and maximum acceleration under a 3 milliseconds time interval $a_{clip3ms}$ are denoted.

2.2 Introduction to Design Optimization

Since multidisciplinary design optimization relies on a general understanding of optimization, a brief background of design optimization follows.

The purpose of product development processes is to design objects which best satisfy a particular need, subject to imposed limitations [15]. In outline, this process traditionally proceeds as follows (with iterations omitted):

1. Problem definition.
2. Concept synthesis.
3. Analysis of proposed concepts.
4. Selection of the best concept.
5. Testing against needs.

The engineers involved in this process utilize their knowledge and experience in the application area at each stage to guide their choices. However, intuitive judgment and trial-and-error can lead to costly outcomes due to the resulting product suboptimality and lengthy development cycles. Design optimization addresses these inefficiencies by introducing a formalized approach to selecting the "best" design within limited means [15]. It is also used to explore the design space and learn about the system under optimization.

After having defined the system and its boundaries, an optimization can be mathematically formulated in so called *negative null form* as:

$$\begin{aligned} &\min\ f(\mathbf{x}, \mathbf{p}) \\ &\text{subject to } \mathbf{h}(\mathbf{x}, \mathbf{p}) = \mathbf{0} \\ &\qquad\quad \mathbf{g}(\mathbf{x}, \mathbf{p}) \leq \mathbf{0} \end{aligned} \tag{2.2}$$

Where the *objective function* $f(\mathbf{x}, \mathbf{p})$ of *design variable* vector $\mathbf{x}$ and parameter vector $\mathbf{p}$ is minimized while respecting *equality* and *inequality constraints* $\mathbf{h}(\mathbf{x}, \mathbf{p})$ and $\mathbf{g}(\mathbf{x}, \mathbf{p})$ respectively. The objective function $f(\mathbf{x}, \mathbf{p})$ is what describes the performance of the design, but it should be noted that it is not always a strictly mathematical function, or functions, but may also be a system of equations or a computer-based procedure [15]. Design variables $\mathbf{x}$ are the inputs that are open for modification while parameters $\mathbf{p}$ are assumed fixed, e.g., physical constants or load cases. Constraints $\mathbf{h}(\mathbf{x}, \mathbf{p})$ and $\mathbf{g}(\mathbf{x}, \mathbf{p})$ are imposed by governing equations and by the designer according to the nature of the engineering problem, e.g., a maximum allowed stress of a structure when mass is minimized, as is common in structural optimization [16]. If the objective and constraint functions are linear functions it is a *linear programming* (LP) problem. If any functions are nonlinear it is a *nonlinear programming* (NLP) problem [17].

Once the problem is formulated, models are chosen to act as bases for the optimization. Models can be fundamental (e.g., mechanics, fluid dynamics), numerical (e.g., FEM, CFD) or data-driven (e.g., Design of Experiments) depending on the needs and limitations of the application. The setup of models, and their antecedent formulation require care to ensure that they represent reality to an acceptably accurate degree. The translation to mathematical formulations can otherwise bring suboptimality in reality even if the optimization yields the mathematically optimal solution [18]. Algorithms are next deployed to solve the optimization problem, i.e., finding the set of optimizers $\mathbf{x}^*$ that yields the optimum f^*.

2.2.1 Optimization Algorithms

In theory, model analysis alone can solve optimization problems, but this is often only the case in simple problems rarely found in engineering contexts. Therefore, optimization practitioners typically employ numerical methods for NLPs after studying the formulation characteristics (e.g., monotonicity and redundancy) [15]. Since no algorithm produces reliable results efficiently for every circumstance, different

computational techniques are used depending on the nature of the optimization problem. In general, the types of optimization algorithms can be divided into *local* and *global algorithms* as well as *gradient* and *nongradient* (also called *gradient-free*) algorithms [19].

Local Optimization Algorithms

Local optimization algorithms search for the optimum from a set starting point and stop the search once it finds a local optimum. Local algorithms are also typically gradient-based in that they use local first and second derivatives to guide the next step in the iteration. However, unless the response is simple (i.e., without multiple extrema), it is uncertain whether or not the point that the algorithm converges upon is also the global optimum. The use of multiple starting points can be a way of circumventing this issue. Gradient-based, local algorithms scale well with the number of design variables due to their computational efficiency, especially if analytic derivatives are used instead of finite-difference approximations [19], [20].

Global Optimization Algorithms

Global optimization algorithms search a larger set of the design space than local algorithms. This family of algorithms is often stochastic (although deterministic methods exist [21]) in that they utilize randomness to select starting points and subsequent evaluation points. Global algorithms are often also nongradient and, in that case, do not require differentiable functions like gradient-based approaches. Nongradient optimization algorithms are typically employed when the responses are multimodal, noisy, or discontinuous. The number of iterations required scales poorly with the number of design variables when compared with gradient-based algorithms [19]. The reader is directed to [22] and [23] for further reading about nongradient optimization algorithms not presented here.

2.2.2 Multi-Objective Optimization

There are many cases where there either is not a single objective function bound by constraints but rather multiple objective functions or that the designer's understanding of the system would be enhanced by exploring trade-offs between different performance criteria before selecting a design. The formulation of min $f(\mathbf{x}, \mathbf{p})$ as stated in Eq. 2.2 is then the following for the objective functions:

$$\min f(\mathbf{x}, \mathbf{p}) = \begin{bmatrix} f_1(\mathbf{x}, \mathbf{p}) \\ f_2(\mathbf{x}, \mathbf{p}) \\ \vdots \\ f_n(\mathbf{x}, \mathbf{p}) \end{bmatrix} \tag{2.3}$$

where n is the number of objective functions in the general multi-objective optimization (MOO) problem. However, typically, only 2 or 3 objective functions are used. Unless the design variables $\mathbf{x}$ are separable, i.e., the objectives are functions of non-overlapping design variable sets, a trade-off is required to select a design for

systems with conflicting requirements.

A design is said to *dominate* another design if the former outperforms the latter in all objectives. A *nondominated* design is said to be *Pareto optimal* and part of the *Pareto set*, which in turn forms the curve called the *Pareto frontier* or *Pareto front* [15], see Figure 2.2.

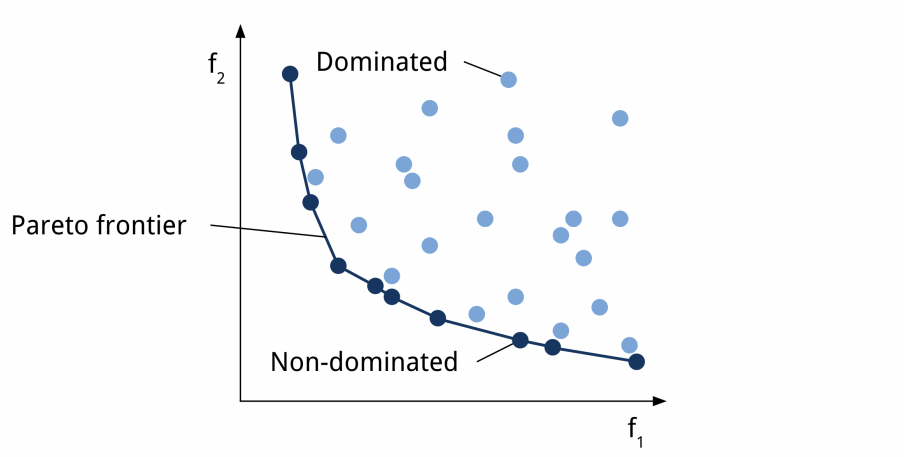


Figure 2.2: Pareto optimality in a min-min problem, i.e., a problem where both objectives are minimized. Points represent evaluations in the design space with the objective functions f_1 and f_2.

A Pareto optimal design may then be chosen from the Pareto set based on the relative weight of f_1 to f_2. The weighting can be expressed informally as a preference for a particular point on the curve that is considered a "good" trade-off between the objectives or formally by using weighted sum techniques that more exactly express preferences about the relative importance of the objectives. Changing the objective units may also be prudent if the relative magnitude of the objectives is large, but this is to be separated from the preference weighting [24].

Multi-Objective Genetic Algorithms

Genetic algorithms are a family of stochastic optimization algorithms and utilize the same principle as Darwin's theory of evolution by natural selection, i.e., where a population of individuals mutates over generations. The best-adapted individuals in the population are selected and influence subsequent generations. In optimization, genetic algorithms use this principle of genetics and evolution to converge on an optimal design for a particular problem formulation [25]. See Figure 2.3 for a general genetic algorithm process.

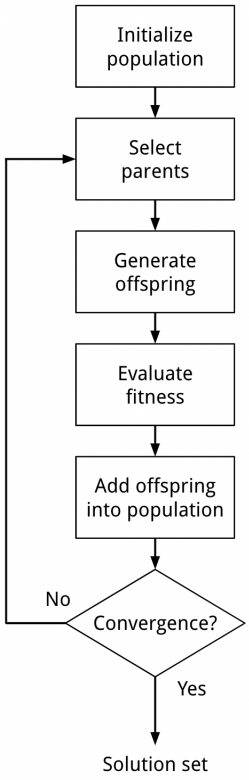


Figure 2.3: A general genetic algorithm process [26].

While there are many variants of genetic algorithms, this work is limited to the Multi-Objective Genetic Algorithm (MOGA-II) which is a non-gradient, global algorithm that operates by reproduction and elitism. Furthermore, MOGA-II uses four different operators for reproduction; classical crossover, directional crossover, mutation, and selection [27], see Figure 2.4.

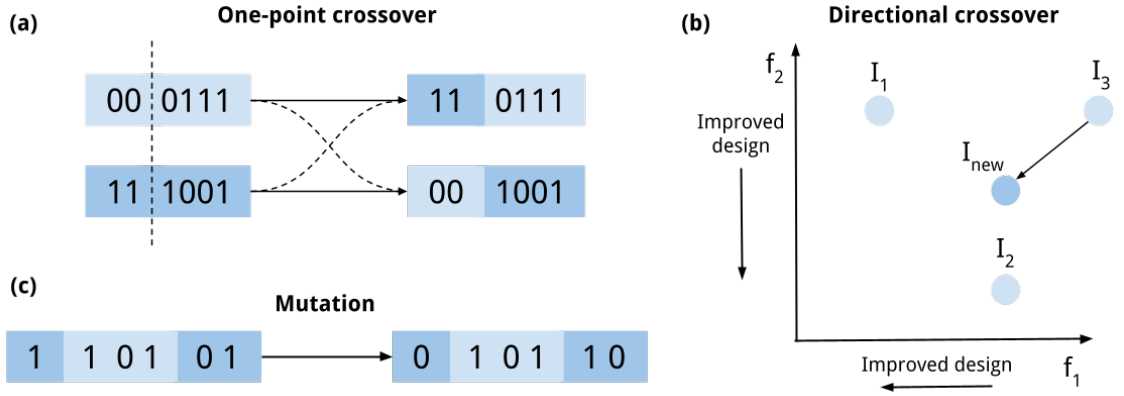


Figure 2.4: Schematic example of (a) one-point crossover where a portion of the genetic material is exchanged between two parent variables, (b) directional crossover where fitness values of two reference individuals (I_1 and I_2) from a population are used to generate a third one I_3, and (c) mutation which induces diversity between generations.

Using these operators, MOGA-II generates and selects superior design points, over generations pushing the design points toward the desired direction of the optimization problem, see Figure 2.5.

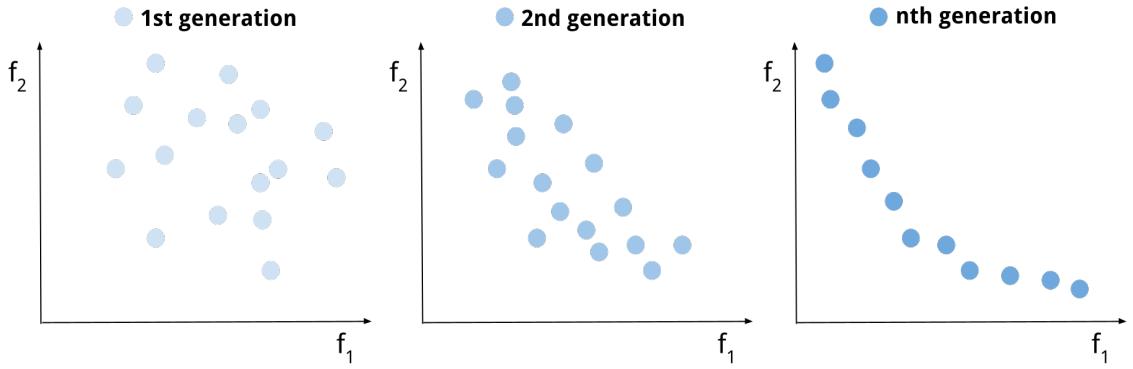


Figure 2.5: Genetic algorithm behavior over generations $1, 2, \ldots, n$ in a multi-objective approach with objectives f_1 and f_2 to be minimized.

The MOGA-II version is an improved version of MOGA by Polini [28] and has proven to be an efficient method for solving multi-objective optimization problems [29] and is recognized for its robustness when employed on various types of optimization problems, including noisy ones [30].

2.3 Statistical Theory

Following Ryberg, Bäckryd, and Nilsson [6], some fundamental statistical theory is included to aid in upcoming topics such as Design of Experiments, variable screening, and metamodeling.

A variable is said to be random if it depends on the outcome of a random, or stochastic, phenomenon. The *expected value* is denoted as $E[x]$ and can be thought of as the arithmetic mean μ. The expected value can be formulated as the following for discrete variables:

$$\mu_x = E[x] = \sum_{i=1}^{n} x_i P_x(x_i) \tag{2.4}$$

Where P is the probability and x_i are all (n) possible values of x. Informally, the expected value is the sum of all possible values of the random variable multiplied by its probability function.

Random variables are often assumed to follow a probability density function of the *normal* or *Gaussian distribution* that can be formulated as:

$$P(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{(x-\mu)^2/2\sigma^2} \tag{2.5}$$

where σ is the *standard deviation* and μ is the mean in a "bell-shaped" curve. The *variance* of x is defined as $Var[x] = \sigma^2$ and the *covariance* between random variables x and y is defined as:

$$Cov[x, y] = E[(x - E[x])(y - E[y])] \tag{2.6}$$

The variance describes the spread around the mean, and covariance describes the degree of directional relationship between two variables.

The *correlation* R between two variables describes the linear relationship between two variables from -1 (perfectly negative) to 1 (perfectly positive) with 0 representing independency. Correlation can be formulated as:

$$R = Cor(x, y) = \frac{Cov(x, y)}{\sigma_x \sigma_y} \tag{2.7}$$

Lastly, statistical theory can be used to detect important or *sensitive* variables from a data set, e.g., in sensitivity analysis. The statistical modeling algorithm *Smoothing Spline Analysis of Variance* (SSANOVA) makes use of the underlying theory of ANOVA, i.e., assessing the size of variance among group means compared to the average variance within groups [31]. The addition of smoothing splines enables the method to become suitable for multivariate modeling or regression when applied to a noisy dataset. For a more detailed description of SSANOVA, see [32].

2.4 Design of Experiments

Design of Experiments (DoE) is a branch of statistics primarily concerned with the planning and analyzing of experiments [33]. DoE is widely used in science to reduce the number of experiments that need to be performed. In this work, concepts from DoE are used in the contexts of variable screening and modeling. In the former case, classical designs are typically favored, while in the latter case, more advanced, space-filling designs are more widely used for reasons described in Section 2.4.2.

2.4.1 Experimental Design for Screening

Variable screening aims to reduce the number of design variables, called *factors* in DoE, by sorting out less significant variables to speed up the optimization process and improve the designers' understanding of the system's behavior [33].

Factorial Designs

The number of samples in the design space, also called the *experimental domain*, varies depending on what design is used. See Figure 2.6 for three common experimental designs in the factorial family.

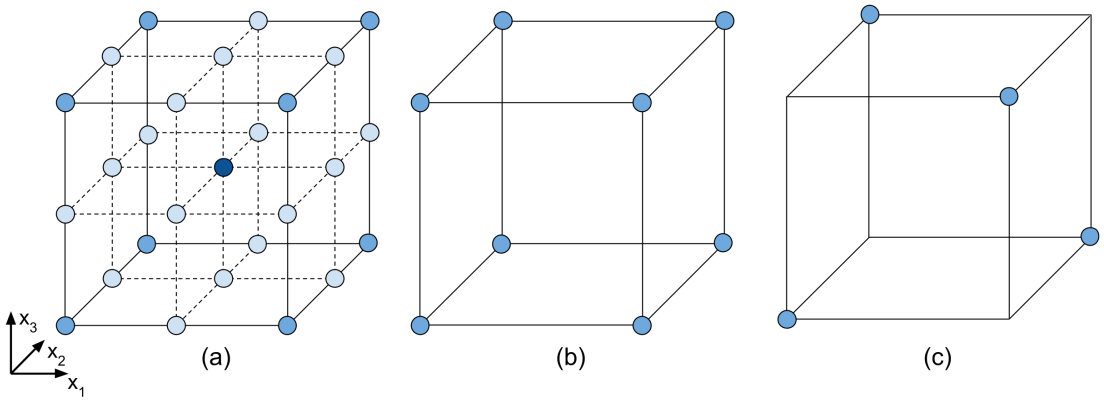


Figure 2.6: Representation of design points in the experimental domain: (a) 3-level, 3-factor full factorial design with 3^3 points; (b) 2-level, 3-factor full factorial design with 2^3 points; (c) 2-level, 3-factor 1/2 fractional factorial design with 2^{3-1} points.

The full factorial with three levels and three factors samples a low, medium, and high value of the factors x_1, x_2, x_3 and can be used to generate a quadratic model of the system behavior. If linear behavior can be assumed, a full factorial design with two levels can be used, reducing the design points from 3^k to 2^k, where k is the number of variables or factors. Both of the aforementioned designs can be used to investigate main effects and second-order interaction effects. The design can be simplified by: (1) assuming higher-order effects to be nonexistent or (2) assuming interaction effects to be insignificant to the variable screening. Both (1) and (2) need to be based on a fundamental understanding of the problem and its physics or tested with a more populous experimental design to verify the assumptions' validity. In the case that second-order, and higher, effects can be ignored, resolution III design can be effective. This design resolution means that no main effects are confounded with each other, but main effects and interaction effects may be confounded, i.e., that they cannot be distinguished between. Fractional factorial with 2^{k-p} runs (where k is the number of factors and p is the fraction of the full factorial 2^k), see Figure 2.6, or Plackett-Burman designs are suitable in this case [33], [34].

Plackett-Burman Designs

The Plackett-Burman (P-B) approach originated from Placket and Burman [35] as a method to determine main effects without the use of a factorial design. Using P-B designs is a popular approach when studying up to $k = (N-1)/(L-1)$ factors where N is the number of runs (a multiple of 4) and L is the number of levels. If N is set as a power of 2 then the resulting designs are called geometric and are equivalent to fractional factorial designs. When N is a multiple of 4, the designs are called non-geometric; these are typically advised to use if interaction effects can be neglected [33]. A distinct advantage of P-B designs is their efficiency in terms of the low number of required runs for screening when only main effects are of interest.

2.4.2 Experimental Design for Modeling

Modeling, covered in Section 2.5, requires a different set of design points to model fit on than the screening designs can provide with reasonable computational efficiency. According to scientific consensus [6], space-filling experimental designs are used to train metamodels.

Uniform Latin Hypercube

Latin Hypercube Sampling (LHS) [36], specifically Uniform Latin Hypercube (ULH) sampling, is recommended by modeFRONTIER documentation to be appropriate for metamodel training [37]. LHS and ULH can be used for multidimensional sampling but are illustrated using a two-dimensional case for clarity.

Generating random points in the design space to use for sampling, e.g., Monte Carlo sampling, has the downside that these points may be unequally distributed. For instance, large areas of the space may have no points while other areas have needlessly many points. It is, therefore, an unsuitable sampling method for representing large design spaces. To avoid large sampling gaps, LHS divides the space into a grid pattern and generates a single point in each row and column. The resolution of the grid, i.e., the number of rows and columns, depends on the number of factors k and the number of runs N. This forms a *Latin Square* in two-dimensional space, however, the same principle is used for multidimensional distribution, then forming a *Latin Hypercube* [38]. ULH is a type of LHS that minimizes the correlations between input variables while also maximizing the distance between points to achieve uniformity [39]. While LHS creates uniform distributions only for continuous variables, ULH can create uniform distributions also in the case of discrete variables [37]. Figure 2.7 illustrates how design point distribution can vary between Monte Carlo sampling, classic LHS and ULH with $N = 10$ points. Note the increased uniformity in ULH.

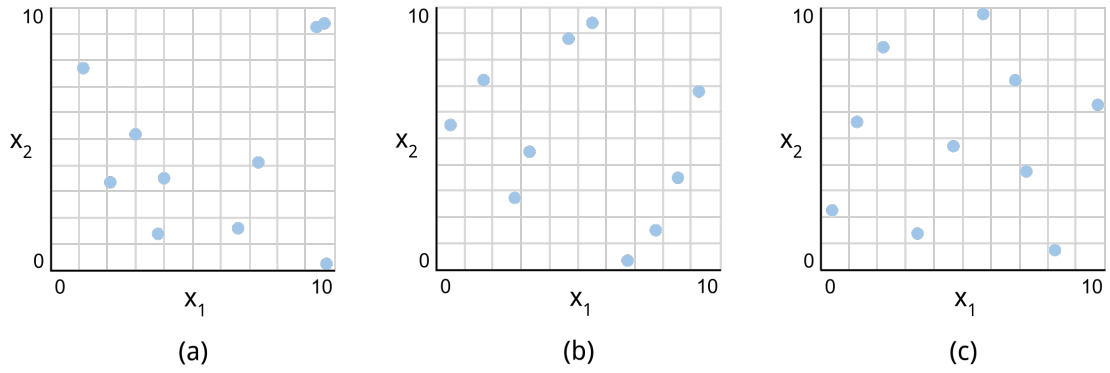


Figure 2.7: Point distribution in a bidimensional space with various sampling algorithms: (a) Simple Monte Carlo sampling; (b) non-space-filling or random Latin Hypercube; (c) space-filling Uniform Latin Hypercube.

The points generated from a space-filling DoE are then used as data sets to train metamodels.

2.5 Metamodels

Metamodels, also called response surface models or surrogate models, are used to approximate system responses. Metamodels are employed in design optimization to (1) minimize computational expense [26], (2) reduce the effects of numerical noise [40], and (3) enhance understanding of the functional relationship between a system's inputs and outputs [41]. To provide context to the metamodeling process, Quipo et al. [42] suggest the key stages illustrated in Figure 2.8.

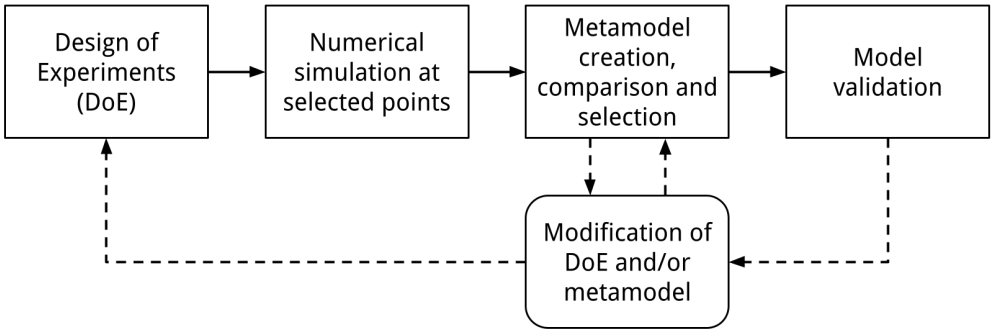


Figure 2.8: Key stages of the metamodeling process, adapted from [42]. Dashed lines indicate steps that may be required if a subprocess arrives at an unsatisfactory result.

As discussed in Section 2.4.2, a space-filling DoE design is first used to sample the design space. Computationally expensive numerical simulations are subsequently run at the points sampled at the DoE stage. The evaluations based on simulations may for this purpose be called *real* and evaluations based on metamodels may be called *virtual* [37]. The responses (i.e., outputs) of the simulation models from the real points (i.e., inputs) can then be used to train metamodels. Metamodel-based optimization can then be used to arrive at optimization results using significantly fewer computational resources than real optimization, assuming simulations are computationally costly [42].

Validation can be performed as a part of the metamodel comparison and selection stage by using a part of the training data set as a validation set; this is also called a *split sample* [42] or *simple cross-validation* [19]. If the fit of the metamodel is insufficiently good, its settings may need to be adjusted, or a different type of metamodel needs to be trained. Insufficient training data may also result in a poor fit, necessitating an increase in DoE resolution. Alternatively, validation can be performed after optimization by comparing the virtual optimization to a real simulation using the optimizers. This is particularly necessary if the metamodel has large local errors because the optimizers may lay in that high-error area.

A comprehensive survey of metamodels is much beyond the scope of this work and as such the reader is directed to [38] and [41] for further reading on the topic of metamodels and their application in design optimization. The types of metamodels

used in this work are described below. A distinction is made between *approximating* and *interpolating* models. The latter passes exactly through the training points while the former does not.

2.5.1 Polynomial Regression

Low-order polynomial regression [43] is a common and simple type of approximating metamodel. A general, n-th order, polynomial regression model can be formulated as:

$$\hat{f} = \beta_0 + \beta_1 x + \beta_2 x^2 + \cdots + \beta_n x^n \tag{2.8}$$

where the approximation $\hat{f} = f + \varepsilon$, where ε is the error, β_0 is the bias and β_i are the weights (where $i = 1, 2, \cdots, n$). Bias and weights are set by minimizing the error. Error minimization is typically done using the method of least squares, i.e., minimizing the sum of the squared errors. Many polynomial models of differing orders can be fitted and subsequently compared with error measures to select the best order. Alternatively, and especially if the design variables are many, some stepwise regression methods can be used to find an appropriate polynomial degree. The order of polynomial models that can be trained is a function of the training data size, i.e., the number of points generated by a space-filling DoE, see Section 2.4.2. Excluding validation points, the minimum training data size required by polynomial regression of degree d for p number of parameters can be formulated as:

$$\text{Minimum no. of points} = \frac{(p+d)!}{p!d!} \tag{2.9}$$

As is evident from the relationship in Eq. 2.9, the number of required samples for anything other than low-order polynomials can be restrictive, e.g., an optimization with 5 DVs being fitted with a 7-degree curve requires 792 runs (excl. validation points), which can be infeasible with computationally costly simulations. Polynomial approaches can be used to identify main trends but may be unsuitable as a metamodel unless responses are simple, unlike multimodal problems [37]. While still a popular method in optimization [44], polynomial regression is being fast replaced by radial basis approaches according to Forrester and Keane [38].

2.5.2 Radial Basis Functions

Radial basis functions (RBFs) are interpolating models that can be described in simple terms as using a weighted sum of multiple simple functions that together exhibits more complex behavior. See Figure 2.9 for a simple illustration in two dimensions. The complexity can be considerably higher with multivariate functions and many points, but the principles are the same.

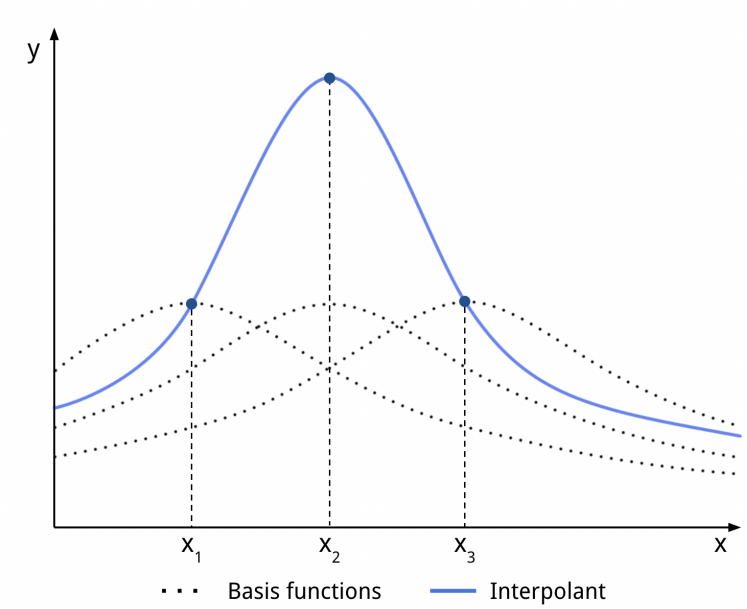


Figure 2.9: RBF interpolation with $n = 3$ using inverse multiquadrics, adapted from [45].

The use of RBFs as approximation functions was first proposed by Hardy [46] in geophysics to fit topological data using scattered data sets. In general, RBFs are expressed in terms of the Euclidean distance, also called the radial distance $r = ||\hat{x} - x_i||$ from an approximating point $\hat{x}$ to a measured data point x_i which is the center point for an RBF. Perhaps the most common form is using a multiquadric function as Hardy [46] originally did:

$$\psi(r) = \sqrt{r^2 - c^2} \tag{2.10}$$

where $c > 0$ is a shape parameter that controls the function's width. The approximation uses the linear combination of the basis functions in all data points:

$$\hat{f}(x) = \sum_{i=1}^{n} \sigma_i \psi(||\hat{x} - x_i||) \tag{2.11}$$

where n is the number of data points, and σ_i are coefficients that are determined by solving the resulting linear system of equations that forms when constraints of interpolation, i.e., $\hat{f} = f$, are included [44]. Many RBF variants exist, the description of which is beyond the scope of this text. See [47] for a survey of commonly used variants.

A considerable benefit with RBFs compared to polynomial regression is that the former is less sensitive to problem dimensionality, i.e., the number of DVs. As covered in Section 2.5.1, polynomial regression requires a minimum number of points to fit a specific function degree, see equation 2.9, this is, however, not the case

with RBFs, at least to an extent. Another significant benefit is their performance in fitting on scattered data in multiple dimensions and their relative ease of use compared with more advanced methods, e.g., Kriging [44]. Without special modification, RBFs are most appropriately used for smooth responses devoid of noise [37].

2.5.3 Kriging

Originally developed by Krige [48] in the field of geostatistics, Kriging is a probabilistic, (often) interpolating, method that uses responses at nearby sample points to estimate the response at a non-sampled point. The many variations and details of Kriging are beyond the scope of the present text. However, some brief introduction is presented, for further details, the reader is referred to [49] and [50].

Operating under the assumption that nearby sample points are more likely to have a similar response than more distant sample points, Kriging involves calculating a weighted sum of nearby sample point responses. This can be formulated as:

$$\hat{f}(x) = \sum_{i=1}^{n} \lambda_i f(x_i) \tag{2.12}$$

where λ_i is the weight at i and otherwise adopting the notation from Section 2.5.2, weights are determined by fitting a so-called *variogram* that describes the variability, or covariance, between a pair of points as a function of distance h. Initially, an *experimental variogram* function $\gamma(h)$ of the sample set is constructed. A *theoretical variogram* is then fitted to the experimental variogram, one of several mathematical models, e.g., exponential, Gaussian, or spheric [51]. Figure 2.10 shows a theoretical variogram fitted to an experimental one.

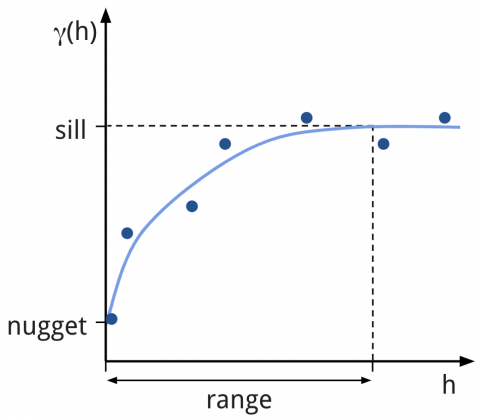


Figure 2.10: Theoretical variogram fitting with spherical model to an experimental variogram with associated notation. Adapted from [51].

In practice, Kriging is suitably used for modeling highly non-linear systems and can be used for both noisy and noise-free systems by changing model parameters. Noise

can be added to make the algorithm approximating. A consideration is that Kriging is computationally expensive relative to other metamodeling approaches described in this text. It may be slow to converge or may fail to do so in cases where $n > 1000$ (n being the number of designs) [37].

2.5.4 Performance Indices for Metamodels

When fitting a metamodel to a number of design points and using a validation set of points n_v, the difference between the observed value x_i and the approximated value $\hat{x}_i$ can be used to evaluate the metamodel's accuracy. The mean absolute error (MAE) provides the actual average value from the comparison between the observed and approximated values and can be formulated as:

$$MAE = \frac{\sum_{i=1}^{n_v} |x_i - \hat{x}_i|}{n_v} \tag{2.13}$$

Using the mean relative error (MRE), also called mean absolute percentage error (MAPE), the error can instead be described in percentage as the ratio between the MAE compared to the real design values:

$$MRE = \frac{\sum_{i=1}^{n_v} \frac{|x_i - \hat{x}_i|}{x_i}}{n_v} \cdot 100 \tag{2.14}$$

Both MAE and MRE are performance indices describing the error of the metamodel, meaning a higher value corresponds to a higher error and thus lower accuracy of the metamodel [6].

2.6 Multidisciplinary Design Optimization

As mentioned in Section 1.1.3, with increasing competition and customer demand, more pressure is placed on companies to produce high-value products in shorter development cycles. Operating in such a competitive environment requires the coordination of complex interactions between multiple disciplines in the product development process. MDO is employed to this end, by for example, minimizing mass while the problem is constrained by performance requirements from several disciplines in the automotive sector. Cost models, manufacturing constraints, and constraints from less traditional disciplines may also be integrated into the MDO setup [52].

By considering several disciplines simultaneously in the optimization process, rather than a single discipline at a time in isolation, a multidisciplinary feasible design can be achieved as a result. While this has been done to some degree in an informal manner by engineering judgment and communication via disciplinary experts, MDO introduces a mathematical and more formal approach to optimization that takes multiple disciplines into account, preferably by exploiting synergies [4].

2.6.1 Terminology and General Formulation

As MDO deals with multiple disciplines, a distinction between local- and shared variables is made. Local variables belong to one specific discipline, while shared variables are variables included in more than one discipline [19]. Discipline-specific data is here denoted to as $()_i$ and [53] is used to denote shared data $()_0$. Furthermore, disciplines may interact in a manner where one discipline's output is required as input for another discipline. This behavior is referred to as a coupling variable, denoted $\mathbf{y}$. Variables affecting more than one discipline, through a shared- or coupling behavior, need to be consistent. This means that variable copies ($\check{\mathbf{y}}$) or shared variables are consistent with their corresponding shared variables for every system it is associated with. Similarly, coupling variables inputs for a certain discipline need to be consistent with their corresponding discipline output. Shared- and coupled variable consistency is required for system consistency to be achieved. Moreover, a state variable ($\bar{\mathbf{y}}$) is the value of a variable at a certain state of the system [53], [54].

The problem formulation is thus formulated as the following:

$$\begin{aligned} \min & f_0(\mathbf{x},\ \mathbf{y}) + \sum_{i=1}^{n} f(\mathbf{x}_0, \mathbf{x}_i, \mathbf{y}_i) \\ & \text{with respect to } \mathbf{x}, \check{\mathbf{y}}, \mathbf{y}, \bar{\mathbf{y}} \\ & \text{subject to } \mathbf{c}_0(\mathbf{x}, \mathbf{y}) \geq \mathbf{0} \\ & \mathbf{c}_i(\mathbf{x}_0, \mathbf{x}_i, \mathbf{y}_i) \geq \mathbf{0} \quad \text{for } i = 1, 2, ..., n \\ & \mathbf{c}_i^c = \check{\mathbf{y}}_i - \mathbf{y}_i = \mathbf{0} \quad \text{for } i = 1, 2, ..., n \\ & \mathbf{R}_i(\mathbf{x}_0, \mathbf{x}_i, \check{\mathbf{y}}_{j \neq i}, \bar{\mathbf{y}}_i, \mathbf{y}_i) = \mathbf{0} \quad \text{for } i = 1, 2, ..., n \end{aligned} \tag{2.15}$$

The general formulation accounts for the interaction between all disciplines in the system by including all behavior in the form of coupling, state, copies, consistency, and residuals. While it is rarely directly applied to optimization problems, its mathematical formulation describes all aspects of a general MDO [53].

2.6.2 Architectures

As a result of the need to coordinate disciplines for a vast variety of design problems and application areas, research in the last decades has been devoted to developing so-called MDO *architectures*. MDO architecture, also called strategy, procedure, or framework refers to how the optimization problem is solved and how the coordination between disciplines (or objects) is organized [53]. This extensive literature field cannot be mapped in its entirety in this work, however, an overview is provided to give context. Since architecture selection is typically made on an ad hoc basis [55], some information about the alternatives may guide an engineer who is new to the area. See Figure 2.11 for an introductory overview of MDO architectures.

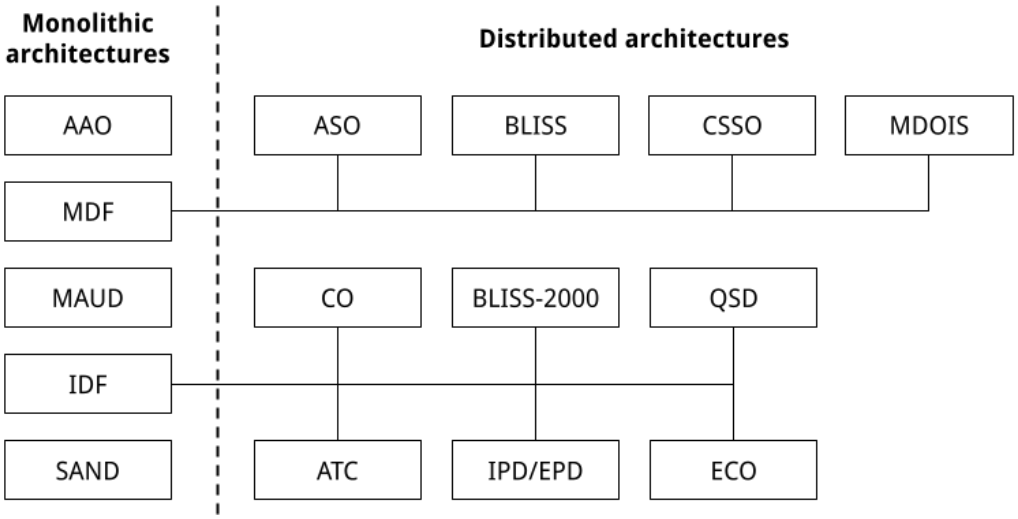


Figure 2.11: A selection of MDO architectures and their categorization.

Architectures are divided into two main categories: *monolithic* and *distributed*. The difference is that monolithic formulations consist of a single optimization problem being solved while distributed formulations divide the problem into several subproblems that contain subsets of associated variables and constraints [53]. Monolithic architectures use a single optimizer and generally perform better if the problem remains at a small scale [55], tentatively in the order of 10^1 design variables. Distributed architectures were developed by modifying already existing monolithic architectures, specifically the monolithic architectures Individual Discipline Feasible (IDF) and Multidisciplinary Feasible (MDF), using more than one optimizer in order to decompose a problem with coupling behavior. The details of each method are not elaborated upon in this report as this is considered beyond the scope. The reader is referred to *A Survey of Architectures* [53] for a survey of the architectures shown in Figure 2.11 and their technical differences, with the exception of *modular analysis and unified derivatives* (MAUD), see [56].

In addition to division by monolithic and distributed, architectures may also be classified by whether they are aimed at *object-* or *aspect-based decomposition*, as identified by Wagner [57].

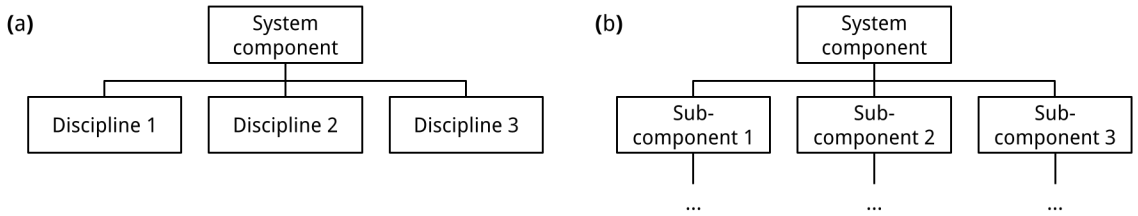


Figure 2.12: Aspect-based decomposition (a) and object-based decomposition (b).

The former is a hierarchical structure in which the top-level represents the whole system, followed by subsystems and their respective components in an inverted tree structure. The latter is a bi-level hierarchy where the system resides at the top

level, and the disciplines (i.e., aspects) are at the lower level. Some architectures are oriented more to one of the aforementioned decomposition types, e.g., *analytical target cascading* (ATC) [58], and may be less suitable for the other type.

Monolithic architectures are commonly employed over distributed architectures in the automotive industry [6] and are generally preferred when the study focuses on a single component. Distributed architectures can be advantageous in medium- to larger systems, where the system favorably is decomposed into a system- and subsystem component(s). Furthermore, distributed architectures are generally applied to larger systems in organizations where automation and in-house expertise are coveted [6]. However, most distributed architectures require a specific structure to be considered, and even so, monolithic approaches tend to converge more rapidly, decreasing the usage of distributed architectures in practical applications [19].

Extended Design Structure Matrix

An *Extended Design Structure Matrix* (XDSM) is used to illustrate the process flow and data dependencies of architectures [59]. XDSM diagrams for the MDF and IDF architectures can be found in Figure 2.13 and Figure 2.14 respectively. The components in an XDSM follow a diagonal layout where the vertical flow in an XDSM represents variable inputs to a component, while the horizontal flow corresponds to the outputs. The small black lines represent the process flow, while the larger grey ones represent the data dependencies. The arrow notations used in the optimization- and multidisciplinary analysis (MDA) components indicate that the process is to be iterated until a certain condition has been met.

Multidisciplinary Feasible

One of the most common ways of posing an MDO problem is using of the multidisciplinary feasible (MDF) architecture [54]. The problem formulation for the MDF excludes the residual and consistency constraints from equation 2.15, making it the smallest problem formulation out of all the monolithic architectures [53]. The MDF architecture runs the discipline analyses sequentially and utilizes an MDA, typically in the form of a Gauss-Seidel multidisciplinary analysis, to solve the governing equations for every discipline [55]. In MDF, the entire optimization problem is thus divided between the MDA, solving the coupling behavior of the disciplines to obtain system consistency, and the optimizer, controlling the objective function, design variables, and design constraints. The process of an MDA is iterated, forcing every discipline analysis to be performed multiple times until system consistency is achieved [53].

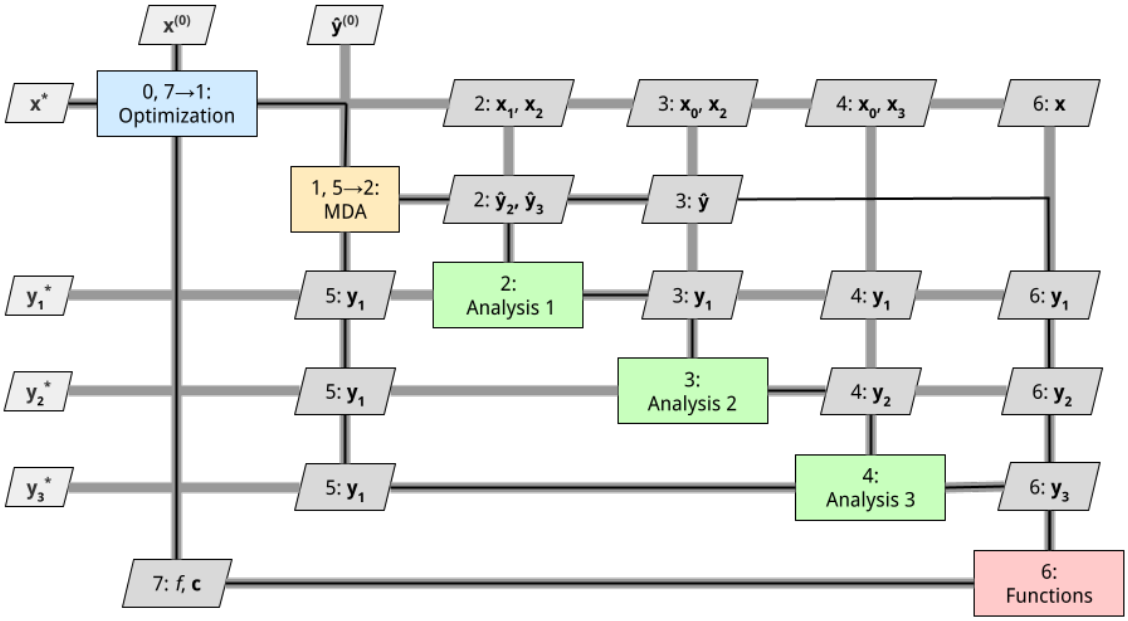


Figure 2.13: Gauss-Seidel multidisciplinary analysis for the MDF architecture [53].

Each optimization iteration requires a full MDA to be performed, requiring multidisciplinary feasibility to be maintained. The exception is if derivative-based methods are used to solve the optimization problem, for which the MDA is not required at every iteration, but instead at every point where the problem is evaluated [54]. Therefore, the choice of optimization algorithm will affect if a feasible result can be expected, should the process be terminated prematurely. Utilizing an MDA for consistency purposes requires a high number of function evaluations and can result in MDF being time-consuming in practice [53]. Nevertheless, MDF has potential in cases of weakly coupled problems with low computational cost to run subsystem analyses [15]. Moreover, the MDF architecture is an attractive option in practical problems since its small size results in lower problem dimensionality, and less implementation complexity to set up [60].

Individual Discipline Feasible

While MDF runs discipline analyses in sequence and utilizes an MDA to maintain multidisciplinary feasibility at every iteration, IDF does not require an MDA and runs the discipline analyses in parallel. The analyses are allowed to run in parallel since independent copies of coupling variables are added to the set of design variables [19], [55]. IDF thereby enforces single discipline feasibility at each design point instead of multidisciplinary feasibility (compared to MDF) [54], [55]. Furthermore, a set of consistency constraints are added to compare estimates to the actual coupling variable values at optimum [19], [55]. IDF thereby enforces multidisciplinary feasibility only at the optimum point and does not in any way guarantee feasibility should the process end prematurely [15], [61].

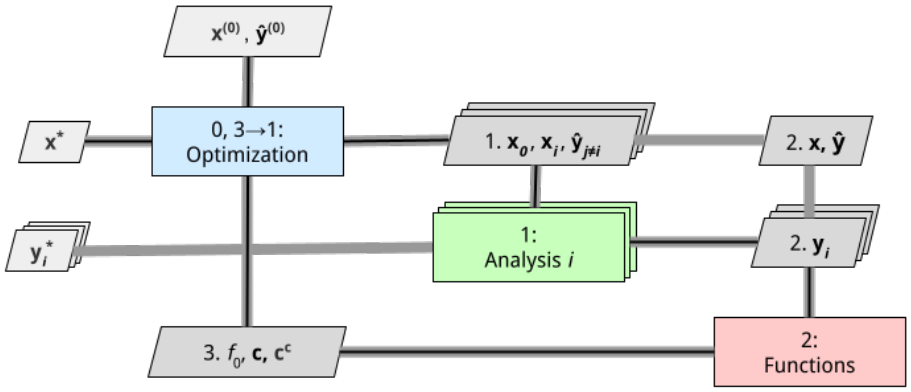


Figure 2.14: Process flow and data dependencies of the IDF architecture [53].

Tedford and Martins [55] illustrated how IDF tends to be the most computationally efficient out of the monolithic architectures. However, it scales worse than MDF when the number of design variables is increased [15]. In general, the advantages of not utilizing an MDA allow IDF to converge several orders of magnitude quicker than MDF [62] while being less computationally costly and more robust [55].

Automotive Considerations

As mentioned to in Section 1.1.3, MDO has been used extensively in the aerospace industry while its use in the automotive industry is relatively more recent. In addition to different development environments, the type of engineering design problems in these industries typically differ. The level of coupling between disciplines is one such difference. As Bäckryd et al. [9] notes: aerospace problems generally exhibit linking by both shared and coupled variables, e.g., in wing design where structural deformations and aerodynamic forces interact strongly. However, in the automotive industry, linking between disciplines is weaker in that disciplines generally are connected only with shared variables. The automotive context may be called a multi-attribute rather than a strictly multi-disciplinary environment, according to Agte et al. [8]. It shall also be noted that the monolithic architectures MDF and IDF mathematically coincide for problems with only shared variables as opposed to coupled variables [9]. In this case, the optimizer forwards variable values to the discipline analyses, which return the contributions to the objective function and respective values for the constraint functions. This parallel approach is used in the present work.

3
Methodology

The thesis adopts a basic research methodology whereby a literature study is performed following a specification of research aim and scope [63]. The literature and theory study aims to elucidate concepts used in the work and to map recent scientific developments that may be built upon. Other activities are specific to the present work. The methodological process, see Figure 3.1, illustrates the steps taken in the project in outline.

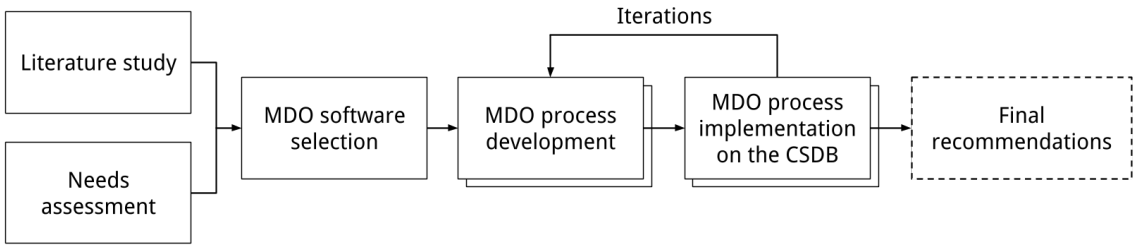


Figure 3.1: Methodology in outline.

The respective steps in the process are elaborated upon in Sections 3.1-3.6.

3.1 Literature Study

A literature study is conducted to map previous literature that is conceptually and practically relevant to this work. This is done to understand the knowledge gaps that may be explored and utilize previous findings to aid the current work. The scope includes relevant parts of the following topics: basic engineering optimization theory, statistics, DoE, optimization algorithms, metamodels, and MDO. Materials included in the literature study are journal articles, dissertations, working papers, books, reports, and manuals.

3.2 Needs Assessment

To ascertain the needs related to an MDO process at VCC, a needs assessment is performed. Brief interviews are performed to collect information on current workflows, optimization experience, and perceived hindrances and benefits with MDO. Semi-structured interviews [64] are chosen to allow for a degree of guidance on the

part of the interviewer while maintaining flexibility in the format to guide the interview to areas of interest that may be difficult to identify in advance.

The sample of interviewees centers on the target demographic, namely CAE engineers at VCC, but other stakeholders such as product owners are also included. The team in which the thesis work is performed has priority. However, CAE engineers in other departments are also interviewed as their needs are likely similar and may be relevant if the selected MDO process is implemented at a larger scale than for a single CAE team. As software and work processes are similar, a larger sample also offers more information that may be pertinent even if scale-up does not occur. Eight subjects were interviewed, and each interview took 20 to 30 minutes.

The collected information from interviews is anonymized and presented as an amalgamated needs list based on subjective interpretation of the collected interviewee responses. Formal analysis techniques, e.g., thematic analysis [65], are deemed not to offer enough benefit to warrant the additional effort due to the limited scope of the needs assessment.

3.3 Software Selection

The term "MDO software" refers to the software in which the MDO process is coordinated, and analysis is done. It is separate and distinct from the software package required for running pre-processing, simulation, and post-processing but must communicate with these. Due to emphasis on ease-of-use and because VCC has licenses to and existing knowledge about MDO-capable software, commercial software is used as a basis for the work.

LS-OPT by Livermore Software Technology Corp. and modeFRONTIER by ESTECO SpA are both viable alternatives. Both LS-OPT and modeFRONTIER have user-friendly Graphical User Interfaces (GUIs), support for external software, and grid computing management. In-built post-processing and visualization tools are also available. Support and regular updates are available for both software. However, modeFRONTIER is widely used among optimization experts at VCC, and existing knowledge can be leveraged to a greater degree than if LS-OPT is selected. ESTECO's modeFRONTIER is therefore used in this work.

The Python-based, open-source platform OpenMDAO [20] was briefly investigated as an alternative, although it is not commercial software. It supports gradient-based optimization with analytic derivatives (as opposed to finite-difference approximations). The number of design variables that can feasibly be used is significantly greater than in commercial software, up to the order of 10^4 has been demonstrated [66]. A basic GUI is under development and plugins for simulation software exist, but implementation difficulty is considerably greater than in commercial software. Reliable support is unavailable due to the open-source nature of the platform. OpenMDAO is therefore not recommended for present purposes. For a further discussion of when the use of OpenMDAO is warranted, see Section 6.3.

3.4 Process Development

In this context, the term "MDO process" is synonymous with "MDO methodology" and refers to the process that receives the parameterized model and requirements as inputs and outputs the optimization result (which can then be interpreted in a CAD model and checked against requirements).

A general workflow is developed based on findings from the literature study and the needs assessment and can be implemented in any software procedure capable of performing the steps outlined in the process. The general workflow is practically developed as follows.

As mentioned in Section 3.3, modeFRONTIER is used for MDO implementation. The process is developed in the selected software using a simplified geometry and mesh to achieve more rapid debugging than would have been possible with the CSDB mesh (and associated geometry required in the analysis, i.e., components that the CSDB is connected to). The faster iterations are possible mainly because less time is spent on running simulations. An illustrative comparison is that design point evaluations with the simplified geometry runs approximately 40 times fast than a crash simulation of the CSDB concept and instrument panel.

Because the focus in this thesis lies on process rather than component development, the geometry need not necessarily reflect the CSDB or any other component at this stage. However, care must be taken to ensure that responses are predictable, e.g., that the first mode shape is kept the same under varying design variables values. Beyond reduced simulation time, a less complex mesh in process development simplifies debugging due to fewer geometric parameters. This places focus on the process development rather than handling issues that might arise with the specific geometry.

The overlapping boxes in Figure 3.1 for "MDO process development" indicate that several subprocesses are developed. Initially, single-discipline processes or workflows are developed separately. Note also that simulation data automation occurs in this step. This refers to scripting that allows for runs to be started, results to be saved in specified directories, nested scripts that post-process simulation results, and extracts responses as outputs in the process. Dynamic waiting for results is also implemented at this automation stage, see Section 4.2. Once single-discipline processes are developed, the combination of their respective parts and associated scripts is straightforward to implement in modeFRONTIER.

3.5 Process Implementation on CSDB

Once a working MDO implementation in modeFRONTIER has been established using simple geometry, the process is tested on the CSDB component for verification. The main differences compared to the simple geometry are the level of geometric complexity and the number of load cases. The overlapping boxes in Figure 3.1

indicate several iterations and formulation variations are tried on the CSDB problem. If unexpected limitations are discovered at this stage, further iteration in process development may take place.

3.6 Final Recommendations

The last major step in the process is to provide final recommendations based on important decisions and other learnings received from developing and verifying the MDO process both conceptually and as a matter of implementation. This includes essential trade-offs to consider, process limitations, and organizational- as well as sustainability-related considerations.

4
Process Development

This chapter presents findings related to the development of the MDO process, both conceptually and as a matter of implementation. Findings from the needs assessment are listed, implementation in modeFRONTIER is detailed, and the resulting process is illustrated using a simple example while decisions made are elaborated upon throughout the chapter.

4.1 Needs Assessment Findings

The needs assessment, see Section 3.2, aims to ascertain the following:

- What is the current practice with regards to achieving component or system requirements across disciplines?
- What software is used for CAE simulation and optimization?
- What is the level of optimization maturity?
- What are the perceived hindrances with MDO?
- What features of an MDO workflow are perceived as desirable?

Sections 4.1.1-4.1.5 present the findings on each topic as an amalgamation of the answers given by all interviewees.

4.1.1 Current Practice

Work in and between CAD and CAE departments forms a major part of the product development process. Once a design is conceived, iterations are performed both on the level of the design engineer(s) and the CAE engineer(s) before proceeding to physical prototypes and verification. The design engineers change the CAD geometry and pass designs onto CAE for testing, while CAE engineers change the finite element (FE) mesh directly to perform iterative looping and evaluate performance changes. The number of loops may exceed 100 on the CAE level for a single component. Knowledge gathered from simulations combined with engineering experience guides each iteration.

Fulfilling requirements across disciplines is coordinated primarily via meetings involving many disciplinary experts. This may be guided by a system architect responsible for balancing disciplines and achieving system compliance to specification. If one discipline requires a change to meet requirements, they inform other disciplines

and looping procedures on a larger scale are performed to converge to a solution.

The description of the current practice above is necessarily incomplete as a complete description of the product development process at VCC in all its detail in practice is beyond the scope of this text. However, it does serve to get a sense of how multidisciplinary work is typically performed currently.

4.1.2 Software

Depending on the disciplinary expertise area, interviewees use different software to perform simulation and analysis. MSC Nastran is used for NVH, Abaqus for durability, and LS-DYNA for crashworthiness. Software for optimization is Altair OptiStruct, 3DS Tosca, and Nastran SOL 200. One respondent report taking a course in Simcenter HEEDS. Some respondents report having used modeFRONTIER, although most have not.

4.1.3 Optimization Maturity

The level of experience with optimization within the sample group varies. Some interviewees reported no optimization experience and forwarded it to more experienced colleagues, while one interviewee, holding a Ph.D. related to optimization, has performed a DoE involving two disciplines and a cost model.

While individual interviewees have experience in size, shape, and topology optimization, the general view is that optimization adoption could be improved within the organization.

4.1.4 Hindrances of an MDO Process

Those familiar with MDO are asked what reasons exist for it not being more widely used. The following perceived and undesirable features are cited (quotes are paraphrased):

- "MDO is time-consuming to set up and run."
  - Especially with crashworthiness simulation as this is typically the most computationally costly simulation type.
- "MDO is complex to implement."
- "Disciplines may use different models and the same mesh may differ in part, node, and element numbering between disciplines."
  - This makes mesh parameterization across disciplines difficult.
- "Nonlinearities, primarily in crashworthiness, makes some simulations unreliable."
  - This issue is not exclusive to MDO but rather a wider issue with crash simulations both due to numerical issues in FEM and inherently unstable behavior of certain automotive designs caused by bifurcations. The latter

may be the most common reason for scattering in crash simulation results [67].

- "Many requirements need to be considered, perhaps too many for MDO to be effective within an acceptable time frame."

4.1.5 Desirable Features of an MDO Process

Respondents report the following perceived opportunities and desirable features of MDO (quotes are paraphrased):

- "In general and as far as feasible, MDO should be easy to employ and integrate well with existing software."
- "It is more important that MDO can be used as a tool to learn more about the design and how it can be improved rather than as a procedure resulting directly in a final design."
- "High automation is desirable at a late stage of the development of an MDO workflow to increase efficiency."

4.2 Process Implementation in modeFRONTIER

The software modeFRONTIER operates through a node-based GUI, where input nodes are used to modify desired input files for the simulation. A simplified example is depicted in Figure 4.1.

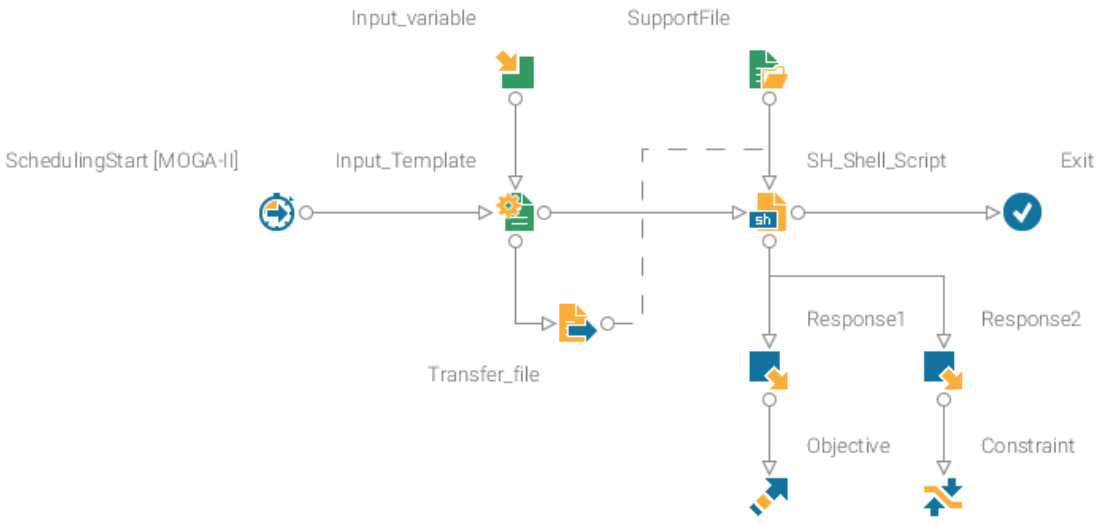


Figure 4.1: Schematic workflow with a selection of nodes used in modeFRONTIER for this implementation.

The process flow is controlled by "Scheduling Start node" and the integration between modeFRONTIER and other software can be controlled, e.g., directly via spe-

cific “CAE node(s)” or by using “SH Shell Script node(s)”. In this case, shell script nodes were used since they are not bound to a specific type of simulation or discipline. A generalized version (due to confidentiality) of the script code used within the shell script node is provided in Appendix B. Finally, output nodes are used to extract desired results which modeFRONTIER automatically creates into a design table for further analyses to be performed. Although the process implementation changes somewhat depending on the problem formulation, which disciplines are used, and the number of load cases for each discipline, an illustration of the modeFRONTIER setup is depicted in Appendix A.

Each time a design from the space-filling DoE is evaluated, a loop is run to evaluate responses at that point. This process repeats until all design points in the DoE have been evaluated, after which other steps in the process, as detailed in Section 4.4, are performed. Beginning with a point in the DoE that corresponds to a set of design variables, a shell script is started for each discipline or load case, see Figure 4.2.

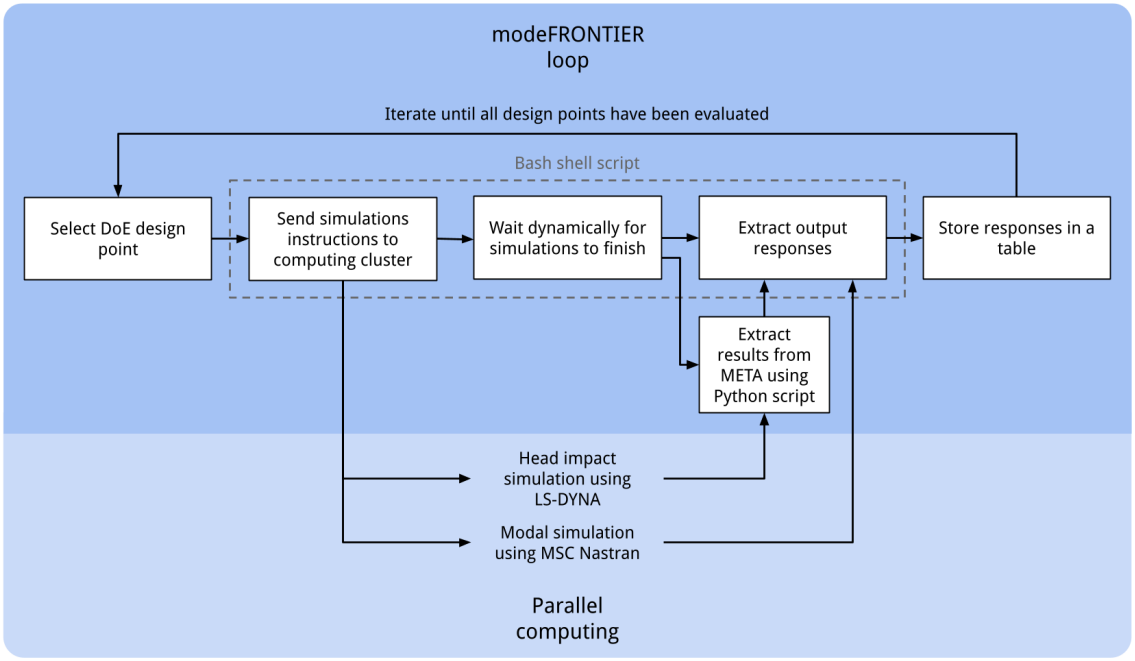


Figure 4.2: Schematic workflow chart of modeFRONTIER loop with associated scripts and subprocesses.

The script parallelly starts all simulation runs by sending instructions to the various software (in this case LS-DYNA and MSC Nastran). While the simulations are run on parallel computing clusters, the script dynamically waits for simulation completion by checking log files at a specified interval using custom code. In the case of head impact, an additional script, in this case, a Python script, is used to run a series of functions in META and write responses to an output file. The script waits for this process before completing and storing the responses in a modeFRONTIER table before iterating the loop with the following design point in the DoE.

A notable benefit with the setup, see Appendix A, is that multiple simulations

can run in parallel, thus reducing total time for simulation as compared to a setup that runs simulations in series. Time is also saved by the custom code dynamically checking for simulation completion. Furthermore, a single "Input Template" node for each discipline is used to ease implementation with a new problem.

4.3 Development Problem

As stated in Section 3.4, a simplified geometry with predictable and stable responses, which need not approximate any real component, is used to develop and illustrate the MDO process. See Figure 4.3 for the geometry, henceforth called "Development Geometry" or DG. Note that shell, rather than volume, elements are used in this model due to it being a thickness-based optimization on mesh level. The compartments of the geometry are to give predictable responses for both crash and modal simulations.

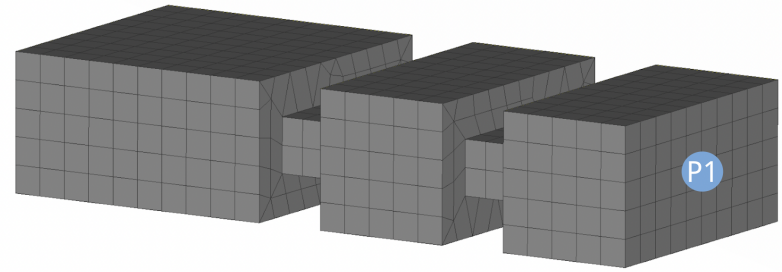


Figure 4.3: Shell mesh of Development Geometry annotated with head impact point $P1$.

Load cases and responses for the DG:

- Load case $P1$ with head impact responses as measured by maximum acceleration a_{max} [G] and maximum acceleration under a 3 milliseconds time interval $a_{clip3ms}$ [G], see Figure 2.1.
- Modal response as measured by the eigenfrequency for the first mode f_{mode1} [hz].
- Mass m [kg] as extracted from the mesh model.

A head impact load case is illustrated in Figure 4.4.

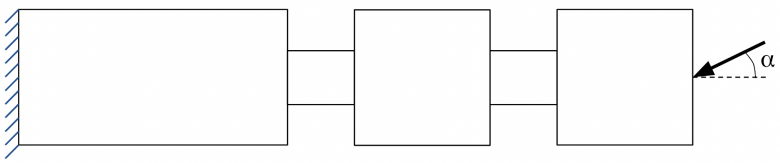


Figure 4.4: Side view of Development Geometry under crash load case $P1$. Impact is at an angle α. Boundary conditions are set as fixed in all translations and rotations, the same as for the modal analysis.

One possible problem formulation is described in equation 4.1 with mass m as a function of design variables $\mathbf{x}$ is minimized subject to the constraints as listed above.

$$\begin{aligned} &\min\ m(\mathbf{x}, \mathbf{p}) \\ &\text{subject to} \\ &g_{1,P1}(\mathbf{x}, \mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}] \\ &g_{2,P1}(\mathbf{x}, \mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}] \\ &g_3(\mathbf{x}, \mathbf{p}) - \mathbf{f}_{mode1} \leq \mathbf{0}\ [\text{Hz}] \\ &b_l \leq x_i \leq b_u\ [\text{mm}] \end{aligned} \tag{4.1}$$

The vector $\mathbf{x}$ contains n variables x_i where $i = 1, 2, \cdots, n$. The vector $\mathbf{p}$ contains parameters that are held constant during the optimization, e.g., Young's modulus and other material properties. The functions g_1, g_2, and g_3 are the true response functions as approximated by metamodels. Vectors $\mathbf{a}_{max}$, $\mathbf{a}_{clip3ms}$, $\mathbf{f}_{mode1}$ are vectors with the dimension n x 1 and all values in the matrix are identical. The design variables are bound between upper and lower bounds, b_u and b_l respectively. No specific values are given for parameters, constraints or bounds as the example is illustrative.

The DG has been used in the development of the MDO process and is used in the current chapter to demonstrate aspects of the process for demonstration clarity. It may also be noted that the material properties used in the model approximate that of mild steel. However, this is of limited relevance as the model is, as stated, used simply for illustrative and development purposes.

4.4 Proposed Process

The proposed MDO process that has been developed in the "MDO process development"-stage, see Figure 3.1, is presented in Figure 4.5. It takes the pre-optimized concept and requirements, including objective(s) and constraints, as inputs and outputs an optimized concept. Additional inputs in terms of choices along the process depending on the application are also required. However, this is explored in the below sections for clarity.

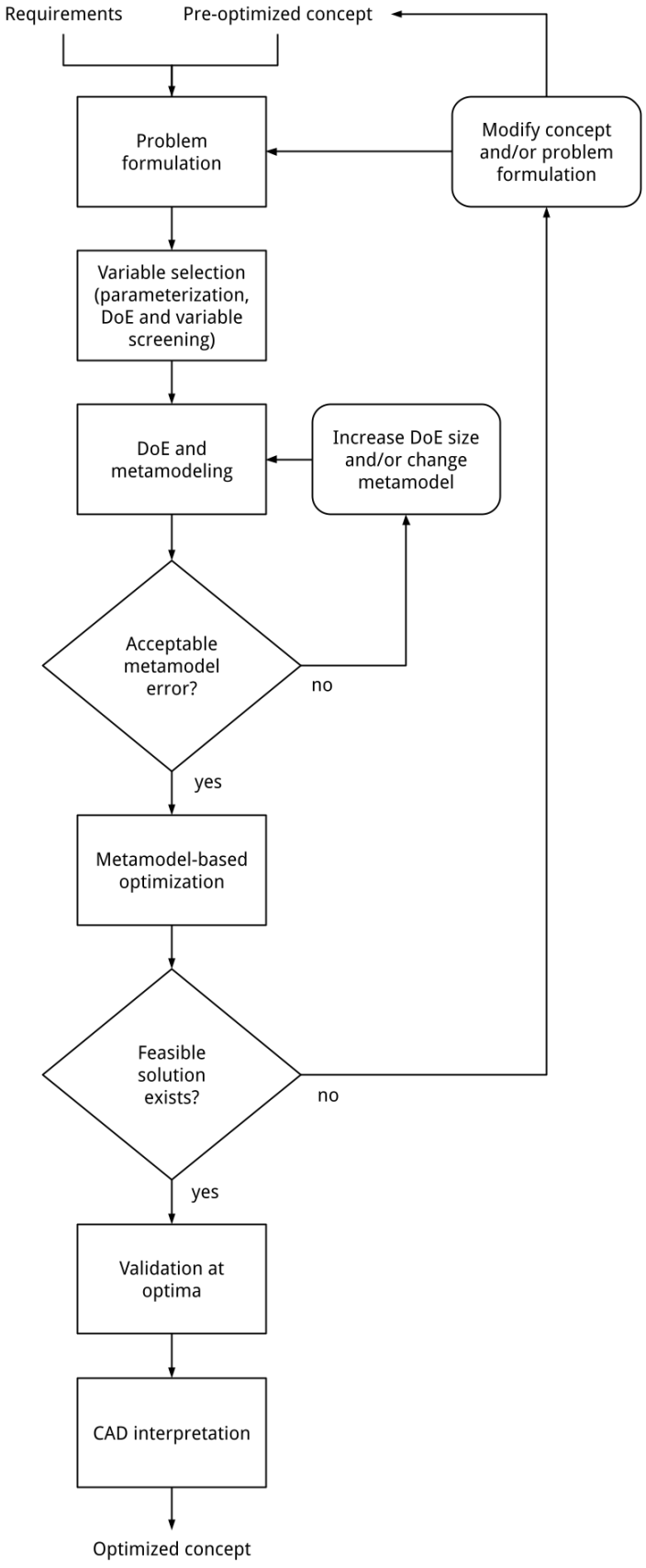


Figure 4.5: Flowchart of the MDO process.

4.4.1 Problem Formulation

Problem formulation involves the setup of objective(s), design variables, constraints, parameters, and design variable intervals to serve as a mathematical representation of the optimization problem and its constituent parts. This formulation must consider the structure of the MDO process as formulations differ between architectures by definition, see Section 2.6.2.

The primary consideration is whether the problem formulation for the proposed process is single- or multi-objective (i.e., SOO or MOO). The proposed workflow allows the user to switch between these objective types with relative ease by changing node types in modeFRONTIER. Intervals of the design variables must also be defined. Limitations can be imposed by, for instance, volume, manufacturing, or cost constraints. Exploration outside the "practical" interval may be beneficial to identify limiting factors, e.g., a manufacturing technique that only allows casting down to a minimum thickness t_{min} while the optimizer is $t^* < t_{min}$ when the design variable interval is broadened, thus indicating that another manufacturing method may be considered.

4.4.2 Variable Selection

A selection of a limited number of features to be varied under the optimization is necessary to keep DoE size at a feasible level. This necessitates the use of parameterization. In the context of sizing optimization of an FE mesh, this means to select groups or "clusters" of elements in the geometry, consisting of shell elements, which thicknesses are varied during the optimization. In this work, parameterization is performed on the mesh in the pre-processor ANSA by grouping sets of elements. After the initial selection, a variable screening is undertaken to determine the statistical contribution of each variable to the response(s), thus making it possible to proceed with the most impactful variables.

Parameterization Approaches

Practically, the parameterization may be performed either on CAD- or mesh-level. The former step may introduce issues with automatic meshing, while the latter may be less flexible. Parameterization at the mesh level is used in this work. Thickness can then be described parametrically, as in by a parameter in a DoE. The parameterized areas can be chosen via a process of iterative sensitivity analysis where the geometry is split into successively smaller areas in regions with a high contribution to the total variance. This approach has been demonstrated in previous work in sizing optimization [68]. The benefits of this approach are that: (1) the selection is more likely to result in a set of relatively sensitive areas than an ad hoc selection and (2) a greater intuitive understanding of the system can be gained from narrowing in on increasingly sensitive areas even before running the MDO. Alternatively, rather than running multiple sensitivity analyses with successively decreasing parameter areas, a single parameterization of the geometry can be performed by fundamental disciplinary knowledge and previous studies of the component under optimization

[9]. This approach is arguably less rigorous; however, it may benefit from being quicker when previous knowledge exists.

Regardless of approach, an important aspect to consider in the parameterization and the subsequent variable screening process is the number of variables, also called *parameters* in this context, to include when proceeding to space-filling DoE(s) for metamodeling. There exists a fundamental trade-off between resolution (i.e., the number of parameter areas in the geometry) and computational expense since both are functions of parameter count but with opposite signs, see Figure 4.6. When the number of parameters is low, results have limited applicability. When the number of parameters is high, the computational expense may be unacceptably high. The optimal number of parameters is located between these extremes, but it is difficult to ascertain where, since the usefulness of results is hard to quantify. For practical purposes, modeFRONTIER practitioners recommend 4-6 DVs as the optimum interval for metamodeling and a maximum of 10 DVs [37]. This is the number of variables, which are the outputs of the screening process.

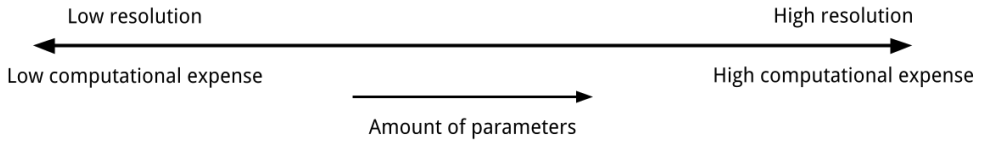


Figure 4.6: Parameter continuum between undesirable extremes.

Screening

Once an initial parameterization has been performed, a statistical DoE, as described in Section 2.4.1, is run and subsequently analyzed to filter out variables with a low contribution to the response(s). Many models have been proposed and demonstrated for variable screening, see Cho et al. [69] for a survey. Recognizing the need for ease-of-use as identified in Section 4.1, this work uses and recommends the built-in sensitivity analysis tool in modeFRONTIER, which uses a proprietary SSANOVA algorithm. This statistical modeling algorithm can estimate the contribution of each variable to the global variance [70].

Subsequent filtering of cumulative effect at, for example, 95% screens out the variables that together contribute to less than 5% of the total response. Alternatively, a maximum number of design variables can be set, e.g., 4-6 as per practitioner recommendations, which filters out variables that contribute less to the total variance than the 4-6 variables that proceed to be used for metamodeling. The latter approach requires that the filtered variables do not exceed some acceptable level of contribution and therefore, the former is to be recommended. The filtered-out variables can be set at a nominal or minimum value (or potentially maximum value depending on the context). If more variables than what can be allowed for a sufficient MDO run-time are left after screening, the parameterization may need to be reconsidered.

Illustrative Example

An illustrative example using the Development Geometry is shown below. A single crash load case is used as per Eq. 4.1, for clarity. However, the principle is the same if more load cases are used. See Figure 4.7 for initial parameterization, Figure 4.8 for a sensitivity analysis based on a Plackett-Burman ($n = 8$) DoE and modeled with SSANOVA. With a limit of 5% for cumulative effect for each response, one variable; "small_box_1", can be filtered out and set at as a constant, e.g., at the minimum value. Parameter resolution then increases in the next iteration and the non-sensitive area is screened out, see Figure 4.9. Observe parameter name changes from Figure 4.7 to Figure 4.9. As for the initial parameterization, a more refined iteration is subjected to a sensitivity analysis using P-B ($n = 12$), see an effect bar chart in Figure 4.10. Again, using a cumulative effect limit of 5%, the following parameters can be screened out: "boundary_side", "impact_side", and "inner_side_1", thus reducing the number of design variables from 8 to 5. The screened-out parameters can be set as constants and are then transferred from the $\mathbf{x}$ vector to the $\mathbf{p}$ vector before optimization is performed.

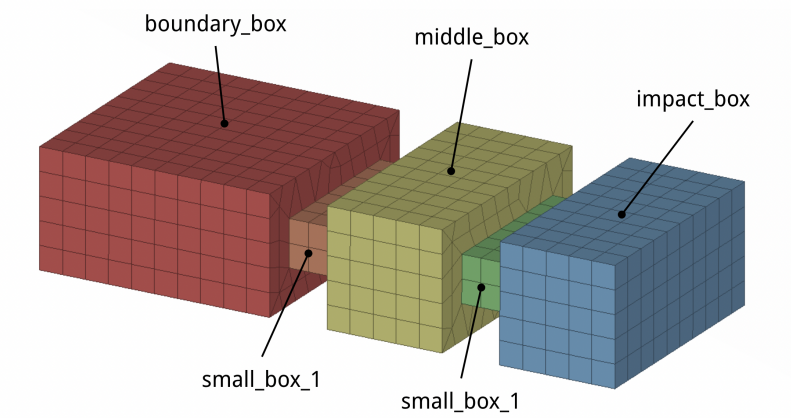


Figure 4.7: First parameterization of DG, $n = 5$.

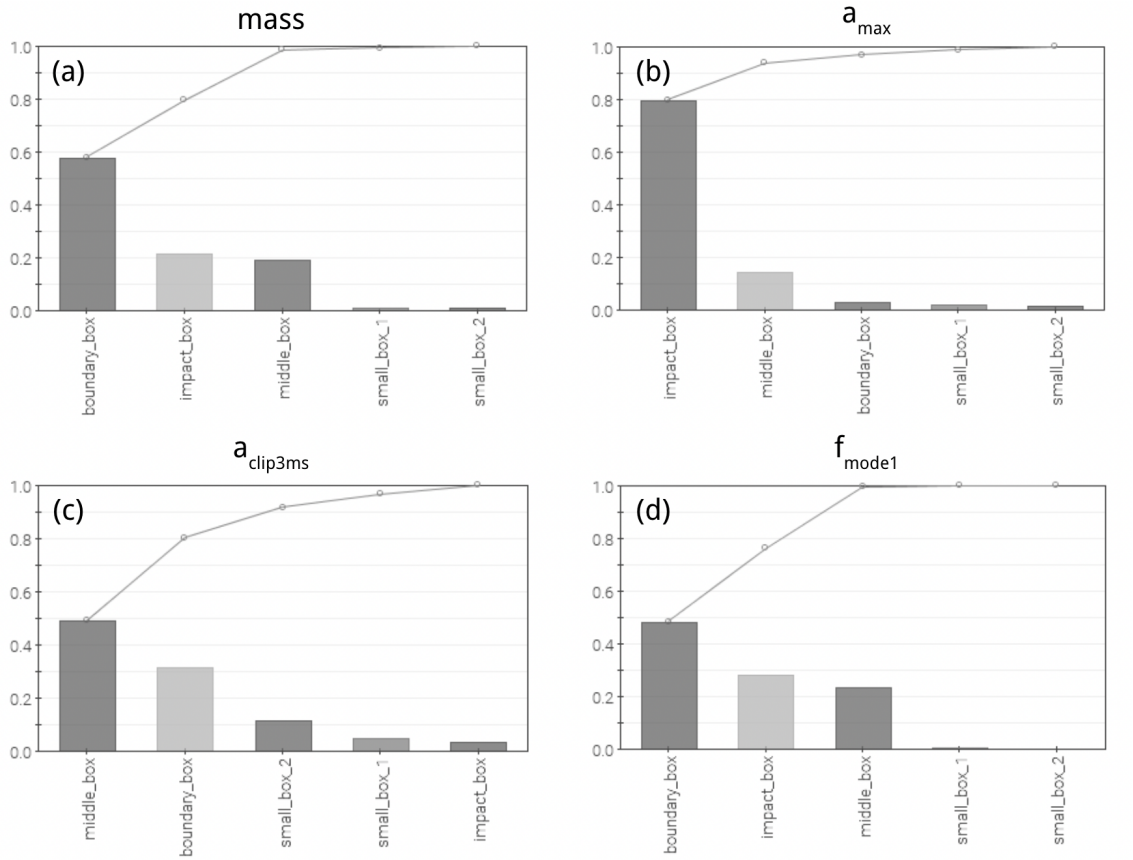


Figure 4.8: Effect bar charts for the first parameterization with four responses: (a) mass, (b) maximum acceleration a_{max}, (c) clip3ms acceleration $a_{clip3ms}$, and (d) eigenfrequency of the first mode f_{mode1}.

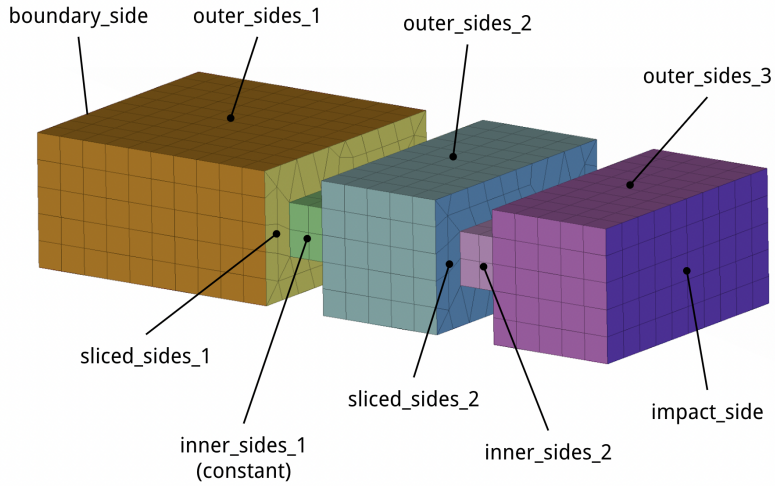


Figure 4.9: Second parameterization of DG, $n = 8$. Note that boundary_side, sliced_sides_1, and sliced_sides_2 are completely or partially obscured.

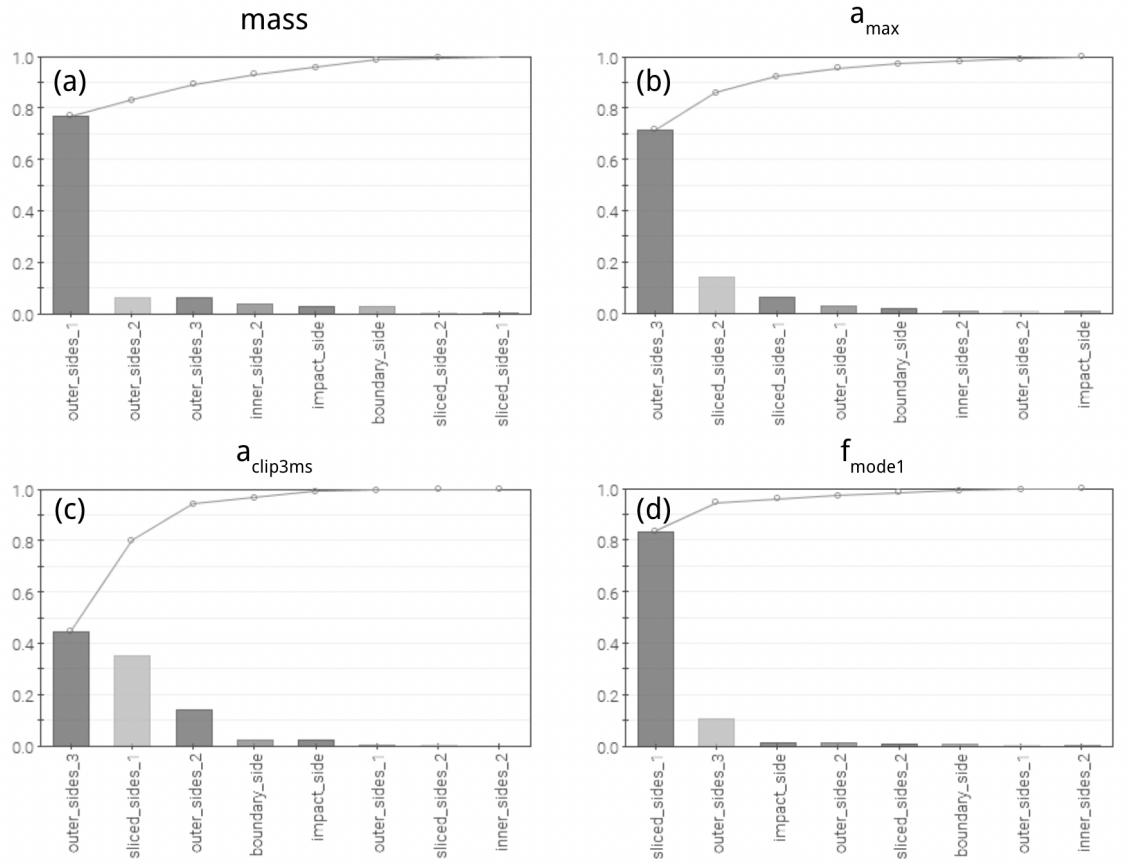


Figure 4.10: Effect bar charts for the second parameterization with four responses: (a) mass, (b) maximum acceleration a_{max}, (c) clip3ms acceleration $a_{clip3ms}$, and (d) eigenfrequency of the first mode f_{mode1}.

4.4.3 Metamodeling

The proposed metamodeling process is in general as described in Section 2.5. The following text expands on the motivation and details for consideration. In particular, aspects affecting the metamodel selection are discussed and a parametric study of metamodel error as a function of DoE size was undertaken.

Motivation for Metamodels

Beyond the reasons why metamodels are used in optimization, as stated in Section 2.5, their use in the automotive industry has also been suggested by Ryberg, Bäckryd, and Nilsson [6], [9], [61]. Furthermore, the authors of this thesis agree that if metamodels can be constructed with acceptable error, they are recommended also due to the increased flexibility. For example, many optimization algorithms can be employed on the metamodel without having to run extra simulations. Decreased computational cost for cases where computationally heavy simulations are required, e.g., crash analysis, as compared to direct optimization, is also a significant benefit. As mentioned, metamodel-based optimization also has noise-reducing properties, which can be beneficial, especially for crash analysis where the response(s) can be

noisy.

Cases when metamodel-based MDO is not recommended are where the above assumptions do not hold or in cases where bifurcations make results highly nonlinear and unpredictable, see interview findings in Section 4.1.4. Crash load cases dominated by bending do not have a tendency to bifurcations [52] but designs that are prone to bifurcation must be avoided if optimization is intended to be used as this is not exclusively an issue with metamodel-based optimization.

Metamodel Algorithm Selection

Several factors influence the choice of metamodel algorithm: the underlying phenomenon being modeled and whether its response is linear or nonlinear, response noisiness, required DoE size and uniformity, and allowable time for training. In general, practitioners are advised to start with simple metamodels, e.g., low-order polynomial regression to survey trends and behavior in low-resolution [37]. However, as covered in Section 2.5.1, more complex behavior cannot be captured by low-order polynomials and higher-order polynomials are costly to train. Since many systems in engineering exhibit complex relationships between inputs (DVs) and outputs (responses), more advanced metamodels, e.g., Kriging or RBFs, can be required to capture this behavior. In general, this work uses RBFs to model low-noise responses due to their lower required DoE size than polynomial regression. Kriging with a noise parameter is used for noisy responses. Responses that are are known to be linear are modeled with first-order polynomial regression.

Design of Experiment Size

Regardless of which metamodels are chosen, one significant aspect in the MDO process is the size of the space-filling DoE, which will influence metamodel accuracy and hence optimization accuracy. It has been suggested in the literature that a sample size of $3n$ (excl. validation samples) constitutes the minimum for a crashworthiness metamodel when fitting using a variety of metamodel algorithms [71]. Although notably, this study concerned a full frontal impact case. Shi et al. [72] confirms the $3n$ recommendation after testing up from $n + 1$ to $10n$ for a Subset Selection Regression [73] and two RBF variants. Here the development geometry, with 5 DVs and a problem formulation described in 4.4.2, is used to give a rough idea of how DoE size affects metamodel error for two disciplines of varying linearity and noisiness. Two polynomials (one first- and one second-order), Kriging and an RBF are compared. Four sample sizes have been used: $3n$, $5n$, $10n$, and $25n$ (again excl. validation points). The validation sets varied between 15-20% for each sample size. A ULH was used for both $clip3ms$- and f_{mode1}-models, see Figure 4.11.

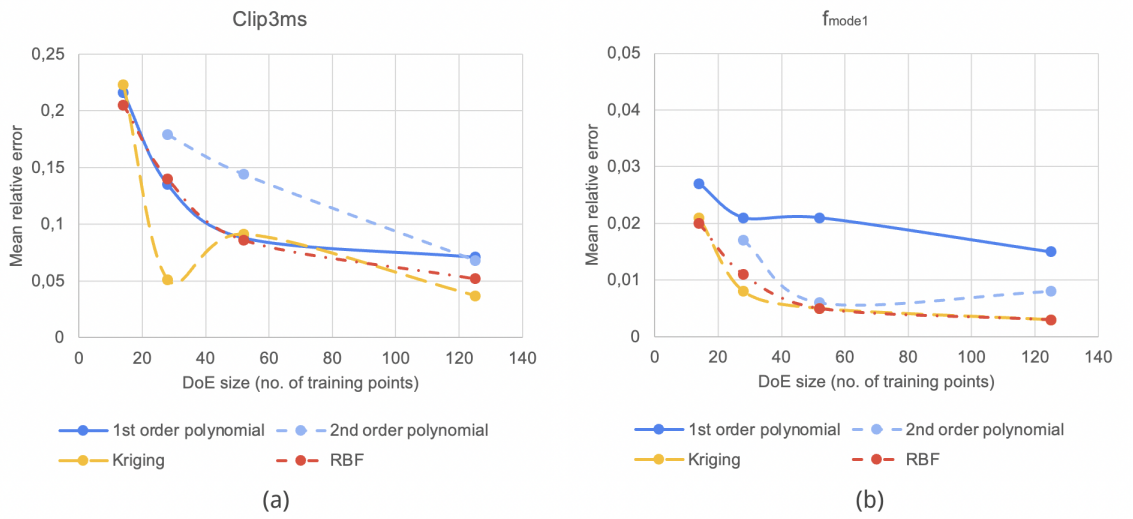


Figure 4.11: Mean relative error as a function of DoE size in $clip3ms$ and f_{mode1} for the Development Geometry. 1st and 2nd order polynomial regression, Kriging, and RBF are compared for $3n$, $5n$, $10n$, and $25n$ with $n = 5$ and 15-20% validation sets for each respective DoE (excluded in DoE size on x-axis).

Since the validation sets of randomly chosen points of 15-20% of the total DoE size corresponds to only a small amount of points, only the rough trend of Figure 4.11 is of importance. In both cases, barring outliers, the general trend is a strictly decreasing function with diminishing returns when the DoE size increases. However, it should be noted that the results are dependent on geometry and load conditions.

With regards to the process as outlined in Figure 4.5, if the metamodel errors are unacceptably high, DoE size can be increased to lower the error, or a different metamodel selection may be made before proceeding to optimization.

4.4.4 Metamodel-Based Optimization

Once a metamodel has been chosen for each discipline and load case based on the procedure described in the preceding sections, optimization algorithms are next applied to perform metamodel-based optimization. The selection of an appropriate algorithm depends on several factors, e.g., whether the problem is convex or multimodal, whether gradient information is available, if variables are continuous or discrete, and what the allowable time for the optimization is. As suggesting a general guide of which optimization algorithm is appropriate for each hypothetical case is beyond the scope of this work, a brief discussion on what guided the optimization algorithm selection in the present work follows.

The problem described in Section 4.3 is partly a nonlinear, multimodal system that is run with discrete variables (in increments of 0.1) and metamodeled with surfaces that have gradient information. Since the optimization is based on a metamodel, the time for the optimization to converge is negligible to the required time to compute

all points in the space-filling DoE. This allows the use of complex, nongradient optimization algorithms with minimal computational cost. Therefore MOGA-II, which is often computationally expensive in direct optimization, can be used. MOGA-II can efficiently optimize in both discrete and continuous variables using multiple objectives [37], i.e., in MOO problems. However, one disadvantage is that this algorithm can be challenging to tune manually, sometimes necessitating a trial-and-error approach. However, modeFRONTIER offers an automated approach termed "self-initializing mode" that takes several evaluations as input and sets remaining parameter settings automatically.

If after optimization it is the case that feasible solutions to the design problem exist, the next step in the MDO process is undertaken. However, if no such feasible solution exists, either the problem formulation or the pre-optimized concept requires modification. This assumes that previous steps of the process have been without error, such that variable selection has not screened out too much variance or that metamodels have too high errors.

4.4.5 Validation at Optima

While the metamodels are compared using simple cross-validation in the metamodeling stage, see Section 4.4.3, validation at every optimum point $\mathbf{x}^*$ can also be undertaken. This is to check the local accuracy at that point to avoid a spurious optimum. Practically this is performed by running a simulation for every discipline and load case at the point $\mathbf{x}^*$ and comparing the responses to the responses given by the metamodel-based optimization. If the error is unacceptable, the metamodel algorithm is changed for another one, or it is kept and modeling parameters are changed. Alternatively, the model's training data set can be expanded, i.e., by increasing the size of the space-filling DoE, as in previous process steps.

A further reason why validation at the optima is necessary is because simple cross-validation points for assessing the error of the metamodels are randomly distributed in the space. It can therefore be the case that the true maximum error between the metamodel and the real response is higher than the measured maximum error because the validation points which are used to calculate errors can be anywhere on the multidimensional function as opposed to exactly at the point of true maximum deviation. See Figure 4.12 for a schematic illustration for a single variable. The same principle holds for multidimensional functions.

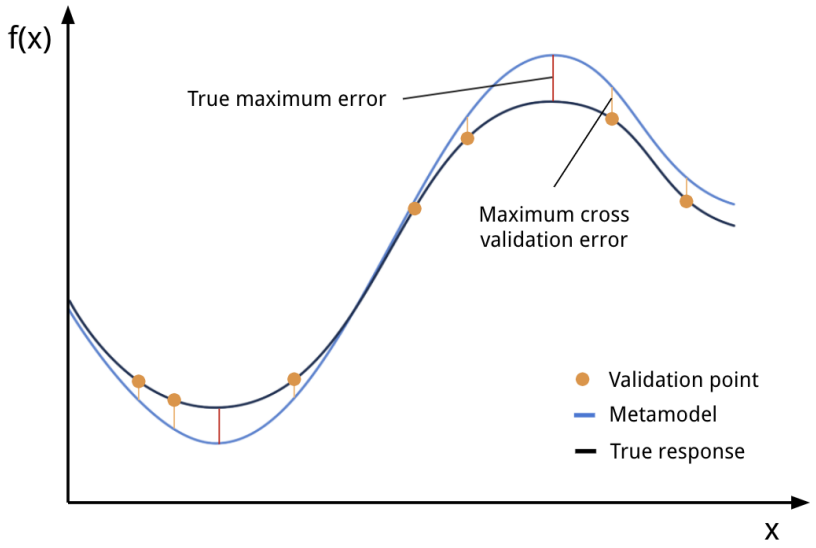


Figure 4.12: Schematic illustration of true maximum metamodel error as compared with maximum simple cross validation error.

Furthermore, the optimization may yield a point slightly outside constraints due to local error in the metamodel. Even if the metamodel performance can be improved by increasing the DoE size or changing the training parameters of the metamodel, it is still possible that the constraint violation will persist. In these cases, a possible solution is to adjust the problem formulation to impose more strict constraints and then re-run the optimization and check for constraint violation again. Rather than changing the actual problem formulation for the design problem, which is defined by the engineering requirements, the constraints limits in the modeFRONTIER nodes are offset to account for the metamodel error. There is to the authors' knowledge no easy way to determine the magnitude of this offset. However, since the metamodel-based optimization problem can be solved quickly, several iterations for fine-tuning may be considered affordable as compared to the DoE, which will require orders of magnitude more time.

4.4.6 CAD Interpretation

As cautioned at the outset, see Section 1.1.3, MDO does not by itself provide a finished design, at least not by the formulation discussed in this work. In addition to constructing the MDO and making decisions throughout the process, engineering intervention is needed to interpret the results of the MDO. A CAD realization of the optimized mesh is made after analyzing the optimization results since the results are typically not directly manufacturable, e.g., due to abrupt changes in thickness for a thickness-based optimization.

It is, therefore, to some degree up to the design engineer what geometric features should be used. For example, a thickness-optimized component to be cast may not take the exact optimized thickness at each point but may potentially utilize rib

structures at select areas, assuming the optimization results are interpreted as that more material is needed in such areas to increase stiffness locally. However, the more one deviates from the optimization results during the interpretation, the less useful the MDO results become.

Subsequent to CAD realization, the geometry is meshed and tested against the initial requirements and objectives defined at the outset. If requirements are not met, some point in the MDO process needs to be reconsidered. Examples include re-interpreting the CAD model, changing the parameterization, or more fundamentally; modifying the underlying pre-optimized concept or reconsidering the problem formulation. However, if the optimized and interpreted geometry meets the requirements, the MDO has been successful.

5

Process Verification on an Automotive Component

In this chapter, the MDO process as developed and described in Chapter 4 is applied to a real design problem, namely the Center Stack Display Bracket (CSDB) problem. The two problem formulation variations that are used are presented, followed by variable selection, metamodeling, optimization, and CAD interpretation as per Figure 4.5.

Note that throughout this chapter, some absolute values (e.g., for constraints) and figures of the component under optimization are omitted for confidentiality, relative values are however presented that retain the magnitude of the results. The geometry of the bracket is also not shown due to ongoing development.

5.1 Problem Formulation

A CSDB is subject to three crash load cases, see Figure 5.1, and a modal analysis. During the verification stage of automotive interior development, many impact points are tested. In this work, a selection of impact points has been chosen to be used after discussion with crash experts for their representativeness. The impact points are $P1$, $P2$, and $P3$ where a head impact is simulated on the CSD. Eigenfrequency of the first mode is of interest as in Section 4.3.

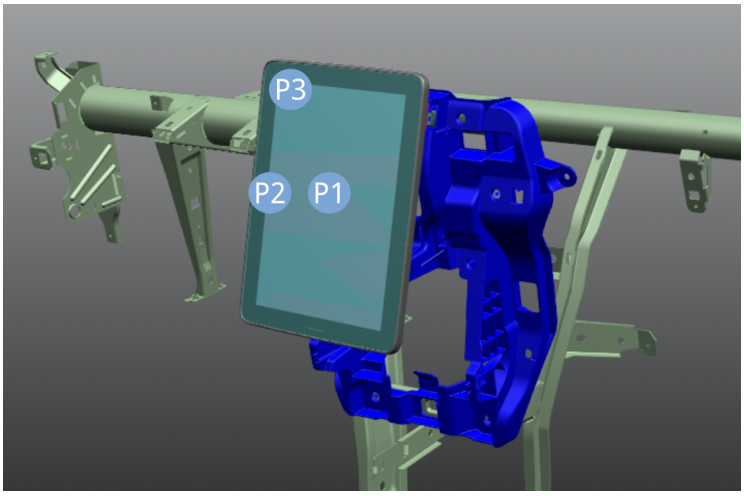


Figure 5.1: Points of head impact on the CSD for load cases $P1$, $P2$, and $P3$.

Load cases and responses for the DG are similar to the development problem in Chapter 4 with the exception of more crash load cases:

- Three load cases $P1$, $P2$, and $P3$ with head impact responses as measured by a_{max} [G] and $a_{clip3ms}$ [G], see Figure 2.1.
- Modal response f_{mode1} [hz].
- Mass m [kg] from the mesh model.

Formulation 1 (min mass) can be formulated as:

$$\begin{aligned}
&\min\ m(\mathbf{x},\mathbf{p})\\
&\text{subject to}\\
&g_{1,P1}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&g_{2,P1}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}]\\
&g_{3,P2}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&g_{4,P2}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}]\\
&g_{5,P3}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&g_{6,P3}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}]\\
&g_{7}(\mathbf{x},\mathbf{p}) - \mathbf{f}_{mode1} \leq \mathbf{0}\ [\text{Hz}]\\
&1.2 \leq x_i \leq 5.0\ [\text{mm}]
\end{aligned} \tag{5.1}$$

where mass m, as a function of design variables $\mathbf{x}$ and parameter constants $\mathbf{p}$ (e.g., material properties), is minimized subject to constraints a_{max}, $a_{clip3ms}$, and f_{mode1} while design variables can vary between lower bound 1.2 mm and upper bound 5.0 mm. Vectors $\mathbf{a}_{max}$, $\mathbf{a}_{clip3ms}$, and $\mathbf{m}_{max}$ are vectors with the dimension n x 1 where n is the number of design variables and all values in the matrix are identical.

Formulation 2 (MOO) can be formulated as:

$$\begin{aligned}
&\min\ f_{clip3ms,P3}(\mathbf{x},\mathbf{p}), 1/f_{f,mode1}(\mathbf{x},\mathbf{p})\\
&\text{subject to}\\
&g_{1}(\mathbf{x},\mathbf{p}) - \mathbf{m}_{max} \leq \mathbf{0}\ [\text{kg}]\\
&g_{2,P1}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&g_{3,P1}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}]\\
&g_{4,P2}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&g_{5,P2}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{clip3ms} \leq \mathbf{0}\ [\text{G}]\\
&g_{6,P3}(\mathbf{x},\mathbf{p}) - \mathbf{a}_{max} \leq \mathbf{0}\ [\text{G}]\\
&1.2 \leq x_i \leq 5.0\ [\text{mm}]
\end{aligned} \tag{5.2}$$

where two objectives; $f_{f,mode1}(\mathbf{x},\mathbf{p})$ i.e., eigenfrequency response is maximized (hence the inversion) and $f_{clip3ms,P3}(\mathbf{x},\mathbf{p})$ i.e., clip3ms response, is minimized with respect to design variables $\mathbf{x}$ under constraints similar to formulation 1, see Eq. 5.1. In both formulations, the vector $\mathbf{x}$ contains n variables x_i where $i = 1, 2, \cdots, n$, initially, $n = 8$. Minimization of clip3ms for $P3$ is chosen as an objective due to it

being the most difficult to lower as compared to $P1$ and $P2$ according to discussion with crash analysis experts.

A multi-objective optimization (MOO) is used to explore the trade-off between the responses that have the most difficult targets to meet. After optimization it has been discovered that no constraints are active in Eq. 5.2 and hence MOO captures the entire trade-off between clip3ms at point $P3$ and f_{mode1}. Due to this, single-objective optimization (SOO) formulations in which either of the aforementioned objectives are optimized individually are unnecessary because their optima are found in the Pareto front in the MOO.

5.2 Variable Selection

As mentioned in Section 4.4.2, there exist multiple strategies for deciding what areas of the geometry are used as design variables for the subsequent optimization. In the case of the CSDB, previous studies and engineering expertise guided the selection of areas to be included in the sensitivity analysis as part of the variable screening. Alternatively, a successive refinement of parameter areas through repeated sensitivity analyses could have been performed as illustrated in Section 4.4.2. In this application, ten parameters, each representing an area of the shell mesh, were selected. An assumption of symmetry along the XZ-plane, i.e., the plane which separates the left and the right side of the car, is made for several areas such that a single parameter controls the thickness on two symmetrical areas of the design. This was done to increase resolution and because the impact points are primarily on the left side, a further discussion follows in Section 6.3. Thicknesses were varied in these areas.

A Plackett-Burmann DoE with 12 runs due to the ten parameters was used to populate the experimental domain, after which an SSANOVA sensitivity analysis was run. The first iteration with screening for 86.5% contribution to the total variance, i.e., a 13.5% threshold, resulted in 5 DVs across all disciplines and load cases. However, after proceeding with this selection, optimization resulted in exclusively infeasible results (i.e., violating constraints) when using a MOGA-II optimization for the “min mass” formulation, see Eq. 5.1. More design variables needed to be included at the cost of metamodel DoE size to capture more contribution to the total variance.

A second iteration of the variable screening was performed. Setting a threshold of 9.5% resulted in 8 DVs, see Table 5.1.

Table 5.1: Contribution to response with a 9.5% threshold. Note that parameter 2 and 6 are below the threshold and thus screened out. Total indicates the amount of variance that is kept for each response after screening.

Var.	**clip3ms**$_{P1}$	**clip3ms**$_{P2}$	**clip3ms**$_{P3}$	$\mathbf{f}_{mode1}$	$\mathbf{a}_{max,P1}$	$\mathbf{a}_{max,P2}$	$\mathbf{a}_{max,P3}$
1	0.4%	6.8%	7.6%	45.1%	1.6%	8.8%	39.3%
2	0.6%	2.6%	1.7%	7.6%	0.0%	8.3%	2.3%
3	0.6%	13.1%	10.0%	4.4%	0.3%	7.0%	6.1%
4	3.7%	74.6%	1.2%	0.2%	0.4%	69.5%	3.5%
5	0.2%	0.0%	0.7%	3.4%	0.2%	0.0%	11.6%
6	0.1%	0.6%	2.8%	0.0%	0.0%	0.0%	9.1%
7	0.6%	1.6%	13.1%	4.1%	0.0%	1.8%	0.5%
8	0.0%	0.0%	3.7%	3.2%	0.1%	1.5%	19.1%
9	93.7%	0.6%	38.2%	31.9%	97.3%	0.9%	0.1%
10	0.1%	0.1%	21.2%	0.1%	0.0%	2.2%	8.4%
Total	**98.4%**	**96.8%**	**95.5%**	**92.4%**	**99.9%**	**91.7%**	**88.6%**

Although outside the optimal scope of 4-6 DVs as recommended by practitioners, see Section 4.4.2, a larger DoE and thus a longer computation time was in this case acceptable for the purposes of attempting to meet constraints. Optimization then proceeded with these 8 DVs.

5.3 Metamodeling

The same metamodeling algorithms used in Section 4.4.3, polynomial regression, Kriging, and RBF for linear, noisy, and noise-free responses respectively, were trained on a data set with a sample size of $6n$ for 8 DVs. $10n$ or above was considered excessive due to already acceptably low errors. In general, the trend illustrated in Figure 4.11 likely holds, but it is difficult to give a general recommendation of what DoE size to favor because it depends on the level of acceptable error and level of nonlinearity and noise in the responses.

As for the Development Geometry (DG), metamodels were compared using mean relative and maximum relative error measures (termed RE_{mean} and RE_{max} here to avoid confusion) as calculated from a simple cross-validation data set with 15% of the ULH DoE being validation points. The training set consisted of 46 sample points. See Table 5.2 for metamodel performance for the selected metamodels.

Table 5.2: Selected metamodels, mean relative error, and maximum relative error measures for each response.

Response	**Metamodel**	$\mathbf{RE}_{mean}$	$\mathbf{RE}_{max}$
$clip3ms_{P1}$	Kriging	2.70%	7.15%
$clip3ms_{P2}$	Kriging	3.46%	7.55%
$clip3ms_{P3}$	Kriging	3.76%	7.41%
f_{mode1}	RBF	0.34%	1.41%
$a_{max,P1}$	Kriging	0.59%	2.10%
$a_{max,P2}$	Kriging	5.40%	10.08%
$a_{max,P3}$	Kriging	2.84%	7.78%

If lower metamodel errors are desirable, a DoE size larger than $6n$ is recommended. For the present purposes, the metamodels are considered to perform acceptably well, so the optimization proceeds with the set of metamodels.

A linear metamodel (first order polynomial regression) was trained for the mass response since modeFRONTIER requires a metamodel for each output to run a metamodel-based optimization; however, it is omitted from in Table 5.2 due to the perfectly linear relationship between mass and thickness.

5.4 Metamodel-Based Optimization

For reasons described in Section 4.4.3, the MOGA-II optimization algorithm was used to perform metamodel-based optimization in both problem formulations. The optimization was completed in under 10 seconds even for over 1000 evaluations due to running on the metamodels. The resulting Pareto front for the MOO formulation is shown in Figure 5.2. The point denoted “PP1” refers to a point on the Pareto front which was considered to be a “good” trade-off, see Figure 5.2.

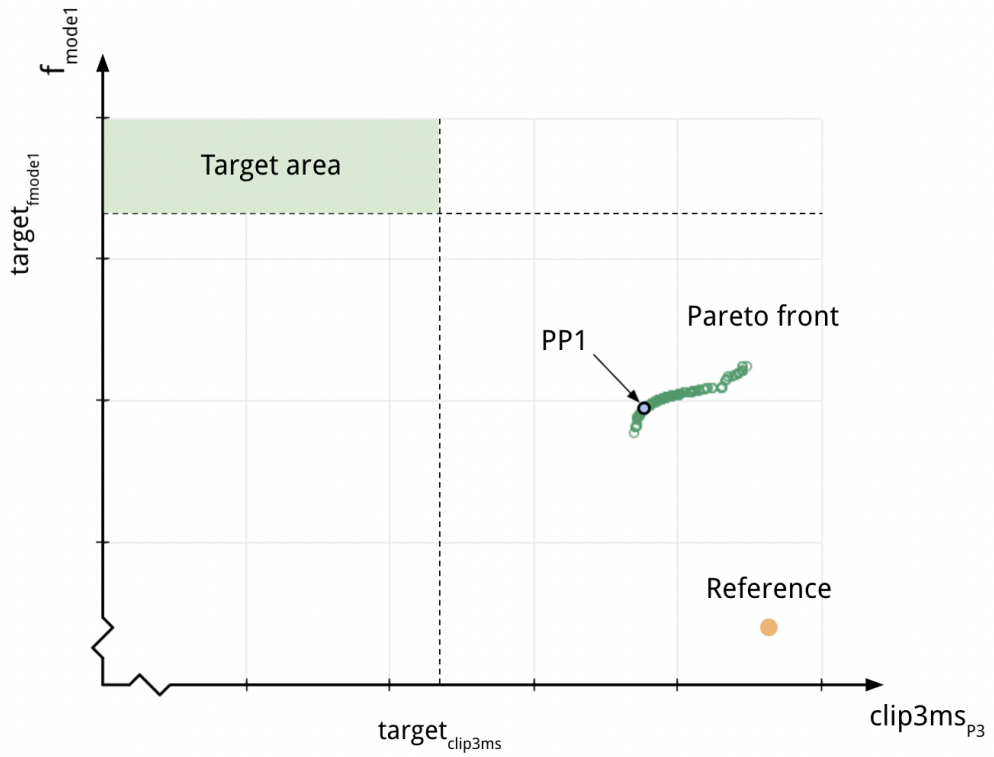


Figure 5.2: Pareto front from metamodels in relation to the reference and the targets on max-min MOO formulation (f_{mode1} is maximized and $clip3ms_{P3}$ is minimized) illustrating the trade-off between clip3ms and modal performance. Values on axes have been removed for confidentiality.

The above trade-off curve is based on the metamodels which are known to have varying error throughout the response space. Therefore, a small set of points (in this case five) are picked from the Pareto front and compared with real simulations to investigate the error at optimum. The points perform similarly on $clip3ms_{P3}$ and f_{mode1} and hence the point with the lowest error in critical responses is picked, here called "PP1". See Table 5.3 for the error (difference between response values as given by the metamodel and by the real optimization) at optimum for each response at the picked point.

Table 5.3: Response error between metamodel and real simulation at optimum for a selected point on the Pareto front. Sign denotes if the real response value is higher or lower than the metamodel response value.

Response	**Error**$_{MOO}$
$clip3ms_{P1}$	-0.60%
$clip3ms_{P2}$	2.00%
$clip3ms_{P3}$	-3.30%
f_{mode1}	-0.60%
$a_{max,P1}$	-0.10%
$a_{max,P2}$	2.70%
$a_{max,P3}$	-3.30%

After validating the optimum points for the MOO and the min mass formulation, the validated values are benchmarked against a reference, in this case, the pre-optimized design, see Table 5.3.

Table 5.4: Comparison between problem formulations as compared to a reference benchmark, i.e., the pre-optimized design. Response improvement (marked in green) or deterioration (marked in red) is compared to the benchmark reference value.

	$\mathbf{clip3ms}_{P1}$	$\mathbf{clip3ms}_{P2}$	$\mathbf{clip3ms}_{P3}$	$\mathbf{f}_{mode1}$	**mass**	$\mathbf{a}_{max,P1}$	$\mathbf{a}_{max,P2}$	$\mathbf{a}_{max,P3}$
Benchmark	Ref	Ref	Ref	Ref	Ref	Ref	Ref	Ref
minimize mass	30.30%	5.71%	-6.12%	5.80%	20.92%	5.63%	0.30%	-4.89%
MOO	43.80%	11.75%	-6.14%	6.36%	35.36%	6.69%	7.26%	-3.63%

Neither formulation (min mass and MOO) has feasible solutions such that clip3ms for $P3$ and f_{mode1} are within the requirement, see "target area" in Figure 5.2. However, the MOO formulation resulted in an improvement of both as compared to the benchmark from the selected point on the Pareto front: circa 6% lower clip3ms at $P3$ and 6% higher f_{mode1}. The min mass formulation had similar performance improvement while less mass was added, see Table 5.4.

All constraints in the MOO formulation were satisfied; however, as shown in Figure 5.2, no point in the Pareto front is in the target area while constraints are respected.

5.5 Optimization Infeasibility

In optimization, the term "feasibility" is used to denote whether there is a solution to the optimization problem. Using the above formulation, see Eq. 5.1, it was shown that no feasible solution exists. The second formulation, see Eq. 5.2, technically has feasible solutions since no constraint is violated. However, it fails to meet the targets and can hence be described as infeasible. There are several reasons why the optimization may fail to arrive at a feasible design, these are briefly explored in this section.

Fundamentally, the design space may be too restrictive to allow the optimization to reach some set performance target because the target lies outside the scope of the optimization. For instance, the interval bounds of the design variables may be too narrow, or the parameterization may be restrictive in that not all relevant geometrical features can be optimized simultaneously. In the case of fine-tuning a mature component in the later stages of development, for instance, by using a thickness-based optimization like in this work, much of the geometry is already defined, thus allowing less design freedom in the optimization process, this may be termed *geometry lock-in*.

It may also be the case that components outside the system under optimization influence the system in a way that unexpectedly limits its performance. If, for instance, some components outside the system under optimization interact in a way that places a limit on the system, no amount of optimization will result in a solution

that satisfies some optimization formulation. Performing the optimization may still be useful because it provides insight into how far the current design can be pushed under the limits of the optimization. If the assumptions imposed by the parameterization and optimization formulation are appropriately set, and a feasible solution is nevertheless not found, issues may be caused by (1) the concept being optimized, (2) some interaction or component outside the system under optimization.

In the case of the CSDB, no design exists within the feasible space, also called the target area in Figure 5.2. Therefore, a second, modified optimization was run. Since some design variables were at their upper limit in many Pareto designs, their upper, as well as their lower bounds were changed to cover a larger area. The upper bound was increased by 40% while the lower bound was decreased by 17% (i.e, to 7 and 1 mm respectively). The selection of these particular values was guided by disciplinary experts who considered manufacturing limitations.

It was also discovered that a nearby component, not within the scope of the optimization, limited crash performance significantly. The mechanism was that this component interacted with other components in the IP assembly in an unexpected way during crash simulations, which effectively placed a lower limit on clip3ms. The complicating aspect that head impact simulations are performed on the level of a large instrument panel assembly with many components obfuscated the mechanism by which clip3ms-performance was limited. This is the reason why the Pareto front in Figure 5.2 is located entirely to the right of the clip3ms target, which in combination with current modal performance motivated additional investigations.

5.6 Second Multi-Objective Optimization

An additional run-through of the MDO process with extended DV bounds and with changes to the associated component that limited clip3ms-performance was performed. The MOO formulation, see Eq. 5.2, but with $1.0 \leq x_i \leq 7.0$ [mm] and with $n = 10$ for $\mathbf{x}$. Previously screened-out variables contained in the $\mathbf{p}$ vector are now in $\mathbf{x}$.

All ten design variables were used in a space-filling DoE to train the metamodels. No variable screening is thus performed to capture all variance. The reason for omitting the variable screening is to identify how close to the target area the optimization can get. An approximately $6n$ run with a 15% validation set was used to train a set of metamodels with the following fit performance, see Table 5.5.

Table 5.5: Selected metamodels and MRE measures for each response for the second MOO.

Response	Metamodel	$\mathbf{RE}_{mean}$	$\mathbf{RE}_{max}$
$clip3ms_{P1}$	Kriging	4.77%	10.25%
$clip3ms_{P2}$	Kriging	8.28%	18.77%
$clip3ms_{P3}$	Kriging	6.39%	10.48%
f_{mode1}	RBF	0.83%	2.16%
$a_{max,P1}$	Kriging	1.06%	2.50%
$a_{max,P2}$	Kriging	8.15%	14.34%
$a_{max,P3}$	Kriging	5.23%	1.17%

The average errors are higher than in Table 5.2 which may be due to the DV bounds extending into a more nonlinear area or due to the additional parameters, i.e., 10 instead of 8 as in the previous MOO. For the purposes of exploring the design space and testing the limits of the current concept and optimization setup, the errors are considered acceptable. Lower errors could be obtained by running a DoE with a larger size as per the general trend described in Figure 4.11. An MOO with a formulation like Eq. 5.2, except the DV bounds, results in the Pareto front as shown in Figure 5.3. The new reference termed "2nd reference" is the pre-optimized concept run in the updated assembly without the complication with an associated component. A point on the Pareto front termed "PP2" is also picked according to the previously described procedure, see Section 5.4.

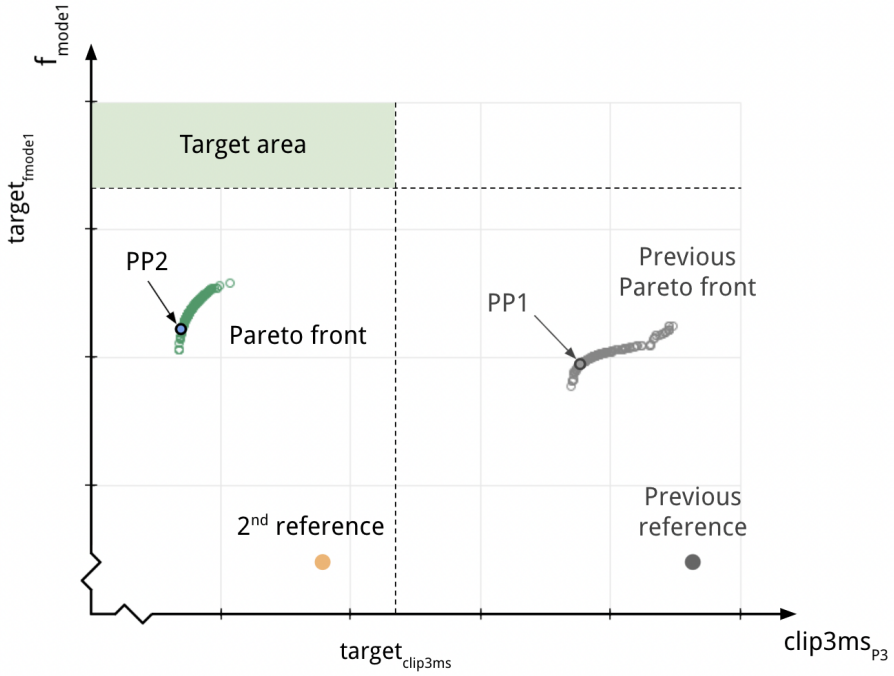


Figure 5.3: Second Pareto front as compared to the previous, see Figure 5.2, after changing DV bounds and fixing a complication with an associated component. Values on axes have been removed for confidentiality.

As is evident in Figure 5.3, the second Pareto front is closer to reaching the per-

formance targets for the most critical metrics as compared to the previous Pareto front, see Figure 5.2. However, all points on the Pareto front lies outside the target area, indicating that a feasible solution has not been reached with the current optimization setup or the current pre-optimized concept but that the former is closer than the latter. The target for clip3ms at $P3$ is however met, indicating that it was limited by the previously described component outside the scope of optimization.

From analyzing the optimization data, it can be ascertained that the constraints regarding maximum and clip3ms acceleration for point $P1$ are active, see Figure 5.4. The infeasible results, shown in yellow, against the feasible results, shown in green, indicate that crash performance is primarily limited by $P1$. A few points in the scatter plot, see points marked in yellow in the bottom left, are infeasible due to $P2$-constraints.

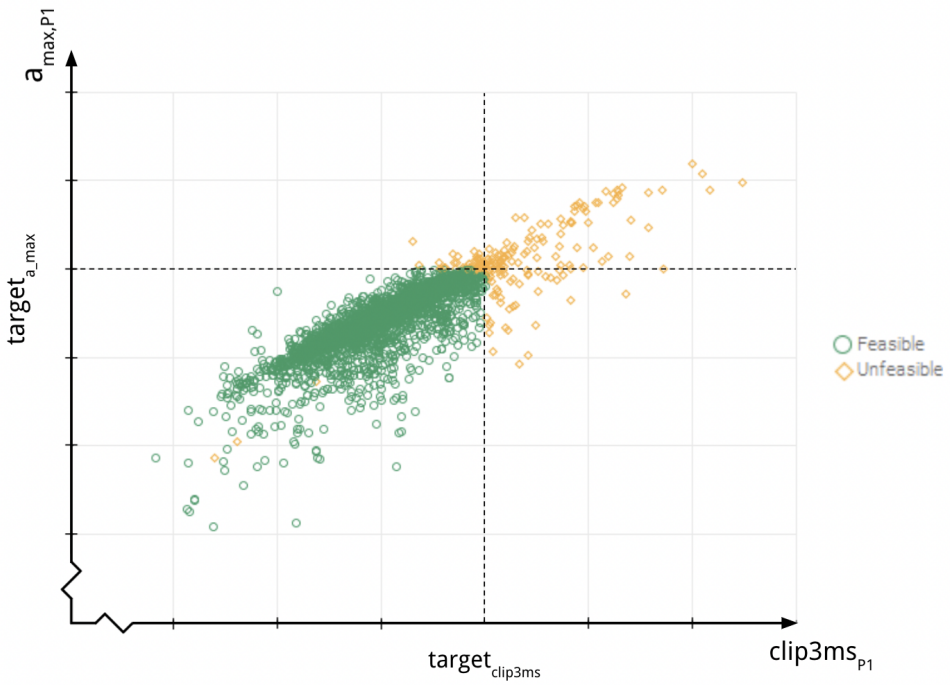


Figure 5.4: MOGA-II crash performance results for second MOO at $P1$. Values on axes have been removed for confidentiality.

Like in Section 5.4, the error at optimum (PP2) is investigated by comparing the metamodel responses to a real simulation using the optimizer thicknesses, see Table 5.6.

Table 5.6: Response error between metamodel and real simulation at optimum for a selected point on the Pareto front. Sign denotes if the real response value is higher or lower than the metamodel response value.

Response	**Error**
$clip3ms_{P1}$	3.70%
$clip3ms_{P2}$	4.70%
$clip3ms_{P3}$	-0.10%
f_{mode1}	-1.30%
$a_{max,P1}$	-1.50%
$a_{max,P2}$	6.80%
$a_{max,P3}$	1.30%

The errors between the metamodels and the validated point PP2 for each response are deemed acceptable. Furthermore, as with the first MOO, all constraints are respected. The validated values, i.e., from real simulations, are compared against the new reference, see Table 5.7.

Table 5.7: Second MOO as compared to previous problem formulations. Response improvement (marked in green) or deterioration (marked in red) is compared to the benchmark reference value.

	clip3ms$_{P1}$	**clip3ms**$_{P2}$	**clip3ms**$_{P3}$	**f**$_{mode1}$	**mass**	**a**$_{max,P1}$	**a**$_{max,P2}$	**a**$_{max,P3}$
Benchmark (2nd reference)	2nd Ref	2nd Ref	2nd Ref	2nd Ref	2nd Ref	2nd Ref	2nd Ref	2nd Ref
2nd MOO	49.93%	12.57%	-3.28%	7.55%	72.47%	5.98%	21.17%	-4.92%

While several responses show worse performance than the previous optimization attempts compared to their appropriate reference, see Table 5.4, all targets (except modal) are met. Compared to the reference, clip3ms at $P3$ is lowered by circa 3% while eigenfrequency at the first mode is increased by over 7%.

Modal Theoretical Maximum and Global Information

Since the more extensive, second MOO also failed to meet the modal requirement, a test was conducted to ascertain the limits of the component under optimization, i.e., the CSD bracket.

Since eigenfrequencies are functions of stiffness according to Eq. 2.1, a manual adjustment of stiffness can determine the limit of the component with respect to modal performance. The theoretical maximum eigenfrequency of the first mode of the bracket was investigated by changing the Young's modulus of the material in the FE modal by a factor of 100 for the pre-optimized concept. It was discovered that the limit was under the set target and only 9.6% higher than the benchmark, as compared to circa 7.6% with the optimization, see Table 5.7. An improvement of only 2 percentage points beyond the optimization results is therefore theoretically possible, meaning that the modal target cannot be reached without modification to other components than the CSDB.

The above results indicate the need for global information about a system and the inherent limitations of optimizing a single component in a larger system affected by many other components and their interactions. The difficulty of finding and the importance of including global information in global optimization has long been noted, e.g., in the article aptly named *Global Optimization Requires Global Information* by Stephens and Baritompa [74] from 1998. Further work with differing scope is therefore needed to meet requirements regarding modal performance.

5.7 CAD Interpretation

As noted in Section 4.4.6, MDO does not output a finished design but rather results that need to be interpreted. As discussed in Section 4.4.6, manufacturing limitations will impose changes to the optimized mesh, e.g., abrupt changes in thickness over the geometry need to be smoothed. A level of ambiguity in the CAD interpretation process is inherent, and therefore multiple CAD realizations can be created and compared if time allows. A detailed CAD realization is not made in the present case as it is outside the scope of the thesis.

6
Discussion

This chapter presents considerations of MDO methods from an organizational-, societal-, ethical-, and ecological perspective. Limitations of the study and the MDO process are both provided as an extension to the initial limitations listed in Section 1.3. Finally, project recommendations to improve the proposed process are presented.

6.1 Organizational Considerations

As described in the needs assessment in Section 4.1, the general perception of MDO seems to be that it is time-consuming, complex, and rigid (i.e., not dynamic to modifications to the system). Moreover, from the MDO implementation in modeFRONTIER it was discovered that organizational hindrances of utilizing MDO can also be related to a lack of FE model standardization for the involved disciplines. The implementation of MDO can be relieved with common FE models, if possible, shared across disciplines using standardized interfaces and numbering. Furthermore, large-scale utilization of MDO introduces difficulties in deciding what design variables to select, seeing as different disciplines aim for different objectives with their design.

Different priorities for the MDO method arise from when in the product development process it is meant to be applied. Early implementation typically faces difficulties related to frequent, drastic design modifications and also sudden requirement changes. Both aspects complicate the timing for when MDO results need to be provided to be useful, as current designs rapidly become outdated in the early stages of development. Thus, MDO implementation at an early stage in the development process should be focused on creating a process that allows for fast optimization loops rather than providing high-accuracy results. On the other hand, if MDO methods are implemented at a late stage in the development process, difficulties with limited design freedom can occur. The cause is that generally, at this stage, component development has arrived at a point where detailed models are tailored to meet each discipline's respective set of requirements.

If the organizational structure allows for it, one strategy for MDO implementation could be to allow each discipline to create and control its own metamodel [6]. The collection of metamodels could then be accessed across disciplinary CAE teams to share knowledge and gain insight into how modification suggestions from one dis-

cipline will affect the other ones. Furthermore, this would allow for geographically disbursed groups to utilize MDO while remaining in control of their domain in which they have expertise. However, as previously argued, this method would assume that the domain involved are able to agree upon unanimous design variables to use for the optimization problem.

A large-scale organizational MDO approach can be hard to implement since it can be too complex to effectively handle or too simplified to give useful results. If a local approach, i.e., implementing MDO for single components of the car, is adopted instead, knowledge can still be gained while optimization implementation is not as comprehensive since the scope is smaller. This can also be a strategy for large corporations to gradually introduce MDO, testing it at a small scale and scaling up if it proves successful. Although results from local MDO methods can sometimes be misguiding since optimization on a local system might not yield the same desired outcome for the global system.

6.2 Sustainability Considerations

While the thesis work and its area of focus are unlikely to have any major sustainability-related impacts, i.e., societal, ethical, or ecological outcomes, it is worth considering any such possible effects in outline.

MDO is fundamentally aimed at improving the efficiency of development efforts by formalizing coordination between different engineering disciplines. If successful, better products and faster development cycles are possible, arguably giving the company that employs such methods a competitive advantage, ceteris paribus. In the transport sector, a common objective in optimization is to minimize mass (subject to various constraints), and MDO has been shown to be effective towards this end [75], [76], [77]. Light-weighting of car components can improve fuel efficiency by approximately 0.4 L/100 km for every 100 kg reduction in cars [78]. Translating to CO_2 and other emissions, it is easy to see the savings in environmental and health impacts in ecological and societal areas, respectively.

Even disregarding weight savings in the automotive sector and adopting a wider frame of reference, it is clear that if MDO can deliver better-optimized products, then this impacts other sectors like life science and energy positively. Improvements in efficiency in the aforementioned areas can yield substantial societal and ecological benefits. It is furthermore unlikely that the introduction of MDO would displace R&D engineers due to more effective ways of coordinating multidisciplinary development since the engineering expertise is required to both set up models, execute the optimization and interpret the results. However, according to Agte et al. [8] acceptance of adopting MDO methods may be relatively low since employees typically are comfortable working in their environment and somewhat fearsome to lose control if their domain boundaries open up.

No direct ethical implications or potential consequences that need to be taken due

consideration can be ascertained at the outset of the thesis work. The work does not contain any ethically salient aspects; however, the outcome may have ethical implications. Indirect ethical consequences may result if safety in the studied component can be improved (for it to be made worse is unlikely due to verification via simulation and otherwise). Although commonly accepted, it can be added that avoidance of harm is an ethical imperative in both deontological and consequentialist approaches to moral philosophy [79].

6.3 Method Limitations

In addition to the limitations and delimitations listed at the outset in Section 1.3, further limitations of the work and the proposed MDO process, respectively, are discussed here. The limitations of the work are listed below:

- Due to the complex nature of MDO and its associated fields of study, e.g., optimization algorithms, metamodeling, DoE, not all aspects of its theory could be included in Chapter 2. Prioritization due to time limitations may have caused useful information from the literature not to have been included in the work.
- The sample of the needs assessment cannot be assured to be representative. However, it can indicate preferences and serves to provide insight into the optimization experience of a subset of engineers at VCC.
- The novelty of the proposed MDO process, as shown conceptually in Figure 4.5, should not be overstated as it builds on previous work. All contributions are too numerous to list but especially noteworthy in their influence on this work are Duddeck [52], Ryberg, Bäckryd, and Nilsson [9], [61], and Agte et al. [8]. Novelty rather lies in the application, see Chapter 5, the practical implementation that is flexible for other optimization problems, the discussion in Chapter 6 and the literature review, see Chapter 2.
- Parameterization, as used in Chapter 5, is dependent on previous studies and engineering judgment and is therefore subject to a degree of uncertainty. Alternatively, iterative sensitivity analyses can be performed as discussed in Section 4.4.2.
- In the verification, see Chapter 5, an assumption of symmetry is used for some parts of the CSDB geometry. This is done to increase resolution and since no impact points on the right side are used in the optimization. However, it may limit the optimization since the design is slightly asymmetrical.

The functionality of the process (Figure 4.5) and modeFRONTIER implementation (Section 4.2) have been verified for one automotive component. However, the following MDO process limitations should be considered before proceeding with other design optimization problems:

- The MDO process as described in Chapter 4 need not necessarily be implemented in modeFRONTIER as has been the case in this work. However, if

this modeFRONTIER implementation is considered, there are a number of limitations specific to it:
  - The availability of optimization and metamodeling algorithms is limited to what modeFRONTIER offers.
  - Choice of error measure used to evaluate the metamodel is limited. Squared error measures such as the RMSE could be preferred to MRE since it is more useful when large errors are undesirable [80].

- The proposed MDO process is likely most appropriate for design variables fewer than ten. Direct optimization is recommended for greater design variable counts [37] but this nullifies the benefits of metamodels as discussed in Section 4.4.3.
  - When the number of design variables is $>10^2$, only gradient-based optimization (for direct optimization) is practical [81] and OpenMDAO may be considered due to its efficiency in handling coupled analytic derivatives as opposed to the less efficient finite-difference approximations used by commercial software according to [20].
- When the number of load cases increases, it is likely that screening becomes more difficult since some aspects of the geometry may only be significant for a particular load case. To screen out the same number of design variables, the threshold value for variance contribution needs to be heightened, which in turn captures less contribution to responses, lowering accuracy. This is a fundamental consideration in variable screening with multiple load cases and not specific to the proposed approach.

6.4 Future Work

The thesis outcome is an MDO process, verified by a practical implementation. However, as the recommended approach utilizes parameter thicknesses as design variables, the design space is somewhat limited. If the software implementation and scripts provided were further developed to allow for more geometrical features to be modified, such as angles and curves (e.g., Beizer splines), larger portions of the design space could be explored. This could be made possible by integrating ANSA's morphing tools, which allows for re-shaping of either FE models into the modeFRONTIER loop depicted in Figure 4.2.

The current approach requires the same design table to be used in all disciplines. As variable screening results generally differ between the aspects, a more efficient approach could be used if the modeFRONTIER implementation was further developed to allow for a nested approach, or a more advanced MDO architecture, as discussed in Section 2.6.2. This would allow for a unique subset of design variables to be selected for each respective discipline, giving the option to either reduce the computational expense required to create the space-filling DoE or to opt for a higher resolution setup.

The current process and modeFRONTIER implementation focus on performing parameter thicknesses study using a constant material parameter. Investigations could be made to modify or further expand the already existing process to evaluate aspects such as e.g., material selection and cost model performance when optimizing the design.

With regards to the implementation as outlined in Chapter 5, other components may need to undergo modification to reach performance targets. Further investigations could also be conducted to verify the usability of the process for other disciplinary functions than crash and NVH, e.g., durability, and also to evaluate the combination of these.

Concerning the statistical DoE for variable screening, see Section 4.4.2, a Plackett-Burman design with foldover can be used to get a resolution IV instead of resolution III experimental design. This has been demonstrated by Tang and Lindqvist [68].

7
Conclusion

This work has investigated the needs and views of a sample of engineers at VCC with regards to MDO and its implementation. It was found that perceived desirable features of MDO are its utility as a learning tool and that its complexity in combination with fundamental challenges related to both organizational and technical aspects are perceived hindrances. Furthermore, it was confirmed that ease-of-use is a highly valued feature in an MDO workflow.

Based on the needs assessment and a review of the literature on MDO and related fields, a process was proposed and implemented in MDO platform modeFRONTIER. The process involves variable selection by parameterization and statistical screening, metamodel-based optimization, and result interpretation steps. Implementation in modeFRONTIER features parallel analysis of multiple disciplines and load cases, dynamic waiting, and response extraction through a set of scripts. The development of the MDO process was illustrated using a simplified geometry and load case and subsequently verified with a sizing optimization of a Center Stack Display bracket. While the verification showed that the process could be used to enhance knowledge about the system under optimization and that performance can be increased, it also indicates important limitations: (1) interactions outside the system boundaries of the optimization can limit system performance, emphasizing the importance of global information, (2) the starting geometry limits design freedom and thus the optimization result, this is especially true for sizing optimization and less so for optimization where the FE mesh is morphed, and (3) parameterization needs to be carefully considered because results are evaluated in the framework imposed by it.

Concerning organizational and sustainability aspects, there is reason to believe that MDO implementation in the automotive industry faces challenges related to the lack of both appropriate models and shared parameters between disciplines. Strategies for large-scale metamodel-based MDO in the automotive sector have been proposed but remain to be implemented at scale. Furthermore, since mass reduction is a common goal in automotive development and MDO has been shown to be effective towards this goal, its use can have a positive environmental and health impact in the areas of ecological and societal sustainability. MDO is deemed unlikely to displace R&D engineers in any significant capacity due to it acting as a tool rather than a replacement for competent engineers. MDO as a technology is arguably ethically neutral, but if it is used to avoid or reduce harm, e.g., improving the safety of automotive vehicles, it can be argued to have a positive ethical impact in many philosophical traditions.

Finally, future work has been recommended. Additional geometrical features, e.g., through mesh morphing, can be used as design variables in the optimization to offer more considerable design freedom. Alternative or additional objectives and constraints can be included in the MDO to make its results more valuable and representative of real trade-offs, e.g., by implementing cost models. Theoretical improvements to the statistical Design of Experiments for variable screening are also suggested.

References

[1] Volvo Car Corporation, "This is Volvo," https://www.media.volvocars.com/global/en-gb/corporate/this-is-volvo, accessed May 10, 2021.

[2] Volvo Car Group, "Annual report 2020," https://investors.volvocars.com/annualreport2020/assets/pdf/VCG_ENG_2020_web_20210317.pdf, accessed May 11, 2021.

[3] S. Eppinger and K. Ulrich, *Product Design and Development*, 6th ed. New York, NY, USA: McGraw-Hill Higher Education, 2015.

[4] J. Giesing and J.-F. Barthelemy, "A summary of industry MDO applications and needs," in *7th AIAA/USAF/NASA/ISSMO Symp. Multidiscip. Anal. Optim.*, St. Louis, MO, USA, 1998, p. 4737.

[5] N. M. Alexandrov, "Multidisciplinary design optimization," *Optim. Eng.*, vol. 6, no. 1, pp. 5–7, 2005.

[6] A.-B. Ryberg, R. Domeij Bäckryd, and L. Nilsson, "Metamodel-based multidisciplinary design optimization for automotive applications," Linköping University, Div. of Solid Mech., Tech. Rep. LIU-IEI-R-12/003, 2012.

[7] L. A. Schmit Jr, "Structural synthesis 1959-1969 - a decade of progress," *Recent Adv. in Matrix Methods of Struct. Anal. and Des.*, pp. 565–634, 1971.

[8] J. Agte, O. De Weck, J. Sobieszczanski-Sobieski, P. Arendsen, A. Morris, and M. Spieck, "MDO: assessment and direction for advancement - an opinion of one international group," *Struct. Multidiscip. Optim.*, vol. 40, no. 1-6, pp. 17–33, 2010.

[9] R. Domeij Bäckryd, A.-B. Ryberg, and L. Nilsson, "Multidisciplinary design optimisation methods for automotive structures," *Int. J. Automot. Mech. Eng.*, vol. 14, no. 1, pp. 4050–4067, 2017.

[10] Z.-F. Fu and J. He, *Modal Analysis*. Oxford, U.K.: Elsevier, 2001.

[11] M. F. M. Alkbir, S. M. Sapuan, A. A. Nuraini, and M. R. Ishak, "Fibre properties and crashworthiness parameters of natural fibre-reinforced composite structure: A literature review," *Compos. Struct.*, vol. 148, pp. 59–73, 2016.

[12] J. Fang, G. Sun, N. Qiu, N. H. Kim, and Q. Li, "On design optimization for structural crashworthiness and its state of the art," *Struct. Multidiscip. Optim.*, vol. 55, no. 3, pp. 1091–1119, 2017.

[13] M. Kiani, I. Gandikota, A. Parrish, K. Motoyama, and M. Rais-Rohani, "Surrogate-based optimisation of automotive structures under multiple crash and vibration design criteria," *Int. J. Crashworthiness*, vol. 18, no. 5, pp. 473–482, 2013.

[14] C. H. Liu, Y. C. Lai, C. H. Chiu, and M. H. Lin, "Interior head impact analysis of automotive instrument panel for unrestrained front seat passengers," *Key Eng. Mater.*, vol. 715, pp. 174–179, 2016.

[15] P. Y. Papalambros and D. J. Wilde, *Principles of Optimal Design: Modeling and Computation*, 3rd ed. Cambridge University Press, 2017.

[16] P. W. Christensen and A. Klarbring, *An Introduction to Structural Optimization.* Springer, 2008.

[17] D. G. Luenberger and Y. Ye, *Linear and Nonlinear Programming*, 2nd ed. Reading, MA, USA: Addison-Wesley, 1984.

[18] P. Pedregal, *Introduction to Optimization.* New York, NY, USA: Springer, 2004.

[19] J. R. R. A. Martins and A. Ning, "Engineering design optimization," 2021, advance book copy. [Online]. Available: https://lnkd.in/e5tn3Xm

[20] J. S. Gray, J. T. Hwang, J. R. R. A. Martins, K. T. Moore, and B. A. Naylor, "OpenMDAO: An open-source framework for multidisciplinary design, analysis, and optimization," *Struct. Multidiscip. Optim.*, vol. 59, no. 4, pp. 1075–1104, 2019.

[21] A. Younis and Z. Dong, "Trends, features, and tests of common and recently introduced global optimization methods," *Eng. Optim.*, vol. 42, no. 8, pp. 691–718, 2010.

[22] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to Derivative-free Optimization.* SIAM Publications, 2009.

[23] L. M. Rios and N. V. Sahinidis, "Derivative-free optimization: a review of algorithms and comparison of software implementations," *J. Global Optim.*, vol. 56, no. 3, pp. 1247–1293, 2013.

[24] R. T. Marler and J. S. Arora, "The weighted sum method for multi-objective optimization: new insights," *Struct. Multidiscip. Optim.*, vol. 41, no. 6, pp. 853–862, 2010.

[25] S. N. Sivanandam and S. N. Deepa, *Introduction to Genetic Algorithms.* Berlin, Germany: Springer, 2008.

[26] S. Shetty, "Optimization of vechicle structures under uncertainties," Ph.D. dissertation, Dept. of Manage. and Eng., Linköping University, Linköping, Sweden, 2017.

[27] E. Rigoni and S. Poles, "NBI and MOGA-II, two complementary algorithms for multi-objective optimizations," in *Dagstuhl Semi. Proc.* Dagstuhl, Germany: Schloss Dagstuhl – Leibniz Center for Informatics, 2005, pp. 1–22.

[28] C. Poloni and V. Pediroda, "GA coupled with computationally expensive simulations: tools to improve efficiency," *Genet. Algorithms and Evol. Strategies Eng. and Comput. Sci.: Recent Adv. and Ind. Appl.*, pp. 267–288, 1997.

[29] S. Poles, "Bench-marking MOGA-II," Esteco, Tech. Rep. 2004-001, 2003.

[30] S. Poles, E. Rigoni, and T. Robic, "MOGA-II performance on noisy optimization problems," in *Int. Conf. Bioinspired Optim. Methods and Appl.*, Ljubljana, Slovenia, 2004, p. 51–62.

[31] H.-Y. Kim, "Analysis of variance (ANOVA) comparing means of more than two groups," *Restor. Dent. Endod.*, vol. 39, no. 1, p. 74, 2014.

[32] C. Gu, *Smoothing spline ANOVA models*, 2nd ed. New York, NY, USA: Springer, 2013, vol. 297.

[33] J. Antony, *Design of Experiments for Engineers and Scientists*, 2nd ed. Elsevier, 2014.

[34] K. Vanaja and R. H. Shobha Rani, "Design of experiments: concept and applications of Plackett Burman design," *Clin. Res. Regul. Aff.*, vol. 24, no. 1, pp. 1–23, 2007.

[35] R. L. Plackett and J. P. Burman, "The design of optimum multifactorial experiments," *Biometrika*, vol. 33, no. 4, pp. 305–325, 1946.

[36] M. D. McKay, R. J. Beckman, and W. J. Conover, "Comparison the three methods for selecting values of input variable in the analysis of output from a computer code," *Technometrics*, vol. 21, no. 2, 1979.

[37] Esteco SpA, *modeFRONTIER User Guide*, 2020, 2020R2 edition.

[38] A. I. J. Forrester and A. J. Keane, "Recent advances in surrogate-based optimization," *Prog. Aerosp. Sci.*, vol. 45, no. 1-3, pp. 50–79, 2009.

[39] M. D. McKay, R. J. Beckman, and W. J. Conover, "A comparison of three methods for selecting values of input variables in the analysis of output from a computer code," *Technometrics*, vol. 42, no. 1, pp. 55–61, 2000.

[40] E. L. Loweth, G. N. De Boer, and V. V. Toropov, "Practical recommendations on the use of moving least squares metamodel building," in *Proc. 13th Int. Conf. Civil, Struct. and Environ. Eng.*, Chania, Crete, Greece, 2011.

[41] G. G. Wang and S. Shan, "Review of metamodeling techniques in support of engineering design optimization," *J. Mech. Eng.*, vol. 129, no. 4, pp. 370–380, 2007.

[42] N. V. Queipo, R. T. Haftka, W. Shyy, T. Goel, R. Vaidyanathan, and P. K. Tucker, "Surrogate-based analysis and optimization," *Prog. Aerosp. Sci.*, vol. 41, no. 1, pp. 1–28, 2005.

[43] G. E. P. Box and K. B. Wilson, "On the experimental attainment of optimum conditions," *J. R. Stat. Soc. Series B Methodol.*, vol. 13, no. 1, pp. 1–38, 1951.

[44] A. A. Mullur and A. Messac, "Extended radial basis functions: more flexible and effective metamodeling," *AIAA J.*, vol. 43, no. 6, pp. 1306–1315, 2005.

[45] M. Buhmann and J. Jäger, "On radial basis functions," Snapshots of Modern Mathematics from Oberwolfach, 2019, SNAP-2019-002-EN.

[46] R. L. Hardy, "Multiquadric equations of topography and other irregular surfaces," *J. Geophys. Res.*, vol. 76, no. 8, pp. 1905–1915, 1971.

[47] H. Fang and M. F. Horstemeyer, "Global response approximation with radial basis functions," *Eng. Optim.*, vol. 38, no. 04, pp. 407–424, 2006.

[48] D. G. Krige, "A statistical approach to some basic mine valuation problems on the Witwatersrand," *J. South Afr. Inst. Min. Metall.*, vol. 52, no. 6, pp. 119–139, 1951.

[49] J.-P. Chilès and N. Desassis, "Fifty years of kriging," in *Handbook of Mathematical Geosciences*. Springer, 2018, pp. 589–612.

[50] J. P. C. Kleijnen, "Kriging metamodeling in simulation: A review," *Eur. J. Oper. Res.*, vol. 192, no. 3, pp. 707–716, 2009.

[51] R. M. Mendes and R. Lorandi, "Indicator kriging geostatistical methodology applied to geotechnics project planning," in *10th Congr. Int. Assoc. for Eng. Geol. and Environ. (IAEG)*, London, U.K., 2006, pp. 1–12.

[52] F. Duddeck, "Multidisciplinary optimization of car bodies," *Struct. Multidiscip. Optim.*, vol. 35, no. 4, pp. 375–389, 2008.

[53] J. R. R. A. Martins and A. B. Lambe, "Multidisciplinary design optimization: a survey of architectures," *AIAA J.*, vol. 51, no. 9, pp. 2049–2075, 2013.

[54] E. J. Cramer, J. E. Dennis, Jr, P. D. Frank, R. M. Lewis, and G. R. Shubin, "Problem formulation for multidisciplinary optimization," *SIAM J. Optim.*, vol. 4, no. 4, pp. 754–776, 1994.

[55] N. P. Tedford and J. R. R. A. Martins, "Benchmarking multidisciplinary design optimization algorithms," *Optim. Eng.*, vol. 11, no. 1, pp. 159–183, 2010.

[56] J. T. Hwang, "A modular approach to large-scale design optimization of aerospace systems," Ph.D. dissertation, Dept. of Aerosp. Eng., University of Michigan, Ann Arbor, MI, USA, 2015.

[57] T. C. Wagner, "A general decomposition methodology for optimal system design," Ph.D. dissertation, Dept. of Mech. Eng. and Appl. Mech., University of Michigan, Ann Arbor, MI, USA, 1993.

[58] H. M. Kim, D. G. Rideout, P. Y. Papalambros, and J. L. Stein, "Analytical target cascading in automotive vehicle design," *J. Mech. Des.*, vol. 125, no. 3, pp. 481–489, 2003.

[59] A. B. Lambe and J. R. R. A. Martins, "Extensions to the design structure matrix for the description of multidisciplinary design, analysis, and optimization processes," *Struct. Multidiscip. Optim.*, vol. 46, no. 2, pp. 273–284, 2012.

[60] J. T. Allison, M. Kokkolaras, and P. Y. Papalambros, "On selecting single-level formulations for complex system design optimization," *J. Mech. Des.*, vol. 129, no. 9, pp. 898–906, 2007.

[61] R. Domeij Bäckryd, "Multidisciplinary design optimization of automotive structures," Ph.D. dissertation, Dept. of Manage. and Eng., Linköping University, Linköping, Sweden, 2013.

[62] R. Pandi Perumal, H. Voos, F. Dalla Vedova, and H. Moser, "Comparison of multidisciplinary design optimization architectures for the design of distributed space systems," in *Proc. 71st Int. Astronautical Congr. 2020*, Online, 2020.

[63] R. Patel and B. Davidson, *Forskningsmetodikens Grunder: Att Planera, Genomföra och Rapportera en Undersökning*, 3rd ed. Lund, Sweden: Studentlitteratur, 2003, in Swedish.

[64] K. F. Punch, *Introduction to Social Research: Quantitative and Qualitative Approaches*, 3rd ed. SAGE Publications, 2014.

[65] V. Braun and V. Clarke, *Thematic Analysis*. American Psychological Association, 2012, pp. 57–71.

[66] J. T. Hwang, D. Y. Lee, J. W. Cutler, and J. R. R. A. Martins, "Large-scale multidisciplinary optimization of a small satellite's design and operation," *J. Spacecr. Rockets*, vol. 51, no. 5, pp. 1648–1663, 2014.

[67] C.-A. Thole and L. Mei, "Reasons for scatter in crash simulation results," in *4th Eur. LS-DYNA Users' Conf.*, Ulm, Germany, 2003.

[68] M. Tang and K. Lindkvist, "Reducing the squeak and rattle risk by improving the dynamic response and geometric variation in an assembly using topometry optimisation," Master's Thesis, Dept. of Des. Sci., Lund University, Lund, Sweden, 2021.

[69] H. Cho, S. Bae, K. K. Choi, D. Lamb, and R.-J. Yang, "An efficient variable screening method for effective surrogate models for reliability-based design optimization," *Struct. Multidiscip. Optim.*, vol. 50, no. 5, pp. 717–738, 2014.

[70] L. Ricco, E. Rigoni, and A. Turco, "Smoothing spline ANOVA for variable screening," *Dolomites Res. Notes Approx.*, vol. 6, 2013.

[71] R. J. Yang, N. Wang, C. H. Tho, J. P. Bobineau, and B. P. Wang, "Metamodeling development for vehicle frontal impact simulation," *J. Mech. Des.*, vol. 127, no. 5, pp. 1014–1020, 2005.

[72] L. Shi, R. J. Yang, and P. Zhu, "A method for selecting surrogate models in crashworthiness optimization," *Struct. Multidiscip. Optim.*, vol. 46, no. 2, pp. 159–170, 2012.

[73] A. Miller, *Subset Selection in Regression*, 2nd ed. New York, NY, USA: CRC Press, 2002.

[74] C. Stephens and W. Baritompa, "Global optimization requires global information," *J. Optim. Theory Appl.*, vol. 96, no. 3, pp. 575–588, 1998.

[75] J. Rakowska, A. Chator, B. Barthelemy, M. Lee, S. Morgans, J. Laya, G. Zinn, C.-H. Chuang, and S. R. Gondipalle, "An iterative application of multidisciplinary optimization for vehicle body weight reduction based on 2015 Mustang product development," *SAE Int. J. Mater. Manuf.*, vol. 8, no. 3, pp. 685–692, 2015.

[76] K. Craig, N. Stander, D. Dooge, and S. Varadappa, "MDO of automotive vehicle for crashworthiness and NVH using response surface methods," in *9th AIAA/ISSMO Symp. Multidiscip. Anal. and Optim.*, Atlanta, GA, USA, 2002, paper 2002-5607.

[77] S. Blum and J. Will, "Combining robustness evaluation with current automotive MDO application," in *Weimar Optimization and Stochastic Days 2006*, vol. 3, Weimar, Germany, 2006.

[78] A. Bandivadekar, K. Bodek, L. Cheah, C. Evans, T. Groode, J. Heywood, E. Kasseris, M. Kromer, and M. Weiss, "On the road in 2035: reducing transportation's petroleum consumption and GHG emissions," Massachusetts Institute of Technology, Tech. Rep., 2008.

[79] L. Alexander and M. Moore, "Deontological Ethics," in *The Stanford Encyclopedia of Philosophy*, E. N. Zalta, Ed. Metaphysics Research Lab, Stanford University, 2020.

[80] J. Wesner, "MAE and RMSE - which metric is better?" https://medium.com/@human-in-a-machine-world/e60ac3bde13d, accessed May 4, 2021.

[81] J. T. Hwang and J. R. R. A. Martins, "A computational architecture for coupling heterogeneous numerical models and computing coupled derivatives," *ACM Trans. Math. Softw.*, vol. 44, no. 4, pp. 1–39, 2018.

A

Appendix - Software Implementation

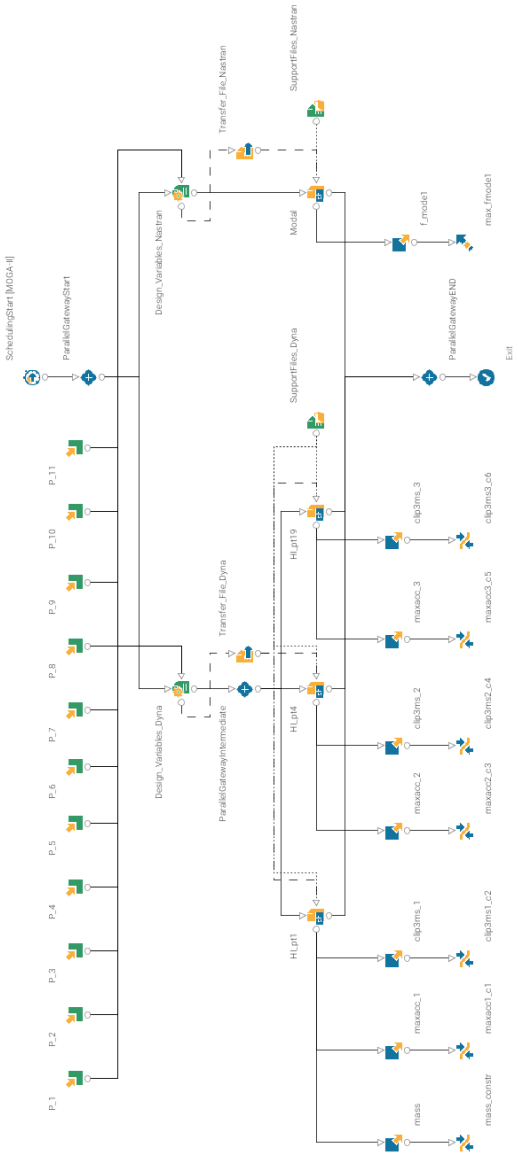


Figure A.1: Optimization loop implementation in modeFRONTIER

B

Appendix - Script Code

Listing B.1: Bash shell script example from LS-DYNA used in the modeFRONTIER implementation. Sensitive information has been removed.

```
#!/bin/bash


###################### Settings ######################

#User
user=write_name_of_user_here

#Define load case
LC=1

#Job name
file=job_name_$LC

#Metascript name
metascript_name=name_of_script_$LC.py

#Sleep intervals
setSleep=50
sleepFactor=5

#Define modeFrontier project directory
main_path=/write/your/project/path/here

#Result path for job
res_path=/write/your/result/path/here

#Mass to be subtracted:
#Adjust manually at bottom of the script (mass_number)

######################################################
```

```
################## Script starting ##################

echo Script Started!
date +"%T.%2N" #time stamp

sleep $setSleep #50s

lsdyna.run ... $file.key
echo " "
echo Job queued!
date +"%T.%2N" #time stamp
echo " "

sleep $setSleep #50s

#Waiting until file modifications happens in res_path
watch -d -n 0.5 -t -g ls -lR $res_path | sha1sum > /dev/
   null && echo "Watch triggered!."
date +"%T.%2N" #time stamp
echo " "

cd $res_path

echo Sleeping $((setSleep * sleepFactor)) sec to ensure
   old log file is removed...
sleep $((setSleep * sleepFactor)) #250s
echo Continuing... Looking for new log file...
date +"%T.%2N" #time stamp
echo " "

while [ ! -f $file.log ];
do
    echo Waiting for log file to be created... date +"%T
       .%2N" #time stamp
    sleep $((setSleep / sleepFactor))
done

echo Log file detected!
date +"%T.%2N" #time stamp
echo " "

while true
do
   echo Grep: $(grep -c 'LS-Dyna done at:' $file.log)

   if [ $(grep -c 'LS-Dyna done at:' $file.log) == 1 ];
```

```
    then
        echo Simulation Finished!
        break;
    fi;
    echo Waiting for simulation to finish... date +"%T.%2N
       " #time stamp
    sleep $((setSleep / sleepFactor)) #10s
done

echo " "
echo Simulation done, running metascript...
date +"%T.%2N" #time stamp
echo " "

cd -

cd $main_path
echo Path for script and metascript
pwd

#Runs MetaPost with scripts that outputs clip3ms and
   maxacc G.
meta -nogui -s $metascript_name $res_path

echo MetaPost script started.
date +"%T.%2N" #time stamp
echo " "

echo Waiting for metascript to finish...
echo " "

sleep $setSleep #50s

echo Result files are put in directory:
pwd
echo " "

source clip3ms_$LC.txt
echo Clip3ms: $clip3ms [G]

source maxacc_$LC.txt
echo Maxacc: $maxacc [G]

#Changing directory to mF default directory
cd -
```

```
#Extract mass
cat $res_path/d3hsp |grep -A 3 't o t a l  m a s s' >
   mass.txt
echo Mass: $(awk 'BEGIN{test=0}{if(NR==1) test=test+$NF}
   END{print test-mass_number}' mass.txt) [kg]
echo " "

#Write outputs to text document
echo clip3ms_$LC=$clip3ms > output_dyna_$LC.txt
echo maxacc_$LC=$maxacc >> output_dyna_$LC.txt
echo mass=$(awk 'BEGIN{test=0}{if(NR==1) test=test+$NF}
   END{print test-mass_number}' mass.txt) >>
   output_dyna_$LC.txt

echo End of script!
echo " "
################## End of script ###################
```

DEPARTMENT OF INDUSTRIAL AND MATERIAL SCIENCE
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY