A Multidisciplinary Design Optimisation Framework for Structural Problems with Disparate Variable Dependence

by

Jonathan Ollar

Submitted to the University of London in partial fulfillment of the requirements of the degree of Doctor of Philosophy

School of Engineering and Materials Science
Queen Mary, University of London
United Kingdom

August 2016

I, Jonathan Ollar, confirm that the research included within this thesis is my own work or that where it has been carried out in collaboration with, or supported by others, that this is duly acknowledged below and my contribution indicated. Previously published material is also acknowledged below.

I attest that I have exercised reasonable care to ensure that the work is original, and does not to the best of my knowledge break any UK law, infringe any third party's copyright or other intellectual property right, or contain any confidential material.

I accept that the college has the right to use plagiarism detection software to check the electronic version of the thesis.

I confirm that this thesis has not been previously submitted for the award of a degree by this or any other university.

Jonathan Ollar

2016-08-17

Details of collaboration and publications:

Chapter 5 contains work published in the following papers. The contribution of the candidate, Jonathan Ollar, is the inclusion of a constraint on the condition number of the correlation matrix during hyper parameter optimisation. Derivation and implementation of the partial derivatives using the adjoint method was carried out by the candidate. The rest of the paper was completed in close collaboration with Mr. Charles Mortished, and under the supervision of Dr. Jones, Prof. Sienz, and Prof. Toropov.

- **Ollar J**, Mortished C, Jones R, Sienz J, Toropov V (2016) Gradient based hyper-parameter optimisation for well conditioned kriging metamodels. Structural and Multidisciplinary Optimization.

- Mortished C, **Ollar J**, Jones R, Benzie P, Toropov V, Sienz J (2016) Aircraft wing optimisation based on computationally efficient gradient-enhanced kriging. 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, San Diego, CA, USA, January 4-8, 2016.

Chapter 6 contains work published in in the following papers. The contribution of the papers are those of the candidate, Jonathan Ollar, under the supervision of Dr. Jones and Prof. Toropov.

- **Ollar J**, Jones R, and Toropov V (2016) Incorporation of bird strike requirements in MDO of an aircraft wing using subspace approximations. 11th ASMO-UK/ISSMO International Conference on Engineering Design Optimisation, Munich, Germany, July 18-20, 2016.

- **Ollar J**, Toropov, V, and Jones R (2016) Sub-space approximations for MDO problems with disparate disciplinary variable dependence. Structural and Multidisciplinary Optimization.

- **Ollar J**, Toropov V, Jones R (2015) Adaptive sub-space approximations in trust-regions for large scale MDO problems. 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Kissimmee, FL, USA, January 5-9, 2015.

- **Ollar J**, Toropov V, Jones R (2014) Mid-range approximations in sub-spaces for MDO problems with disparate discipline attributes. 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, GA, USA, June 16-20, 2014.

Abstract

Multidisciplinary design optimisation incorporates several disciplines in one integrated optimisation problem. The benefit of considering all requirements at once rather than in individual optimisations is that synergies between disciplines can be exploited to find superior designs to what would otherwise be possible. The main obstacle for the use of multidisciplinary design optimisation in an industrial setting is the related computational cost which may become prohibitively large.

This work is focused on the development of a multidisciplinary design optimisation framework that extends the existing trust-region based optimisation method known as the mid-range approximation method.

The main novel contribution is an approach to solving multidisciplinary design optimisation problems using metamodels built in sub-spaces of the design variable space. Each metamodel is built in the sub-space relevant to the corresponding discipline while the optimisation problem is solved in the full design variable space. Since the metamodels are built in a space of reduced dimensionality, the computational budget for building them can be reduced without compromising their quality.

Furthermore, a method for efficiently building kriging metamodels is proposed. This is done by means of a two-step hyper parameter tuning strategy. The first step is a line search where the set of tuning parameters is treated as a single variable. The solution of the first step is used in the second step, a gradient based hyper parameter optimisation where partial derivatives are obtained using the adjoint method.

The framework is demonstrated on two examples, a multidisciplinary design optimisation

of a thin-walled beam section subject to static and impact requirements, and a multidisciplinary design optimisation of an aircraft wing subject to static and bird strike requirements. In both cases the developed technique demonstrates a reduced computational effort compared to what would typically be achieved by existing methods.

Acknowledgments

Firstly, I would like to express my gratitude to my supervisors. Dr. Royston Jones and Prof. Vassili Toropov.

Dr. Jones presented me with the opportunity of pursuing this work and has been supporting it since. He has constantly been interested in the application of the research to industrial problems. Without his drive for innovation this work would not have been the same.

I cannot thank Prof. Toropov enough for his supervision, inspiration and guidance. As an academic supervisor Prof. Toropov was excellent at giving advice and support, and simultaneously allowing enough freedom in the choice of direction of the research. His constant interest, ideas and thorough reviews has significantly contributed to the quality of the work presented in this thesis.

I am also grateful to my examiners, Prof. Alastair Wood and Prof. Fred van Keulen, for their comments and valuable advice which has improved the quality of this thesis.

I wish to thank Charles Mortished, Yury Korolev, and Ralf Schlaps with whom I had the fortune to collaborate, for all our discussions and shared ideas.

I would like to thank everyone involved in the AMEDEO project, in particular Tim Surry for managing all administrative tasks of the research project, coordinator Prof. Harvey Thompson, project manager Juliet Jopson, all principal investigators and early stage researchers.

There are many people at Altair Engineering whom I wish to thank, these include

but are not limited to Tim Willment, Alex Quirk, Lou Zhifan, Stephan Koerner, Joseph Pajot, Ming Zhou, Michal Stefuca, Pierre Rousseau, Richard Boyd, Marco Fließer, Peter Benzie, Jerome Rousseau, and Jamie Buchanan.

I would like to thank my mother, grandmother and uncle for their continuous love and support and my friends in Sweden whom I have not visited nearly enough in the past few years.

To Sandra, thank you for your love, understanding, encouragement, support and most of all, for taking my mind off the research project now and again.

Finally, I wish to acknowledge the financial support from the European Union Seventh Framework Programme FP7-PEOPLE-2012-ITN under grant agreement 316394, Aerospace Multidisciplinarity Enabling DEsign Optimization (AMEDEO) Marie Curie Initial Training Network.

Jonathan Ollar,
London, August 2016.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

DOE	Design of experiments
GA	Genetic algorithm
GS	Golden Search
GEK	Gradient enhanced kriging
GE-MLSM	Gradient enhanced moving least squares method
IDF	Individual discipline feasible
LOOCV	Leave-one-out cross-validation
MAM	Mid-range approximation method
MELS	Modified Extensible lattice sequences
MLSM	Moving least squares method
MDA	Multidisciplinary analysis
MDO	Multidisciplinary design optimisation
MDF	Multidisciplinary feasible
PSO	Particle swarm optimisation
RMSE	Root mean squared error
SQP	Sequential quadratic programming
SAND	Simultaneous analysis and design
SPH	Smooth particle hydrodynamics
ULH	Uniform latin hypercube

List of Symbols

a	Regression coefficient
$\mathbf{a}$	Vector of regression coefficients
$\mathbf{A}$	Vector of lower bounds
$\mathbf{A}^k$	Vector of lower bounds for current trust-region
b	Regressor
$\mathbf{b}$	Vector of regressors
$\mathbf{B}$	Vector of upper bounds
$\mathbf{B}^k$	Vector of upper bounds for current trust-region
b_r	Ratio of recycle region size to trust region size
b_s	Ratio of sample region size to trust region size
$\mathbf{c}$	Vector of consistency constraints
$\mathbf{f}$	vector of function values
f_0	Objective function
f_j	j-th ineqtionuality constraint
G	Least squares problem
h_k	k-th equality constraint
m	Number of constraints
N	Number of disciplines
n	Number of design variables

P	Projection operator
p	Number of points
$\mathbf{r}$	Vector of residuals
$\mathbf{R}$	Correlation matrix
$\bar{\mathbf{R}}$	Adjoint of matrix $\mathbf{R}$
$\mathbf{u}$	Vector of state variables
$\mathbf{w}$	Vector of coupling variables
w	Weighting factor
$\mathbf{W}$	Matrix of weight factors
$\mathbf{x}$	Vector of design variables
$\mathbf{y}$	Vector of response functions
ϵ	Discrepancy
θ	Tuning parameter
$\hat{\mu}$	Estimated mean
$\hat{\sigma}$	Estimated variance

Superscripts

$+$	Variable copy
$*$	Optimal value
$\sim$	Approximation
T	Transpose
-1	Inverse
$-T$	Transpose of inverse
(i)	Discipline i
(k)	Iteration k

Chapter 1

Introduction

This chapter gives an overview to the research that is presented in this work. The chapter commences by a motivation to the research in Section 1.1, followed by a definition of research objectives in Section 1.2. The chapter concludes with an outline of the thesis in Section 1.3.

1.1 Motivation of research

Multidisciplinary design optimisation (MDO) considers requirements from several disciplines in one integrated optimisation problem. By including all disciplines in one optimisation problem, synergies between disciplines can be exploited to find designs that are superior to what could possibly be obtained if attempting separate optimisations for each discipline.

One of the main challenges of MDO is the associated computational cost. In the last couple of years there has been an explosion in computing power. It is now possible to have thousands of cores in a high-performance computing facilities dedicated to numerical

simulations. However, due to the ever increasing fidelity of the computational models used in industry it is still very time consuming to carry out certain types of simulations. As MDO requires multiple simulations across multiple disciplines, the computational budget can easily become unaffordable.

Much research focus is devoted to efficiently obtaining accurate gradients of the response functions with respect to the design variables. The adjoint method is particularly efficient to use for obtaining gradients when the number of design variables is large compared to the number of response functions. This is because the computational cost is proportional to the number of response functions, as opposed to the number of design variables as with direct differentiation. When used together with gradient based optimisation algorithms, such as sequential quadratic programming, the optimisation process can be made very efficient. For further information on obtaining gradients for multidisciplinary systems the reader is referred to Martins and Hwang (2013).

For certain types of numerical simulations, e.g. structural crashworthiness or impact simulations, it may not be possible to obtain gradients at an acceptable computational cost, and even if the computational cost of obtaining the gradients was acceptable, they may not be of practical use if the response functions are noisy. Examples of such simulations can be found in both the automotive industry, e.g. crashworthiness analysis, and the aerospace industry, e.g. bird strike simulation.

There are a number of options for optimisation without the use of gradients such as methods inspired by biological processes or behaviour. Examples of such methods are genetic algorithms (GA) that mimic natural selection and particle swarm optimisation (PSO) that mimic social behaviour of animals in flock. Such algorithms are usually popular for finding the best of several local optima of optimisation problems, however, at a high computational cost.

Another possibility is the use of metamodel-based optimisation which is one of the main topics of this work. Metamodels can be used to replace expensive numerical experiments with cheaper mathematical representations and many of them inherently, or by simple modifications, smooths noisy responses. Gradients are not required, however, if available, they can be used to enhance the quality of the approximations by simple modifications to the metamodel. In order to construct metamodels, the function values at a number of points within the design domain are needed. The number of required simulations are dictated by the severity of the non-linearity of the response, the desired accuracy of the metamodel, and the number of design variables. Unless the response is linear with respect to the design variables, the required number of points increases super-linearly with the number of design variables. This is commonly known as the *curse of dimensionality*.

1.2 Research objectives

The objective of this thesis is to propose and develop a metamodel-based MDO framework suitable for industrial MDO problems with a large number of design variables, at least 100. The framework should be able to incorporate crash-worthiness or impact responses, for which gradients may or may not be available. This objective is broken down into the following tasks:

1. Identify existing techniques with promising attributes for metamodel-based MDO.
2. Identify bottlenecks of existing techniques and suggest improvements.
3. Develop a metamodel-based MDO framework.
4. Propose methods to reduce the computational cost for MDO problems with a large number of design variables (>100).

5. Demonstrate the applicability of the developed framework on problems including impact or crashworthiness requirements.

1.3 Thesis outline

The following chapters are outlined as follows:

Chapter 2 presents an introduction to MDO and the terminology that will be used throughout the thesis followed by a discussion on MDO architectures that leads to a choice of a suitable architecture.

Chapter 3 introduces metamodel-based optimisation and presents information on design of experiments, approximation techniques, and optimisation techniques used throughout the work.

Chapter 4 outlines the details of a trust-region and metamodel-based optimisation method, known as the mid-range approximation method, which has been used throughout this work.

Chapter 5 presents a novel method for reducing the computational effort of the parameter tuning related to building kriging metamodels.

Chapter 6 presents an MDO framework based on the mid-range approximation method. Changes are made to individually handle disciplines within the framework and take advantage of any disparities between disciplines. A method for reducing the computational effort of solving MDO problems including disciplines with varying variable

dependence is presented and demonstrated on a benchmark optimisation example of a thin-walled beam structure and on an industry-related optimisation example of an aircraft wing.

Chapter 7 concludes the thesis with a summary, conclusions, and suggestions for future work.

Chapter 2

Multidisciplinary design optimisation

This chapter discusses the choice of MDO architecture. For comprehensive reviews on MDO architectures, see for instance Cramer *et al.* (1993), Sobieszczanski-Sobieski and Haftka (1997), and most recently Martins and Lambe (2013). The chapter commences with a short introduction outlining the terminology that will be used throughout the work in Section 2.1. This is followed by a discussion of MDO architectures in Section 2.2 and finally concludes with a motivation and choice of architecture in Section 2.3.

2.1 Introduction and terminology

Multidisciplinary design optimisation includes several disciplines in one integrated optimisation problem. This section provides definitions and terminology relating to the definition of a discipline, interactions between disciplines, and their use in a multidisciplinary design optimisation problem.

2.1.1 Disciplines

It is common and intuitive to divide a problem into physical disciplines, e.g. structural mechanics, fluid mechanics, electromagnetic, etc. In this work, however, the division is related to the practicalities of analysing the computational models which are included in the multidisciplinary design optimisation problem. A discipline is defined here as a process, depicted in Figure 2.1, often a commercial finite element (FE) or computational fluid dynamics (CFD) software product, which takes several input arguments $\mathbf{x}_i \in \mathbb{R}^{n_i}$, and outputs a set of function values $\mathbf{y}_i \in \mathbb{R}^{m_i}$. n_i is the number of design variables and m_i is the number of responses for discipline i. In certain situations the partial derivatives of the response functions with respect to the design variables, hereafter referred to as gradients, are available as output from the software product. However, it is assumed here that gradient output is not available in the general case.

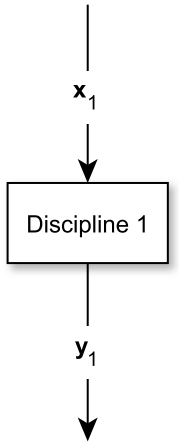
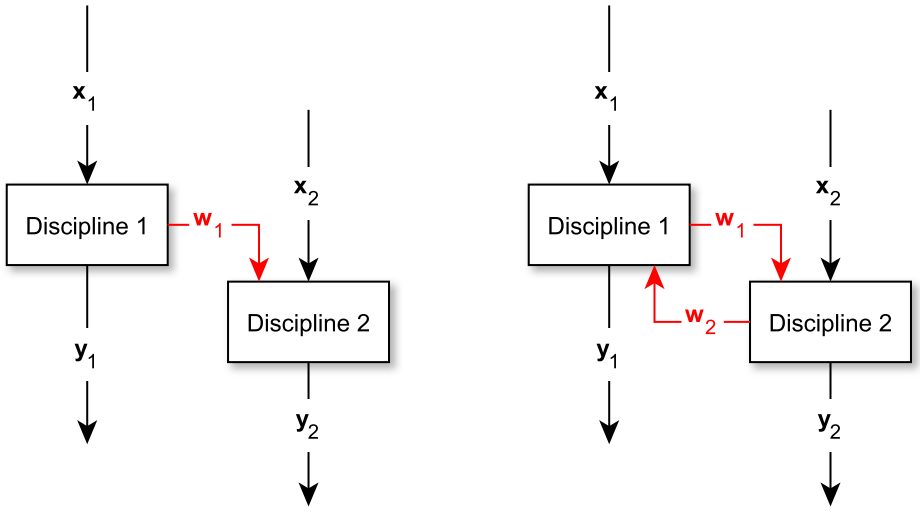


Figure 2.1: A discipline

2.1.2 Multi-physics coupling

In order to capture some physical phenomena, e.g. fluid structure interaction (FSI), it is required to share information between disciplines. There are two conceptually different types of coupling, one or two way coupling, as shown in Figure 2.2. The input variables, which can be overlapping between the disciplines, are denoted x_d where d denotes the discipline number. Similarly the output is denoted y_d. The coupling vectors containing the shared information are denoted as $\mathbf{w}_d$, where d denotes which discipline the shared variable belongs to.

A common way of handling such physics coupling between multiple disciplines is multidisciplinary analysis (MDA). For a one way coupled analysis MDA is carried out by evaluating the first discipline followed by evaluation of the second discipline by using intermediate information from the first discipline. For a two way coupling this becomes more complicated. For a two way coupling MDA is commonly carried out by iteratively



(a) One way coupling (b) Two way coupling

Figure 2.2: Multidisciplinary analysis.

evaluating the individual disciplines until multidisciplinary feasibility has been achieved, usually meaning that the changes between two iterations in the result or intermediate variables between two following iterations are below a certain threshold value (Martins and Lambe, 2013).

An example where a two way coupling is required is aeroelastic analysis of an aircraft wing. The fluid (air) flow leads to a loading and hence deformation of the structure (wing) which in turns leads to changes in the fluid flow, and the circle is complete. As such the fluid pressure from the flow analysis is shared with the structural analysis which in term shares the computed displacement field with the flow analysis. This is repeated until multidisciplinary feasibility has been achieved.

Although multi-physics simulation is not part of the scope of this work, it will be part of the discussion in this chapter in order to accommodate future attempts to extend this work to that area.

2.1.3 MDO terminology

An example of a five discipline MDO problem with two types of multidisciplinary analyses is depicted in Figure 2.3. The design vector for the complete MDO problem is denoted as $\mathbf{x} \in \mathbb{R}^n$ where n is the number of design variables. In order to allow for different parametrisation of disciplines the design vector related to a particular discipline is defined as a projection of the design vector $\mathbf{x}_d = P_d\mathbf{x}$, where P is the projection $P_d : \mathbb{R}^n \mapsto \mathbb{R}^{n_d}$, n is the number of design variables for the MDO problem, and n_d the number of design variables for discipline i. The full set of response functions in the MDO problem is defined as $\mathbf{y} = [\mathbf{y}_1, ..., \mathbf{y}_N]$ where N is the number of disciplines.

The objective function and constraints are written as functions of the complete design variable vector as $f_j(\mathbf{x}) = f_j(\mathbf{y}_1(\mathbf{x}_1)), ..., \mathbf{y}_N(\mathbf{x}_N))$ where $j = 1, ..., m$. $f_0(\mathbf{x})$ denotes the

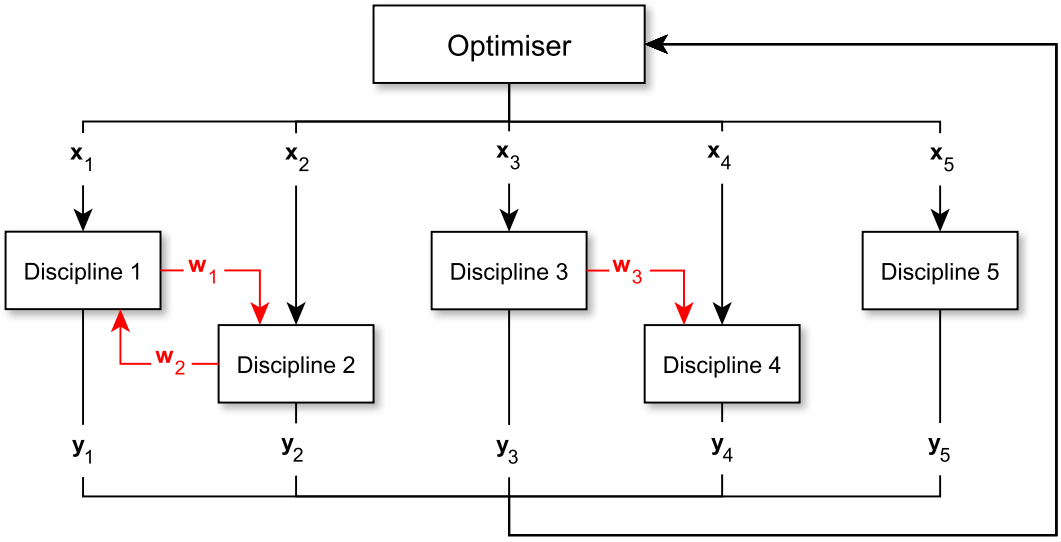


Figure 2.3: MDO problem with five disciplines.

objective function and $f_j(\mathbf{x})$ for $j = 1, ..., m$ denotes inequality constraint functions. In reality they can be functions of one or several disciplines, however the given notation is used for generality and brevity.

2.2 MDO architectures

MDO architectures can be divided into two groups, namely *monolithic* and *distributed* architectures. Monolithic architectures pose a single optimisation problem while distributed architectures decompose the optimisation problem to sub-problems with a top level optimiser to enforce consistency constraints. Distributed architectures have mainly been developed in order to fit organisational structures with teams of experts in charge of their own disciplines. These disciplinary teams are meant to carry out optimisation of their discipline separately whilst occasionally receiving information from the others in order to enforce multidisciplinary feasibility. The penalty associated with

the use of of distributed architectures is usually an increased computational cost (Martins and Lambe, 2013), which is why this work will be focused on monolithic, rather than distributed, architectures. In the following sections the three most common monolithic MDO architectures will be presented and discussed.

2.2.1 Multidisciplinary feasible

The multidisciplinary feasible (MDF) architecture, first formulated by Cramer *et al.* (1993), is the most common and intuitive way of formulating an MDO problem for designers and engineers. It does not differ conceptually from a single discipline optimisation problem, apart from the fact that response functions are evaluated from several disciplines. Figure 2.4 shows the MDF architecture for the previously presented five discipline MDO problem. In MDF the problem is treated as a problem with three disciplines rather than five. Any disciplines which are connected through multi-physics

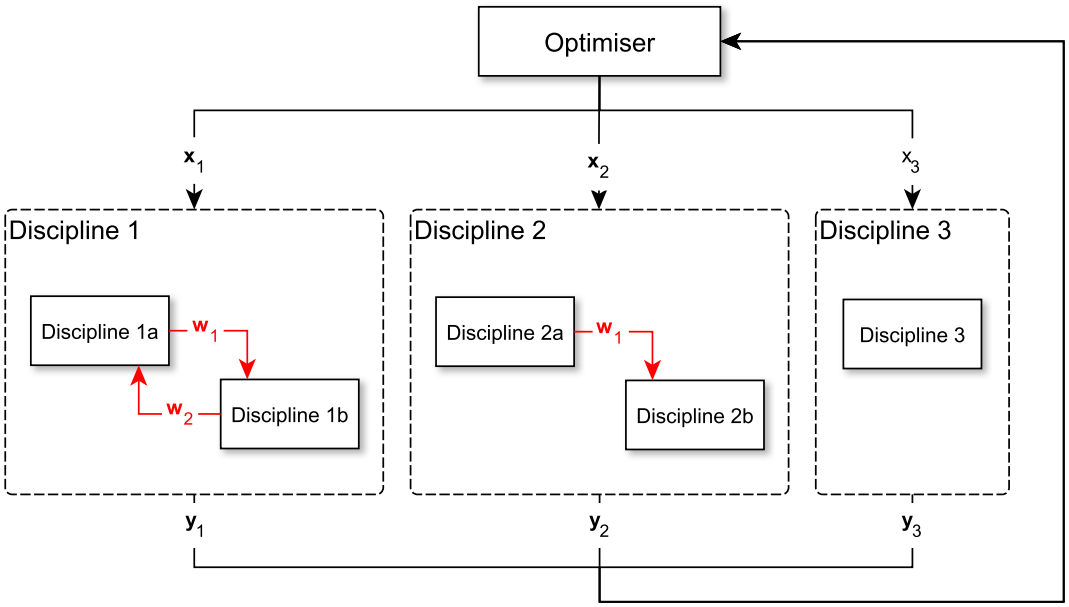


Figure 2.4: Multidisciplinary feasible (MDF) architecture

coupling are grouped together as a super discipline and MDA is carried out internally within the super discipline to guarantee the multidisciplinary feasibility in each iteration. The optimisation problem for MDF can be written, analogous to a mono disciplinary optimisation problem as

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimise}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_j(\mathbf{x}) \leq 1, \qquad j = 1, \ldots, m \\ & A_i \leq x_i \leq B_i, \quad i = 1, \ldots, n \end{aligned} \tag{2.1}$$

$f_0(\mathbf{x}) = f_0(\mathbf{y}_1(\mathbf{x}_1), .., \mathbf{y}_N(\mathbf{x}_N))$ is the objective function, $f_j(\mathbf{x}) = f_j(\mathbf{y}_1(\mathbf{x}_1), .., \mathbf{y}_N(\mathbf{x}_N))$ is the j-th constraint, $\mathbf{x}$ is the vector of design variables and $\mathbf{A}$ and $\mathbf{B}$ are the upper and lower bounds respectively on the design variables. As the optimisation problem does not differ from a mono disciplinary optimisation problem, it is, theoretically, trivial to implement within existing optimisation methods given that all disciplines and multidisciplinary analysis schemes have been set up *a priori*.

2.2.2 Individual discipline feasible

Individual discipline feasible (IDF) also formulated by Cramer *et al.* (1993), enforces multidisciplinary feasibility through explicit constraints in the optimisation problem which eliminates the need for multidisciplinary analysis. A conceptual flowchart for the three discipline discipline problem is shown in Figure 2.5.

In order to enforce multidisciplinary feasibility, copies, $\mathbf{w}_d^+$, of the coupling variables, $\mathbf{w}_d$, where d denotes the discipline number, are created which are treated as design variables in the optimisation problem and passed to the relevant disciplines. Multidisciplinary feasibility is achieved by constraining the coupling variable copies to

equal the coupling variables. The optimisation problem for IDF becomes

$$\begin{aligned} \underset{\mathbf{x},\mathbf{w}^+}{\text{minimise}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_j(\mathbf{x}) \leq 1, && j = 1, \ldots, m \\ & A_i \leq x_i \leq B_i, && i = 1, \ldots, n \\ & \mathbf{c}_d = \mathbf{w}_d^+(\mathbf{x}_d) - \mathbf{w}_d(\mathbf{x}_d) = \mathbf{0}, && d = 1, \ldots, N \end{aligned} \tag{2.2}$$

where $\mathbf{c}_d$ denotes the consistency constraints for coupling variable copies $\mathbf{w}_d^+$ corresponding to coupling variables $\mathbf{w}_d$.

The benefit of IDF is that advantage can be taken of not requiring multidisciplinary analysis at each design point and hence the computational budget can be decreased. It also means that multidisciplinary feasibility is not ensured until convergence, something that may be prohibitive for industrial use where the available time may not be enough for a converged solution. Furthermore, the number of both design variables and constraints increases by the number of coupling variables.(Martins and Lambe, 2013).

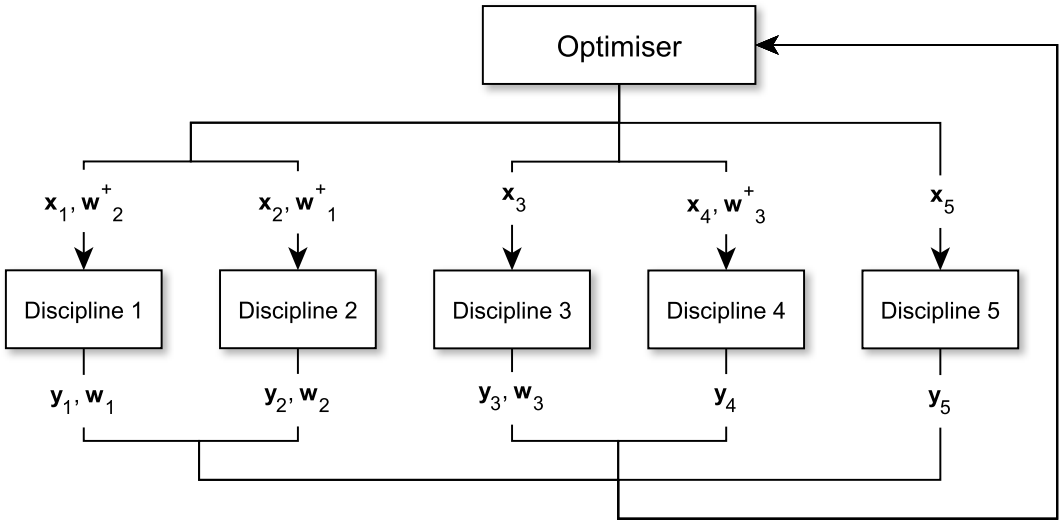


Figure 2.5: Individual discipline feasible (IDF) architecture

2.2.3 Simultaneous analysis and design

Simultaneous analysis and design (SAND), presented by Haftka (1985), also known as *All at once (AAO)* by Cramer *et al.* (1993), enforces multidisciplinary and interdisciplinary feasibility by explicit constraints in the optimisation problem. In other words, the discrete equations of the discipline analyses is included in the optimisation problem as constraints and only residuals are computed by the discipline simulations. Interdisciplinary feasibility is then enforced by equality constraints. Just like IDF, multidisciplinary feasibility is handled with consistency constraints. The SAND architecture is shown in Figure 2.6, and the optimisation problem is given as

$$\begin{aligned}
\underset{\mathbf{x},\mathbf{u}}{\text{minimise}} \quad & f_0(\mathbf{x}) \\
\text{subject to} \quad & f_j(\mathbf{x}) \leq 1, && j = 1, \ldots, m \\
& A_i \leq x_i \leq B_i, && i = 1, \ldots, n \\
& \mathbf{c}_d = \mathbf{w}_d^+(\mathbf{x}_d) - \mathbf{w}_d(\mathbf{x}_d) = \mathbf{0}, && d = 1, \ldots, N \\
& \mathbf{r}_d(\mathbf{u}_d) = \mathbf{0}, && d = 1, \ldots, N
\end{aligned} \tag{2.3}$$

where $\mathbf{r}_i$ are the residuals of discipline i with state variables $\mathbf{u}_i$.

The motivation for SAND is that the computational budget can be further reduced by including both multidisciplinary and interdisciplinary feasibility as optimisation constraints. This however means that interdisciplinary and multidisciplinary feasibility is not ensured until convergence, the number of variables in the optimisation problem increases dramatically and it limits the use of software products to the ones which can output residuals (Martins and Lambe, 2013).

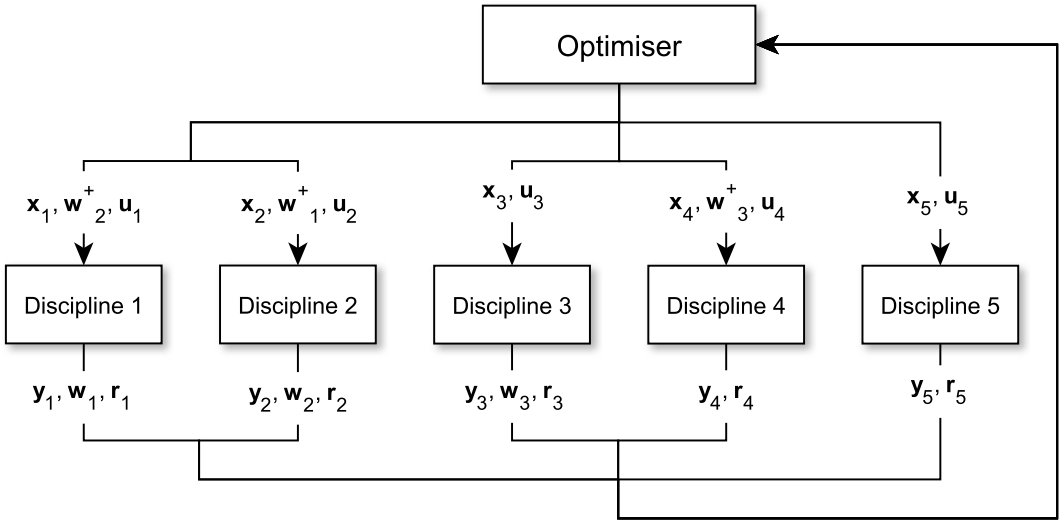


Figure 2.6: Simultaneous analysis and design (SAND) architecture

2.3 Choice of architecture and motivation

The SAND architecture is appealing as it, in the same way as MDO brings together all disciplines in one optimisation problem, brings together all constraints into one optimisation problem. However, most commercial software products would not have the option to output residuals which are required for the use of SAND. Furthermore the number of optimisation variables and responses would increase dramatically for large problems which is a problem for optimisation techniques that do not require gradients. It is therefore concluded that SAND is not a suitable architecture for the purpose of this work.

In the current research multi-physics simulations are not considered, making the MFD and IDF architectures equivalent as there are no coupling variables present. However, the rationale for deciding which architecture to use is based on the possibility of solving such problems within the planned framework.

The IDF architecture is interesting for problems including multi-physics coupling as it could potentially be made computationally efficient by not requiring multidisciplinary feasibility in each iteration of the optimisation process. Unlike SAND it is possible to use with commercial software as no residual output is needed. However, the number of design variables increase with the number of introduced coupling variables. A large number of coupling variables could therefore make the problem unaffordable if gradients are not available.

Despite the penalty in computational cost compared to IDF and SAND the MDF architecture is chosen for use throughout this work. The motivation is that the MDF architecture requires none or very little change to an already existing process for evaluation the requirements that will be used in the MDO, has already been set up. Furthermore it keeps both interdisciplinary and multidisciplinary feasibility enforced throughout the optimisation process which is advantageous for industrial use as time constraints might force premature termination of the optimisation process. It is also the author's opinion that MDF is the most suitable architecture for the use of metamodels as it keeps the number of variables to a minimum.

2.4 Summary

In multidisciplinary design optimisation several disciplines are included in one integrated optimisation problem. This chapter served as an introduction to MDO and the related terminology. Three monolithic architectures, multidisciplinary feasible, individual design feasible, and simultaneous analysis and design, were introduced and described.

It was argued that monolithic architectures are more efficient in terms of computational cost than distributed architectures, and therefore various monolithic MDO architectures were covered and their potential impact on the planned framework

discussed. The multidisciplinary feasible architecture was chosen as the most suitable for the new MDO framework as it is compatible with existing commercial finite element software, keeps interdisciplinary and multidisciplinary feasibility enforced throughout the optimisation process. Furthermore, it is advantageous to use with metamodels because it keeps the number of variables to a minimum.

Chapter 3

Metamodel-based optimisation

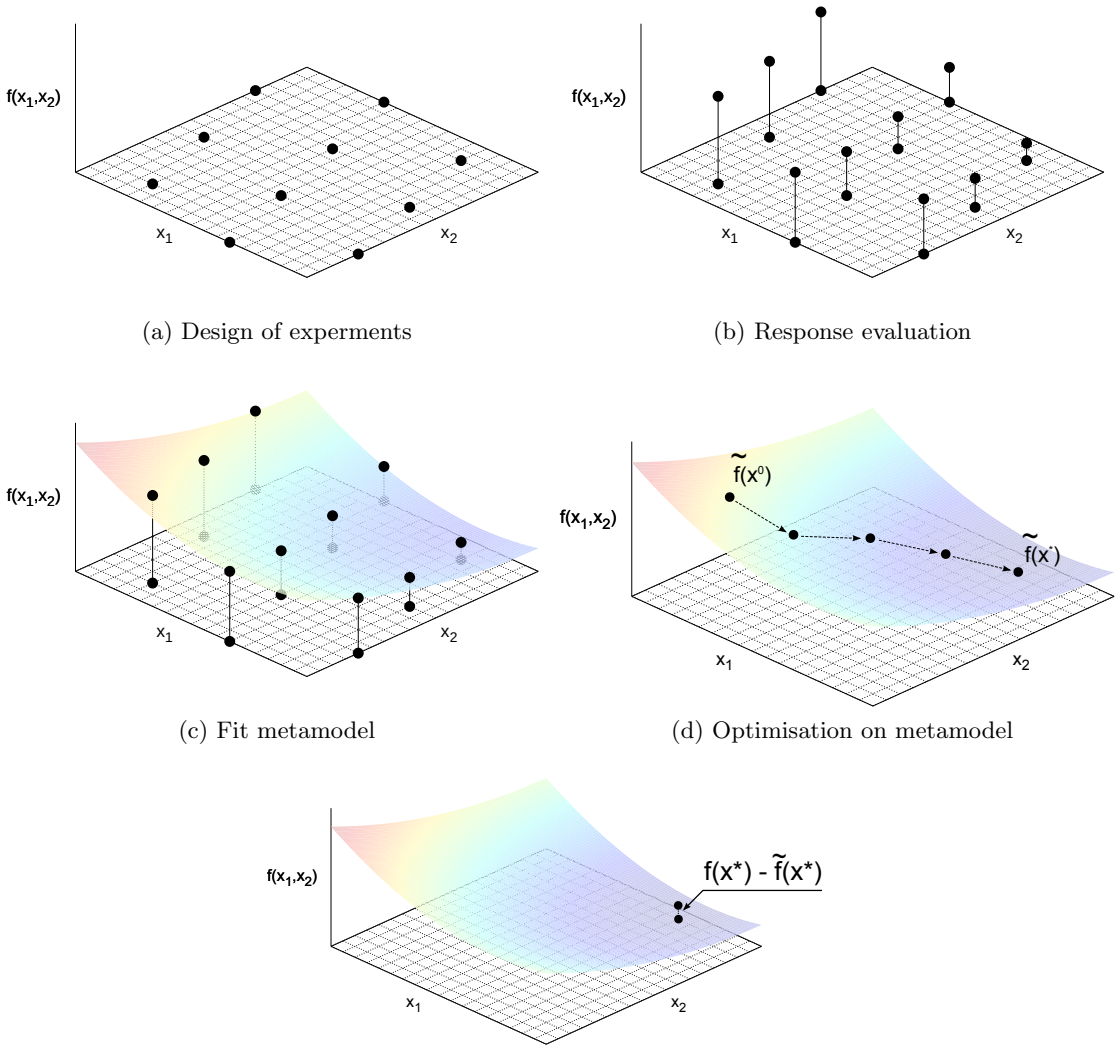
Metamodels are frequently used to replace computationally expensive simulations with cheaper mathematical models. Metamodels are also commonly referred to as surrogate models, response surfaces, or approximations. Using metamodels to replace response functions in optimisation is called metamodel-based optimisation and will be discussed in this chapter.

The chapter starts with an introduction to metamodel-based optimisation in Section 3.1 and Section 3.2 presents a short overview of design of experiments method. Section 3.3 outlines two approximation methods used throughout this work, namely the moving least squares and kriging. Section 3.4 describes two gradient based optimisation algorithms used in this work, the method of feasible directions and sequential quadratic programming. The chapter concludes by describing a novel method of carrying out hyper parameter tuning for kriging metamodels in Chapter 5.

3.1 Introduction to metamodel-based optimisation

Several review articles have been written on the subject of metamodel-based optimisation, for instance, Barthelemy and Haftka (1993), Wang and Shan (2007), Forrester and Keane (2009), and Viana *et al.* (2014, 2010). Barthelemy and Haftka (1993) categorise approximations depending on their intended range of use within the design space. Local approximations are valid in the immediate vicinity of a point. Examples of local approximations are Taylor series expansions. Global approximations are intended to be used throughout the entire design space, while mid-range approximations are intended to be used in a sub-region of the design space. This section focuses on the two latter categories as local approximations are not within the scope of the research. More information on local approximations can be found in, for instance, Haftka and Gurdal (1992).

The process of metamodel-based design optimisation is depicted in Figure 3.1. The first step (a) is to decide the location of a set of training points, the i-th point is denoted $\mathbf{x}^{(i)} = (x_1^{(i)}, ..., x_n^{(i)})$, where n denotes the number of design variables. This is followed by (b) evaluation of the response at the training points $f^{(i)} = f(\mathbf{x}^{(i)})$ and (c) fitting a suitable model $\tilde{f}(\mathbf{x})$ to the evaluated points. Once the metamodel has been created it can be used to calculate an approximate response at new points at a much reduced computational cost. For instance, optimisations (d) can be carried out using the metamodel to find an approximate optimum $\tilde{f}(\mathbf{x}^*)$, which is verified by evaluating the true function at the same point, i.e. $f(\mathbf{x}^*)$. It is common to return to adding more training points and repeating the process if the metamodel has not reached desired accuracy.



(a) Design of experments

(b) Response evaluation

(c) Fit metamodel

(d) Optimisation on metamodel

(e) Verify discrepancy of model and true function

Figure 3.1: The metamodel-based optimisation process.

3.2 Design of experiments

The task of determining the locations of the training points within the design space is commonly known as Design of Experiments (DOE) and has its origin in the planning of physical experiments, see for instance, Box and Draper (1987). For physical experiments, classical methods were used with the emphasis on estimating the effects of variables and

reducing the effect of noise due to uncontrollable factors (Grove and Davis, 1992). Several classical methods exist, such as factorial designs (Fisher, 1960), central compsite designs (Box and Wilson, 1951), Box-Behnken (Box and Behnken, 1960) and Plackett-Burman (Plackett and Burman, 1946), to name a few.

For building response surfaces of deterministic computer simulations, DOE methods with space-filling properties are preferred. A particular type of space filling design that has a good distribution when projected onto any of the individual variable sub-spaces is the Latin hypercube design (Audze and Eglajs, 1977; M. D. McKay, 1979). One of the properties that are desired for computer experiments is a uniform distribution of points throughout the design space. Therefore it is very common to use the so called uniform latin hypercube design (ULH) proposed by Audze and Eglajs (1977), where optimisation is used to generate uniform Latin hypercube designs according to an objective function related to uniformity. This optimisation process has been made more efficient since by using a permutation genetic algorithm (Bates *et al.*, 2003). Further improvements to the ULH were proposed by Kianifar *et al.* (2016) in order to sequentially add training and validation points to the design space while keeping the combined set of points optimal. This is a desirable feature as one seldom knows the number of required points to reach desired accuracy beforehand.

Another set of space filling designs are based on quasi-random low discrepancy sequences such as Halton (Halton, 1964), Sobol (Sobol', 1967) and Hammersley (Hammersley and Handscomb, 1964) sequences. In the present work a technique developed within Altair HyperStudy (Altair Engineering, Inc., 2014a), called modified extensible lattice sequences (MELS), is used. The technique is based on extensible lattice sequences proposed by Hickernell *et al.* (2000) and allows for creating lattice sequences that do not require knowing the number of points *a priori*. This makes the technique, just like the ULH proposed by Kianifar *et al.* (2016), suitable for sequential use.

3.3 Metamodel techniques

There is a large number of modern metamodel techniques ranging from polynomial regression techniques such as the moving least squares method (Lancaster and Salkauskas, 1981), interpolating techniques such as radial basis functions (Broomhead and Lowe, 1988) and kriging (Sacks *et al.*, 1989), to machine learning techniques such as support vector regression (Vapnik and Vapnik, 1998) and artificial neural networks (Rojas, 2013). In this section two metamodel techniques, used in this work, are presented. The moving least squares method is based on regression around a point (the evaluation point) and Kriging is an interpolating metamodel technique based on spatial correlation.

3.3.1 Moving least squares method

The moving least squares method (MLSM) was initially proposed by Lancaster and Salkauskas (1981) for smoothing and interpolation of scattered data and later used in the mesh-free form of the Finite Element Method (Liszka, 1984). Choi *et al.* (2001) later proposed the use of MLSM as a metamodelling technique. Just like simple polynomial regression, the following model is used:

$$f(\mathbf{x}^{(i)}) = \tilde{f}(\mathbf{x}^{(i)}) + \epsilon_i = \sum_{j=1}^{h} b_j(\mathbf{x}^{(i)})a_j + \epsilon_i, \quad (i = 1, ..., p), \tag{3.1}$$

where $b_j(\mathbf{x}^{(i)})$ is the j-th regressor, a_j the corresponding regression coefficient, and ϵ_i are assumed to be normally distributed independent errors, e.g. noise. In the case of a linear basis polynomial the number of regressors is $h = n + 1$, and a resulting vector of

regressors can be written as

$$\mathbf{b}(\mathbf{x}^{(i)}) = \left[\begin{array}{cccc} 1 & x_1^{(i)} & \dots & x_n^{(i)} \end{array} \right], \quad (i = 1, ..., p). \tag{3.2}$$

Higher order polynomial basis functions can be used by adding additional terms. However, a quadratic polynomial basis requires $h = (n+1)(n+2)/2$, which may be difficult to obtain for problems with a large number of design variables and expensive function evaluations. The regression coefficients are obtained by means of a weighted least squares problem defined as

$$\underset{\mathbf{a}}{\text{minimise}}\ G(\mathbf{a}) \equiv \sum_{i=1}^{p} \left\{ w_i(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) \left[\tilde{f}(\mathbf{x}^{(i)}, \mathbf{a}) - f(\mathbf{x}^{(i)}) \right] \right\}^2, \tag{3.3}$$

where the weight w_i depends on both the associated training point $\mathbf{x}^{(i)}$ and the evaluation point $\mathbf{x}^{(e)}$ according to a weight decay formula, discussed in Section 3.3.1.2. The stationary point of (3.3) can be found through the system of normal equations written in matrix form as

$$\mathbf{B}^T\mathbf{W}\mathbf{B}\mathbf{a} = \mathbf{B}^T\mathbf{W}\mathbf{f}, \tag{3.4}$$

where $\mathbf{B}$ contains the polynomial terms for each training point according to

$$\mathbf{B} = \left[\begin{array}{ccc} b_1(\mathbf{x}^{(1)}) & \dots & b_h(\mathbf{x}^{(1)}) \\ \vdots & \ddots & \vdots \\ b_1(\mathbf{x}^{(p)}) & \dots & b_h(\mathbf{x}^{(p)}) \end{array} \right]. \tag{3.5}$$

$\mathbf{W}$ is a diagonal matrix with the weights between each training point and the evaluation point

$$\mathbf{W} = diag \left[\begin{array}{cccc} w_1(\mathbf{x}^{(1)}, \mathbf{x}^{(e)}) & \dots & w_p(\mathbf{x}^{(p)}, \mathbf{x}^{(e)}) \end{array} \right], \tag{3.6}$$

and $\mathbf{f}$ contains the function values at each of the training points

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}^{(1)}) & \dots & f(\mathbf{x}^{(p)}) \end{bmatrix}^T. \tag{3.7}$$

Solving the system of equations (3.4) for the coefficients in $\mathbf{a}$ yields the moving least squares approximation at a point $\mathbf{x}_e$ as

$$\tilde{f}(\mathbf{x}^{(e)}) = \mathbf{b}(\mathbf{x}^{(e)})^T\mathbf{a}. \tag{3.8}$$

It is worth noting that as the weights matrix (3.6) is a function of the evaluation point, the system of equations needs to be resolved for every evaluation point.

3.3.1.1 Gradient-enhanced moving least squares method

In order to utilise available partial derivatives, gradients, with respect to the design variables one can make use of the gradient enhanced moving least squares method (GE-MLSM) described by Choi *et al.* (2001).

The least squares problem (3.3) is modified to contain not only the function values but also the derivatives as

$$\begin{aligned} \underset{\mathbf{a}}{\text{minimise}}\ G(\mathbf{a}) \equiv \sum_{i=1}^{p} \Bigg\{ & w_i(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) \left[\tilde{f}(\mathbf{x}^{(i)}, \mathbf{a}) - f(\mathbf{x}^{(i)}))\right]^2 \\ & + \sum_{j=1}^{n} w_i^j \left[\frac{\partial \tilde{f}(\mathbf{x}^{(i)}, \mathbf{a})}{\partial \mathbf{x}_j} - \frac{\partial f(x^{(i)})}{\partial x_j}\right]^2 \Bigg\}. \end{aligned} \tag{3.9}$$

The weights for the derivative terms w_i^j are the weights for the corresponding funcion value multiplied by a factor according to

$$w_i^j = \delta w_i, \quad 0 < \delta \leq 1\cdot \tag{3.10}$$

δ is a user defined value indicating how much importance is given to the derivative terms relative to the function values. Just like the non-derivative case, the stationary point is found by solving the system (3.4), where the matrix $\mathbf{B}$ now not only contains the basis polynomial for each training point but also the derivatives with respect to the design variables as

$$\mathbf{B} = \begin{bmatrix} b_1(\mathbf{x}^{(1)}) & \dots & b_h(\mathbf{x}^{(1)}) \\ \partial_1 b_1(\mathbf{x}^{(1)}) & \dots & \partial_1 b_h(\mathbf{x}^{(1)}) \\ \vdots & \ddots & \vdots \\ \partial_n b_1(\mathbf{x}^{(1)}) & \dots & \partial_n b_h(\mathbf{x}^{(1)}) \\ \vdots & \ddots & \vdots \\ b_1(\mathbf{x}^{(p)}) & \dots & b_h(\mathbf{x}^{(p)}) \\ \partial_1 b_1(\mathbf{x}^{(p)}) & \dots & \partial_1 b_h(\mathbf{x}^{(p)}) \\ \vdots & \ddots & \vdots \\ \partial_n b_1(\mathbf{x}^{(p)}) & \dots & \partial_n b_h(\mathbf{x}^{(p)}) \end{bmatrix}. \tag{3.11}$$

∂_k denotes the partial derivative with respect to the k-th design variable. $\mathbf{W}$ is updated to contain the weights for the derivative terms according to

$$\mathbf{W} = diag \begin{bmatrix} \mathbf{w}_1(\mathbf{x}^{(1)}, \mathbf{x}^{(e)}) & \dots & \mathbf{w}_p(\mathbf{x}^{(p)}, \mathbf{x}^{(e)}) \end{bmatrix}, \tag{3.12}$$

where

$$\mathbf{w}_i(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) = diag \begin{bmatrix} \mathbf{w}_i^1(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) & \dots & \mathbf{w}_i^n(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) \end{bmatrix}, \tag{3.13}$$

and the vector f now contains the function values and gradients at each of the training points.

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}^{(1)}) \\ \partial_1 f(\mathbf{x}^{(1)}) \\ \vdots \\ \partial_n f(\mathbf{x}^{(1)}) \\ \vdots \\ f(\mathbf{x}_p) \\ \partial_1 f(\mathbf{x}^{(p)}) \\ \vdots \\ \partial_n f(\mathbf{x}^{(p)}) \end{bmatrix}. \tag{3.14}$$

Solving the system of equations (3.4) for the coefficients in $\mathbf{a}$ yields the gradient-enhanced moving least squares approximation at a point $\mathbf{x}_e$ according to (3.8).

3.3.1.2 Weight decay function

The weights, appearing in (3.6) and (3.13), are calculated according to a weight decay function. Points that are close to the evaluation point are given high weighting while points far away are given low weighting. A popular choice for the weight decay function is the Gaussian function

$$w_i(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}) = e^{-\theta||\mathbf{x}^{(i)} - \mathbf{x}^{(e)}||^2}. \tag{3.15}$$

$||\cdot||$ denotes the Euclidean norm. θ is commonly referred to as the closeness of fit parameter and controls the rate by which the weight decays in the Gaussian function. A low value of the closeness of fit parameter will assign high weights across all training points, resulting in a loose fit, whilst a high value will lead to a rapid weight decay and a close fit. Figure 3.3.2.1 shows the Gaussian weight decay function for four different

values of θ. It is important to carefully choose the value of θ as an underestimated value may result in a very loose fit that does not represent the general trend of the function well and an overestimation may result in over-fitting. This is illustrated in Figure 3.3 where three MLSM fits using different values of the closeness of fit parameter are shown.

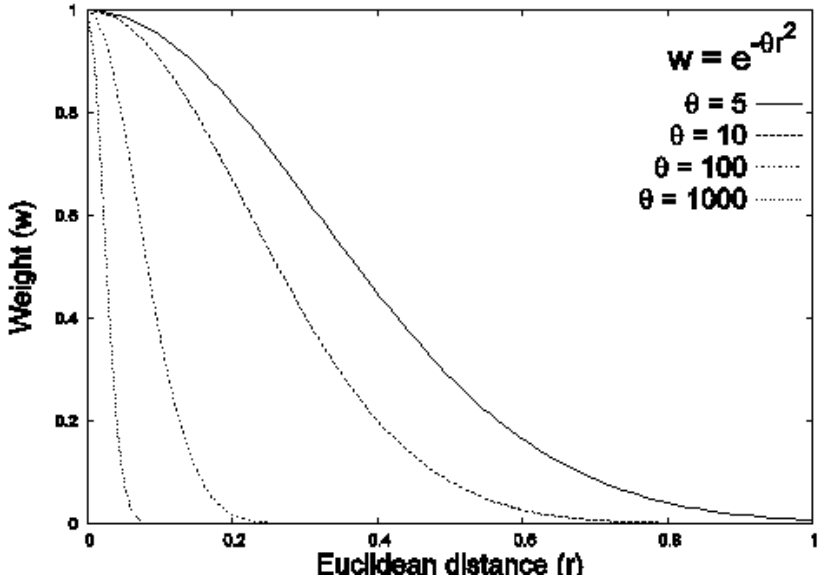


Figure 3.2: Gaussian weight decay function with varying closeness of fit parameter.

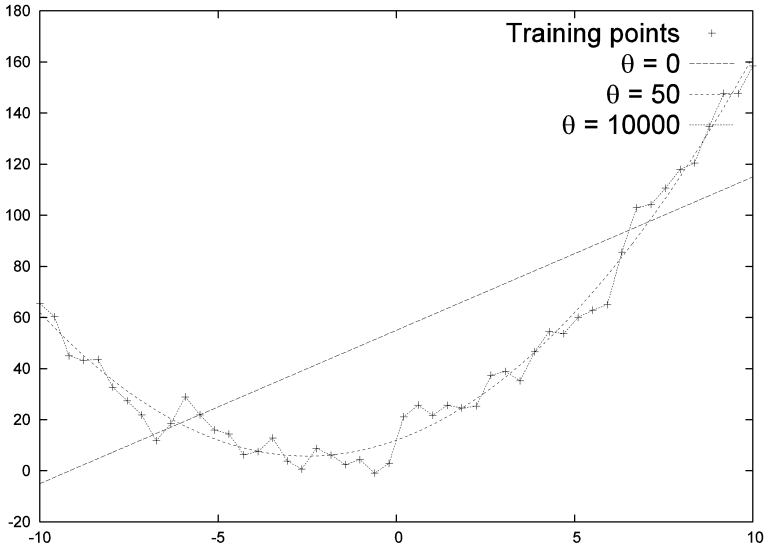


Figure 3.3: Three MLSM fits with different closeness of fit parameter values.

3.3.1.3 Closeness of fit parameter tuning

In order to determine the closeness of fit parameter in a systematic way, cross-validation can be used. Cross validation is a way of estimating the quality of the approximation and it is usually implemented by evaluating a set of points that are not included in the set of training points used to build the approximation. The drawback of this procedure is that additional evaluations are needed and hence will add to the cost of building the approximation.

As demonstrated by Tu and Jones (2003), a Leave-One-Out Cross-Validation (LOOCV) methodology can be used to estimate the error of the metamodel without the need for additional training points. This is done by successively leaving one point out of the training set, constructing the surrogate model on the remaining set and evaluating the approximate function value, at the point left out. The discrepancy between the approximate function value $\tilde{f}_{p-1}(\theta, \mathbf{x}^{(i)})$, and the true function value, $f(\mathbf{x}^{(i)})$, is evaluated as an estimated approximation error at that point. This is done for all of the points in the set and used to calculate a Root Mean Square Error (RMSE) for the current closeness of fit parameter value, as:

$$RMSE(\theta) = \sqrt{\frac{\sum_{i=1}^{p} \left[f(\mathbf{x}^{(i)}) - \tilde{f}_{p-1}(\theta, \mathbf{x}^{(i)}) \right]^2}{p}}. \tag{3.16}$$

Any one dimensional optimisation technique can then be used in order to find the value that minimises $RMSE(\theta)$. Figure 3.4 illustrates a typical function $RMSE(\theta)$ and the corresponding location of the minimum.

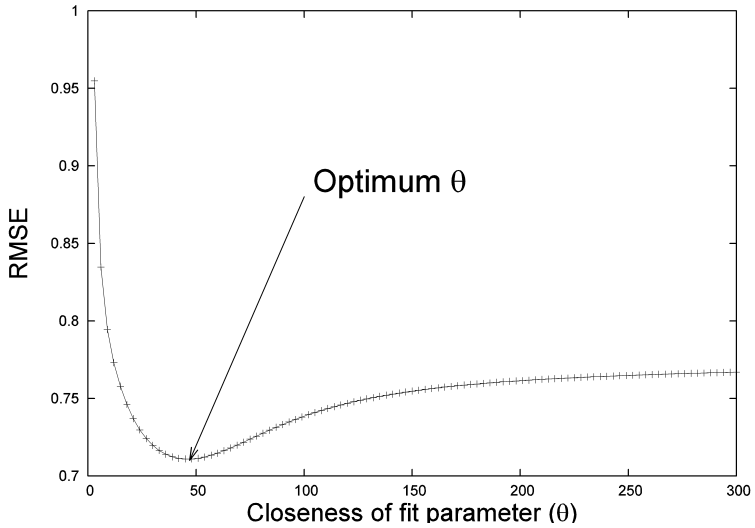


Figure 3.4: RMSE as a function of the closeness of fit parameter.

3.3.1.4 Variable ranking

Another application of the Cross-Validated Moving Least Squares Method is design variable ranking. A backwards elimination ranking proposed by Tu and Jones (2003) is carried out by calculating an impact factor for each design variable based on successively leaving out each variable. An impact factor for a particular variable, x_j, can be calculated by building an approximation that ignores the effect of x_j. The RMSE of this approximation is then compared to the error of an approximation built with the full set of variables as

$$I_j = \frac{RMSE^j - RMSE}{RMSE}, \tag{3.17}$$

where $RMSE^j$ denotes the RMSE for an approximation built without the variable x_j, and $RMSE$ is the error for an approximation built on the full set of variables. If the impact factor is a measure of the importance of each variable on the response, a small or negative value for a variable indicates that the variable is a candidate for elimination. The main benefit of this design variable ranking technique is that, unlike methods such analysis of variance (ANOVA), it can account for coupling effects between variables.

3.3.2 Kriging

Kriging is an interpolating metamodel technique based on spatial correlation that was first proposed by Krige (1951) and later implemented by Matheron (1963) for use within the mining industry. The use of kriging for approximation of expensive computational models was shown by Sacks *et al.* (1989).

Following the notation of Jones (2001), kriging is derived from the assumption that computer simulations are entirely deterministic and any error in the fit of a metamodel is entirely down to missing terms in the model. Hence, the error term in (3.1) can be written as a function of $x^{(i)}$ and becomes

$$f(\mathbf{x}^{(i)}) = \sum_{j=1}^{h} b_j(\mathbf{x}^{(i)})a_j + \epsilon(x^{(i)}), \quad (i = 1, ..., p). \tag{3.18}$$

Furthermore it is assumed that the error, $\epsilon_i(x^{(i)})$, is continuous for any continuous function $f(\mathbf{x}^{(i)})$, and that the error at two points $\mathbf{x}^{(i)}$ and $\mathbf{x}^{(j)}$ are correlated with their distance according to a model $\psi(\mathbf{x}_i, \mathbf{x}_j)$. As the error is modelled explicitly in kriging, the model will exactly interpolate the training points.

The first part of the model (3.18), the polynomial regression, can be of arbitrary order. However, the order of the regression model will dictate the number of required points which must be at least as many as the number of regressors. Kriging with zero-th order polynomials is usually referred to as "ordinary" kriging while using first order or higher order polynomials are termed "universal" kriging. Ordinary kriging tends to be the most popular method as trends are not usually known beforehand (Forrester and Keane, 2009) and hence this will be the focus of this section. For ordinary kriging the

model can be written as

$$f(\mathbf{x}^{(i)}) = \hat{\mu} + \epsilon_i(\mathbf{x}^{(i)}), \quad (i = 1, ..., p), \tag{3.19}$$

where the estimated mean $\hat{\mu}$ is determined by solving the weighted least squares problem

$$\hat{\mu} = \left(\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{R}^{-1}\mathbf{f}, \tag{3.20}$$

and the matrix of regressors, representing a zero-th order basis function in ordinary kriging, is reduced to a vector of ones according to

$$\mathbf{B} = \mathbf{1} = [\underbrace{1, ..., 1}_{\text{p}}]^T. \tag{3.21}$$

The matrix $\mathbf{R}$ is discussed shortly. The error, treated as a stochastic process, is modelled as

$$\epsilon(\mathbf{x}^{(i)}) = \mathbf{w}^T\mathbf{r}(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}), \tag{3.22}$$

where $\mathbf{r}$ contains basis functions depending on some specified spatial correlation between the evaluation point and the training points

$$\mathbf{r} = \left[\psi(\mathbf{x}^{(e)}, \mathbf{x}^{(1)}), ..., \psi(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})\right]^T. \tag{3.23}$$

This is commonly modelled as a Gaussian function according to

$$\psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) = exp\left[\sum_{k=1}^{n} -\theta_k(x_k^{(i)} - x_k^{(j)})^2\right], \tag{3.24}$$

where $\theta_k, k = 1, ..., n$ are tuning parameters, often denoted hyper parameters that needs to be determined through optimisation in order to produce a good quality metamodel. This is further discussed in Chapter 5.

The weights are calculated as:

$$\mathbf{w} = \mathbf{R}^{-1} \left(\mathbf{f} - \mathbf{B}\hat{\mu} \right), \tag{3.25}$$

where $\mathbf{R}$ contains the estimated spatial correlation between all the training points, including between themselves on the diagonal,

$$\mathbf{R} = \begin{bmatrix} \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}) & \dots & \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(p)}) \\ \vdots & \ddots & \vdots \\ \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(1)}) & \dots & \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(p)}) \end{bmatrix}, \tag{3.26}$$

and $\mathbf{f}$ contains the function values at the training points,

$$\mathbf{f} = \left[f(\mathbf{x}^{(i)}), ..., f(\mathbf{x}^{(p)}) \right]^T. \tag{3.27}$$

The estimated system variance is calculated as

$$\hat{\sigma}^2 = \frac{1}{p} \left(\mathbf{f} - \mathbf{B}\mu \right)^T \mathbf{R}^{-1} \left(\mathbf{f} - \mathbf{B}\mu \right), \tag{3.28}$$

the final predicted kriging estimation at a point $\mathbf{x}^{(e))}$ is given by

$$\tilde{f}(\mathbf{x}^{(e)}) = \hat{\mu} + \mathbf{w}^T \mathbf{r}(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}), \tag{3.29}$$

and a predicted mean squared error of the predictor is

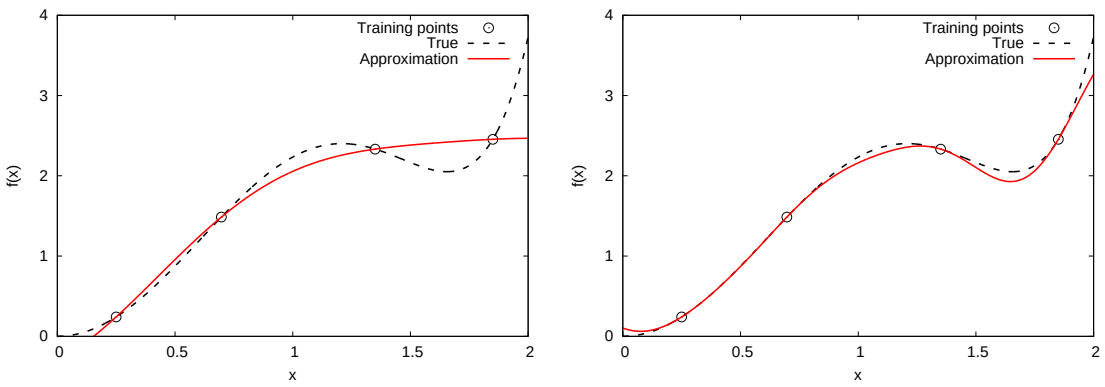
$$s^2(\mathbf{x}^{(e)}) = \hat{\sigma}^2 \left[1 - \mathbf{r}^T \mathbf{R}^{-1} \mathbf{r} + \frac{(1 - \mathbf{B}^T \mathbf{R}^{-1} \mathbf{r})^2}{\mathbf{B}^T \mathbf{R}^{-1} \mathbf{B}} \right]. \tag{3.30}$$

3.3.2.1 Gradient-enhanced kriging

If, in addition to the function values, gradients are available, they may be used to improve the accuracy of the kriging metamodel. The method of incorporating gradients in kriging is called gradient enhanced kriging (GEK) and is described in this section. Figure 3.5 shows a one-dimensional example with a kriging and gradient enhanced kriging fit respectively. It is clear that the gradient enhanced fit is of superior quality. The presented implementation is as described in Han *et al.* (2013) to which the reader is referred for further information.

In order to create a gradient enhanced kriging fit the correlation matrix needs to be extended to include derivative terms according to

$$\mathbf{R} = \begin{bmatrix} \mathbf{Q}^{1,1} & \mathbf{Q}^{1,2} \\ (\mathbf{Q}^{1,2})^T & \mathbf{Q}^{2,2} \end{bmatrix}, \tag{3.31}$$



(a) Kriging (b) Gradient-enhanced kriging

Figure 3.5: An illustrative Comparison between kriging and GEK metamodel in one dimension.

where $\mathbf{Q}^{1,1}$ is the same as the correlation matrix used in the non-gradient case

$$\mathbf{Q}^{1,1} = \begin{bmatrix} \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(1)}) & \dots & \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(p)}) \\ \vdots & \ddots & \vdots \\ \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(1)}) & \dots & \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(p)}) \end{bmatrix}. \tag{3.32}$$

$\mathbf{Q}^{1,2}$ contains the first derivatives of $\mathbf{R}$ according to

$$\mathbf{Q}^{1,2} = \begin{bmatrix} \frac{\partial \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(1)})}{\partial \mathbf{x}^{(1)}} & \dots & \frac{\partial \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(p)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(1)})}{\partial \mathbf{x}^{(1)}} & \dots & \frac{\partial \psi(\mathbf{x}^{(p)} \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(p)}} \end{bmatrix}, \tag{3.33}$$

where

$$\frac{\partial \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial \mathbf{x}^{(j)}} = \begin{bmatrix} \frac{\partial \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(i)})}{\partial x_1^{(j)}} \dots \frac{\partial \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(i)})}{\partial x_n^{(j)}} \end{bmatrix}. \tag{3.34}$$

and, using the Gaussian function (3.24), leads to

$$\frac{\partial \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_k^{(j)}} = 2\theta_k \left(x_k^{(i)} - x_k^{(j)} \right) \psi(\mathbf{x}_k^{(i)}, \mathbf{x}_k^{(j)}). \tag{3.35}$$

The sub-matrix $\mathbf{Q}^{2,2}$ contains the second derivatives

$$\mathbf{Q}^{2,2} = \begin{bmatrix} \frac{\partial^2 \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(1)})}{\partial \mathbf{x}^{(1)} \partial \mathbf{x}^{(1)}} & \dots & \frac{\partial^2 \psi(\mathbf{x}^{(1)}, \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(1)} d\mathbf{x}^{(p)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi(\mathbf{x}^{(p)}, \mathbf{x}^{(1)})}{\partial \mathbf{x}^{(p)} \partial \mathbf{x}^{(1)}} & \dots & \frac{\partial^2 \psi(\mathbf{x}^{(p)} \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(p)} \partial \mathbf{x}^{(p)}} \end{bmatrix}, \tag{3.36}$$

where

$$\frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial \mathbf{x}^{(i)} \partial \mathbf{x}^{(j)}} = \begin{bmatrix} \frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_1^{(i)} \partial x_1^{(j)}} & \dots & \frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_n^{(i)} \partial x_1^{(j)}} \\ \vdots & \ddots & \vdots \\ \frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_1^{(i)} \partial x_n^{(j)}} & \dots & \frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_n^{(i)} \partial x_n^{(j)}} \end{bmatrix}, \tag{3.37}$$

and, using the Gaussian function (3.24), leads to

$$\frac{\partial^2 \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)})}{\partial x_l^{(i)} \partial x_k^{(j)}} = \begin{cases} 2\theta_k \left[-2\theta_k \left(x_k^{(i)} - x_k^{(j)} \right)^2 + 1 \right] \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \ , \ k = l \\ -4\theta_k \theta_l \left[\left(x_k^{(i)} - x_k^{(j)} \right) \left(x_l^{(i)} - x_l^{(j)} \right) \right] \psi(\mathbf{x}^{(i)}, \mathbf{x}^{(j)}) \ , \ k \neq l \end{cases} . \quad (3.38)$$

The vector of spatial correlations between the evaluation points and the training points is extended in the same manner to

$$\mathbf{r} = \begin{bmatrix} \psi(\mathbf{x}^{(e)}, \mathbf{x}^{(1)}) \\ \vdots \\ \psi(\mathbf{x}^{(e)}, \mathbf{x}^{(p)}) \\ \frac{\partial \psi(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(1)}} \\ \vdots \\ \frac{\partial \psi(\mathbf{x}^{(e)}, \mathbf{x}^{(p)})}{\partial \mathbf{x}^{(p)}} \end{bmatrix} , \quad (3.39)$$

using the expression in (3.34) and (3.35). The vector of function values is extended to include gradients as well as the function values

$$\mathbf{f} = \begin{bmatrix} f(\mathbf{x}^{(1)}) \\ \vdots \\ f(\mathbf{x}^{(p)}) \\ \frac{\partial f(\mathbf{x}^{(1)})}{\partial \mathbf{x}} \\ \vdots \\ \frac{\partial f(\mathbf{x}^{(p)})}{\partial \mathbf{x}} \end{bmatrix} , \quad (3.40)$$

where

$$\frac{\partial f(\mathbf{x}^{(i)})}{\partial \mathbf{x}} = \begin{bmatrix} \frac{\partial f(\mathbf{x}^{(i)})}{\partial x_1} \\ \vdots \\ \frac{\partial f(\mathbf{x}^{(i)})}{\partial x_n} \end{bmatrix} . \quad (3.41)$$

Similarly, the basis polynomial is also extended to include a vector of zeros of length equal to the number of derivative terms

$$\mathbf{B} = [\mathbf{1} \quad \mathbf{0}] = [\underbrace{1, ..., 1}_{\text{p}}, \underbrace{0, ..., 0}_{\text{pn}}]^T, \tag{3.42}$$

recalling that n is the number of design variables and p is the number of training points. The predicted mean is now calculated as

$$\hat{\mu} = \left(\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}\right)^{-1}\mathbf{B}^T\mathbf{R}^{-1}\mathbf{f}, \tag{3.43}$$

the predicted system variance is calculated as

$$\hat{\sigma}^2 = \frac{1}{n(1+p)}\left(\mathbf{f} - \mathbf{B}\mu\right)^T\mathbf{R}^{-1}\left(\mathbf{f} - \mathbf{B}\mu\right), \tag{3.44}$$

and the weights vector takes the form

$$\mathbf{w} = \mathbf{R}^{-1}\left(\mathbf{f} - \mathbf{B}\hat{\mu}\right). \tag{3.45}$$

The final predicted kriging estimate at a point $\mathbf{x}^{(e)}$ is calculated as

$$\tilde{f}(\mathbf{x}^{(e)}) = \hat{\mu} + \mathbf{w}^T\mathbf{r}(\mathbf{x}^{(i)}, \mathbf{x}^{(e)}), \tag{3.46}$$

and a predicted mean squared error of the kriging prediction is given by

$$s^2(\mathbf{x}) = \hat{\sigma}^2\left[1 - \mathbf{r}^T\mathbf{R}^{-1}\mathbf{r} + \frac{(1 - \mathbf{B}^T\mathbf{R}^{-1}\mathbf{r})^2}{\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B}}\right]. \tag{3.47}$$

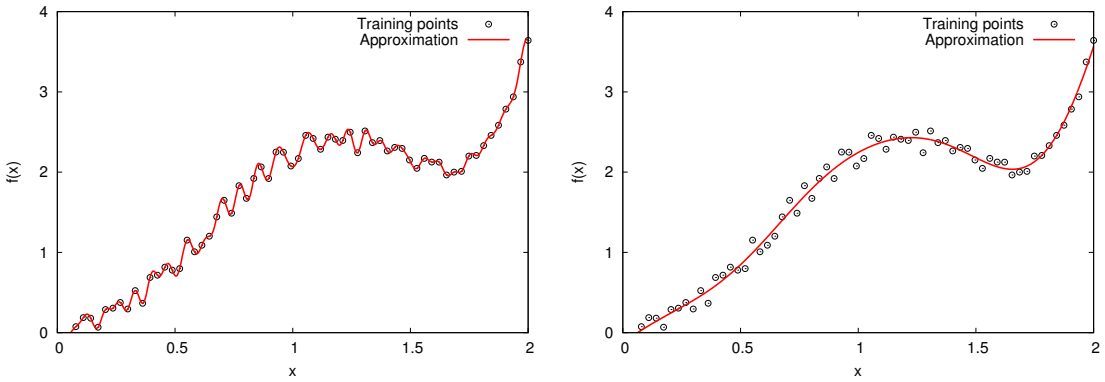
3.3.2.2 Noise regularisation

Although kriging is an interpolation technique, it is possible to create a regression-like metamodel through regularisation as suggested by Forrester *et al.* (2006). This is performed by adding another tuning parameter, commonly denoted regularisation parameter, λ to the diagonal elements of the correlation matrix as

$$\mathbf{R} = \mathbf{R} + \lambda \mathbf{I}. \tag{3.48}$$

This parameter, like the hyper parameters in (3.24), is to be determined through optimisation to produce a good quality fit. This is further discussed in Chapter 5. Figure 3.6 demonstrates a one-dimensional fit of a noisy function with and without regularisation. It can be seen that the regularisation allows the metamodel to deviate from the training points in order to follow the underlying trend.

For the gradient-enhanced case there is a possibility of noise in both the function values and the gradients values. As such it is beneficial to have two regularisation parameters, one relating to the function values and one for the gradients, as proposed



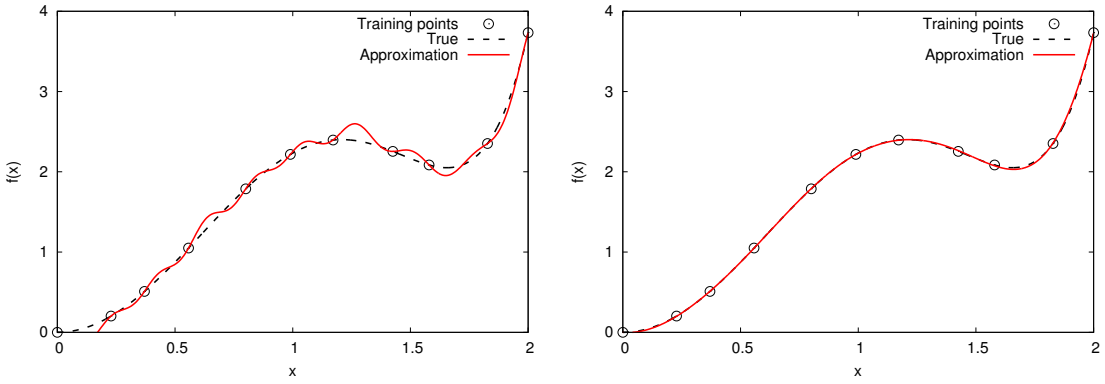
(a) Without regularisation (b) With regularisation

Figure 3.6: One-dimensional fit of a noisy function.

by Lukaczyk *et al.* (2013). This leads to an augmentation of the correlation matrix according to

$$\mathbf{R} = \begin{bmatrix} \mathbf{Q}^{1,1} & \mathbf{Q}^{1,2} \\ (\mathbf{Q}^{1,2})^T & \mathbf{Q}^{2,2} \end{bmatrix} + \begin{bmatrix} \mathbf{I}\lambda_1 & \mathbf{0} \\ \mathbf{0} & \mathbf{I}\lambda_2 \end{bmatrix}. \tag{3.49}$$

By having two regularisation parameters, the regularisation of noisy function values and gradients can be addressed separately. Figure 3.7 shows a one-dimensional fit of data containing noisy gradients, with and without regularisation. As with the non-gradient case, the regularisation parameters are determined through optimisation which is further discussed in Chapter 5.



(a) Without regularisation (b) With regularisation

Figure 3.7: One-dimensional fit of a function with noisy gradients.

3.4 Optimisation techniques

Once metamodels have been created, the optimisation problem can be solved approximately. As the optimisation is carried out using metamodels rather than expensive function evaluations, the optimisation problem can be solved much more

quickly. However, as the function evaluations are not free, particularly for the MLSM which requires a matrix decomposition for every evaluation, the choice of optimisation algorithm is still an important consideration.

This section discusses two gradient based optimisation algorithms, the method of feasible directions and sequential quadratic programming, which are both used in this work. Even though both are gradient based optimisation algorithms they have individual properties which makes them attractive for the tasks that they have been assigned to.

For problems with several local minima, it is common to either use global algorithms such as the genetic algorithm, or to use multiple start points of gradient based methods. The latter option has been used in this work.

3.4.1 The Karush-Kuhn-Tucker conditions

Before discussing the optimisation algorithms, the Karush-Kuhn-Tucker (KKT) conditions are presented as outlined in Haftka and Gurdal (1992). Consider a general constrained optimisation problem in the form

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimise}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_j(\mathbf{x}) \leq 0, \quad j = 1, \ldots, m \end{aligned} \tag{3.50}$$

where f_0 is the objective function to be minimised, and f_j is the *j-th* inequality constraint. Even though only upper bound constraints are present, this formulation allows lower bound constraints by a simple sign switch and equality constraints by a pair of inequality constraints. The Lagrangian function for an equality constrained optimisation problem can be written as

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f_0(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j h_j(\mathbf{x}), \tag{3.51}$$

where h_j are equality constraints and λ_j are Lagrangian multipliers. The inequality constraints in (3.50) are transformed to equality constraints as

$$h_j(\mathbf{x}) = f_j(\mathbf{x}) - v_j^2, \quad j = 1, \ldots, m. \tag{3.52}$$

where v_j are so called slack variables that define how close the constraint functions are to becoming critical. The Lagrangian function becomes

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f_0(\mathbf{x}) + \sum_{j=1}^{m} \lambda_j (f_j(\mathbf{x}) - v_j^2). \tag{3.53}$$

Differentiation with respect to $\mathbf{x}$, $\boldsymbol{\lambda}$, and $\mathbf{v}$ yields the necessary conditions for a stationary point

$$\begin{aligned} \frac{\partial \mathcal{L}}{\partial x_i} &= \frac{\partial f_0}{\partial x_i} - \sum_{j=1}^{m} \lambda_j \frac{\partial f_j}{\partial x_i} = 0, \quad && i = 1, \ldots, n \\ \frac{\partial \mathcal{L}}{\partial \lambda_j} &= f_j - v_j^2 = 0, && j = 1, \ldots, m \\ \frac{\partial \mathcal{L}}{\partial v_j} &= 2\lambda_j v_j = 0, && j = 1, \ldots, m \end{aligned} \tag{3.54}$$

which leads to the conclusion that the Lagrange multipliers are zero when their corresponding slack variables are non-zero, i.e. when the constraint is not critical.

Finally, the KKT conditions as proposed by Karush (1939) and Kuhn and Tucker (1951) state that $\mathbf{x}$ is a local minima if

$$-\frac{\partial f_0}{\partial x_i} = -\sum_{j=1}^{m} \lambda_j \frac{\partial f_j}{\partial x_i}, \quad \lambda_j > 0, \tag{3.55}$$

i.e. if the gradient of the objective function can be expressed as a linear combination of the gradients of the active, non-degenerate, constraints with $\lambda_j > 0$ as geometrically illustrated in Figure 3.8.

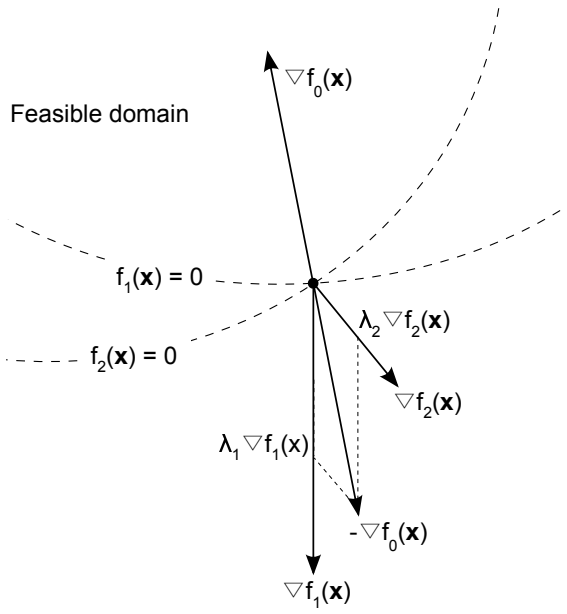


Figure 3.8: Geometrical representation of the KKT conditions.

3.4.2 Method of feasible directions

The method of feasible directions (MFD) solution proposed by Zoutendijk (1960) is presented here as described in Haftka and Gurdal (1992). The FORTRAN program known as CONMIN, developed by Vanderplaats (1973), is used in the research.

Let $\mathbf{x}^{(k)}$ denote the starting point for iteration k. The next point $\mathbf{x}^{(k+1)}$ is to be found by determining a search direction $\mathbf{s}$ and a step length α according to

$$\mathbf{x}^{k+1} = \mathbf{x}^k + \alpha \mathbf{s}. \tag{3.56}$$

In the MFD the search direction $\mathbf{s}$ is to be determined as a feasible direction

$$\mathbf{s}^T \nabla f_j(\mathbf{x}^k) \leq 0, \quad j = 1, \ldots, m, \tag{3.57}$$

and a direction in which the objection function descends

$$\mathbf{s}^T \nabla f_0(\mathbf{x}^k) \leq 0 \quad . \tag{3.58}$$

In practise this is achieved by solving a linear optimisation problem defined as

$$\begin{aligned} \underset{\beta,\mathbf{s}}{\text{minimise}} \quad & \beta \\ \text{subject to} \quad & \mathbf{s}^T \nabla f_0(\mathbf{x}^k) \leq \beta \\ & \mathbf{s}^T \nabla f_j(\mathbf{x}^k) \leq \theta_j \beta, \quad j = 1, \ldots, m \\ & -1 \leq s_i \leq 1, \qquad\quad i = 1, \ldots, n \end{aligned} \tag{3.59}$$

where $\boldsymbol{\theta}$ are so called *push-off* factors which determine the angle between the tangent of the constraint boundary and the search direction. If the resulting objective function is zero, $\beta = 0$, then the KKT conditions (3.55) are met. If $\beta \leq 0$ then $\mathbf{s}$ is a descending feasible direction. Once the search direction has been established, the step length is to be obtained by solving the line search

$$\begin{aligned} \underset{\alpha}{\text{minimise}} \quad & f_0(\mathbf{x}^k + \alpha \mathbf{s}) \\ \text{subject to} \quad & f_j(\mathbf{x}^k + \alpha \mathbf{s}) \leq 0, \quad j = 1, \ldots, m \end{aligned} \quad . \tag{3.60}$$

One of the main benefits of MFD is that it stays feasible throughout the optimisation.

3.4.3 Sequential quadratic programming

The sequential quadratic programming (SQP) solution procedure proposed by Powell (1978) is presented here as shown by Haftka and Gurdal (1992). The FORTRAN program known as MINCF, developed by Madsen *et al.* (2002), based on the work of Powell (1978).

In the SQP solution procedure, the search direction $\mathbf{s}$ is obtained by solving the quadratic sub-problem defined as

$$\begin{aligned} \underset{\mathbf{s}}{\text{minimise}} \quad & \mathbf{s}^T \nabla f_0(\mathbf{x}_k) + \frac{1}{2}\mathbf{s}^T \tilde{\boldsymbol{H}} \mathbf{s} \\ \text{subject to} \quad & f_j(\mathbf{x}_k) + \mathbf{s}^T \nabla f_j(\mathbf{x}_k) \geq 0, \quad j = 1, \ldots, m \end{aligned} \tag{3.61}$$

where $\tilde{\boldsymbol{H}}$ is an approximation of the Hessian of the Lagrangian function, which in the first iteration is initialised as the identify matrix and from then on is updated using the BFGS update (Broyden, 1970; Fletcher, 1970; Goldfarb, 1970; Shanno, 1970)

$$\tilde{\boldsymbol{H}}_{k+1} = \tilde{\boldsymbol{H}}_k - \frac{\tilde{\boldsymbol{H}}_k \Delta\mathbf{x} \Delta\mathbf{x}^T \tilde{\boldsymbol{H}}_k}{\Delta\mathbf{x}^T \tilde{\boldsymbol{H}}_k \Delta\mathbf{x}} + \frac{\Delta\mathbf{l}\Delta\mathbf{l}^T}{\Delta\mathbf{x}^T \nabla\mathbf{x}}, \tag{3.62}$$

where

$$\Delta\mathbf{x} = \mathbf{x}_{k+1} - \mathbf{x}_k \quad , \tag{3.63}$$

and

$$\Delta\mathbf{l} = \nabla\boldsymbol{\mathcal{L}}(\mathbf{x}_{k+1}, \boldsymbol{\lambda}_k) - \nabla\boldsymbol{\mathcal{L}}(\mathbf{x}_k, \boldsymbol{\lambda}_k). \tag{3.64}$$

If the condition

$$\Delta\mathbf{x}^T \Delta\mathbf{l} \leq 0.2 \Delta\mathbf{x}^T \tilde{\boldsymbol{H}}_k \Delta\mathbf{l}, \tag{3.65}$$

is met $\Delta\mathbf{l}$ is replaced by the following expression to ensure that the approximation of the Hessian is positive definite

$$\Delta\mathbf{l}' = \theta\Delta\mathbf{l} + (1-\theta)\tilde{\boldsymbol{H}}_k \Delta\mathbf{x}, \tag{3.66}$$

where

$$\theta = \frac{0.8\Delta\mathbf{x}^T \tilde{\boldsymbol{H}}_k \Delta\mathbf{x}}{\Delta\mathbf{x}^T \tilde{\boldsymbol{H}}_k \Delta\mathbf{x} - \Delta\mathbf{x}^T \Delta\mathbf{l}}. \tag{3.67}$$

Once the search direction has been established, the step length is to be calculated

using the line search

$$\underset{\alpha}{\text{minimize}} \quad f_0(\mathbf{x}_k + \alpha\mathbf{s}) + \sum_{j=1}^{m} \mu_j \left| min(0, f_j(\mathbf{x}_k + \alpha\mathbf{s}) \right| \quad , \tag{3.68}$$

where

$$\mu_j = max\left[\left|\lambda_j\right|, \frac{1}{2}\left(\mu_j^{i-1} + \left|\lambda_j^{i-1}\right|\right)\right]. \tag{3.69}$$

The process is repeated until the KKT conditions (3.55) are met.

The advantage of the SQP method is that it is associated with faster convergence than other gradient based optimisation methods Haftka and Gurdal (1992). However, unlike the MFD, it does not necessarily generate a sequence of feasible points throughout the optimisation.

3.5 Summary

Metamodels are used to replace computationally expensive simulations with mathematical models. Once the metamodels are built, they can be used to approximate the response of the expensive simulations to a very low computational cost. In order to use metamodels in optimisation, three things are needed: design of experiments, a metamodel technique and an optimiser, all of which are introduced in this chapter.

The choice of the DOE technique was chosen as modified extensible lattice sequences (MELS) which is an infinite quasi-random low discrepancy sequence. The method produces an infinitely extensible sequence of points. This is a desired feature for metamodel based optimisation since one seldom know the number of required points for desired metamodel accuracy *a priori*.

Two metamodel techniques, the moving least squares method and Kriging, were

introduced as they are used throughout this work. The moving least squares method is based on weighted regression, where the weights are (descending) functions of the distance from the evaluation point to the training points. Kriging is an interpolation technique based on spatial correlation of metamodel errors according to some continuous model, but can easily be allowed to deviate from the training points in order to smooth noisy functions. Both techniques can be modified to accommodate partial derivatives of the response functions with respect to the design variables in order to enhance the quality of the metamodels.

Two optimisation techniques, method of feasible directions and sequential quadratic programming, were covered in this chapter and are used throughout the research. Sequential quadratic programming is associated with faster convergence than other gradient based optimisation methods. However, the method of feasible directions is more likely to generate a sequence of feasible points throughout the optimisation.

Chapter 4

Mid-range approximation method

This chapter discusses the metamodel-based optimisation framework known as the mid-range approximation method (MAM). The MAM is an iterative optimization technique based on approximations built in trust regions. A trust region is a sub-domain of the design space in which a set of design points, treated as a small-scale DOE, are evaluated. These and a subset of previously evaluated design points are used to build approximations of the objective and constraint functions that are considered to be valid in the current trust region. The trust region will then translate and change size as the optimization progresses.

The chapter begins with an overview of the mid-range approximation method in Section 4.1. The following sections describe each part of the framework in more detail. Section 4.2 describes how DOE points are generated within the trust regions which are then used in Section 4.3 to construct metamodels. This is followed by Section 4.4 which presents the use of optimisation to obtain the current best point in each iteration. Section 4.5 outlines the trust-region strategy which represents the decision making part of the process. The chapter concludes with a summary in Section 4.6.

4.1 Introduction

The mid-range approximation method (Toropov, 1989, 1992; Toropov *et al.*, 1993), also known as the multi-point approximation method, solves a typical constrained optimisation problem in the form:

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & f_0(\mathbf{x}) \\ \text{subject to} \quad & f_j(\mathbf{x}) \leq 1, \qquad j = 1, \ldots, m \\ & A_i \leq x_i \leq B_i, \quad i = 1, \ldots, n \end{aligned} \tag{4.1}$$

where $f_0(\mathbf{x})$ is the objective function, $f_j(\mathbf{x})$ is the j-th constraint, $\mathbf{x}$ is the vector of design variables and A_i and B_i are the lower and upper bounds respectively for the design variable x_i. The optimisation problem (4.1) is replaced by a sequence of approximate sub-problems defined as

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \tilde{f}_0^{\,k}(\mathbf{x}) \\ \text{subject to} \quad & \tilde{f}_j^{\,k}(\mathbf{x}) \leq 1, \qquad j = 1, \ldots, m \\ & \left.\begin{aligned} & A_i^k \leq x_i \leq B_i^k \\ & A_i^k \geq A_i \\ & B_i^k \leq B_i \end{aligned}\right\} \quad i = 1, \ldots, n \end{aligned} \quad . \tag{4.2}$$

k denotes the current iteration number, $\tilde{f}_j^{\,k}(\mathbf{x})$ is a metamodel of $f_j^k(\mathbf{x})$, and A_i^k and B_i^k are the bounds of the current trust region where the sub-problem (4.2) is solved for the current iteration. The solution procedure for each sub-problem consists of sampling, creating metamodels, solving the optimisation problem and determining a new location and size of the trust region for the next iteration. The trust region will move and change size after each iteration, as illustrated in Figure 4.1, until the termination criteria are

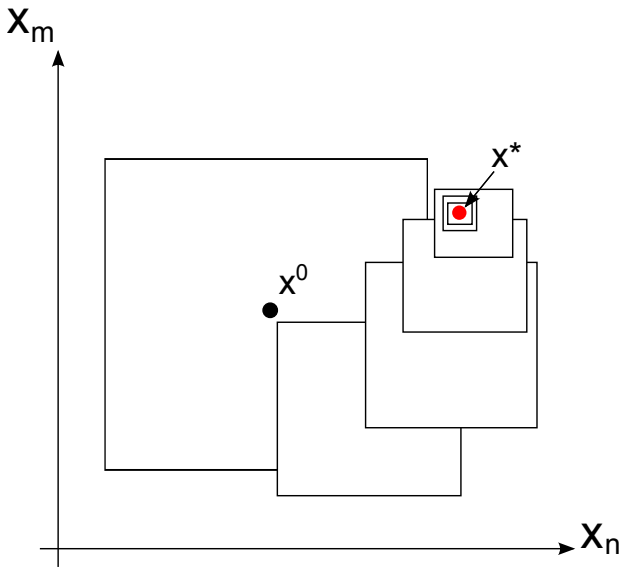


Figure 4.1: Typical history of the trust regions.

reached. The trust region strategy has gone through several developments to account for the presence of numerical noise in the response function values (van Keulen *et al.*, 1996; Toropov *et al.*, 1996) and occasional simulation failures (Toropov *et al.*, 1999).The flowchart in Figure 4.2 outlines the major steps of the optimisation process, some of which deserve extra attention and will be covered in the following sections.

4.2 DOE points

At the beginning of each iteration a small-scale design of experiments (DOE) is carried out. The DOE points are to be used as training points for building metamodels which are valid only in the current trust region and should therefore be placed in its vicinity. A flowchart of the process of obtaining the training points, and a table of corresponding parameters, are shown in Figure 4.3 and Table 4.1. The number of desired training points p_{opt} in each iteration is set by the user. However, the value cannot be less than the minimum number of training points required to build metamodels, p_{min}.

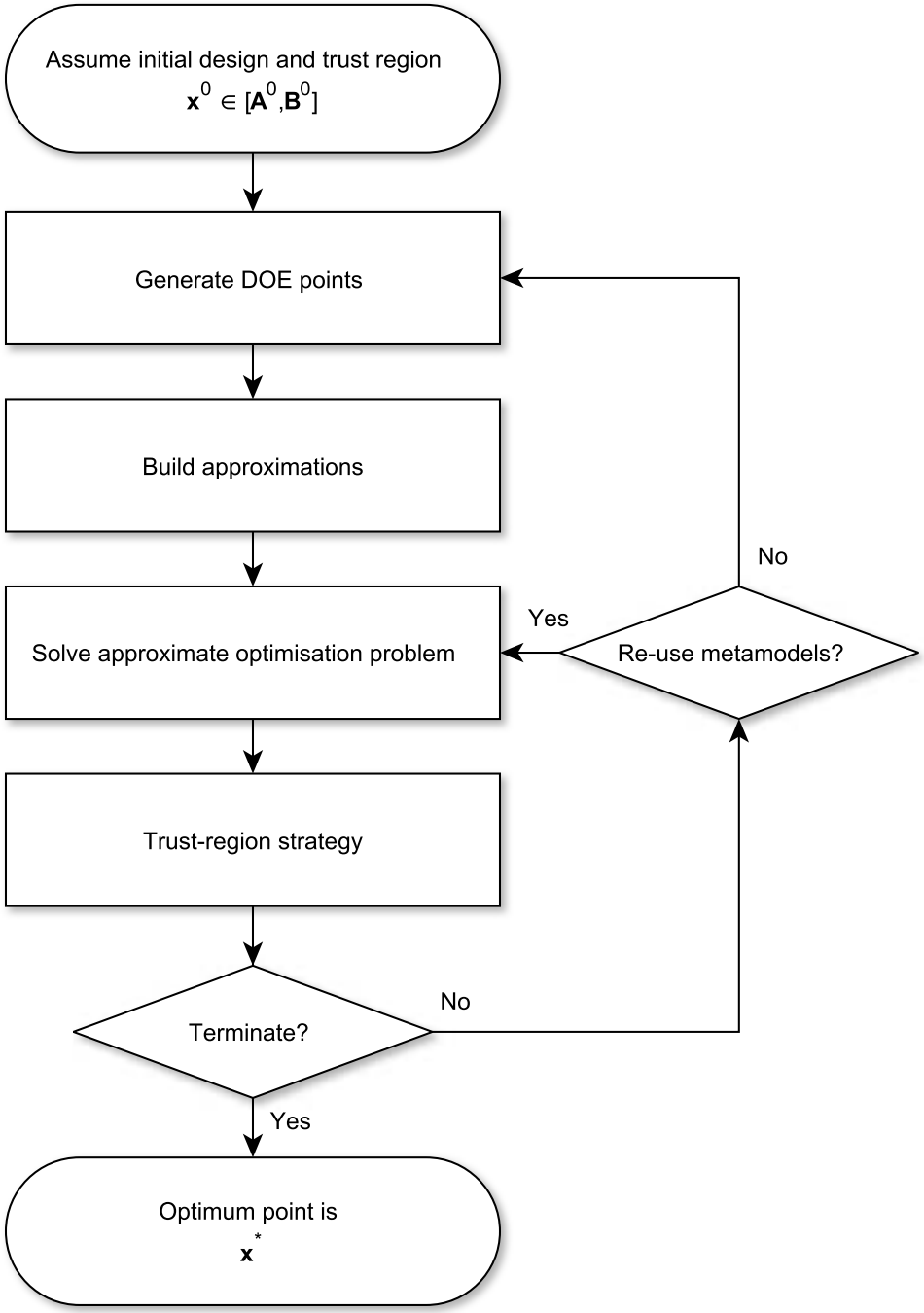


Figure 4.2: Flowchart of the MAM optimisation process.

Table 4.1: Settings of the MAM relating to design of experiments and corresponding default values.

Description	Variable	Default value
Ratio doe-region to trust-region	b_s	1.2
Ratio recycle-region to trust-region	b_r	1.9
Desired number of points per iteration	p_{opt}	$1.5n$
Minimum number of points per iteration	p_{min}	$1.9n$
Number of available processes	N_{ap}	p_{opt}

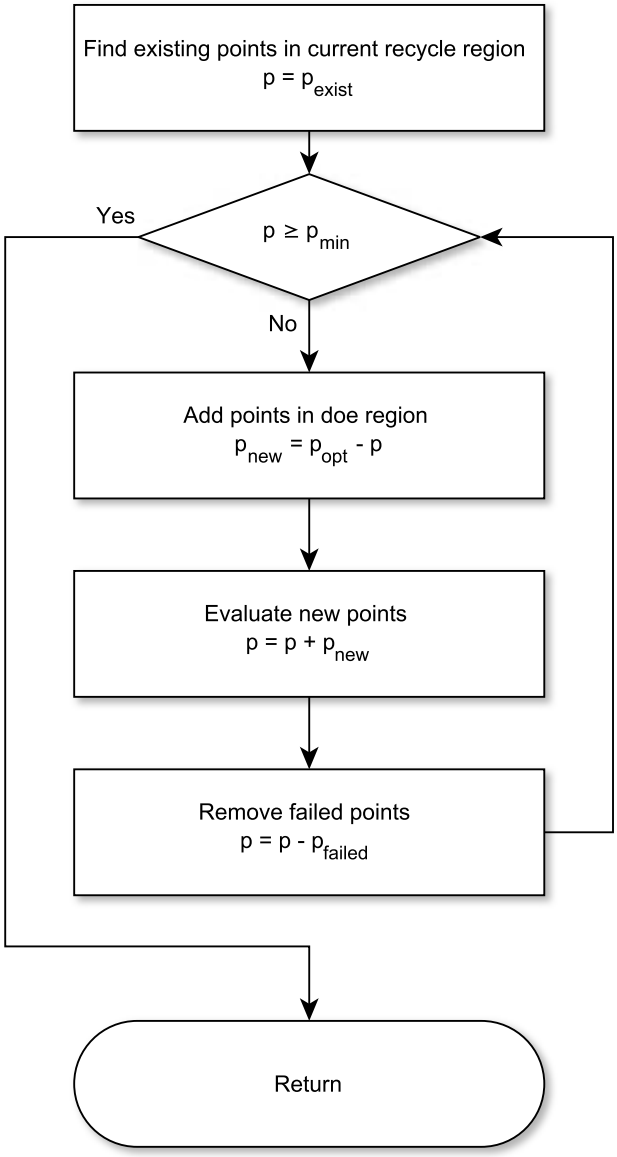


Figure 4.3: Flowchart of process for generation of DOE points.

4.2.1 Existing points

The process begins by checking if any existing points, evaluated in previous iterations, can be used in the current iteration. Points located outside the trust region may be used, however, points that are far away from the trust region may spoil the metamodel and are not considered. The region in which points are considered is referred to as the *recycle region*, shown in Figure 4.4, and is defined as an enlargement of the trust region by a factor b_r as

$$\bar{B}_i^k - \bar{A}_i^k = b_r \cdot \left(B_i^k - A_i^k \right), \quad b_r \geq 1, \tag{4.3}$$

where $\bar{B}_i^k$ and $\bar{A}_i^k$ denotes the upper and lower bounds, respectively, of the recycle region.

4.2.2 Supplementary points

If there are not enough existing points to build metamodels, new points are generated such that the total number of existing and new points is equal to the number of desired points p_{opt}. The new points are positioned using one of the available sampling techniques discussed in Section 3.2.

In order to fully utilise parallel hardware Korolev *et al.* (2015) introduced a number of available processes, N_{ap}, set by the user such that the number of DOE points will be forced to a multiple of the chosen value. It is desired to allow points to be located slightly outside the trust region in order to promote interpolation rather than extrapolation, but not as far outside as the bounds of the recycle region. Therefore another region, denoted as the *DOE region*, shown in Figure 4.4, is introduced as an enlargement of the trust region by a factor b_s as

$$\tilde{B}_i^k - \tilde{A}_i^k = b_s \cdot \left(B_i^k - A_i^k \right), \quad b_s \geq 1, \tag{4.4}$$

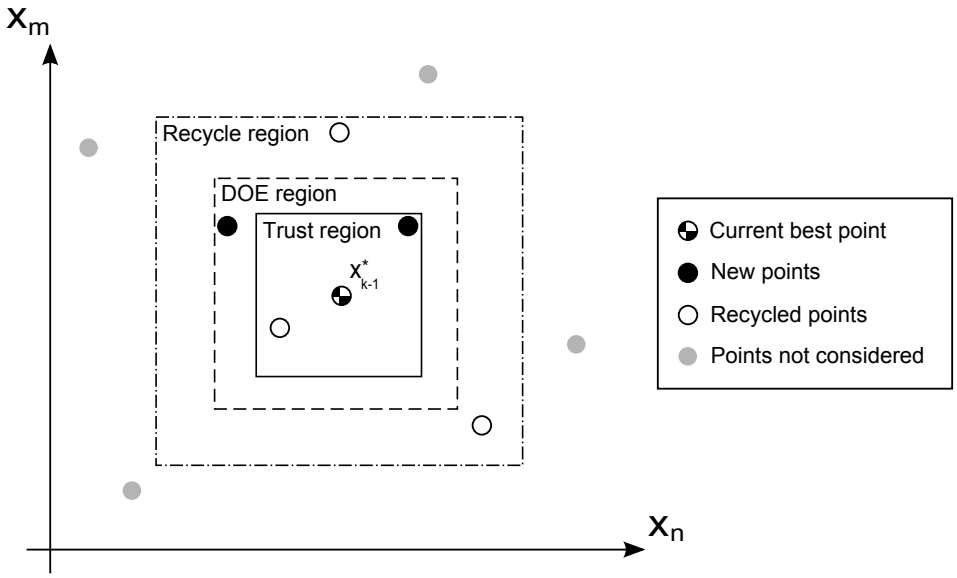


Figure 4.4: Geometrical representation of the trust, DOE, and recycle regions.

where $\tilde{B}_i^k$ and $\tilde{A}_i^k$ denotes the upper and lower bounds, respectively, of the DOE region. None of the regions are allowed outside the design variable upper and lower bounds B_i and A_i.

After sampling, the DOE points are evaluated. DOE points that lead to simulation failure are identified and removed as introduced by Toropov *et al.* (1999). If there is not a sufficient number of remaining DOE points to build metamodels, i.e. less than p_{min}, the process of adding points is repeated until at least p_{min} successful points have been obtained

4.3 Metamodels

Once a sufficient number of DOE points have been successfully evaluated metamodels are created for each response. Available metamodel techniques include metamodel

assemblies by Polynkin and Toropov (2012), where intrinsically linear and rational functions are assembled into a single metamodel using linear regression, the moving least squares method as outlined in Section 3.3.1, and kriging as described in Section 3.3.2. Gradient enhanced versions of all metamodel techniques are available.

4.4 Candidate points

The next step is to obtain candidate points that potentially can become the current best point. The process of obtaining candidate points is outlined in Figure 4.5 and the corresponding parameters are presented in Table 4.2. The number of candidate points to be obtained in each iteration is user defined and denoted by p_{cand}. To fully utilise available parallel hardware the number of candidate points will be set to N_{ap} or a multiple of the same.

The primary way of obtaining candidate points is by solving the approximate sub-problem (4.2) by using the SQP optimisation procedure, as outlined in Section 3.4.3. The SQP is started from several starting points in order to increase the chance of finding a good solution for problems with several local optima. The number of starting points is user-defined and denoted p_{sqp}. The SQP solutions are ranked and duplicate solution points are removed such that only p_{unique} unique points are left. The p_{cand} best ones are used as candidate points. If p_{unique} is less than p_{cand}, the remaining desired candidates are found using the chosen DOE technique. All candidate points are evaluated in parallel and any failed candidates are removed, but not replaced by new points. The best candidate point is compared to the current best point and, if superior, replaces the current best point.

Table 4.2: Settings of the MAM relating to design of experiments and corresponding default values.

Description	Variable	Default value
Number of SQP start points	p_{sqp}	20
Number of desired candidate points	p_{cand}	N_{ap}

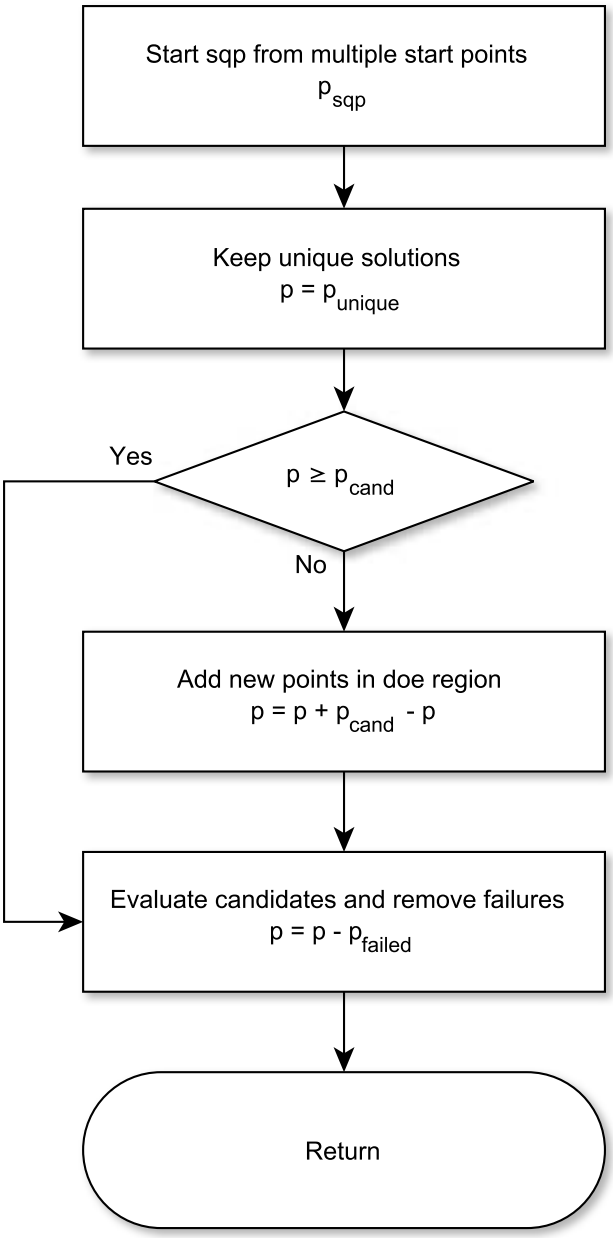


Figure 4.5: Flowchart outlining process of obtaining candidate points.

4.5 Trust region strategy

The trust region strategy defines the decision making process of the MAM. The convergence criteria are checked and adjustments are made to the trust region. The centre of the trust region will always be taken as the current best point x_k^*, however, the size of the trust region will be decided based on the trust region strategy outlined in Figure 4.6 and parameters presented in Table 4.3.

Table 4.3: Settings of the MAM relating to design of experiments and corresponding default values.

Description	Variable	Default value
Max. approximation error for sufficient quality	ϵ_{good}	5%
Max. approximation error for excellent quality	$\epsilon_{verygood}$	0.5%
Sufficiently small trust region size	r_{suff}	5%
Minimum trust region size	r_{min}	1%
Indicator for move angle (oscillations)	Θ_{min}	0
Indicator for move angle (same direction)	Θ_{max}	0.8
No. iterations considered for Θ_{max}	l	3
Trust region reduction factor	β_1	1/1.5
Trust region reduction factor	β_2	1/4.0
Trust region reduction factor	β_3	1/2.0
Trust region reduction factor	β_4	1/1.5
Trust region enlargement factor	β_5	1.25

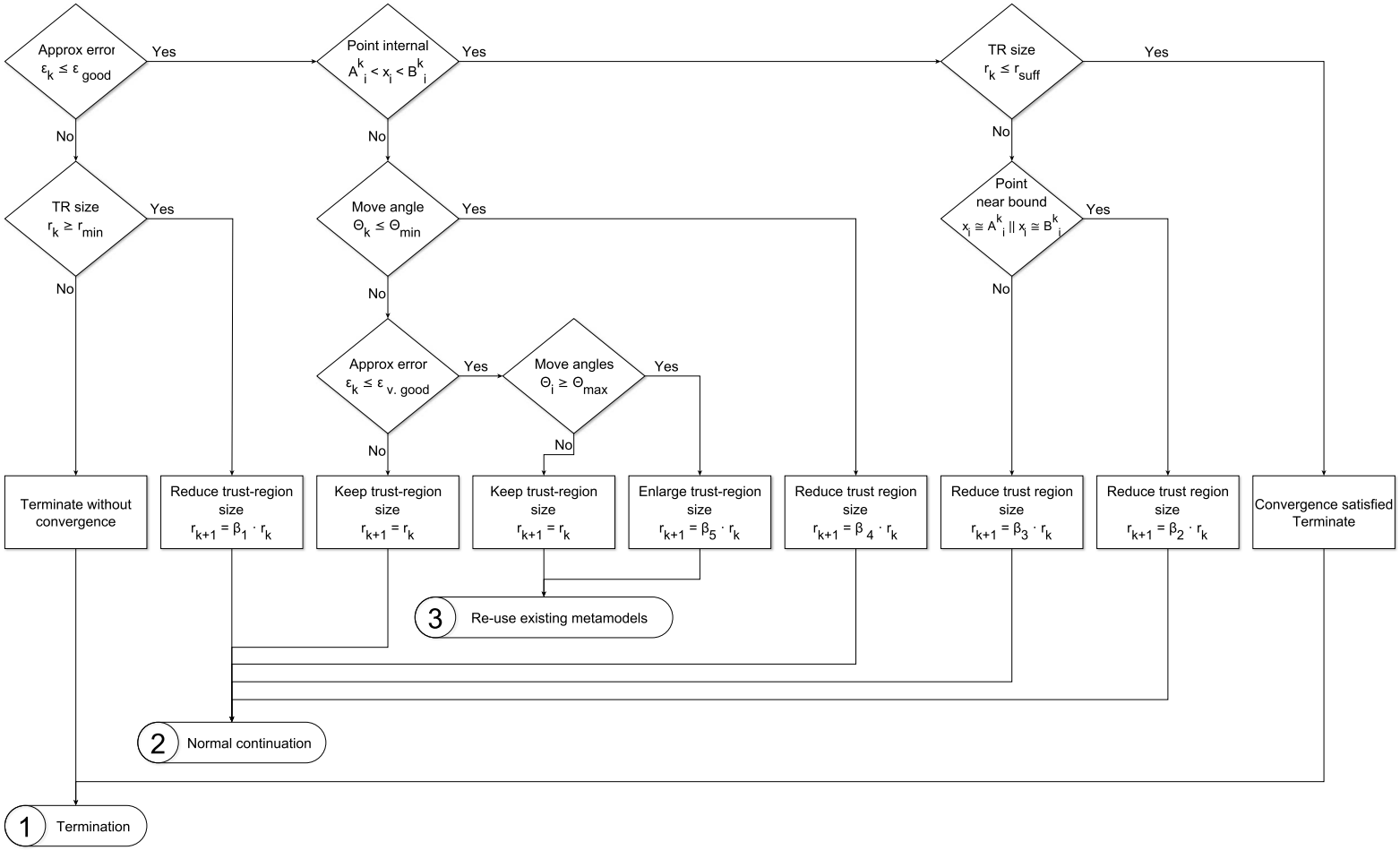


Figure 4.6: Flowchart describing the trust region strategy. The trust region strategy determines whether the process should terminate or continue, the size and location of the next trust region, and whether metmodels should be re-used or not. This is done based on the approximation error, location of current best point, move angles, and the size of the current trust region.

4.5.1 Metamodel quality

The discrepancy between the approximated function values and the function values of the candidate points are used to evaluate the metamodel quality for the current iteration. The metamodel error is calculated for each response as the root mean squared error (RMSE)

$$\epsilon_k^j = \sqrt{\frac{1}{n_k}\sum_{i=1}^{n_k}\left[\tilde{f}_j^{\,k}(\mathbf{x}_i) - f_j^k(\mathbf{x}_i)\right]^2}, \quad j = 0, .., m. \tag{4.5}$$

In order to partially satisfy the convergence criterion the metamodel error of any response may not exceed ϵ_{good}. If it does then the trust region strategy will commence to decrease the size of the trust region by a factor β_1 until the criterion is met. If the trust region size falls below r_{min} without passing the metamodel error criteria, the optimisation is terminated (exit point 1 in Figure 4.6) without convergence. This situation is extremely rare and typically indicates deficiency in the problem formulation.

4.5.2 Location of the current best point

If the metamodel quality criterion is satisfied, the location of the current best point x_k^* relative to the trust region is checked. If it is on a boundary of the trust region, it is an indication that the solution might be outside the trust region. The search is then continued by moving the trust region. In this situation it is necessary to check whether the solution is oscillating as this can result in endlessly moving between similar solutions without getting much closer to the stationary point. This is evaluated by calculating the angle between the last two move vectors, described by van Keulen *et al.* (1996), as

$$\Theta_k = \cos(\alpha_k) = \frac{(x_k^* - x_{k-1}^*)}{||x_k^* - x_{k-1}^*||} \cdot \frac{(x_{k-1}^* - x_{k-2}^*)}{||x_{k-1}^* - x_{k-2}^*||}. \tag{4.6}$$

If Θ_k is positive, the angle α_k shown in Figure 4.7 is acute, indicating that the optimisation is progressing somewhat in the same direction. If Θ_k is negative it may be an indication of oscillations.

If oscillations are identified, by Θ_k being less than the user defined parameter Θ_{min}, the trust region size is reduced by a factor β_2 and the process continues. If oscillations are not present and the metamodel quality is deemed very good, i.e. the metamodel error, calculated in (4.5), does not exceed the user defined value $\epsilon_{v.good}$ for any of the responses, the metamodels will be reused in the next iteration (exit point 3 in Figure 4.6), thus eliminating the need for additional simulations to be carried out. Otherwise the trust region size will be kept and the process continued as normal (exit point 2 in Figure 4.6). If, in addition to a very good metamodel quality, the optimisation has progressed in almost the same direction for the last l iterations, i.e.

$$\Theta_i \geq \Theta_{max}, \quad i = k + 1 - l, k, \tag{4.7}$$

the trust region is enlarged to promote faster convergence, otherwise the size of the trust region will be kept the same.

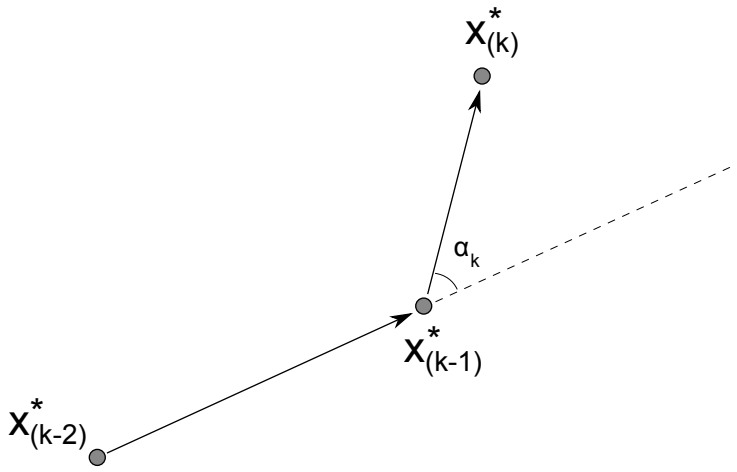


Figure 4.7: The angle between move vectors for iterations k-2, k-1 and k.

4.5.3 Trust region size

If, in addition to satisfying the two previous partial convergence criteria, the trust region size does not exceed r_{suff}, then the optimisation converges and terminates (exit point 1 in Figure 4.6). Otherwise the trust region size will be reduced. If the current best point is near a boundary,'the trust region will be reduced by the factor β_4, otherwise β_3 is used.

4.6 Summary

The MAM is an iterative optimization technique based on mid-range approximations built in trust regions. A trust region is a sub-domain of the design space in which a set of design points, treated as a small-scale DOE, are evaluated. These and a subset of previously evaluated design points are used to build approximations of the objective and constraint functions that are considered to be valid in the current trust region. The trust region will then translate and change size according to a trust region strategy as the optimization progresses. The trust region strategy has gone through several developments to account for the presence of numerical noise in the response function values, occasional simulation failures and, most recently, developments for deployment within high performance computing facilities.

In this work the mid-range approximations used in the trust regions are either the the moving least squares method as outlined in Section 3.3.1 or kriging as described in Section 3.3.2. The approximated optimisation problem is solved using the SQP optimisation procedure, as outlined in Section 3.4.3.

Chapter 5

Parameter tuning for well conditioned kriging metamodels

This chapter discusses the efficiency of building of kriging metamodels, as introduced in Section 3.3.2. One of the main challenges of kriging, and gradient enhanced kriging in particular, is the computational cost associated with the parameter tuning, necessary for building the metamodel. In this chapter a novel method for efficient parameter tuning is presented.

5.1 Introduction

In kriging, parameter tuning requires optimisation of a condensed log likelihood function with respect to a set of hyper parameters, one for each design variable. Every evaluation of the condensed log likelihood function requires decomposition of a square correlation matrix, $\mathbf{R} \in \mathbb{R}^{d \times d}$. For kriging $d = p$, where p is the number of training points, and for gradient enhanced kriging $d = p \times (n + 1)$ where n is the number of design variables.

For problems with a small number of design variables, gradient based algorithms such as sequential quadratic programming have shown good performance (Zimmermann, 2013; Lockwood and Anitescu, 2010). To increase the probability of finding a better solution for problems with several optima, multiple start-points have also been proposed, with as few as five points (Lockwood and Anitescu, 2010) or as many ten times the number of hyper parameters (Liu and Batill, 2002). For larger problems the optimisation is often carried out using global algorithms such as simulated annealing (Xiong *et al.*, 2007) and genetic algorithm (Forrester *et al.*, 2008). Toal *et al.* (2011) proposed a Hybrid optimisation scheme where promising points from a particle swarm optimisation were used as starting points for gradient based optimisations using sequential quadratic programming. In the same paper it was also shown how the adjoint method can be used to obtain partial derivatives of the condensed log likelihood function with respect to the correlation matrix, which greatly reduces the computational effort required when compared to finite differences and the direct method.

Ill-conditioning of the correlation matrix can become an issue when building metamodels where training points are located near each other (Haaland *et al.*, 2011), especially for Gaussian correlation matrices (Zimmermann, 2015). Attempts have been made to reduce ill conditioning by, for instance, using uniform subsets of the training points Rennen (2008), adding regularisation terms along the diagonal of the correlation matrix which makes the kriging metamodel approximate rather than interpolate the data, and constraining the condition number explicitly during optimisation Dalbey (2013).

In this work partial derivatives of the condition number of the correlation matrix with respect to the hyper parameters are obtained, making it possible to constrain the condition number directly in a gradient based optimisation approach. A two-step approach is suggested for optimisation of the hyper parameters. In the first step,

the optimisation problem is considered as a single variable problem by treating all hyper parameters as one variable. The solution to this problem is then used as a starting point for a gradient based optimisation algorithm. In both cases an upper bound constraint is enforced on the condition number of the correlation matrix. The approach is tested on several analytical examples using two types of gradient based optimisation algorithms, the sequential quadratic programming and the method of feasible directions. The approach is compared to gradient based optimisations starting from random points, multiple starting points and, a genetic algorithm followed by gradient based optimisations. Finally a case study is presented where the responses of an aircraft wing-box with 126 design variables is approximated using the suggested approach and compared to a selection of optimisation methods.

5.2 Parameter tuning

To obtain a good kriging fit it is important to determine suitable values of the hyper parameters, $\boldsymbol{\theta}$, and regularisation parameters, $\boldsymbol{\lambda}$, as introduced in Section 3.3.2. Failing to do so may result in a sub-standard fit. Figure 5.1 shows an example of (a) an overestimated hyper parameter and (b) an optimised hyper parameter.

MLE would accomplish this by taking the mean and variance as parameters and finding particular parametric values that make the observed results the most probable given the model.

The hyper parameters and regularisation parameters are determined such that they make the observed results, the response values at the training points, the most probable for the model. This is usually referred to as a maximum likelihood estimator (MLE). The MLE for kriging given a Gaussian distribution is determined through maximisation

of the condensed log likelihood function (Jones, 2001).

$$\phi(\boldsymbol{\theta}, \boldsymbol{\lambda}) = -\frac{p}{2}\ln(\hat{\sigma}^2(\boldsymbol{\theta}, \boldsymbol{\lambda})) - \frac{1}{2}\ln(|\mathbf{R}(\boldsymbol{\theta}, \boldsymbol{\lambda})|), \tag{5.1}$$

and for the gradient enhanced case (Han *et al.*, 2013)

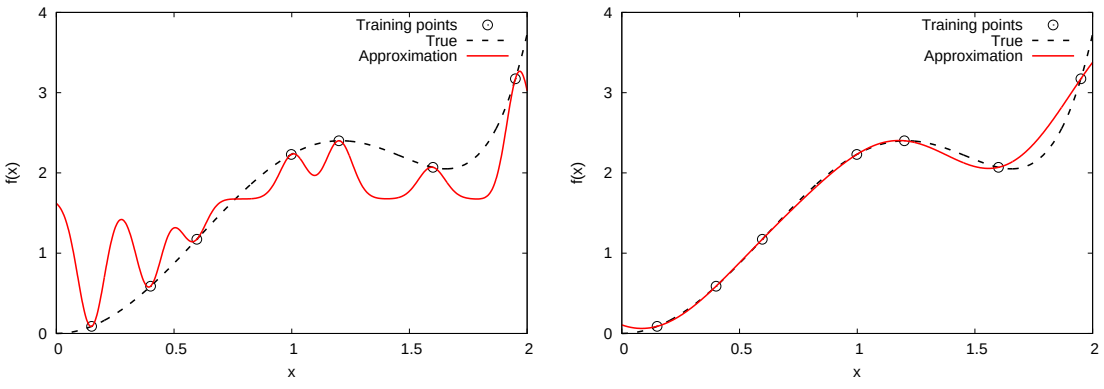
$$\phi(\boldsymbol{\theta}, \boldsymbol{\lambda}) = -\frac{p(n+1)}{2}\ln(\hat{\sigma}^2(\boldsymbol{\theta}, \boldsymbol{\lambda})) - \frac{1}{2}\ln(|\mathbf{R}(\boldsymbol{\theta}, \boldsymbol{\lambda})|), \tag{5.2}$$

where $|\mathbf{R}(\boldsymbol{\theta}, \boldsymbol{\lambda})|$ denotes the determinant of the correlation matrix. To prevent ill conditioning of the correlation matrix the condition number is constrained to be lower than some threshold during optimisation. The condition number is obtained as

$$k(\mathbf{R}) = \|\mathbf{R}^{-1}\|\|\mathbf{R}\|, \tag{5.3}$$

where $\|\mathbf{R}\|$ denotes the norm of the correlation matrix which is here calculated as the Frobenius norm

$$\|\mathbf{R}\| = \|\mathbf{R}\|_F = \sqrt{\sum_i \sum_j R_{i,j}^2}, \tag{5.4}$$



(a) Overestimated hyper parameter θ

(b) Optimised hyper parameter θ

Figure 5.1: Importance of parameter optimisation.

and $\|\mathbf{R}^{-1}\|$ denotes the norm of the inverse correlation matrix which is calculated using the matrix inversion (*DPOTRI*) routine from the Intel Math Kernel Library 11.2 (Intel, 2015) using the matrix decomposition previously obtained for Kriging. Formally, the tuning parameter optimisation problem takes the form

$$\begin{aligned} \underset{\boldsymbol{\theta},\boldsymbol{\lambda}}{\text{maximise}} \quad & \phi(\boldsymbol{\theta},\boldsymbol{\lambda}) \\ \text{subject to} \quad & \kappa(\boldsymbol{\theta},\boldsymbol{\lambda}) \leq \kappa_{max} \end{aligned} \tag{5.5}$$

where κ_{max} is the upper bound constraint on the condition number. Here, $\kappa_{max} = 10^7$ is used.

Because of the computational expense related to tuning parameter optimisation this work is concerned with developing a hyper parameter optimisation approach which is efficient in terms of computational performance. This is done using gradient based optimisation techniques. In the following section it is shown how to obtain the gradients of the condensed log likelihood function and of the condition number with respect to the hyper parameters and regularisation parameters. These are then used for a hyper parameter optimisation approach outlined in the subsequent section.

5.3 Obtaining gradients

This section describes how the gradients of the condensed log likelihood function and the condition number with respect to the hyper and regularisation parameters are obtained in a computationally efficient manner. These can be obtained using different methods, with varying associated computational cost, depending on the problem at hand. For a large number of design variables, it may be prohibitively expensive to use finite differences or the direct method as the cost is proportional to the number of design variables.

The computational cost of the adjoint method is proportional to the number of response functions, which in this case are two; the condensed likelihood function and the condition number.

Using the chain rule the gradients of the condensed likelihood function with respect to the hyper parameters can be written as

$$\frac{\partial \phi}{\partial \boldsymbol{\theta}} = \sum_{i=1}^{p} \sum_{j=1}^{p} \frac{\partial \phi}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{\theta}} \quad , \tag{5.6}$$

and with respect to the regularisation parameters

$$\frac{\partial \phi}{\partial \boldsymbol{\lambda}} = \sum_{i=1}^{p} \sum_{j=1}^{p} \frac{\partial \phi}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{\lambda}} \quad . \tag{5.7}$$

Similarly the gradients of the condition number with respect to the hyper parameters can be written as

$$\frac{\partial \kappa}{\partial \boldsymbol{\theta}} = \sum_{i=1}^{p} \sum_{j=1}^{p} \frac{\partial \kappa}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{\theta}} \quad , \tag{5.8}$$

and with respect to the regularisation parameters

$$\frac{\partial \kappa}{\partial \boldsymbol{\lambda}} = \sum_{i=1}^{p} \sum_{j=1}^{p} \frac{\partial \kappa}{\partial R_{ij}} \frac{\partial R_{ij}}{\partial \boldsymbol{\lambda}} \quad . \tag{5.9}$$

In total there are four types of derivatives to establish. The gradients of the condensed likelihood function with respect to the correlation matrix, the gradients of the condition number with respect to the correlation matrix, and the gradients of the correlation matrix with respect to the hyper parameters and regularisation parameters. These are discussed in the following sections.

5.3.1 Gradients of the condensed likelihood function w.r.t. the correlation matrix

The partial derivatives of the condensed likelihood function with respect to the correlation matrix can be obtained using the adjoint method as shown by Toal *et al.* (2011) according to

$$\frac{\partial \phi}{\partial \mathbf{R}} = \bar{\mathbf{R}} = \frac{1}{2\hat{\sigma}^2}\mathbf{R}^{-T}\left(\mathbf{y} - \mathbf{F}\hat{\mu}\right)\left(\mathbf{y} - \mathbf{F}\hat{\mu}\right)^T\mathbf{R}^{-T} - \frac{1}{2}\mathbf{R}^{-T}, \tag{5.10}$$

where $\bar{\mathbf{R}}$ is the adjoint of the correlation matrix. This is applicable to both the non-gradient and gradient-enhanced case.

5.3.2 Gradients of the condition number w.r.t. the correlation matrix

The adjoint method can also be used for obtaining gradients of the condition number with respect to the correlation matrix as described in Ollar *et al.* (2016b). Using the chain rule and recalling (5.3) the derivatives of the condition number with respect to the correlation matrix can be written as

$$\frac{\partial \kappa}{\partial \mathbf{R}} = \frac{\partial \|\mathbf{R}^{-1}\| \|\mathbf{R}\|}{\partial \mathbf{R}} = \frac{\partial \|\mathbf{R}\|}{\partial \mathbf{R}}\|\mathbf{R}^{-1}\| + \frac{\partial \|\mathbf{R}^{-1}\|}{\partial \mathbf{R}}\|\mathbf{R}\|. \tag{5.11}$$

With this result the intermediate variables for reversed differentiation of the condition number with respect to the correlation matrix can be determined. The intermediate variable for the first term can, given that the intermediate variable for the condition

number itself has been initialised to $\bar{\kappa} = 1$, be written as

$$\overline{\|\mathbf{R}\|} = \bar{\kappa}\|\mathbf{R}^{-1}\| = \|\mathbf{R}^{-1}\|. \tag{5.12}$$

Using the results presented by Giles (2008) which are based on the work of Dwyer and MacPhail (1948) the adjoint of the Frobenius norm can be determined according to

$$\overline{\mathbf{R}} = \overline{\|\mathbf{R}\|}\frac{1}{\|\mathbf{R}\|}\mathbf{R}, \tag{5.13}$$

which together with (5.12) leads to the adjoint of the correlation matrix for the first term in (5.11)

$$\overline{\mathbf{R}}_1 = \frac{\|\mathbf{R}^{-1}\|}{\|\mathbf{R}\|}\mathbf{R}. \tag{5.14}$$

In the second term the intermediate variable from the product rule can be obtained as

$$\overline{\|\mathbf{R}^{-1}\|} = \bar{\kappa}\|\mathbf{R}\| = \|\mathbf{R}\|. \tag{5.15}$$

Again, using the adjoint of the Frobenius norm leads to

$$\overline{\mathbf{R}^{-1}} = \overline{\|\mathbf{R}^{-1}\|}\frac{1}{\|\mathbf{R}^{-1}\|}\mathbf{R}^{-1}. \tag{5.16}$$

Giles (2008) also presents the adjoint of the inverse as

$$\overline{\mathbf{R}} = -\mathbf{R}^{-T}\overline{\mathbf{R}^{-1}}\mathbf{R}^{-T}, \tag{5.17}$$

which together with (5.16) and (5.15) leads to the adjoint of the correlation matrix for the second term in (5.11)

$$\overline{\mathbf{R}}_2 = -\mathbf{R}^{-T}\frac{\|\mathbf{R}\|}{\|\mathbf{R}^{-1}\|}\mathbf{R}^{-1}\mathbf{R}^{-T}. \tag{5.18}$$

Adding (5.14) and (5.18) yields the gradients of the condition number with respect to the hyper parameters as

$$\frac{\partial \kappa}{\partial \mathbf{R}} = \overline{\mathbf{R}}_1 + \overline{\mathbf{R}}_2 = \frac{\|\mathbf{R}^{-1}\|}{\|\mathbf{R}\|}\mathbf{R} - \frac{|\mathbf{R}\|}{\|\mathbf{R}^{-1}\|}\left(\mathbf{R}^{-T}\ \mathbf{R}^{-1}\mathbf{R}^{-T}\right), \tag{5.19}$$

which is applicable both for the non-gradient and gradient-enhanced case.

5.3.3 Gradients of the correlation matrix w.r.t. regularisation parameters

The gradients of the correlation matrix with respect to the regularisation parameters can easily be obtained from (3.48) for the non-gradient case as

$$\frac{\partial R}{\partial \lambda} = \mathbf{I}_p, \tag{5.20}$$

where $\mathbf{I}_p \in \mathbb{R}^{p \times p}$, is the identity matrix with the number of diagonal elements of p. For the gradient-enhanced case from (3.49), for the first regularisation parameter as

$$\frac{\partial R}{\partial \lambda_1} = \begin{bmatrix} \mathbf{I}_p & \mathbf{0} \\ \mathbf{0} & \mathbf{0} \end{bmatrix}, \tag{5.21}$$

and for the second regularisation parameter as

$$\frac{\partial R}{\partial \lambda_2} = \begin{bmatrix} \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{I}_d \end{bmatrix}, \tag{5.22}$$

where $d = p \times n$.

5.3.4 Gradients of the correlation matrix w.r.t. the hyper parameters

For the non-gradient case, the partial derivatives of the correlation matrix with respect to the hyper parameters can be calculated as

$$\frac{\partial R_{i,j}}{\partial \theta_m} = -(x_i^m - x_j^m)^2 R_{i,j}. \tag{5.23}$$

For the gradient enhanced case they can be calculated as

$$\frac{\partial \mathbf{R}}{\partial \theta_m} = \begin{bmatrix} \frac{\partial \mathbf{Q}^{1,1}}{\partial \theta_m} & \frac{\partial \mathbf{Q}^{1,2}}{\partial \theta_m} \\ \frac{\partial (\mathbf{Q}^{1,2})^T}{\partial \theta_m} & \frac{\partial \mathbf{Q}^{2,2}}{\partial \theta_m} \end{bmatrix}, \tag{5.24}$$

where the first quadrant can be calculated according to (5.23) as

$$\frac{\partial \mathbf{Q}^{1,1}}{\partial \theta_m} = -(x_i^m - x_j^m)^2 R_{i,j}, \tag{5.25}$$

and, through the derivation shown in Appendix A, to the following expression for the upper right quadrant

$$\frac{\partial Q_{i,jk}^{1,2}}{\partial \theta_m} = \begin{cases} \left[\frac{1}{\theta_m} - (x_i^m - x_j^m)^2 \right] Q_{i,jk}^{2,1} & , m = k \\ -(x_i^m - x_j^m)^2 Q_{i,jk}^{1,2} & , m \neq k \end{cases}, \tag{5.26}$$

and the lower right quadrant

$$\frac{\partial Q^{2,2}_{il,jk}}{\partial \theta_m} = \begin{cases} -(x_i^m - x_j^m)^2 Q^{2,2}_{il,jk} & , m \neq k, m \neq l \\ \left[\frac{1}{\theta_k} - (x_i^m - x_j^m)^2\right] Q^{2,2}_{il,jk} & , m = k, m \neq l \\ \left[\frac{1}{\theta_l} - (x_i^m - x_j^m)^2\right] Q^{2,2}_{il,jk} & , m \neq k, m = l \\ \left[2 - 8\theta_k (x_i^m - x_j^m)^2\right] R_{i,j} - (x_i^m - x_j^m)^2 Q^{2,2}_{il,jk} & , m = k, m = l \end{cases} \tag{5.27}$$

for $i = 1, ..., p$, $j = 1, ..., p$, $k = 1, ..., n$ and $l = 1, ..., n$.

5.4 Computational performance

In order to get an idea of the computational cost of obtaining the function values and the partial derivatives of the parameter tuning problem a benchmark example was carried out. The benchmark was carried out using a 76 design variable analytical function with 100 training points. The computational cost of the various routines for the gradient enhanced case are outlined in Table 5.1. It is shown that the cost of calculating the condensed log likelihood function value and the condition number of the correlation matrix adds up to 4.8 seconds while their partial derivatives with respect to the hyper parameters and regularisation parameters takes 45 seconds. This means that for this particular case the cost of the partial derivatives are 9.4 times more expensive than the function values themselves. This is of course less costly than obtaining the gradients through the direct method or finite differences which would incurr a computational cost of around 76 (the number of design variables) times the cost of performing a function evaluation.

Table 5.1: Computational cost for evaluation of the various variables in kriging for a test problem with 76 design variables and 100 training points.[1]

Variables	Description	Time [s]
R,B,f	Pre-processing	0.3
$\mathbf{L}\mathbf{L}^T$	Cholesky decomposition	1.7
$\lvert\mathbf{R}\rvert$	Determinant	<0.1
$\hat{\mu}$	System mean	<0.1
$\hat{\sigma}^2$	System variance	<0.1
ϕ	Condensed likelihood function	<0.1
$\mathbf{R}^{-1}$	Inverse of $\mathbf{R}$	4.7
κ	Condition number of $\mathbf{R}$	0.1
		4.8
$\partial\phi/\partial\mathbf{R}$	Partial derivatives of ϕ w.r.t. $\mathbf{R}$	0.7
$\partial\kappa/\partial\mathbf{R}$	Partial derivatives of κ w.r.t. $\mathbf{R}$	20.8
$\partial\mathbf{R}/\partial\boldsymbol{\theta}$	Partial derivatives of $\mathbf{R}$ w.r.t. $\boldsymbol{\theta}$	23.9
$\partial\mathbf{R}/\partial\boldsymbol{\lambda}$	Partial derivatives of $\mathbf{R}$ w.r.t. $\boldsymbol{\lambda}$	<0.1
		45.3

[1] *The study was carried out on a computer with the following specifications: Intel(R) Core(TM) i7-4800MQ CPU @ 2.70GHz, and using Intel Math Kernel Library 11.2 (Intel, 2015) for matrix multiplication (DGEMM, DSYRK, DSYMM), Cholesky decomposition (DPOTRF) and backsubstitution (DPOTRS), matrix inverse (DPOTRI), norm (DLANGE) and vector multiplications (DGEMV).*

5.5 Proposed optimisation approach

In the proposed approach the aim is to use the gradients of the log likelihood function and the condition number with respect to the tuning parameters in gradient based tuning parameter optimisation. In this section a two step approach to tuning parameter optimisation is presented. The first step is finding a suitable starting point for the gradient based optimisation using a simplification of the optimisation problem to a one dimensional problem, and the second step is the gradient based optimisation.

5.5.1 Finding a suitable starting point

As seen in, for instance, (Chung and Alonso, 2002) it is possible to reduce the complexity of the parameter optimisation problem by considering the set of hyper parameters as a single variable according to

$$\boldsymbol{\theta} = [\theta_1, ..., \theta_n] = \gamma\,[1, ..., 1]\,, \tag{5.28}$$

where γ is the single considered variable. This is more commonly known as a radial basis function (RBF). The resulting, reduced, optimisation problem can be solved using a one dimensional line search, in this case a golden search (GS). This greatly reduces the computational cost of the optimisation problem but also limits the optimisation to find a solution on the hyper-diagonal of the design space. Here, instead of accepting the resulting point as the final solution, it is used as a starting point for optimisation in full space. Any regularisation parameters are set to zero during this stage of the optimisation approach. The reduced optimisation problem is defined as

$$\begin{aligned} \underset{\gamma}{\text{maximise}} \quad & \phi(\boldsymbol{\theta}, \boldsymbol{\lambda}) \\ \text{subject to} \quad & \kappa(\boldsymbol{\theta}, \boldsymbol{\lambda}) \leq \kappa_{max} \\ & \boldsymbol{\theta} = \gamma\,[1, ..., 1] \\ & 0 < \gamma \leq \gamma_{max} \\ & \boldsymbol{\lambda} = \mathbf{0} \end{aligned} \tag{5.29}$$

where κ_{max} is the upper bound constraint on the condition number, chosen as a user input, and γ_{max} is the upper bound of the single hyper parameter, chosen such that all off diagonal elements of the correlation matrix can become sufficiently small, i.e. such that $min(\mathbf{R}_{ij}) = R_{min}, i \neq j$, where R_{min} is a user input. In this work $R_{min} = 10^{-6}$.

5.5.2 Gradient based optimisation

After a starting point has been found through the golden search a gradient based method is to be used in order to explore the full hyper parameter and regularisation parameter space. Two gradient based optimisation methods are considered, the method of feasible directions (MFD) developed by Vanderplaats (1973) based on the work of Zoutendijk (1960) and sequential quadratic programming (SQP) developed by Madsen *et al.* (2002) based on the work of Powell (1978).

To ensure a well conditioned correlation matrix at the solution, the condition number is constrained throughout the optimisation. This is enabled through using the gradients of the condition number with respect to the hyper and regularisation parameters as outlined in Section 5.3.

5.6 Comparative study of optimisation approaches

The proposed approach is here compared to a selection of optimisation approaches, listed in Table 5.2. These approaches include sequential quadratic programming (SQP) and

Table 5.2: Considered optimisation methods and corresponding abbreviations

Abbreviation	Optimisation method
GS	Golden search
R-MFD	Random start MFD
R-SQP	Random start SQP
GS-MFD	MFD starting from GS result
GS-SQP	SQP starting from GS result
M-MFD	Multi-start MFD
M-SQP	Multi-start SQP
GA	Genetic algorithm
GA-MFD	MFD starting from GA result
GA-SQP	SQP starting from GA result

method of feasible directions (MFD) from one random (R-) start point, 10 multi start-points (M-) and the proposed method whereby the start-point is found by a golden search (GS-). Furthermore a genetic algorithm (GA) with 5000 evaluations and MDF and SQP starting from the resulting GA solution is included in the study. The study consists of two parts. The first one is carried out on two dimensional functions and the second on a dimensionally scalable problem.

5.6.1 Two dimensional benchmark study

This study compares the performance of the optimisation approaches on a suite of two dimensional analytical functions. The functions used in the case study, selected from those presented in Jamil and Yang (2013), are presented in Table 5.3. In order to reduce the risk of sporadic solutions 50 design of experiments (DOEs) were generated using different seed. Each of these were used in the optimisation of the tuning parameters for the metamodel.

Table 5.3: Two dimensional benchmark functions

Function name	Equation
Six-hump	$f(\mathbf{x}) = (4 - 2.1x_1^2 + \frac{x_1^4}{3})x_1^2 + x_1x_2 + (4x_2^2 - 4)x_2^2$
Branin-Hoo	$f(\mathbf{x}) = (x_2 - \frac{5.1x_1^2}{4\pi^2} + \frac{5x_1}{\pi} - 6)^2 + 10(1 - \frac{1}{8\pi})cos(x_1) + 10$
Himmelblau	$f(\mathbf{x}) = (x_1^2 + x_2 - 11)^2 + (x_1 + x_2^2 - 7)^2$
Ursem	$f(\mathbf{x}) = -sin(2x_1 - 0.5\pi) - 3cos(x_2) - 0.5x_1$
Adjiman	$f(\mathbf{x}) = cos(x_1)sin(x_2) - \frac{x_1}{x_2^2+1}$
Keane	$f(\mathbf{x}) = \frac{sin^2(x_1-x_2)sin^2(x1+x2)}{\sqrt{x_1^2+x_2^2}}$

The mean time spent, the mean resulting condensed log likelihood and generalisation error of the metamodels, for each of the optimisation approaches, is shown in Table 5.4. The GA, GA-MFD and GA-SQP provide the highest condensed log likelihood values. However, these methods take the longest of the tested methods as the number of evaluations carried out by GA within the 2D design variable space is exhaustive. The GS, R-MFD and R-SQP provide the worst results. It is possible to increase the likelihood

Table 5.4: Condensed log likelihood of gradient enhanced kriging metamodel built using 12 training points averaged over 50 training DoEs.

	Six Hump			Branin Hoo			Himmelblau		
	Time (ms)	Mean ϕ	RMSE %	Time (ms)	Mean ϕ	RMSE %	Time (ms)	Mean ϕ	RMSE %
GS	4.8	-41.05	12.89%	6.2	-33.21	12.78%	3.8	-27.29	3.69%
R-MFD	3.5	-35.20	10.90%	3.7	-25.13	10.46%	3.6	-29.68	5.23%
R-SQP	34.2	-34.49	10.78%	28.6	-25.18	10.37%	20.2	-32.38	6.26%
GS-MFD	7.5	-33.04	9.86%	9.9	-23.39	9.57%	7.8	-26.86	3.72%
GS-SQP	12.1	-33.38	10.07%	12.4	-23.56	9.67%	8.6	-26.86	3.71%
M-MFD	19.7	-33.27	9.81%	15.5	-23.36	9.69%	15.4	-25.88	2.94%
M-SQP	373.8	-33.04	9.91%	253.9	-23.24	9.64%	156.6	-26.48	3.10%
GA	376.4	-33.04	9.92%	376.3	-23.24	9.64%	376.2	-25.42	2.50%
GA-MFD	379.9	-33.03	9.92%	380.0	-23.24	9.64%	379.1	-25.39	2.49%
GA-SQP	383.3	-33.03	9.92%	382.6	-23.24	9.64%	382.2	-25.39	2.49%

	Ursem			Adjiman			Keane		
	Time (ms)	Mean ϕ	RMSE %	Time (ms)	Mean ϕ	RMSE %	Time (ms)	Mean ϕ	RMSE %
GS	4.6	-2.26	1.71%	4.3	51.20	0.19%	4.9	-31.25	10.10%
R-MFD	4.6	8.70	4.29%	4.1	12.86	5.49%	3.5	-31.11	10.18%
R-SQP	20.6	17.27	1.60%	12.2	34.68	2.56%	16.5	-32.65	11.44%
GS-MFD	8.8	23.19	0.68%	7.1	51.97	0.19%	6.5	-31.11	10.17%
GS-SQP	10.4	22.28	0.70%	9.4	52.02	0.18%	7.7	-31.11	10.17%
M-MFD	16.5	23.19	0.68%	20.7	47.86	0.47%	15.3	-31.09	9.77%
M-SQP	160.6	23.19	0.68%	113.7	52.04	0.19%	137.3	-31.11	10.17%
GA	373.9	23.19	0.68%	374.5	52.02	0.19%	378.4	-31.11	10.17%
GA-MFD	377.3	23.19	0.68%	377.7	52.04	0.19%	381.9	-31.11	10.17%
GA-SQP	378.9	23.19	0.68%	379.9	52.03	0.19%	385.2	-31.11	10.17%

that a good value is found by the MFD and SQP by using a multi-start strategy as shown. However, this increases the amount of time required to build the metamodel. The GS-MFD and GS-SQP provide similar resulting values of the condensed log likelihood function to the GA-MFD and GA-SQP results at a far lower computation cost. In this case the M-MFD is also finding high values of the condensed log likelihood function to a relatively low computational cost, albeit higher than the GS-MFD and GS-SQP. It is worth noting that for these functions there seems to be a good correlation between a high log likelihood and a low generalisation error.

5.6.2 Dimensionally scalable benchmark study

This study aims to benchmark the optimisation techniques for functions with higher dimensionality. This was done using the following dimensionally scalable polynomial function where n is the total number of design variables, chosen as 10, 40 and 60 respectively in this benchmark study.

$$\begin{aligned} f(\mathbf{x}) &= \frac{i}{n}\sum_{i} c_1 x_i^3 + c_2 x_i^2 + c_3 x_i + c_4 + c_5 sin(x_i) \\ c_1 &= 0.5 \\ c_2 &= -2.02(i - \frac{n}{2}) \\ c_3 &= 7.0(i - \frac{n}{2}) \\ c_4 &= 1.0 \\ c_5 &= 35.0 \end{aligned} \tag{5.30}$$

The function has varying degrees of non-linearity between the different design variables and is evaluated in the range 0 to 5. As with the 2D function 50 different training DOEs were evaluated for each of the three cases in order to reduce the risk of sporadic solutions.

Tables 5.5, 5.6 and 5.7 show the results of the parameter tuning for the dimensionally scalable polynomial in the cases of 10, 40 and 60 design variables respectively. For the 10 design variable case, Table 5.5, it can be seen that the GS-MFD and GS-SQP perform

Table 5.5: Scalable polynomial 10 design variables

	10 Training Points			20 Training Points			50 Training Points		
	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %
GS	<00:00:01	39.12	16.32%	<00:00:01	107.50	14.93%	<00:00:01	428.49	10.76%
R-MFD	<00:00:01	66.90	16.44%	00:00:02	182.21	12.73%	00:00:10	618.52	9.77%
R-SQP	<00:00:01	54.22	16.05%	<00:00:01	142.08	15.17%	00:00:03	548.38	12.61%
GS-MFD	<00:00:01	76.97	13.88%	00:00:01	202.91	10.66%	00:00:04	729.28	8.40%
GS-SQP	<00:00:01	77.19	13.59%	<00:00:01	197.48	10.99%	00:00:03	729.72	8.39%
M-MFD	00:00:01	71.72	15.57%	00:00:11	193.84	11.28%	00:01:13	671.56	8.86%
M-SQP	<00:00:01	74.74	13.96%	00:00:01	171.87	13.62%	00:00:31	718.10	8.58%
GA	00:00:03	74.19	14.47%	00:00:13	183.26	12.07%	00:01:06	656.52	8.84%
GA-MFD	00:00:03	74.74	14.37%	00:00:14	199.83	10.85%	00:01:10	728.86	8.39%
GA-SQP	00:00:03	77.81	13.44%	00:00:13	193.35	11.39%	00:01:08	729.68	8.39%

Table 5.6: Scalable polynomial 40 design variables

	10 Training Points			20 Training Points			50 Training Points		
	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %
GS	<00:00:01	519.24	15.67%	00:00:01	1153.24	14.60%	00:00:11	3190.31	13.85%
R-MFD	00:00:05	610.67	17.44%	00:00:23	1328.32	17.19%	00:03:07	3647.07	16.38%
R-SQP	00:00:01	566.24	17.51%	00:00:12	1329.35	17.22%	00:03:01	3937.76	12.99%
GS-MFD	00:00:06	664.50	15.54%	00:00:25	1435.77	14.31%	00:03:10	3945.74	12.87%
GS-SQP	00:00:03	669.91	15.98%	00:00:12	1437.64	14.51%	00:01:18	3946.86	12.92%
M-MFD	00:00:49	632.29	17.38%	00:03:43	1350.05	17.14%	00:29:34	3697.19	14.91%
M-SQP	00:00:13	603.41	17.40%	00:02:10	1355.04	16.83%	00:30:59	3946.38	12.92%
GA	00:00:41	622.84	17.48%	00:03:00	1321.30	17.26%	00:20:47	3564.98	17.01%
GA-MFD	00:00:43	633.20	17.41%	00:03:14	1347.24	17.18%	00:23:43	3696.73	14.97%
GA-SQP	00:00:44	659.21	16.42%	00:03:04	1352.33	16.84%	00:22:39	3933.32	13.07%

Table 5.7: Scalable polynomial 60 design variables

Optimisation method	10 Training Points			20 Training Points			50 Training Points		
	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %	Time $(hh:mm:ss)$	Mean ϕ	RMSE %
GS	<00:00:01	926.49	15.03%	00:00:03	2048.53	14.13%	00:00:26	5487.08	13.58%
R-MFD	00:00:12	1040.72	16.91%	00:01:03	2291.84	16.66%	00:07:54	6133.60	16.20%
R-SQP	00:00:07	1041.60	16.88%	00:00:43	2329.69	16.59%	00:10:27	6562.21	13.00%
GS-MFD	00:00:12	1137.31	14.98%	00:01:07	2463.50	14.05%	00:08:04	6559.71	12.92%
GS-SQP	00:00:09	1156.69	16.00%	00:00:36	2471.26	14.38%	00:04:13	6563.42	13.00%
M-MFD	00:02:12	1082.86	16.87%	00:10:42	2332.23	16.63%	01:21:39	6167.31	15.79%
M-SQP	00:01:16	1101.63	16.62%	00:07:33	2376.14	15.71%	01:45:10	6563.21	13.00%
GA	00:01:30	1050.56	16.95%	00:07:09	2241.87	16.70%	00:54:27	5898.04	16.56%
GA-MFD	00:01:40	1080.82	16.92%	00:09:47	2320.88	16.67%	01:03:26	6126.89	16.27%
GA-SQP	00:01:36	1093.91	16.59%	00:08:55	2309.32	16.68%	01:02:19	6562.34	13.00%

very well in comparison to the other algorithms, providing the highest log likelihood together with the hybrid GAs. In this case the remaining algorithms do not perform as well. The GS-MFD and GS-SQP provide the lowest generalised error over the 50 validation DoEs, followed by the GA-MFD and GA-SQP.

When increasing dimensionality of the scalable polynomial function to 40 design variables the benefit of the proposed approach becomes more apparent. The proposed approach delivers a solution with high mean condensed log likelihood value for a low computational effort when compared to the other evaluated methods. The GS-SQP provides the highest mean condensed log likelihood value for all numbers of training points. The GS-MFD provides the second highest mean log likelihood over all of the number of training points, followed by the GA-SQP. In the 50 training point case, the M-SQP provides the second highest mean condensed log likelihood, however for the 10 and 20 training point cases does not perform as well. For the 10 and 20 training point cases the GS-MFD and GS-SQP take slightly longer to build than the R-MFD and R-SQP. However, for the 50 training point case the GS-SQP takes less than half the time taken to build the R-SQP. The lowest generalisation error is provided by the GS, GS-MFD and GS-SQP. As more training points are used the GS-MFD and GS-SQP provide a better generalisation error.

In the final case with 60 design variables the GS-MFD and GS-SQP also perform very well. For 10 and 20 training points they provide the highest mean log likelihood values. For 50 training points the GS-SQP provides the highest mean condensed log likelihood followed closely by the M-SQP, then the GA-SQP and R-SQP, at 25, 15 and 2.5 times the computational effort respectively.

Of the proposed methods the GS-MFD and GS-SQP provide the best results for the time required to build the metamodels. They consistently outperform the random start point, GA-MFD and GA-SQP methods. As the dimensionality of the scalable

polynomial function increases, the benefit of using the solution of the GS as a starting point for the MFD or SQP increases. Overall the GS-SQP provides the best mean log likelihood for the time required to build the metamodel, as such it will be used in Section 5.7 for an industrial sized test case.

5.7 Case Study: Aircraft wing example

This section presents a study where gradient enhanced kriging metamodels are created for a finite element model of an aircraft wing. The GS-MFD and GS-SQP methods are compared to the GS, R-MFD and R-SQP methods. The multi-start and GA start point methods are not included as the computational effort would be too great. The study was first shown using an early implementation in Mortished *et al.* (2016) and later shown with the current implementation in Ollar *et al.* (2016b).

5.7.1 The wing model

The wing model consists of 126 aluminium sheet panels with designable thickness; as shown in Figure 5.2. Each of the design variables are aluminium sheets which are modelled with shell elements with a mesh size of 18 mm. The allowable thickness range is from 0.5-5 mm with a nominal thickness of 2.5 mm.

The wing is fully constrained on the wider end to represent attachment to the fuselage. Forces and moments are applied to nodes located at the centroid of each rib, as shown in Figure 5.3a and 5.3b, and using one dimensional (RBE3) elements equally distributed to the edges of the rib. An example of the deformation due to the loading is shown in Figure 5.4. Two responses are considered: vertical deflection and rotation of the wing tip. Both are measured at the horizontal centre and vertical top of the wing tip.

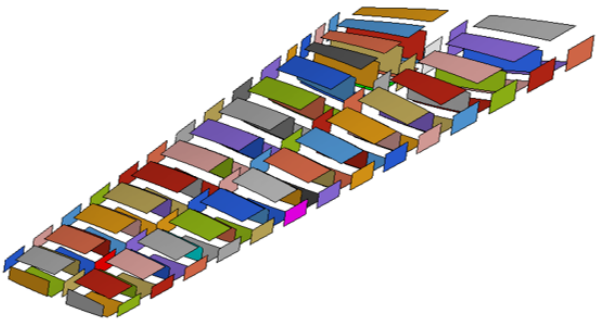
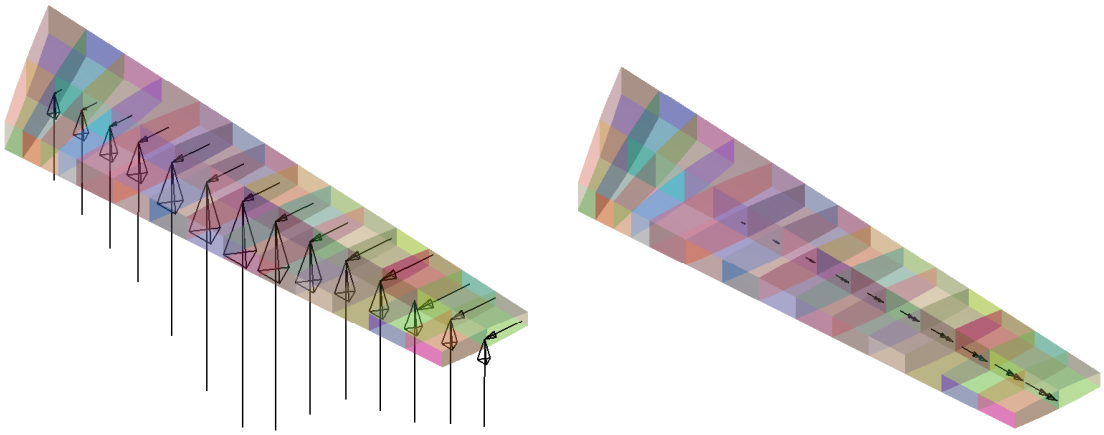


Figure 5.2: Wing panel design variables.



(a) Applied forces.

(b) Applied moments.

Figure 5.3: Wing loading.

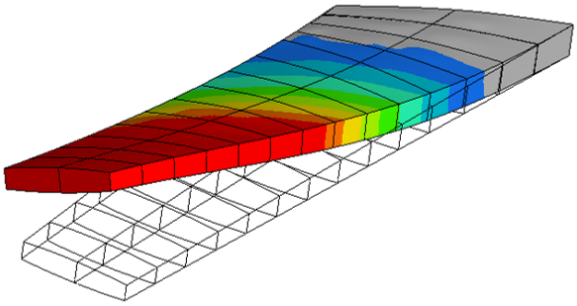


Figure 5.4: Magnified deformation due to loading.

The model is analysed using OptiStruct v13.0.210 (Altair Engineering, Inc., 2014b). OptiStruct provides analytical gradients via either the direct or the adjoint method depending on which is the more efficient choice for the case. In this case the adjoint method is used as the number of design variables is far greater than the number of responses; evaluating the gradients took roughly the same time as evaluating the function values, doubling the total analysis time.

5.7.2 Study setup

The study was performed by building the metamodels with 5 points at first, followed by 10, 20, 50 points. For this purpose, sampling was performed using MELS. One benefit of MELS is that any subset of the DoE in sequence from the first point is suitably spaced. This allows the user to assess the approximation quality interactively allowing for a far more flexible approach than would be possible with other space filling techniques such as the Optimal Latin Hypercube (Audze and Eglajs, 1977). To leverage this feature a single DoE was created. 50 points were reserved for training the metamodels and 500 points were reserved for validation.

5.7.3 Results

Table 5.8 and 5.9 show the performance of the GS-MFD and GS-SQP is compared with that of the GS, R-MFD and R-SQP. In Table 5.8a the condensed log likelihood obtained by the different optimisation methods is shown for the wing tip displacement. It can be seen that the GS-SQP outperforms the other methods, closely followed by the GS-MFD which are second best in all cases apart from in the 10 point case where R-SQP provides a slightly better solution.

In Table 5.8b the generalisation error obtained for the different optimisation methods

Table 5.8: Results for wing tip displacement.

No. points	GS	R-MFD	R-SQP	GS-MFD	GS-SQP
a) Condensed log likelihood (ϕ)					
5	732	798	1082	1731	1780
10	2197	2173	4069	3915	4241
20	5406	5113	6434	8275	8725
50	14696	13912	17415	21737	21853
b) Generalisation error (RMSE)					
5	14.77%	14.78%	14.78%	13.27%	10.69%
10	6.56%	12.57%	10.84%	5.17%	7.66%
20	6.34%	12.64%	12.59%	5.01%	6.66%
50	5.82%	12.37%	12.32%	6.33%	7.26%

is shown. It can be seen that GS, GS-MFD and GS-SQP provide solutions which outperform R-MFD and R-SQP. It can also be seen that even though GS-MFD and GS-SQP provide condensed log likelihood values which are higher than the one for GS, the generalisation error is not necessarily improved.

Table 5.9a shows that, for wing tip rotation, the GS-MFD and GS-SQP outperform the other evaluated methods, which is reflected in the generalisation error, Table 5.9b. Similarly to the wing tip displacement, the golden search optimisation method shows lower resulting generalisation error than the R-MFD and R-SQP.

Table 5.9: Results for wing tip rotation.

No. points	GS	R-MFD	R-SQP	GS-MFD	GS-SQP
a) Condensed log likelihood (ϕ)					
5	1360	1384	1500	2091	2157
10	2768	2621	3682	3949	3910
20	5982	5293	6371	7824	7916
50	15998	13494	16003	19123	19236
b) Generalisation error (RMSE)					
5	10.10%	8.16%	8.16%	9.07%	9.06%
10	4.50%	7.80%	7.64%	4.81%	6.32%
20	4.55%	7.62%	7.60%	3.91%	4.16%
50	4.07%	7.18%	7.18%	3.49%	3.68%

5.8 Summary

One of the main challenges of kriging, and gradient enhanced kriging in particular, is the computational cost associated with the tuning parameter optimisation necessary for building the metamodel. In this chapter an approach was suggested for efficient tuning parameter optimisation for building well conditioned gradient-enhanced kriging metamodels. The approach consists of two steps, namely a one-dimensional line search where all hyper parameters are treated as one variable, and a gradient based optimisation starting from the solution of the initial line search.

In order to ensure a suitable condition number of the correlation matrix, an upper bound constraint was enforced. Partial derivatives of the condition number with respect to the correlation matrix were derived in order to use this constraint in the gradient based optimisation approach. Both the method of feasible directions and sequential quadratic programming were evaluated within the approach.

The approach was compared to random start point gradient based algorithms, multiple start point gradient based algorithms and a genetic algorithm followed by

gradient based algorithms from promising points. It was shown that the approach outperforms random start-point and multi-start gradient based algorithms in terms of both computational performance and quality of solutions. The comparative study shows the SQP to be the better choice of algorithm within the approach as it provides slightly higher condensed log likelihood values than the MFD for a similar time to build.

The proposed approach, using both the SQP and MFD, was compared to a selection of the other optimisation approaches using an aircraft wing model comprising of 126 thickness design variables. The GS-SQP consistently provides the highest condensed log likelihood value closely followed by the GS-MFD.

In some case it was shown that a big improvement in log likelihood did not necessarily translate to an improvement in generalisation error. This was particularly apparent for the wing tip displacement metamodels where the GS-SQP provided a higher condensed log likelihood than the GS case but the generalisation error was of comparable magnitude.

Chapter 6

An MDO framework for problems with discipline disparities

This chapter presents an MDO framework based on the mid-range metamodel method as outlined in Chapter 4. Section 6.1 discusses the use of existing metamodel-based optimisation techniques, such as the MAM for MDO. It is suggested to introduce the concept of disciplines within the optimisation framework to take advantage of disparities between the disciplines. Section 6.2 outlines a method for reducing the computational effort related to solving MDO problems with disparate design variable dependences of the disciplines. The chapter concludes with two case studies. The first case study is an MDO benchmark example of a thin-walled beam in Section 6.4 and the second is an MDO of an aircraft wing subject to stength and stiffness as well as bird strike requirements in Section 6.3.

6.1 Separation of disciplines

It would be straightforward to use metamodel based optimisation algorithms for MDO by simply treating the responses of all disciplines as if they were produced by a single discipline. Figure 6.1 shows how a metamodel-based optimisation algorithm such as the MAM could be used to solve MDO problems without any modifications. The optimisation algorithm would simply produce a set of DOE points for which response function values are to be returned. Whether they are evaluated from a single discipline

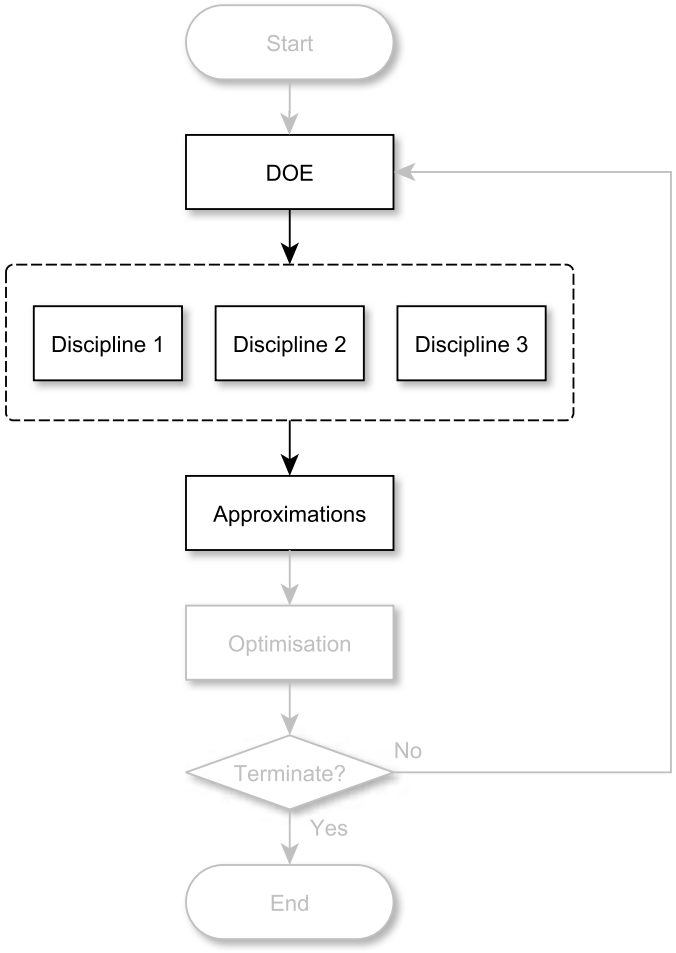


Figure 6.1: MDO using single-disciplinary approach.

or several would not be relevant to the optimisation algorithm. Once the points are evaluated metamodels are created for each response, which are then used to find an approximate optimum using an optimisation algorithm. Finally, termination criteria are checked. If the termination criteria are met the process ends, otherwise a new iteration is started at the point of creating DOE points. Note that, as mentioned in Chapter 2, any disciplines including multi-physics coupling are handled with MDAs and labelled as a single discipline.

In this section it is argued that the framework can be made more computationally efficient if advantage is taken of disparities in the disciplinary attributes. This is made possible by separately considering the disciplines within the optimisation framework and, in particular, creating individual DOEs for each discipline as shown in Figure 6.2.

6.1.1 Advantages of separation

There are several advantages to using individual DOEs for each of the disciplines. This section outlines the ones that have been taken advantage of in the current framework.

Required number of points

As the functions belonging to different disciplines can be arbitrarily complex there might be a discrepancy across disciplines in terms of how many points are needed to obtain metamodels of required accuracy. With individual DOEs for each discipline the number of points can be independently controlled. Furthermore, once the metamodels for a particular discipline reaches required accuracy, computational resources could be saved or used to improve accuracy of the disciplines that have not yet met their required accuracy.

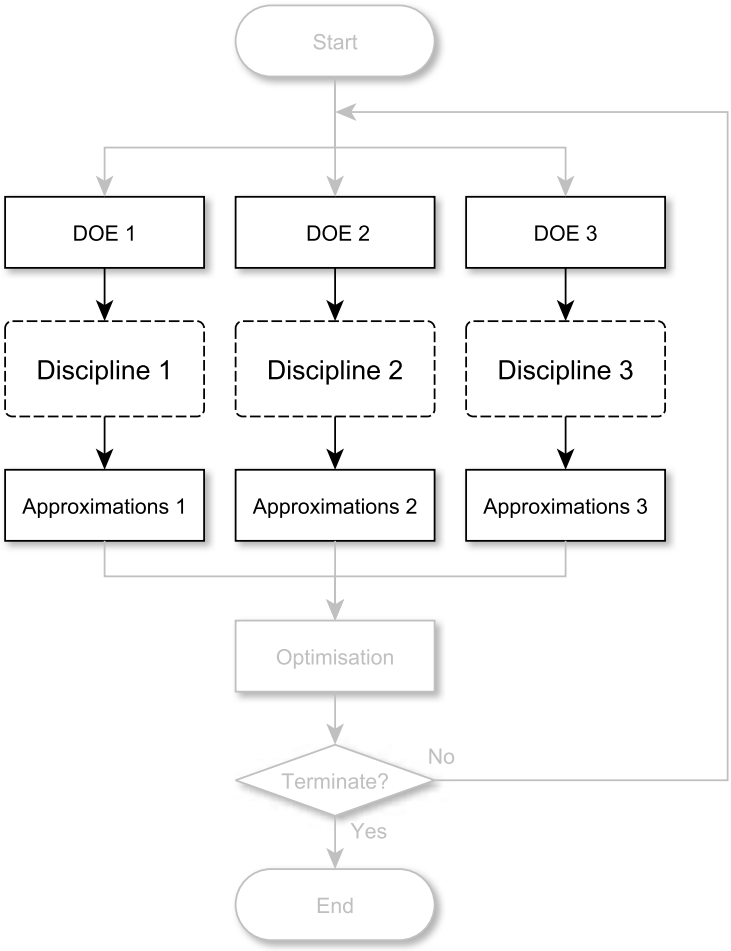


Figure 6.2: MDO using individual DOE's for each discipline.

Available gradients

As previously mentioned, gradients may not be available for some of the disciplines. However, for the disciplines which do have available gradients, it is desired to take advantage of this when building (gradient-enhanced) metamodels. As gradients improve the quality of the metamodels for a given computational budget, this allows reduction of the computational budget in each iteration.

Simulation failures

It is not uncommon for simulation failures to occur due to numerical, software, hardware or network issues. In the event of failure of numerical simulations, using the single discipline approach it is common to discard the failed point for all responses to keep a consistent set of points across all responses. However when using individual DOEs the failed point will only have to be discarded for the affected discipline.

Disparate variable dependence

There are cases where the full set of design variables are not present in all the models. In such situations it is beneficial to create DOEs and build metamodels in the space of the variables related to the individual disciplines rather than in full design variable space as the number of design variables influences the metamodel quality as discussed in Section 1.1. This can be taken further to include only variables that have influence on the responses of individual disciplines and is the main idea behind the research presented in Section 6.2.

6.1.2 Disadvantages of separation

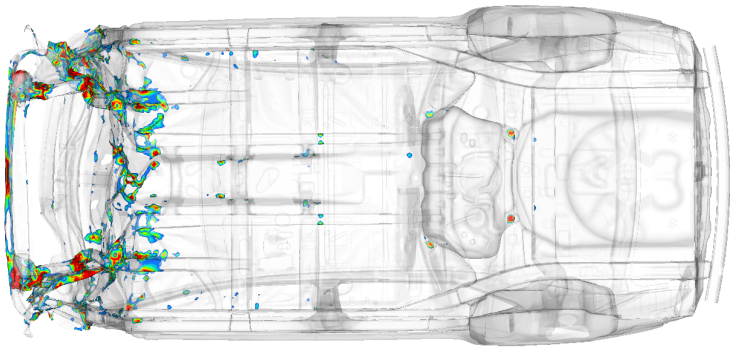
The disadvantage of the proposed separation is that a consistent set of points will not be evaluated across all disciplines. With a consistent set of points across all disciplines it is straightforward to assess all points against the posed optimisation problem. Points from the set of training points that are superior to points found by the optimiser could therefore be chosen by the algorithm as the current best point. However, finding such a point would be purely chance and for problems with a large number of variables is is very unlikely.

6.2 Metamodels in sub-spaces

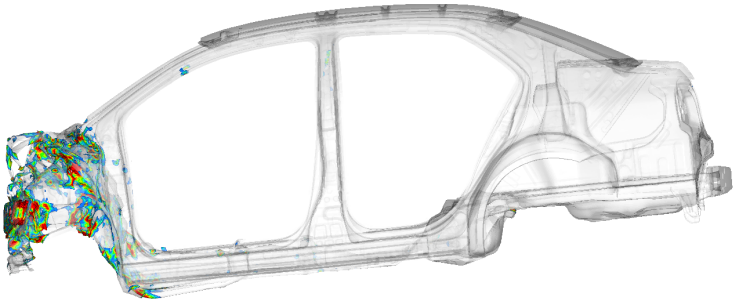
This section describes a technique for reducing the computational budget related to solving multidisciplinary design optimisation problems with disparate design variable dependences of the disciplines.

Suppose that the responses belonging to a discipline in the MDO problem only depend on a subset of the full set of design variables, i.e. a set of the variables has none or very little influence on the responses of the particular discipline. An example of this from the automotive industry can be seen in Figure 6.3 which shows a front crash simulation of an automotive structure. As can be expected, it can be concluded that the internal energy is concentrated in the front of the vehicle. It can be assumed that variables in the rear, e.g. the thickness of the rear bumper, will have very little effect on related responses. With such information for the responses of all disciplines, a partitioning of design variables, such as the one in Figure 6.4, can be created. Instead of building metamodels in the space of the full set of design variables, the response belonging to each discipline is built in the space of its related subset of variables. The resulting metamodels will be built in a space of reduced dimensionality and hence will require a reduced computational budget.

There are several examples of the use of this approach, e.g. by Sobieszczanski-Sobieski *et al.* (2001), Kodiyalam *et al.* (2004), Ollar *et al.* (2014) and Ryberg *et al.* (2015). The benefit is that sampling and metamodel building can be carried out in a space of reduced dimensionality which allows for a reduction in computational budget for obtaining an metamodel of sufficient quality. However, poor assumptions made when identifying significant variables can lead to metamodel errors that cannot be reduced by additional sampling. As will be described in Section 6.2.2 this can be remedied with a recovery mechanism by taking advantage of the iterative range reduction in the trust region strategy of the MAM.



(a) Top view



(b) Side view

Figure 6.3: Automotive model subject to front crash load case. Each element is colored according to its level of internal energy.[1]

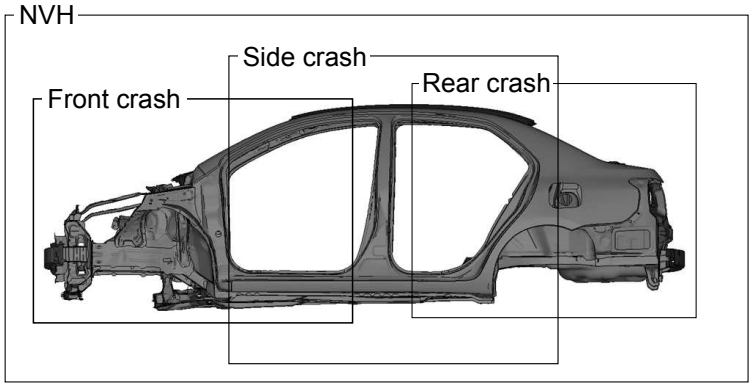


Figure 6.4: Conceptual partitioning of design variables for an automotive model into significant design variable sets related to each of the disciplines: *Front Crash*, *Side Crash*, *Rear Crash* and *Noise Vibration and Harshness* (NVH)

[1]The model was developed by the National Crash Analysis Center (NCAC), The George Washington University, Washington, USA.

6.2.1 Formulation of sub-space metamodels

Here a mathematical formulation for introducing sub-space metamodels in metamodel assisted MDO is given. Unlike previous work (Sobieszczanski-Sobieski *et al.*, 2001; Kodiyalam *et al.*, 2004; Ollar *et al.*, 2014; Ryberg *et al.*, 2015), all response functions are assumed to be defined in the full variable space of the optimisation problem in order to control insignificant variables. It will be shown that when used within a trust region framework, this formulation becomes necessary to account for possible errors in assumptions on partitioning.

Consider solving the optimisation problem (4.1) using metamodels. The optimisation problem becomes

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \tilde{f}_0\left(\mathbf{x}\right) \\ \text{subject to} \quad & \tilde{f}_j\left(\mathbf{x}\right) \leq 1, \qquad j = 1, \ldots, m \\ & A_i \leq x_i \leq B_i, \quad i = 1, \ldots, n \end{aligned} \tag{6.1}$$

where $\tilde{f}_0(\mathbf{x})$ is an metamodel of the objective function and $\tilde{f}_j(\mathbf{x})$ is an metamodel of the j-th constraint function. Given that the design variables in the optimisation problem can be categorised either as significant or insignificant for each related response, a projection can be defined for each response j from the design variable space onto the space of the significant variables for that particular response. This is denoted as

$$\left.\begin{aligned} &\boldsymbol{\xi}_j = P_j^{\xi}\mathbf{x} \\ &P_j^{\xi} : \mathbb{R}^n \mapsto \mathbb{R}^{s_j} \end{aligned}\right\}, \quad j = 0, ..., m, \tag{6.2}$$

where n is the number of design variables in the optimisation problem and s_j is the number of significant variables for the response j. A projection onto the space of the

insignificant variables is defined in the same manner as

$$\left.\begin{aligned} \boldsymbol{\psi}_j &= P_j^{\psi}\mathbf{x} \\ P_j^{\psi} &: \mathbb{R}^n \mapsto \mathbb{R}^{n-s_j} \end{aligned}\right\}, \quad j = 0, ..., m. \tag{6.3}$$

From here on the projections are described according to the following convention

$$\mathbf{x} = \begin{bmatrix} \boldsymbol{\xi}_j \\ \boldsymbol{\psi}_j \end{bmatrix}, \quad j = 0, ..., m, \tag{6.4}$$

noting that the components of $\boldsymbol{\xi}_j$ and $\boldsymbol{\psi}_j$ are present in $\mathbf{x}$ in arbitrary order. The responses in the optimisation problem can then be described as

$$f_j(\mathbf{x}) = f_j\left(\begin{bmatrix} \boldsymbol{\xi}_j \\ \boldsymbol{\psi}_j \end{bmatrix}\right), \quad j = 0, ..., m, \tag{6.5}$$

where the values of $\boldsymbol{\psi}_j$ can be chosen arbitrarily since they are deemed to be insignificant to the response. The metamodels of the responses, therefore, may now be defined in the space of only the significant variables which allows a re-writing of the approximate optimisation problem as

$$\begin{aligned} \underset{\mathbf{x}}{\text{minimize}} \quad & \tilde{f}_0\left(\boldsymbol{\xi}_0\right) \\ \text{subject to} \quad & \tilde{f}_j\left(\boldsymbol{\xi}_j\right) \leq 1, \qquad j = 1, \ldots, m \\ & \boldsymbol{\xi}_j = P_j^{\xi}\mathbf{x}, \qquad j = 0, \ldots, m \\ & A_i \leq x_i \leq B_i, \quad i = 1, \ldots, n \end{aligned} \tag{6.6}$$

where each response is defined only in the space of variables that are significant to the response. The optimisation problem, however, is defined in the full design variable space. This has the benefit that as each metamodel is defined only in the space of the significant

variables, the sampling of training points only needs to be carried out in that space and projected onto the full space as demonstrated by Figure 6.5. Hence if the number of significant variables is small compared to the number of design variables, the density of the training points will increase leading to a better quality metamodel as compared to what would have been achieved otherwise. Note that even though there is one projection per response, practicalities may require groups of responses to use the same projection, e.g. due to several responses being evaluated from the same discipline.

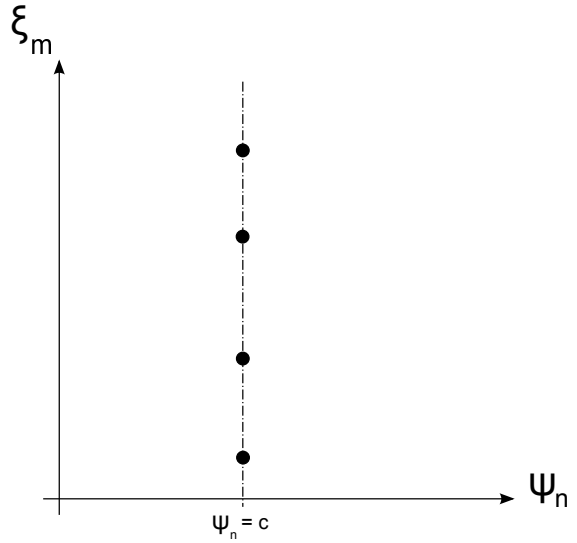


Figure 6.5: Sub-space sampling shown in two dimensions. Vertical axis corresponds to the significant variable ξ_m and horizontal axis to the insignificant variable ψ_n. Sampling is carried out in the space of the significant variable while the insignificant variable is kept at a constant value, c.

6.2.2 Integration in trust region framework

In this section an approach to building sub-space metamodels within the MAM is proposed. A recovery mechanism for erroneous assumptions for sub-space partitioning is also suggested. Sub-space metamodels can be introduced in the MAM framework by re-writing the sequence of optimisation problems (4.2) as:

$$\begin{aligned}
\underset{\mathbf{x}}{\text{minimize}} \quad & \tilde{f}_0^{\,k}\left(\boldsymbol{\xi}_0\right) & & \\
\text{subject to} \quad & \tilde{f}_j^{\,k}\left(\boldsymbol{\xi}_j\right) \leq 1, & & j = 1, \ldots, m \\
& \boldsymbol{\xi}_j = P_j^{\xi}\mathbf{x}, & & j = 0, \ldots, m \\
& \left.\begin{aligned} A_i^k \leq x_i \leq B_i^k \\ A_i^k \geq A_i \\ B_i^k \leq B_i \end{aligned}\right\} & & i = 1, \ldots, n
\end{aligned} \tag{6.7}$$

Note that the mid-range metamodels created here in each iteration are functions of the significant variables only. The significant variables for each discipline are identified by the designer. Such judgement may be based on, for instance, engineering experience or design variable ranking studies. In the techniques presented by by Sobieszczanski-Sobieski *et al.* (2001), Kodiyalam *et al.* (2004), Ollar *et al.* (2014) and Ryberg *et al.* (2015) deficiencies in sub-space partitioning, i.e. by failing to identify a significant variable, can result in metamodel errors that cannot be resolved by additional sampling.

Regardless of how carefully the partitioning of variables is made, there is always a risk that significant variables will be incorrectly identified as insignificant. Therefore a recovery mechanism for such errors is needed. This can be implemented in the trust region strategy by making sure that the vector of insignificant variables for each response is updated at the end of the iterations depending on the current best point as proposed by Ollar *et al.* (2015, 2016c).

Let $\mathbf{x}^*_{k-1}$ denote the solution vector to the previous iteration $(k-1)$ for the optimisation problem (6.11). For each response this can be written in accordance to (6.5) as

$$\mathbf{x}^*_{k-1} = \begin{bmatrix} \boldsymbol{\xi}^*_{k-1} \\ \boldsymbol{\psi}^*_{k-1} \end{bmatrix} \quad , \tag{6.8}$$

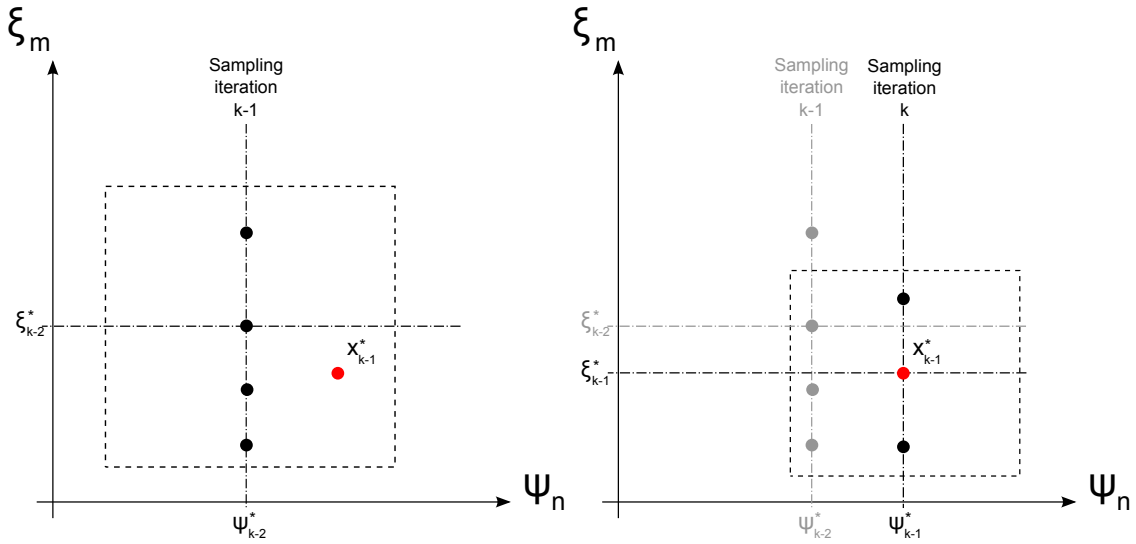
where $\boldsymbol{\xi}^*_{k-1}$ denotes the projection of the solution vector onto the space of the significant variables and $\boldsymbol{\psi}^*_{k-1}$ onto the space of insignificant variables. The subscript j, denoting the response number of the projections, has been omitted for brevity.

The values of $\boldsymbol{\psi}^*_{k-1}$ are then used as the constant values for the insignificant variables for sampling in the current iteration, k, according to

$$\boldsymbol{\psi}_k = \boldsymbol{\psi}^*_{k-1} \quad , \tag{6.9}$$

where $\boldsymbol{\psi}_k$ denotes the values of the insignificant variables for sampling. Figure 6.6 demonstrates how the value of $\boldsymbol{\psi}_n$ changes from the previous iteration to the current for the two dimensional case. The change is due to updating the value according to the current best solution. As the metamodels for the new iteration are built using the sampling including this update, any changes in response values due to changes in insignificant variables from the previous iteration, will be accounted for in the new iteration.

With the possibility of significant variables being identified as insignificant, there is a possibility that existing points located within the recycle region but far away from the current value of the insignificant variables may spoil the resulting metamodel. The recycle-region size is therefore multiplied by a reduction factor b_s, for the insignificant variables, as shown in Figure 6.7.



(a) Sub-problem in iteration k-1 (b) Sub-problem in iteration k

Figure 6.6: The values of the insignificant variables in the solution of iteration $k-1$ differ from the same values in the sampling of iteration k-1. Potential changes in the function values as a consequence of this is taken into account in iteration k by updating the values of the insignificant variables for sampling according to the solution of iteration $k-1$.

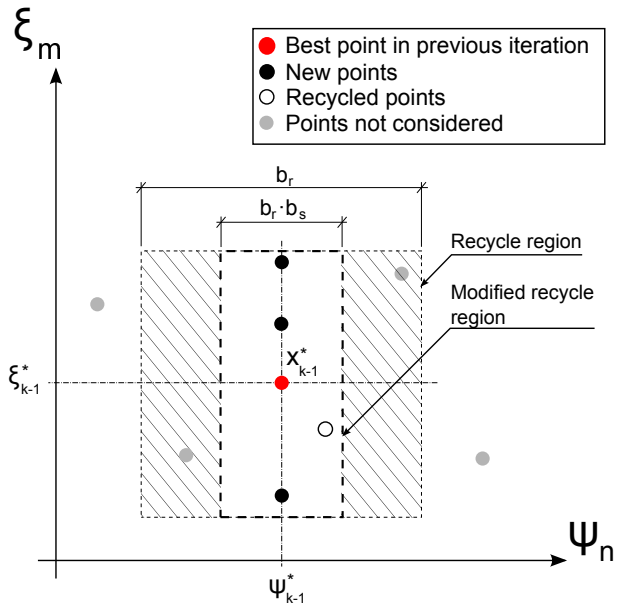


Figure 6.7: Reduction of recycle region (see Figure 4.4) for insignificant variables by multiplication by a factor b_s.

6.2.3 Automatic variable selection

Although made more robust by the recovery mechanism, the approach relies on the user to make the assumption of which variables are significant for each discipline. This section explores the idea of automatically determining the set of significant and insignificant variables based on design variable ranking at the start of each iteration during the optimisation rather than by the user before starting the optimisation.

Not only would this eliminate human error in determining the variable dependence but it would also allow for a different variable dependence to be identified for each iteration. This could be beneficial as one can easily imagine that different regions of the design space can have different variable dependence.

In Section 3.3.1.4 it was described how the cross-validated moving least squares method could be used to carry out a backwards elimination ranking, proposed by Tu and Jones (2003). This is carried out by calculating an impact factor for each design variable based on successively leaving out each variable. An impact factor for a particular variable, x_j, can be calculated by building an approximation that ignores the effect of x_j. The RMSE of the leave one out cross validation error, outlined in Section 3.3.1.3, of this approximation is then compared to the error of an approximation built with the full set of variables as

$$I_j = \frac{RMSE^j - RMSE}{RMSE}, \tag{6.10}$$

where $RMSE^j$ denotes the RMSE for an approximation built without the variable x_j, and $RMSE$ is the error for an approximation built on the full set of variables. If the impact factor is a measure of the importance of each variable on the response, a small or negative value for a variable indicates that the variable is a candidate for elimination.

In this approach, the described design variable ranking is carried out at the start

of each iteration using the information contained within the trust-region from previous iterations. Using this information a set of significant variables with impact factor greater than a perscribed value, $\boldsymbol{\xi}_j^k(I_j \geq I_j^s)$, and a set of insignificant variables, $\boldsymbol{\psi}_j^k(I_j < I_j^s)$, for the current iterations, are obtained.

Points are then added for the current iteration, in the plane of significant variables, and within the trust-region. However, in order to prepare for the design variable ranking in the next iteration, variation of the insignificant variables needs to be introduced in the DOE. This is done by sampling in full design variable space but with reduced bounds of the trust-region for the insignificant variables by a factor b_a, similar (or equal) to the recycle region reduction factor b_s, shown in Figure 6.7.

After sampling, approximations are created in the space of the significant variables and the optimisation problem is carried out in full space as

$$\begin{aligned}
\underset{\mathbf{x}}{\text{minimize}} \quad & \tilde{f}_0{}^k\left(\boldsymbol{\xi}_0^k\right) & & \\
\text{subject to} \quad & \tilde{f}_j{}^k\left(\boldsymbol{\xi}_j^k\right) \leq 1, & & j = 1, \ldots, m \\
& \boldsymbol{\xi}_j^k = P_j^{\xi,k}\mathbf{x}, & & j = 0, \ldots, m \\
& \left.\begin{aligned} A_i^k \leq x_i \leq B_i^k \\ A_i^k \geq A_i \\ B_i^k \leq B_i \end{aligned}\right\} & & i = 1, \ldots, n
\end{aligned} \tag{6.11}$$

Initial attempts on a simple model by Ollar *et al.* (2015), showed promising results using the described approach. However, further testing indicated that for more complicated problems there was no computational benefit of this approach. This is believed to be because in each iteration, enough information needs to be present in order to carry out variable ranking in full design space. As a minimum, this method

requires enough points in each iteration to carry out linear regression in full space plus two points. In other words, the number of points needs to be at least the number of design variables plus two points. This increased computational budget diminishes the computational gain from using sub-space approximations.

Many attempts of varying the parameters I_j^s and b_a were carried out without finding a combination that would show a consistent computational reduction. Therefore this approach is not used throughout the rest of this thesis, but the author remains hopeful that future research will see this implementation improved to the point where this automatic approach will show consistent computational gains.

6.3 MDO of a thin-walled beam section

This case study presents a multidisciplinary design optimisation benchmark example of a thin-walled beam structure subject to both a strength and stiffness load case as well as impact load cases. Advantage is taken of the local nature of the impact load cases with the use of sub-space metamodels as outlined in Section 6.2. This enables the study to be performed at a much reduced computational cost than would otherwise be possible. Furthermore the recovery mechanism for deficiencies in sub-space partitioning outlined in Section 6.2.2 is demonstrated.

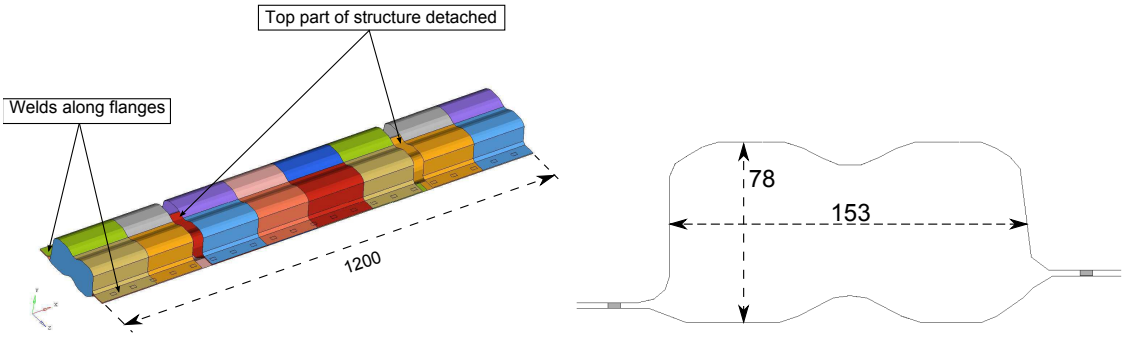
6.3.1 Beam model

The dimensions of the beam are $L = 1200mm$, $W = 153mm$, $H = 78mm$, and is made of two thin-walled hat sections spot-welded together along two flanges as shown in Figure 6.8. It is modelled using shell elements with an average mesh size of 10 mm. The welds are modelled using single hexahedron elements connected to the shell elements using a

tied contact formulation. The structure is divided into 26 components, shown in Figure 6.9, with individual thickness values which are to be determined. The starting thickness is 3 mm for each component and the allowable thickness range is 1-5 mm.

6.3.2 Load cases

The beam is subjected to one static load case and three dynamic load cases as outlined below and shown in Figure 6.10.



(a) 3D view. (b) Section perpendicular to longitudinal direction.

Figure 6.8: The thin-walled beam consisting of two hat sections spot-welded together along the flanges. Top part of structure has been detached at two locations.

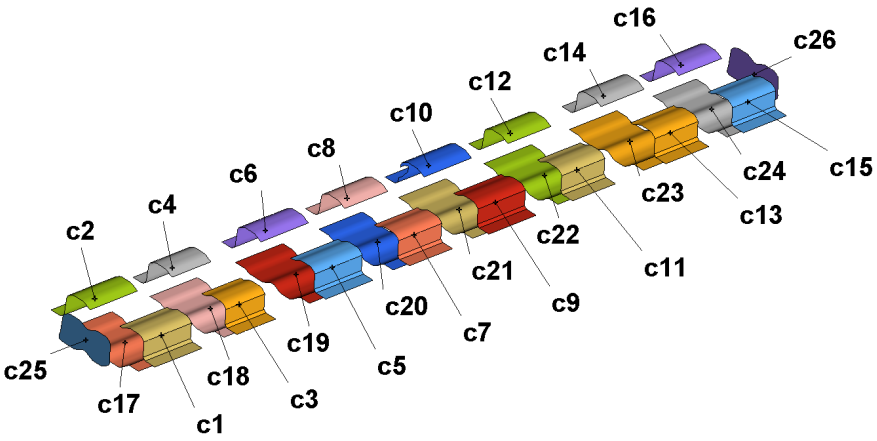
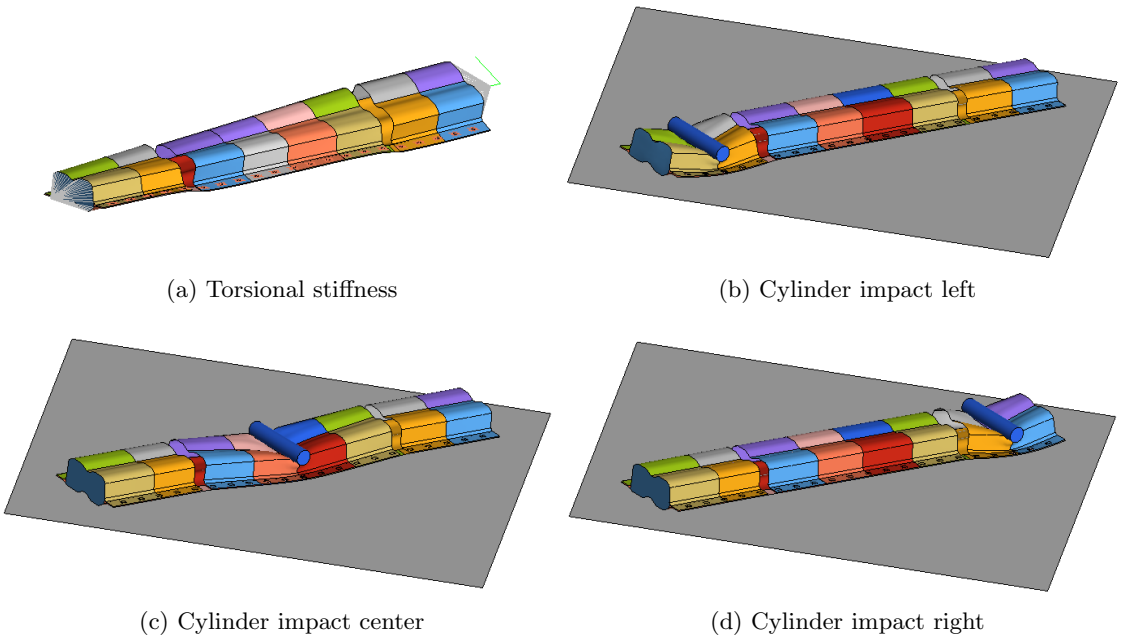


Figure 6.9: Exploded view of the 26 panels included in the MDO. Every panel has designable thickness range between 1-5 mm.

(a) *Torsional stiffness*

A free node on the left of the beam is constrained in all translational degrees of freedom as well as from rotation around the longitudinal axis of the beam. The node is connected to the edges of component c25 (see Figure 6.9) using one dimensional elements (RBE3) that distributes forces from the free node to the structure. At the right part of the beam a force is applied to a lever arm, connected to the beam with the same type of connection, resulting in a moment around the longitudinal axis of the beam.

The response is the resulting rotation at the point where the moment is applied. The mass of the structure is also used as a response and is extracted from this load case. The load case which is a static implicit load case is analysed using Altair OptiStruct (Altair Engineering, Inc., 2014b).



(a) Torsional stiffness

(b) Cylinder impact left

(c) Cylinder impact center

(d) Cylinder impact right

Figure 6.10: Load cases included in the MDO.

(b) *Cylinder impact left (CL)*

A heavy cylinder with an imposed velocity impacts the left section of the beam in vertical direction. The beam is supported in vertical direction by a rigid plane. At impact the beam is compressed by the heavy cylinder. The response is measured as the vertical deformation of the top part of the beam, caused by the cylinder impact. The load case, which is a dynamic explicit load case, is analysed using Altair RADIOSS (Altair Engineering, Inc., 2014c).

(c) *Cylinder impact center (CM)*

The load case definition is the same as for (b) except for the impact position of the cylinder which is now at the centre of the beam.

(d) *Cylinder impact right (CR)*

The load case definition is the same as for (b) and (c) except for the impact position of the cylinder which is now at the right section of the beam.

6.3.3 Optimisation problem setup

The objective of this MDO problem is to minimise the mass of the beam subject to sufficient torsional stiffness and not exceeding maximum intrusion from the cylinder impact. The target values are summarised in Table 6.1 together with initial values for each response. The mass and torsion response, evaluated using OptiStruct, have available gradients which are used during the optimisation to build gradient-enhanced approximations. As the mass response is linear in the space of the design variables a simple linear regression using the least squares method (LSM) is used to approximate this response. The remaining responses are approximated using the moving least squares method (MLSM).

Four types of optimisations were carried out as outlined below. All optimisations were carried out using the MAM with 10 candidate points in each iteration. The number of significant variables identified for each discipline and the corresponding number of points per iteration is presented in Table 6.2 for each optimisation. The convergence criteria was set such that the approximation quality must be below 5% and the trust region size must be smaller than 10% of the design region. The DOE method used for sampling of training points and start points for the SQP is based on a random number generator and will hence produce a different set of points depending on the initially chosen seed. In order to account for this uncertainty 50 optimisations with varying seeds were carried out for each of the optimisation types described below.

Table 6.1: Responses included in the optimisation problem and corresponding initial and target value, whether gradients are available, and the choice of approximation technique.

Response	Initial	Target	Gradients	Approximations
Mass	1.00	*min*	Yes	LSM
Torsion	0.98	≤ 1.00	Yes	MLSM
CL	1.18	≤ 1.00	No	MLSM
CM	1.18	≤ 1.00	No	MLSM
CR	1.17	≤ 1.00	No	MLSM

Table 6.2: Number of significant variables identified for each discipline and corresponding number of points per iteration.

Opt.	1		2		3		4	
	ns	np	ns	np	ns	np	ns	np
Torsion	26	39	26	39	26	39	26	39
CL	26	39	7	39	7	10	7	10
CM	26	39	12	39	12	18	11	18
CR	26	39	7	39	7	10	7	10

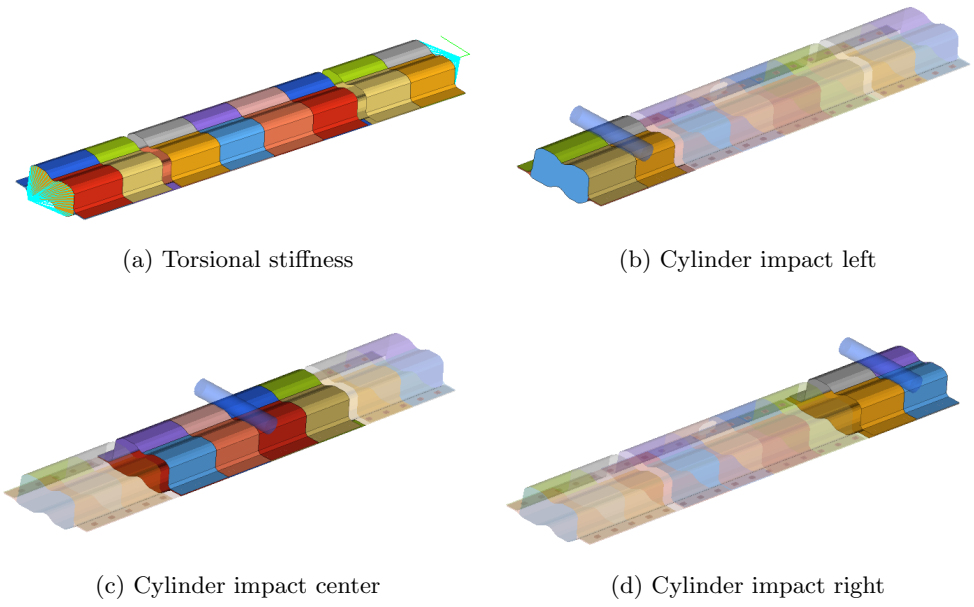
ns - number of significant variables
np - number of points per iteration

Optimisation 1 - Full space approximations

The first optimisation is set up as a benchmark using the MAM without sub-space approximations. The number of points evaluated per iteration is chosen as $np = 1.5 \times n = 39$ per load case, leading to a total of 156 points per iteration.

Optimisation 2 - Same budget

In the second optimisation sub-space approximations with the proposed recovery mechanism is used within the MAM. Partitioning of the beam model is shown in Figure 6.11, where insignificant variables are shown transparent. The optimisation is carried out with the same computational budget per iteration as *Optimisation 1* in an attempt to increase the quality of the approximations in each iteration. This will hopefully lead to fewer required iterations for convergence



(a) Torsional stiffness

(b) Cylinder impact left

(c) Cylinder impact center

(d) Cylinder impact right

Figure 6.11: Sub-space partitioning for the beam structure. Insignificant variables are showed as transparent.

Optimisation 3 - Decreased budget

In the second optimisation sub-space approximations with the proposed recovery mechanism is used within the MAM. The partitioning of variables is the same as in the second optimisation. Here the number of points are determined individually for each discipline as $ns = 1.5 \times ns$, where ns is the number of significant variables for the discipline. The aim of the optimisation is to keep the number of required evaluations per iteration to a minimum.

Optimisation 4 - Erroneous partitioning

In the final optimisation sub-space approximations with the proposed recovery mechanism is used within the MAM. The partitioning of variables is the same as in the second and third optimisation with one exception. Variable c7, shown in Figure 6.12, which is significant to discipline CM, is deliberately identified as insignificant in order to test the proposed recovery mechanism outlined in Section 6.2.2.

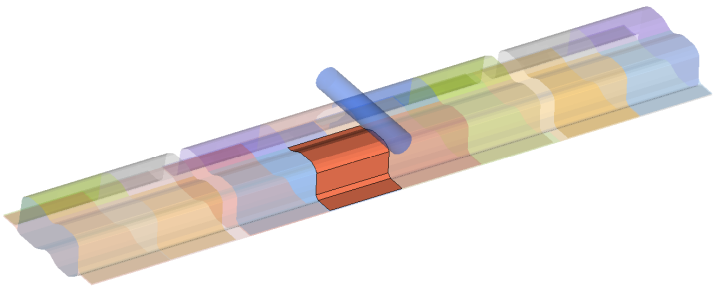


Figure 6.12: Significant variable erroneously identified as insignificant.

6.3.4 Results

The results of the study are presented in Figure 6.13. For each optimisation the median, upper and lower quartiles, and the minimum and maximum value are shown for the number of iterations, the number of evaluations, the objective function, and the maximum constraint violation.

Optimisation 1 - Full space approximations

The first optimisation which was used as a comparison for the other three optimisations finished on average in 10.5 iterations, having required 1407 evaluations, and with an average reduction in objective function of 13.2%.

Optimisation 2 - Same budget

The second optimisation, which was using sub-space approximations with the same number of evaluations per iteration as the first optimisation, finished on average in 8.5 iterations, 2 iterations less than the first, having spent 1242 evaluations, 165 less than the first optimisation, and with an average reduction in the objective function of 13.5%, 0.3% more than the first optimisation.

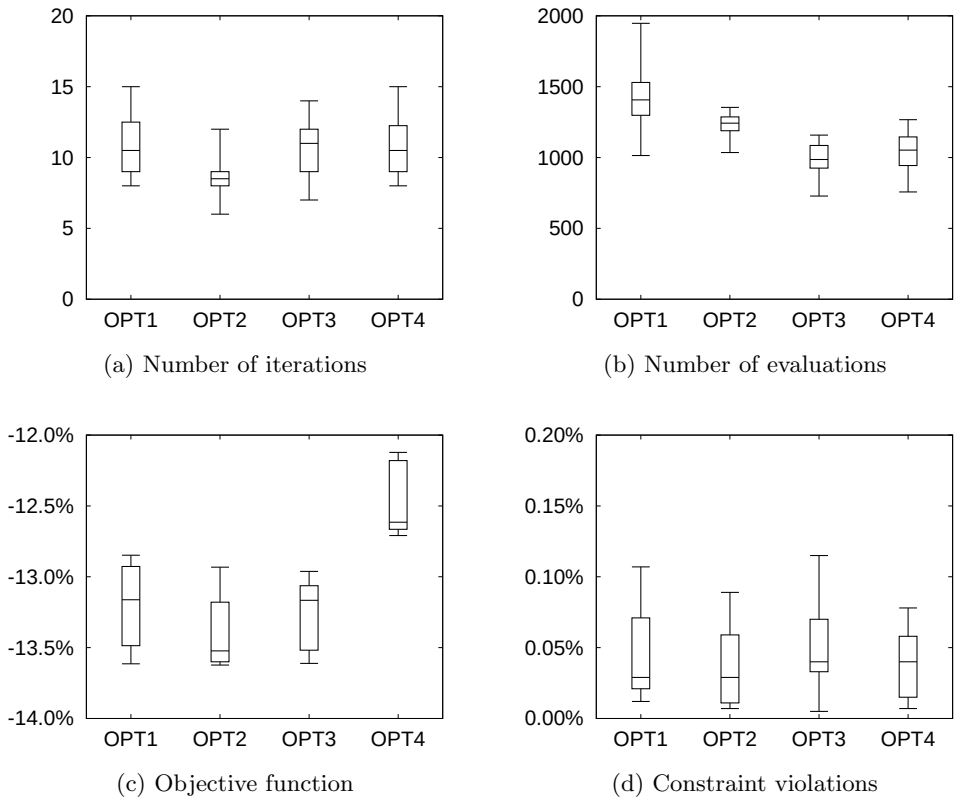
Optimisation 3 - Decreased budget

The third optimisation, which was carried out using sub-space approximations with individual allocation of the number of points per iteration for each discipline, finished on average in 11 iterations, 0.5 iterations more than the first optimisation, and 2.5 more than the second. However, having spent 986 evaluations, 421 less than the first

optimisation and 256 less than the second. The average reduction in objective function was 13.2%, just like in the first optimisation.

Optimisation 4 - Erroneous partitioning

The fourth optimisation, where a significant variable was deliberately identified as insignificant, finished on average after 10.5 iterations, having spent 1052 evaluations, with an objective reduction of 12.6%, 1% less than the first optimisation, but with equally low constraint violations as the first three optimisations.



(a) Number of iterations

(b) Number of evaluations

(c) Objective function

(d) Constraint violations

Figure 6.13: Statistical results from 50 runs with varying seed of each optimisation type, 1-4. OPT 1: Full space approximations, OPT 2: Sub space approx with full budget, OPT 3: Sub space approx with decreased budget, OPT 4: Sub space approx with erroneous partitioning.

6.3.5 Conclusions

It can be concluded from this test problem that, by using sub-space approximations for the presented example, it is possible to reduce the number of required iterations and/or the number of evaluations for carrying out the optimisation without compromising the results. By maintaining the number of points that are needed for full space optimisation when using sub-space approximations, as in optimisation 2, the number of iterations can be reduced. If instead the number of evaluations per iteration is individually allocated for each discipline, as in optimisation 3, a reduction in the number of evaluations can be achieved. It can also be concluded from the fourth optimisation that the proposed recovery mechanism makes sure that erroneous identification of significant variables does not lead to constraint violations, but is instead recovered to a cost for the objective function.

6.4 MDO of an aircraft wing subject to bird strike requirements

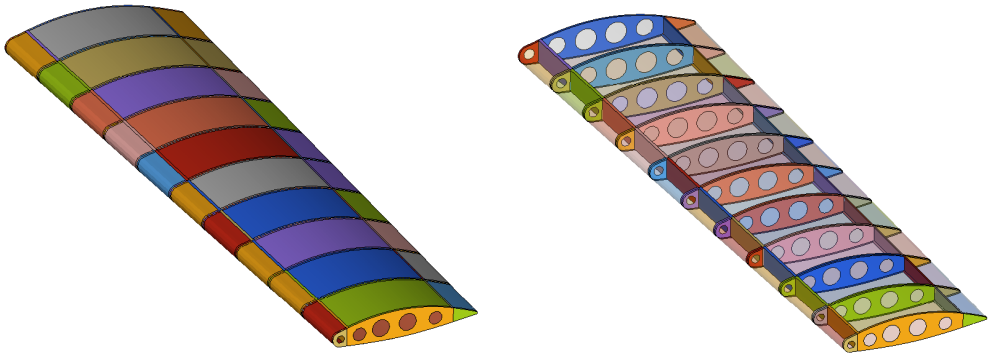
This case study, presented in Ollar *et al.* (2016a), outlines a multidisciplinary design optimisation of a wing structure subject to both strength and stiffness as well as bird strike requirements. In order to account for all critical locations of bird impact, 10 separate bird strike simulations are considered. Advantage is taken of the local nature of the bird impact with the use of sub-space metamodels. This enables the study to be performed at a much reduced computational cost than would otherwise be possible

6.4.1 Wing structure

The considered wing is a 3 m long aluminium structure with a root chord of 830 mm and tip chord of 670 mm. It has two longitudinal spars and 11 ribs as shown in Figure 6.14. The material is precipitation-hardened aluminium (6061-T6) with properties outlined in Table 6.3.

6.4.1.1 Strength and stiffness

The strength and stiffness requirements are evaluated using a linear static finite element model. The loading consist of forces and moments applied to a single point per rib, which is then distributed to the edges of the rib using one dimensional distributing elements as



(a) Complete structure. (b) Internal structure.

Figure 6.14: The wing model.

Table 6.3: Material characteristics for aluminium (6061-T6).

Property	Constant	Value	Unit
Material density	ρ	2.8	g/cm^3
Young's modulus	E	68.3	GPa
Poisson's ratio	E	0.33	–
Yield strength	σ_y	241.1	MPa
Ultimate tensile strength	σ_u	279.0	MPa

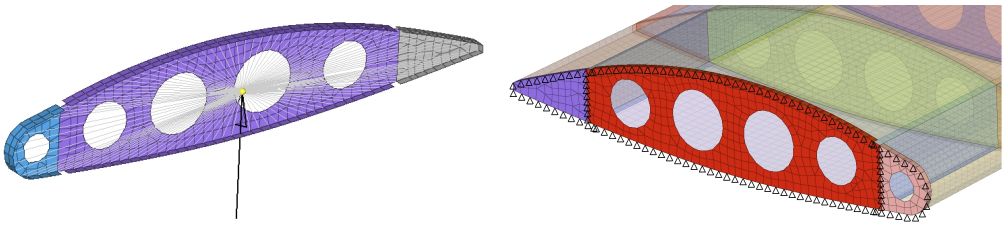
shown in Figure 6.15a. The wing is rigidly constrained at the fuselage end of the wing in degrees of freedom 1-3 of the nodes around the edges of the rib as shown in Figure 6.15b.

Two load cases, shown in Figure 6.16, are considered. In the first one the wing is bent upwards by applying forces at each of the previously discussed rib loading points. The displacement at the tip of the wing due to the loading is used as a response. In the second case the wing is twisted by applying moments at the rib loading points. The twist of the wing at the tip, due to the loading, is used as a response.

The analysis is carried out using Altair OptiStruct (Altair Engineering, Inc., 2014b) with the assumption of infinitesimal strain theory and an isotropic linear-elastic material model. Analytical gradients can be efficiently obtained using the adjoint method.

6.4.1.2 Bird Strike

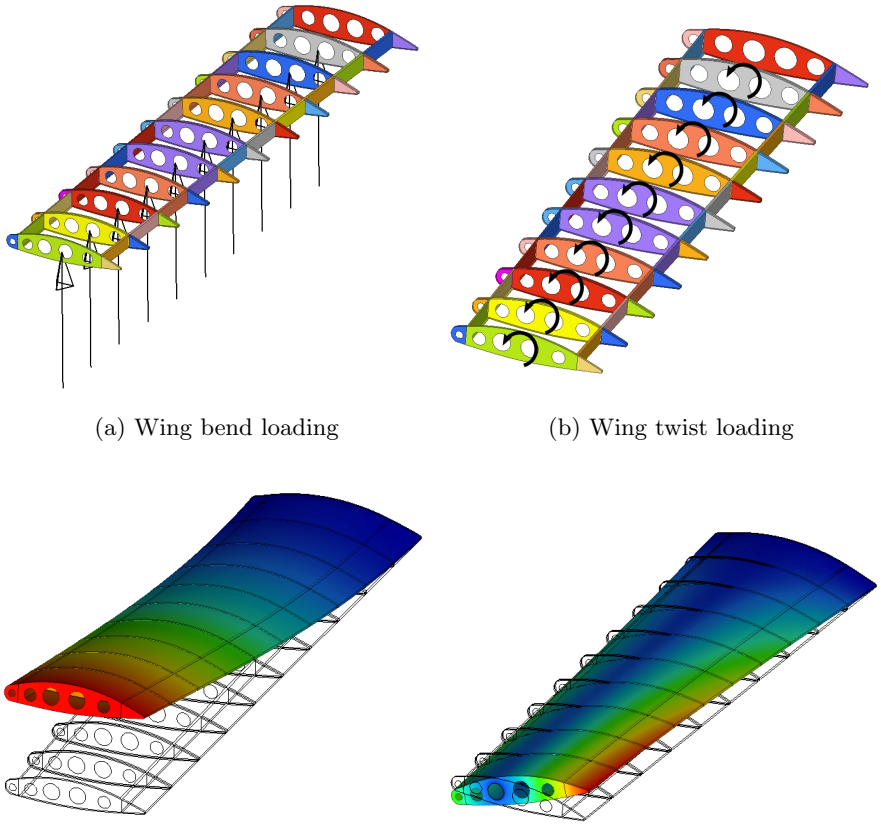
Bird strikes are high speed impact events and are thus evaluated using an explicit time-stepping scheme. In this work Altair RADIOSS (Altair Engineering, Inc., 2014c), an explicit finite element analysis software, is used.



(a) Forces/moments distributed using 1-D elements.

(b) Boundary conditions

Figure 6.15: Details on load application and boundary conditions.



(a) Wing bend loading

(b) Wing twist loading

(c) Wing bend results

(d) Wing twist results

Figure 6.16: Loading and results for the wing bend and twist cases.

Constitutive model

As the structure is expected to both yield and fail in certain areas, a suitable material model need to be used. The constitutive model used, shown in Figure 6.17, with parameters shown in Table 6.4, is elasto-plastic with isotropic hardening and failure known as /MAT/PLAS_TAB (Altair Engineering, Inc., 2014c). The work-hardening part of the curve is defined using tabular data of plastic strain versus stress shown in Table 6.5, and the failure criterion is defined as a constant rate decrease of stress from the point of maximum tensile failure strain, ϵ_u, until reaching zero stress at the point of maximum tensile failure damage, ϵ_m. The elements are deleted as they reach the tensile strain for element deletion, ϵ_d.

Table 6.4: Parameters of consitutive model for aluminium (6061-T6).

Property	Constant	Value	Unit
Material density	ρ	2.8	g/cm^3
Young's modulus	E	68.3	GPa
Max tensile failure strain	ϵ_u	0.08	–
Max tensile failure damage	ϵ_m	0.12	–
Tensile strain for element deletion	ϵ_d	0.13	–

Table 6.5: Tabular data for isotropic hardening of aluminium (6061-T6).

Plastic strain ϵ^{pl} –	Stress σ MPa
0.000	241.067
0.005	248.598
0.010	252.232
0.016	255.548
0.021	258.840
0.026	262.079
0.031	265.241
0.036	268.267
0.041	271.155
0.046	273.887
0.051	276.404
0.056	278.565
0.080	279.000

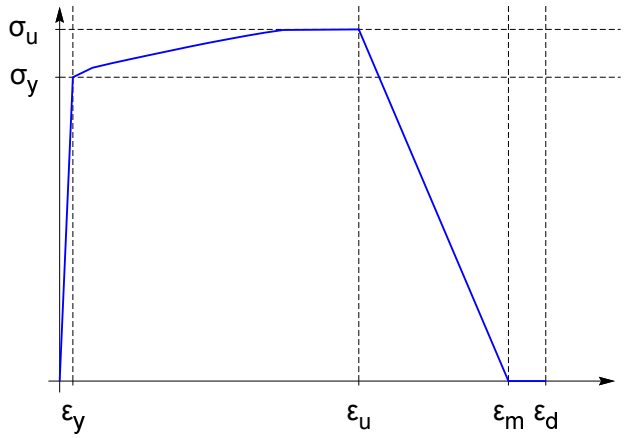


Figure 6.17: Elasto-plastic constitutive model with isotropic hardening and failure.

Bird model

The bird strike requirement is for a 4lb, or 1.81kg, bird impacting the leading edge of the wing at a speed of 150 m/s. The bird is modelled using smooth particle hydrodynamics (SPH) which is a meshless Lagrangian method based on interpolation theory. SPH is commonly used to model fluid structure interaction problems where the arbitrary Lagrangian Eulerian (ALE) formulation is expected to fail because of excessive mesh distortion. A bird exhibits fluid like behaviour at high impact speeds and can therefore be modelled realistically using the SPH formulation (Heimbs, 2011).

The bird model, developed by Altair RADIOSS (Altair Engineering, Inc., 2014c), has the shape of a cylinder with hemispherical ends as shown in Figure 6.18. The radius R is 57 mm which leads to a total volume of 1939 cm^3. The model contains 41544 cells weighing approximately 0.0437 g each, adding up to a total mass of 1.81 kg and an initial density of 0.935 g/cm^3. The average distance between neighbouring particles is 4.03 mm. The constitutive model is a polynomial equation of state (EOS), known as representing a hydrodynamic viscous fluid material defined as

$$P = C_1 \cdot \left(\frac{\rho}{\rho_0} - 1 \right). \tag{6.12}$$

P is the pressure, $C_1 = 2.106$ GPa a material constant (the bulk modulus), and ρ and

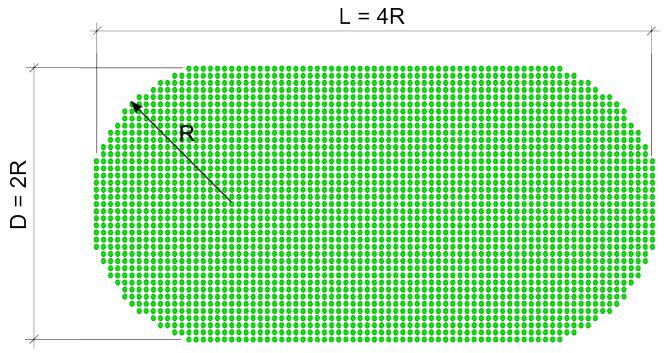


Figure 6.18: SPH bird model.

ρ_0 represents the current and initial density respectively. The properties for the SPH model are summarised in Table 6.6.

Load cases

As the impact location of the bird along the leading edge is arbitrary, several simulations need to be performed altering the impact location. To reduce the number of simulations needed, it is assumed that the critical location for bird impact is at the centre of each wing section, between the ribs. This means that in total 10 simulations, with varying start points of the bird, as shown in Figure 6.19, are to be carried out to assess the requirements for bird strike.

A bird strike simulation with start position 6 is shown in Figure 6.20. It shows that the bird impacts the leading edge skin which provides the initial energy absorption. The

Table 6.6: Parameters of consitutive model for SPH model.

Property	Constant	Value	Unit
Material density	ρ	0.935	g/cm^3
Bulk modulus	C_1	2.106	GPa
Particle mass	m_p	43.67	mg
Particle distance	h_p	4.03	mm
Number of particles	n_p	41544	-

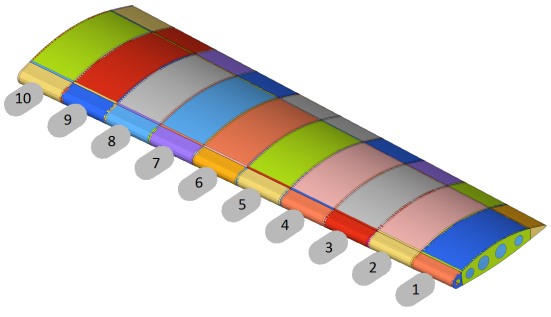
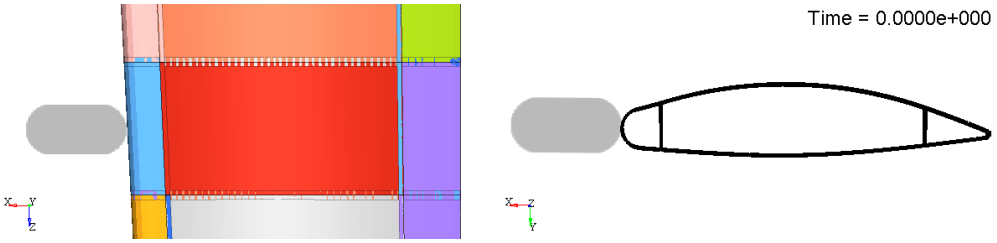
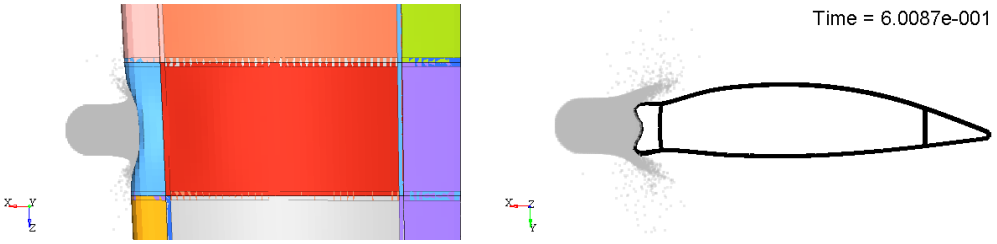


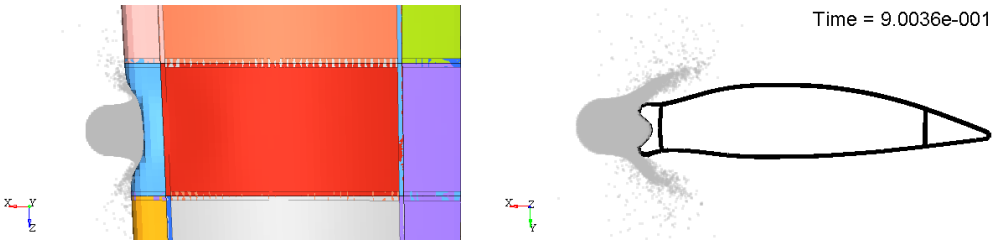
Figure 6.19: Critical impact locations along the leading edge.



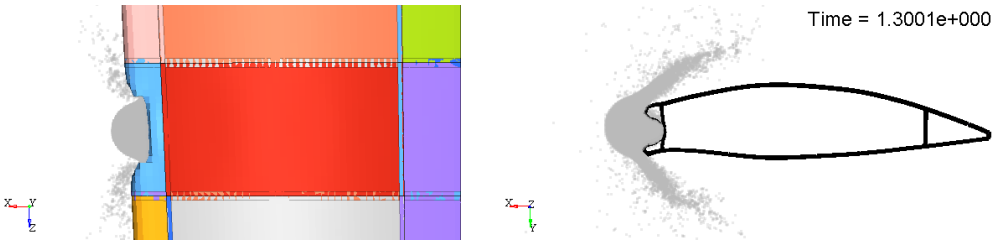
(a) t=0.0ms - Simulation start



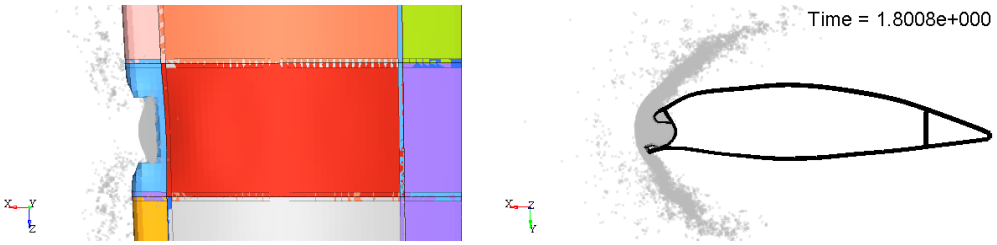
(b) t=0.6ms - Leading edge skin deformed



(c) t=0.9ms - Leading edge skin rupturing



(d) t=1.3ms - Leading edge skin contacts leading edge spar



(e) t=1.8ms - Maximum deformation leading edge spar

Figure 6.20: History of the bird strike simulation with starting position 6. Left side shows top view of the impact and right side shows a section at the point of impact.

leading edge skin ruptures and makes contact with the leading edge spar which deforms as a consequence. The requirements for the impact is that the structural integrity of the wing is not to be compromised. This is interpreted such that the leading edge skin is allowed to fail, however the leading edge spar must remain intact. In this study the magnitude of intrusion into the wing, measured at the location of impact as shown in Figure 6.21 are used as constraints for simplicity. Of course any type of response, suitable for optimisation, could be used.

6.4.2 Optimisation problem

The design variables are the thicknesses of the 100 components shown in Figure 6.22. The starting thickness for all components are 3 mm with a lower bound of 2 mm and upper bound of 5 mm. Note that the leftmost rib is not designable as in this study it is constrained by boundary conditions. The objective of the optimisation is to minimise the weight of the structure subject to the structural requirements outlined in the previous section and summarised in Table 6.7.

6.4.3 Optimisation procedure

The minimum number of points required by the MAM per iteration is set to the number of points needed for linear regression, $n + 1$, recalling that n is the number of design variables. It is set to this value regardless of how many points are needed by the chosen

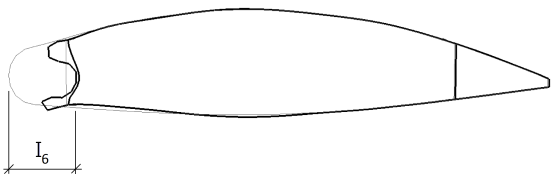
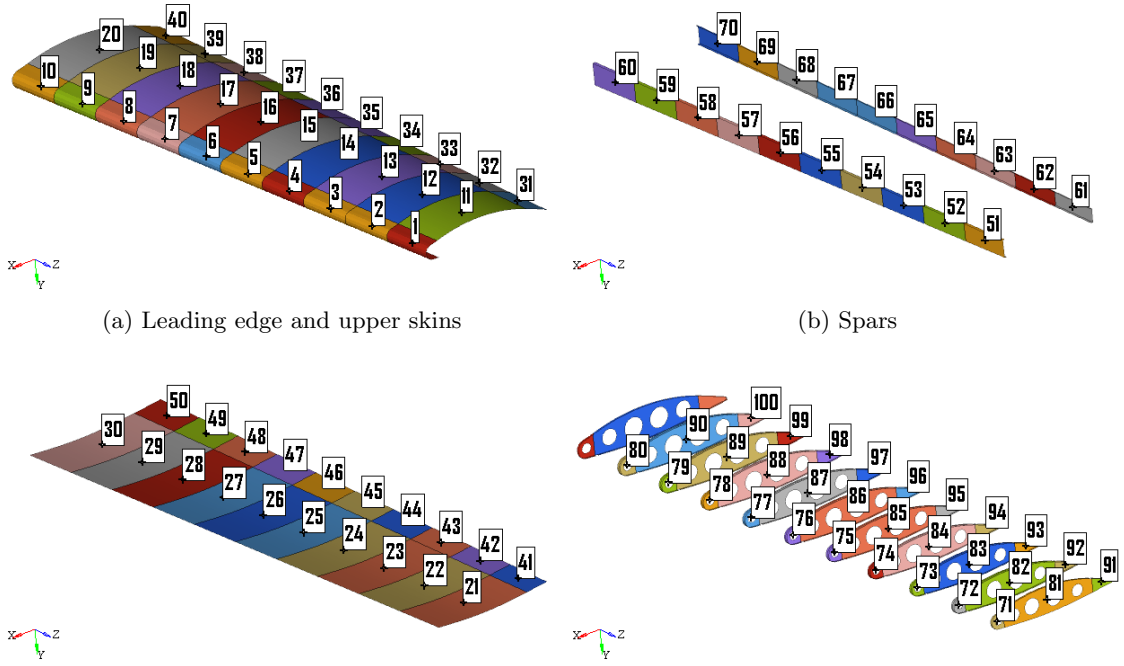


Figure 6.21: Maximum intrusion response.



(a) Leading edge and upper skins

(b) Spars

(c) Lower skins

(d) Ribs

Figure 6.22: Sizing design variables.

Table 6.7: Initial constraint violations, normalised to percentage exceeding target value.

Response	Init. constr. viol.
wing bend	11.9%
wing twist	0.5%
Intrusion 1	3.9%
Intrusion 2	3.9%
Intrusion 3	4.0%
Intrusion 4	4.0%
Intrusion 5	3.9%
Intrusion 6	4.1%
Intrusion 7	3.8%
Intrusion 8	4.2%
Intrusion 9	3.9%
Intrusion 10	4.1%

metamodel technique, in this case Kriging. It is useful to increase the number of points per iteration slightly in order to obtain better metamodels. The number of desired points per iteration is therefore chosen as $1.5n$. As the stiffness simulations have available gradients, gradient-enhanced metamodel building is used. This allows the number of points per iteration to be significantly reduced. Here the number of points required for simulations that have available gradients is chosen as $1.5n/\sqrt{n}$, resulting in 15 points per iteration. For the bird strike simulations no gradients are available which means that, for 100 design variables, the minimum number of points required is 101 while the desired number is 152. The total number of desired points for the 10 bird strike simulations would hence be 1520 points per iteration, a prohibitively large number.

In this problem sub-space metamodels can be used since the bird strike simulations have local design variable dependence. For each simulation, an assumption is made on which variables are significant to the response using engineering judgement. For each impact location of the bird, eight design variables, shown in Figure 6.23 are assumed to have most of the influence on the response. Other variables may have a slight influence, and could have been considered, but for the price of an increase in computational cost. Instead, any influence from other variables will be taken care of by the recovery

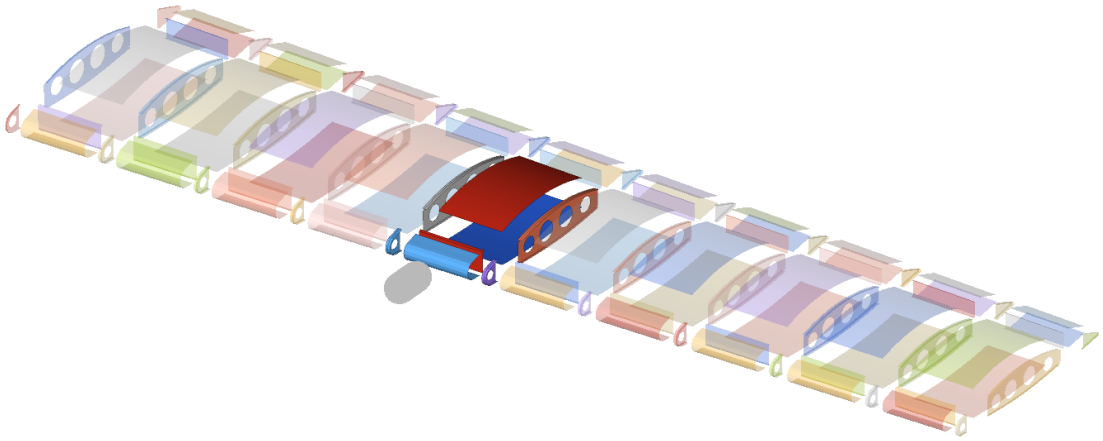


Figure 6.23: Assumed significant variables for bird at position 6.

mechanism outlined in Section 6.2.2. This leads to a minimum number of 9 points and a desired number of points of 12 points per simulation and iteration. For the load case where the bird impacts the leading edge skin adjacent to the rigidly constrained rib, there are only 6 significant variables which leads to a minimum of 7 points and a desired number of 9 points. In total, a minimum of 78 and a desired number of 117 bird strike simulations per iteration.

The number of variables, together with the corresponding minimum number of points and desired number of points is shown in Table 6.8. The minimum number of points are calculated as $n_{sgft} + 1$ and the number of desired points as $1.5n_{sgft}$. This leads to a minimum number of 9 points and a desired number of points of 12 points per simulation and iteration, except for the load case where only 6 design variables are considered, leading to a minimum of 7 points and a desired number of 9 points. In total 117 points for the bird strike requirements. However, as specified before a few extra points are added per iteration to improve accuracy and account for simulation failures.

Table 6.8: Number of points per iteration for each load case based on the number of signifiant variables.

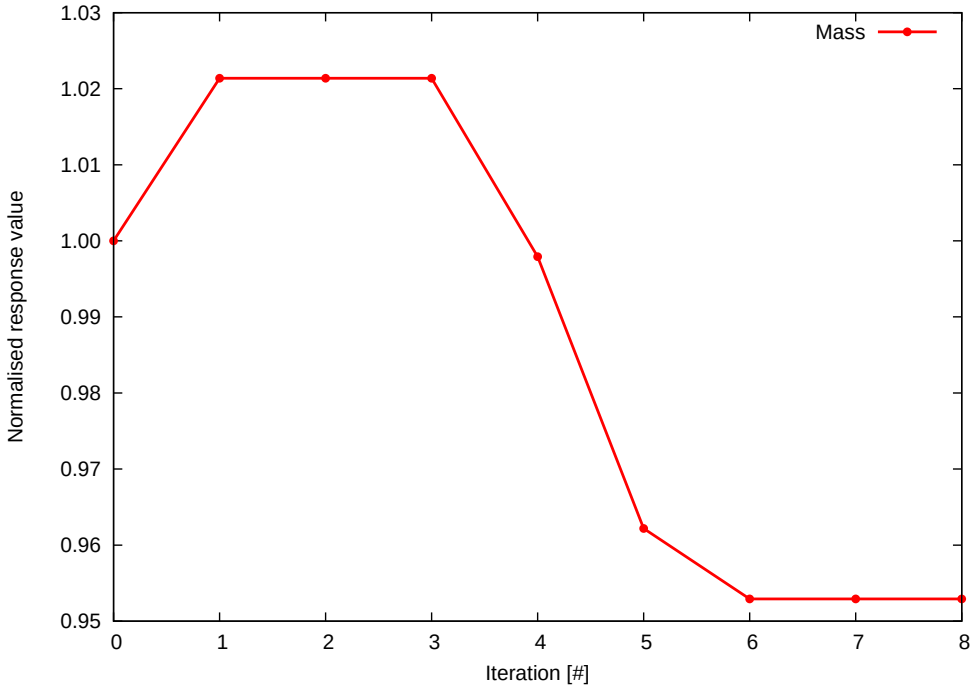
#	Load case	n_{sgft}	p_{min}	p_{opt}
1	Strength and stiffness	100	1	15
2	Bird strike 1	8	9	12
3	Bird strike 2	8	9	12
4	Bird strike 3	8	9	12
5	Bird strike 4	8	9	12
6	Bird strike 5	8	9	12
7	Bird strike 6	8	9	12
8	Bird strike 7	8	9	12
9	Bird strike 8	8	9	12
10	Bird strike 9	8	9	12
11	Bird strike 10	6	7	9

6.4.4 Results

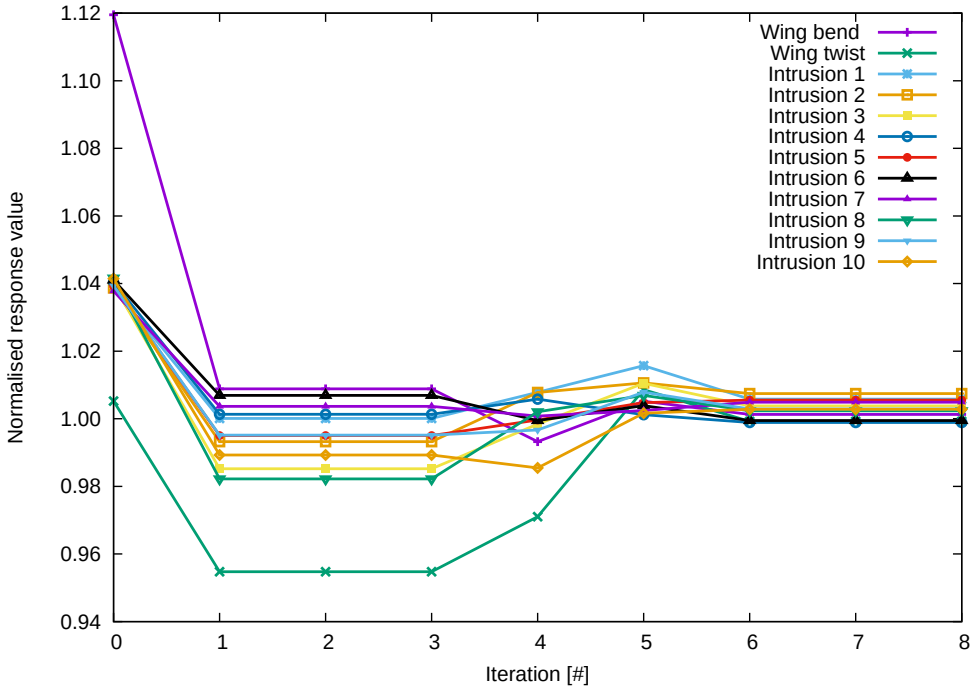
The history of the objective function and constraints during the optimisation is shown in shown in Figure 6.24. The optimisation required 8 iterations, 139 stiffness simulations, and 1276 bird strike simulations, in total for the 10 bird locations, less than would be required per iteration had sub-space metamodels not been used. The final mass is 4.7% less than the initial design and all constraint violations were reduced to less than 1%. The initial and final response values are shown in Table 6.9 and the final thickness distribution is shown in Figure 6.25. From the result it can be noted that none of the panels have gone to the upper thickness of 5 mm, but some to the lower one of 2 mm. Many of the ribs have a resulting thickness which is in the thinner part of the thickness range. This is most likely because of the very simplistic set of static requirement used for the optimisation. As can be expected, all leading edge skins have high thickness whilst leading edge ribs are thinner. This is most likely because the leading edge skin is more likely to rupture if the leading edge rib is less compliant.

Table 6.9: Results of the optimisations. Objective function is normalised to initial value and constraints are normalised to percentage exceeding target.

#	Response	Initial design	Final iteration
1	Mass		−4.7%
2	Wing bend	11.9%	0.1%
3	Wing twist	0.5%	0.0%
4	Intrusion 1	3.9%	0.6%
5	Intrusion 2	3.9%	0.7%
6	Intrusion 3	4.0%	0.4%
7	Intrusion 4	4.0%	0.0%
8	Intrusion 5	3.9%	0.6%
9	Intrusion 6	4.1%	0.0%
10	Intrusion 7	3.8%	0.5%
11	Intrusion 8	4.2%	0.2%
12	Intrusion 9	3.9%	0.3%
13	Intrusion 10	4.1%	0.3%



(a) Objective function.



(b) Constraint functions.

Figure 6.24: Optimisation history.

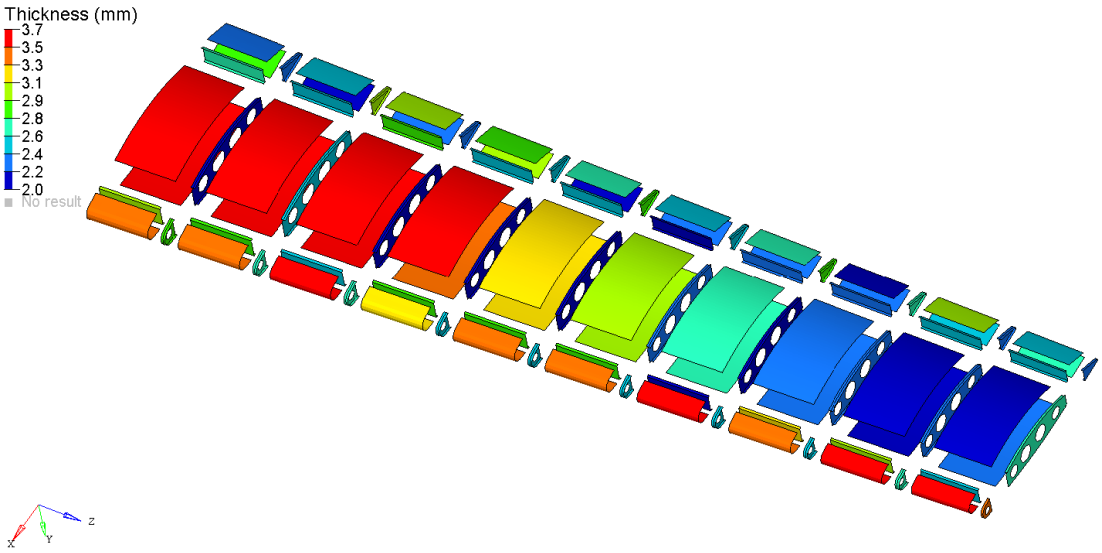


Figure 6.25: Final thickness for each of the considered components.

6.4.5 Conclusions

A multidisciplinary design optimisation of a wing structure was carried out. The considered load cases was static bending and twisting stiffness as well as bird strike requirement for impact at 10 locations. The computational cost of evaluation of the bird strike requirements is many times larger than the one of the static requirements. The optimisation was carried out using an approach previously proposed by the authors for solving MDO problems using metamodels built in individual sub-spaces of the design variable space. The approach uses existing knowledge of design variable dependence for each of the disciplines to decrease the number of required evaluations, and hence the related computational budget, in each iteration of a tust-region based optimisation procedure. The optimisation finished in 8 iterations having evaluated 139 stiffness simulations and 1276 bird strike simulations in total, less than would be required per iteration had sub-space metamodels not been used. The final result is a mass save of 4.7% and a reduction of all previously violated constraints to less than 1%.

6.5 Summary

An approach to carrying out multidisciplinary design optimisation using sub-space approximations has been proposed. Sub-space approximations are built in individual sub-spaces for each discipline while carrying out the optimisation in the full variable space has been proposed. The sub-spaces in which the approximations are built are defined by the sets of significant variables for the individual disciplines. The main benefit of the technique is the dimensionality reduction of the approximations and sampling. This enables reducing the computational budget required to obtain approximations of sufficient quality.

The method relies on the designer to make assumptions on which variables are significant for each response. If such assumptions are deficient, approximation errors can occur that cannot be reduced by additional sampling. Therefore a technique was proposed that can recover from such errors within a trust region based optimisation framework. By updating the values of the variables identified as insignificant, but remain present in full space, according to the best current solution in each iteration, the technique can recover from such errors.

The approach was demonstrated on two finite element examples. The first was an MDO of a thin-walled beam where a reduction in the computational budget was shown for optimisations carried out using the sub-space approximation approach compared to conventional optimisations. In addition, a test where a significant variable was deliberately identified as insignificant was carried out and showed the validity of the developed recovery mechanism.

The second example was an MDO of an aircraft wing subjected to bird strike at several locations along the leading edge as well as static stiffness requirements. By using sub-space approximations it was shown that less evaluations were required for the entire

optimisation than what would be required per iteration had sub-space metamodels not been used.

An automatic approach in which the sub-space partitioning is automatically determined at the start of each iteration using a design variable ranking algorithm has also been implemented and tested, however without showing a consistent computational gain.

Chapter 7

Summary, conclusions and recommendations for future work

This chapter concludes this work by summarising and discussing the achievements in Section 7.1 followed by suggestions on future work in Section 7.2.

7.1 Summary

The objective of this work was to propose and develop a metamodel-based multidisciplinary design optimisation framework suitable for incorporation of crash-worthiness or impact simulations, of which gradients may not be available. This objective was broken down into the following tasks:

1. Identify existing techniques with promising attributes for metamodel-based MDO.

2. Identify bottlenecks of existing techniques and suggest improvements.

3. Develop a metamodel-based MDO framework.

4. Propose methods to reduce the computational cost for MDO problems with a large number of design variables (>100).

5. Demonstrate the applicability of the developed framework on problems including impact or crashworthiness requirements.

This section outlines conclusions and discussions related to each fo the tasks.

Identify existing techniques with promising attributes for metamodel-based MDO

It was decided that the multidisciplinary feasible MDO architecture was to be used because it keeps the problem size to a minimum which is important for metamodel-based frameworks. Furthermore it ensures both multidisciplinary as well as interdisciplinary feasibility at each design point which is attractive for computationally expensive industrial applications, where the interest might be in design improvement within a reasonable time frame rather than a fully converged solution.

With the target of being able to handle problems with hundreds of variables without necessarily having access to analytical gradients meant that the choice of optimisation framework would have to fall on a local optimisation methods rather than design exploration approaches. The mid-range approximation method, which is a trust region-based framework, was chosen as a suitable framework as a starting point for the research.

A design of experiments technique known as modified extensible lattice sequences was chosen because of its ability to generate good quality space filling design of experiments while taking into account existing points. The moving least squares method was used for metamodelling in the initial stages of the research and was later replaced by kriging. Both techniques have the ability to use available gradients for building the metamodels

with the aim of increasing the quality of the fit for a given budget. In order to solve optimisation problems using the metamodels, the sequential quadratic programming method was used.

Identify bottlenecks of existing techniques and suggest improvements

Building kriging metamodels can be very costly because of the need to carry out an optimisation of a set of tuning parameters. A method for efficiently carrying out the required optimisation by using a two step approach consisting of a line search followed by a gradient based optimisation was suggested. Partial derivatives was efficiently obtained using the adjoint method. The approach was tested on several analytical examples and on a industry size test problem.

Develop a metamodel-based MDO framework

As a starting point for the MDO framework, the modified extensible lattice sequences method and kriging was added to the mid-range approximation method framework which already had an implementation of the moving least squares method.

It was suggested that the new MDO framework should handle each discipline individually and take advantage of any disparities in attributes of the disciplines. As a consequence individual design of experiments for each discipline was introduced which allows for different number of points required per iteration for the individual disciplines. It allows exploiting gradients if available for individual disciplines. Finally, simulation failure would previously lead to discarding a sample point across all disciplines to keep a consistent set of points. With individual sampling it would only have to be discarded from the related discipline.

Propose methods to reduce the computational cost for MDO problems with a large number of design variables (>100)

The main research effort was spent on a method for efficiently incorporating crashworthiness or impact requirements in the multidisciplinary design optimisation process using the MDO framework. It was identified that many crashworthiness and impact requirements are somewhat local in terms of the design variable dependence and that it can be used to an advantage.

The computational cost of building a metamodel is directly related to dimensionality of the design variable space in which the metamodel is built. Hence if one can reduce the number of design variables the computational cost can be reduced. By identifying the design variables which has most of the influence on each discipline, so called sub-space metamodels, can be built in the space of only those variables, hence saving computational budget. However, if the design variable dependence is misjudged and significant variables are not identified, this results in metamodel errors that cannot be resolved by additional sampling.

In this work sub-space metamodels was introduced within the mid-range approximation method, together with a mechanism that recovers from such metamodelling errors. This is done at the end of each iteration by updating the values of the insignificant variables such that the starting point for the next iteration will include any changes in the response function that cannot be taken into account by the sub-space metamodels. This results in a robust technique that allows reduction in computational cost of multidisciplinary design optimisation problems with disparate design variable dependence of the individual disciplines.

Demonstrate the applicability of the developed framework on problems including impact or crashworthiness requirements

The proposed techniques were demonstrated on a multidisciplinary design optimisation benchmark example of a thin-walled beam structure and on an optimisation example of an aircraft wing subject to bird strike as well as strength and stiffness requirements. It was shown that by using sub-space metamodels the computational budget of the optimisation problem could be greatly reduced. The results presented indicates that the use of sub-space metamodels is advantageous for industr y size multidisciplinary design optimisation problems with disparate design variable dependence, such as in the example.

7.2 Recommendations for future work

As the current work is concluded there are many potential paths for further research that would be interesting to pursue. The first is further testing of the framework by adding additional disciplines, such as fluid dynamics and electromagnetics and to test the framework on industrial applications in the automotive and aerospace industry. Further more, it would be interesting to investigate the applicability to other industries.

In terms of further development of the framework there are several potential areas of future research. Development of an extension to the current sub-space metamodels framework that can automatically determine design variable dependence of each discipline during optimisation would be interesting. Such a technique would need to efficiently estimate the set of significant design variables of each discipline without consuming the reduction in computational effort from building sub-space metamodels. It was briefly tested during the research but without observing a reduction in overall computational effort of the optimisation problem.

The framework would benefit from being tested on disciplines that includes multidisciplinary analysis, also known as multi-physics simulations. How to efficiently handle multidisciplinary analysis within the framework could be an area of future research. It may be of interest to revisit architectures such as IDF to research the applicability of sub-space metamodels.

Furthermore, an interesting area of research is multi-fidelity optimisation, which would fit well within the optimisation framework and could be researched in terms of synergies with the sub-space metamodels.

Appendix A

Partial derivatives of the correlation matrix for gradient enhanced kriging

This section explains the derivation of the derivatives of the covariance matrix with respect to the hyper parameters for the quadrants containing information relating to the design sensitivities.

Quadrant $Q^{1,2}$ and $Q^{2,1}$

The derivation for the derivatives of quadrant $Q^{1,2}$ that contains the covariance between the design sensitivities and the design function evaluations are shown below:

$$\frac{\partial Q^{1,2}_{i,j\times k}}{\partial \theta_m} = \frac{\partial (2\theta_k (x_i^k - x_j^k)\psi(\mathbf{x}_i, \mathbf{x}_j))}{\partial \theta_m} \tag{A.1}$$

- For the case $k = m$:

$$\begin{aligned}
&\frac{\partial(2\theta_m \cdot (x_i^m - x_j^m) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j))}{\partial \theta_m} \\
&= 2(x_i^m - x_j^m) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j) + 2\theta_m \cdot (x_i^m - x_j^m) \cdot (-(x_i^m - x_j^m)^2) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j) \\
&= \underbrace{2\theta_m (x_i^m - x_j^m) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j)}_{Q_{i,j\cdot k}^{1,2}} \left[\frac{1}{\theta_m} - (x_i^m - x_j^m)^2 \right] \\
&= \left[\frac{1}{\theta_m} - (x_i^m - x_j^m)^2 \right] \cdot Q_{i,j\cdot k}^{1,2}
\end{aligned} \tag{A.2}$$

- For the case $k \neq m$:

$$\begin{aligned}
&\frac{\partial(2\theta_k \cdot (x_i^k - x_j^k) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j))}{\partial \theta_m} \\
&= 2\theta_k \cdot (x_i^k - x_j^k) \cdot (-(x_i^m - x_j^m)^2) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j) \\
&= \underbrace{2\theta_k (x_i^k - x_j^k) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j)}_{Q_{i,j\cdot k}^{1,2}} \left[-(x_i^m - x_j^m)^2 \right] \\
&= \left[-(x_i^m - x_j^m)^2 \right] \cdot Q_{i,j\cdot k}^{1,2}
\end{aligned} \tag{A.3}$$

Quadrant $Q^{2,2}$

The derivation of the derivatives of quadrant $Q^{2,2}$ that contains the covariance between the design sensitivities are shown below.

$$\frac{\partial Q^{2,2}_{i\cdot l,j\cdot k}}{\partial \theta_m} = \begin{cases} \dfrac{\partial(2\theta_k(-2\theta_k \cdot (x_i^k - x_j^k)^2 + 1) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j))}{\partial \theta_m} & , k = l \\ \dfrac{\partial(-4\theta_k\theta_l \left[(x_i^k - x_j^k)(x_i^l - x_j^l)\right] \psi(\mathbf{x}_i, \mathbf{x}_j)}{\partial \theta_m} & , k \neq l \end{cases} \tag{A.4}$$

- For the case $k = l = m$

$$\begin{aligned} &\frac{\partial(2\theta_m(-2\theta_m \cdot (x_i^m - x_j^m)^2 + 1) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j))}{\partial \theta_m} \\ &= (4\theta_m^2(x_i^m - x_j^m)^4 - 10\theta_m(x_i^m - x_j^m)^2 + 2) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j) \\ &= (x_i^m - x_j^m)^2 \underbrace{(-2\theta_m(2\theta_m(x_i^m - x_j^m)^2 - 1)) \cdot \psi(\mathbf{x}_i, \mathbf{x}_j)}_{Q^{2,2}_{i\cdot l,j\cdot k}} \\ &+ (-8\theta_m(x_i^m - x_j^m)^2 + 2) \underbrace{\psi(\mathbf{x}_i, \mathbf{x}_j)}_{R_{i,j}} \\ &= \left[2 - 8\theta_k(x_i^m - x_j^m)^2\right] \cdot R_{i,j} - (x_i^m - x_j^m)^2 \cdot Q^{2,2}_{i\cdot l,j\cdot k} \end{aligned} \tag{A.5}$$

- For the case $k \neq m, l \neq m$:

$$\begin{aligned}
&\frac{\partial(-4\theta_k\theta_l\left[(x_i^k - x_j^k)(x_i^l - x_j^l)\right]\psi(\mathbf{x}_i, \mathbf{x}_j)}{\partial\theta_m} \\
&= (x_i^m - x_j^m)^2 \underbrace{(4\theta_k\theta_l\left[(x_i^k - x_j^k)(x_i^l - x_j^l)\right]\psi(\mathbf{x}_i, \mathbf{x}_j))}_{-Q^{2,2}_{i\cdot l, j\cdot k}} \\
&= -(x_i^m - x_j^m)^2 \cdot Q^{2,2}_{i\cdot l, j\cdot k}
\end{aligned} \tag{A.6}$$

- For the case $k = m, l \neq m$:

$$\begin{aligned}
&\frac{\partial(-4\theta_k\theta_l\left[(x_i^k - x_j^k)(x_i^l - x_j^l)\right]\psi(\mathbf{x}_i, \mathbf{x}_j)}{\partial\theta_m} \\
&= \Bigg[\frac{1}{\theta_k}\underbrace{(-4\theta_k\theta_l\left((x_i^k - x_j^k)(x_i^l - x_j^l)\right)}_{Q^{2,2}_{i\cdot l, j\cdot k}/\psi(\mathbf{x}_i, \mathbf{x}_j)} \\
&+(x_i^m - x_j^m)^2 \underbrace{4\theta_k\theta_l\left((x_i^k - x_j^k)(x_i^l - x_j^l)\right)}_{-Q^{2,2}_{i\cdot l, j\cdot k}/\psi(\mathbf{x}_i, \mathbf{x}_j)}\Bigg]\psi(\mathbf{x}_i, \mathbf{x}_j) \\
&= \left[\frac{1}{\theta_k} - (x_i^m - x_j^m)^2\right] \cdot Q^{2,2}_{i\cdot l, j\cdot k}
\end{aligned} \tag{A.7}$$

- The case $k \neq m, l = m$ may be derived in the same manner as the previous case, leading to:

$$\begin{aligned}
&\frac{\partial(-4\theta_k\theta_l\left[(x_i^k - x_j^k)(x_i^l - x_j^l)\right]\psi(\mathbf{x}_i, \mathbf{x}_j)}{\partial\theta_m} \\
&= \left[\frac{1}{\theta_l} - (x_i^m - x_j^m)^2\right] \cdot Q^{2,2}_{i\cdot l, j\cdot k}
\end{aligned} \tag{A.8}$$

Appendix B

Author's publications

Journal papers - Published

- **Ollar J**, Mortished C, Jones R, Sienz J, Toropov V (2016) Gradient based hyper parameter optimisation for well conditioned kriging metamodels. Structural and Multidisciplinary Optimization.
- **Ollar J**, Toropov V, Jones R (2016) Sub-space approximations for MDO problems with disparate disciplinary variable dependence. Structural and Multidisciplinary Optimization.

Conference papers - Proceedings

- Schlaps R, **Ollar J**, Shahpar S, Toropov V (2016) Multi-disciplinary optimization of compressor rotor subjected to ice impact. 11th ASMO-UK/ISSMO International Conference on Engineering Design Optimisation, Munich, Germany, July 18-20, 2016.

- **Ollar J**, Jones R, Toropov V (2016) Incorporation of bird strike requirements in MDO of an aircraft wing using subspace approximations. 11th ASMO-UK/ISSMO International Conference on Engineering Design Optimisation, Munich, Germany, July 18-20, 2016.

- Mortished C, **Ollar J**, Jones R, Benzie P, Toropov V, Sienz J (2016) Aircraft wing optimisation based on computationally efficient gradient-enhanced kriging. 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, San Diego, CA, USA, January 4-8, 2016.

- **Ollar J**, Toropov V, Jones R (2015) Adaptive sub-space approximations in trust-regions for large scale MDO problems. 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Kissimmee, FL, USA, January 5-9, 2015.

- **Ollar J**, Toropov V, Jones R (2014) Mid-range approximations in sub-spaces for MDO problems with disparate discipline attributes. 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, GA, USA, July 16-20, 2014.

Bibliography

Altair Engineering, Inc (2014a) HyperStudy 13.0 User's Guide. Altair Engineering, Inc.

Altair Engineering, Inc (2014b) OptiStruct 13.0 User's Guide. Altair Engineering, Inc.

Altair Engineering, Inc (2014c) RADIOSS 13.0 User's Guide. Altair Engineering, Inc.

Audze P, Eglajs V (1977) New approach for planning out of experiments (in Russian), vol 35. Problems of Dynamics and Strengths

Barthelemy JFM, Haftka RT (1993) Approximation concepts for optimum structural design — a review. Structural optimization 5(3):129–144

Bates S, Sienz J, Langley D (2003) Formulation of the audzeeglais uniform latin hypercube design of experiments. Advances in Engineering Software 34(8):493 – 506

Box GE, Behnken DW (1960) Some new three level designs for the study of quantitative variables. Technometrics 2(4):455–475

Box GE, Draper NR (1987) Empirical model-building and response surfaces, vol 424. Wiley New York

Box GEP, Wilson KB (1951) On the experimental attainment of optimum conditions. Journal of the Royal Statistical Society Series B (Methodological) 13(1):1–45

Broomhead DS, Lowe D (1988) Multi-variable functional interpolation and adaptive networks. Tech. rep., DTIC Document

Broyden CG (1970) The convergence of a class of double-rank minimization algorithms 1. general considerations. IMA Journal of Applied Mathematics 6(1):76–90

Choi K, Youn BD, Yang RJ (2001) Moving least square method for reliability-based design optimization. 4th world congress of structural and multidisciplinary optimization

Chung HS, Alonso JJ (2002) Using gradients to construct cokriging approximation models for high-dimensional design optimization problems. AIAA paper 317:14–17

Cramer EJ, Dennis JE, Jr, Frank PD, Lewis RM, Shubin GR (1993) Problem formulation for multidisciplinary optimization. SIAM journal on optimization 4:754–776

Dalbey KR (2013) Efficient and robust gradient enhanced kriging emulators. Technical Report SAND2013-7022, Sandia National Laboratories

Dwyer PS, MacPhail M (1948) Symbolic matrix derivatives. The annals of mathematical statistics pp 517–534

Fisher SRA (1960) The design of experiments, vol 12. Oliver and Boyd Edinburgh

Fletcher R (1970) A new approach to variable metric algorithms. The Computer Journal 13(3):317–322

Forrester A, Sobester A, Keane A (2008) Engineering design via surrogate modelling: a practical guide. Wiley

Forrester AI, Keane AJ (2009) Recent advances in surrogate-based optimization. Progress in Aerospace Sciences 45(13):50 – 79

Forrester AIJ, Keane AJ, Bressloff NW (2006) Design and analysis of "noisy" computer experiments. AIAA Journal 44(10):2331–2339

Giles M (2008) An extended collection of matrix derivative results for forward and reverse mode automatic differentiation. Tech. Rep. 08/01, Oxford University Computing Laboratory

Goldfarb D (1970) A family of variable-metric methods derived by variational means. Mathematics of Computation 24(109):23–26

Grove D, Davis T (1992) Engineering, quality, and experimental design. Longman Scientific & Technical

Haaland B, Qian PZ, *et al.* (2011) Accurate emulators for large-scale computer experiments. The Annals of Statistics 39(6):2974–3002

Haftka R (1985) Simultaneous analysis and design. AIAA Journal 23:1099–1103

Haftka R, Gurdal Z (1992) Elements of Structural Optimization. Springer

Halton JH (1964) Algorithm 247: Radical-inverse quasi-random point sequence. Communications of the ACM 7(12):701–702

Hammersley JM, Handscomb D (1964) Monte Carlo Methods. Springer

Han ZH, Görtz S, Zimmermann R (2013) Improving variable-fidelity surrogate modeling via gradient-enhanced kriging and a generalized hybrid bridge function. Aerospace Science and Technology 25:177–289

Heimbs S (2011) Computational methods for bird strike simulations: A review. Computers & Structures 89(2324):2093 – 2112

Hickernell FJ, Hong HS, LÉcuyer P, Lemieux C (2000) Extensible lattice sequences for

quasi-Monte Carlo quadrature. SIAM J Sci Comput 22(3):1117–1138

Intel (2015) Reference Manual for Intel Math Kernel Library 11.2. Intel

Jamil M, Yang XS (2013) A literature survey of benchmark functions for global optimisation problems. International Journal of Mathematical Modelling and Numerical Optimisation 4(2):150–194

Jones DR (2001) A taxonomy of global optimization methods based on response surfaces. Journal of Global Optimization 21(4):345–383

Karush W (1939) Minima of Functions of Several Variables with Inequalities as Side Constraints. Master's thesis, Dept. of Mathematics, Univ. of Chicago

van Keulen F, Toropov V, Markine V (1996) Recent refinements in the multi-point approximation method in conjunction with adaptive mesh refinement. ASME Design Engineering Technical Conferences and Computers in Engineering Conference, 18-22 August 1996, Irvine, CA

Kianifar MR, Campean F, Wood A (2016) Application of permutation genetic algorithm for sequential model building–model validation design of experiments. Soft Computing 20(8):3023–3044

Kodiyalam S, Yang R, Gu L, Tho CH (2004) Multidisciplinary design optimization of a vehicle system in a scalable, high performance computing environment. Structural and Multidisciplinary Optimization 26(3-4):256–263

Korolev Y, Toropov V, Shahpar S (2015) Large-scale CFD optimization based on the FFD parametrization using the multipoint approximation method in an HPC environment. 16th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, AIAA Aviation, 22-26 June 2015, Dallas, TX

Krige D (1951) A statistical approach to some mine valuation and allied problems on the witwatersrand. Master's thesis, University of Witwatersrand

Kuhn HW, Tucker AW (1951) Nonlinear programming. In: Proceedings of the Second Berkeley Symposium on Mathematical Statistics and Probability, University of California Press, Berkeley, Calif., pp 481–492

Lancaster P, Salkauskas K (1981) Surfaces generated by moving least squares methods. Mathematics of Computation 37(155):141–158

Liszka T (1984) An interpolation method for an irregular net of nodes. International Journal for Numerical Methods in Engineering 20:1599–1612

Liu W, Batill S (2002) Gradient-enhanced response surface approximations using kriging models. In: 9th AIAA/ISSMO Symposium on Multidisciplinary Analysis and Optimization, American Institute of Aeronautics and Astronautics, Atlanta, Georgia

Lockwood BA, Anitescu M (2010) Gradient-enhanced universal kriging for uncertainty propagation. Preprint ANL/MCS-P1833-0111

Lukaczyk T, Taylor T, Palacios F, Alonso J (2013) Managing gradient inaccuracies while enhancing optimal shape design methods. Proceedings of the 51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition

M D McKay WJC R J Beckman (1979) A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. Technometrics 21(2):239–245

Madsen K, Nielsen HB, Sondergaard J (2002) Robust subroutines for non-linear optimization. University of Denmark IMM-REP-2002-02

Martins JRRA, Hwang JT (2013) Review and unification of methods for computing

derivatives of multidisciplinary computational models. AIAA Journal 51(11):2582–2599

Martins JRRA, Lambe AB (2013) Multidisciplinary design optimization: A survey of architectures. AIAA Journal 51(9):2049–2075

Matheron G (1963) Principles of geostatistics. Economic Geology 58:1246–1266

Mortished C, Ollar J, Jones R, Benzie P, Toropov V, Sienz J (2016) Aircraft wing optimization based on computationally efficient gradient-enhanced ordinary kriging metamodel building. 57th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, San Diego, CA, USA, January 4-8, 2016

Ollar J, Toropov V, Jones R (2014) Mid-range approximations in sub-spaces for MDO problems with disparate discipline attributes. 15th AIAA/ISSMO Multidisciplinary Analysis and Optimization Conference, Atlanta, GA, USA, July 16-20, 2014

Ollar J, Toropov V, Jones R (2015) Adaptive sub-space approximations in trust-regions for large scale MDO problems. 56th AIAA/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference, Kissimmee, FL, USA, January 5-9, 2015

Ollar J, Jones R, Toropov V (2016a) Incorporation of bird strike requirements in MDO of an aircraft wing using subspace approximations. 11th ASMO-UK/ISSMO International Conference on Engineering Design Optimisation, Munich, Germany, July 18-20, 2016

Ollar J, Mortished C, Jones R, Sienz J, Toropov V (2016b) Gradient based hyper-parameter optimisation for well conditioned kriging metamodels. Structural and Multidisciplinary Optimization

Ollar J, Toropov V, Jones R (2016c) Sub-space approximations for MDO problems

with disparate disciplinary variable dependence. Structural and Multidisciplinary Optimization

Plackett RL, Burman JP (1946) The design of optimum multifactorial experiments. Biometrika 33(4):305–325

Polynkin A, Toropov V (2012) Mid-range metamodel assembly building based on linear regression for large scale optimization problems. Structural and Multidisciplinary Optimization 45(4):515–527

Powell MJD (1978) Numerical Analysis: Proceedings of the Biennial Conference Held at Dundee, June 28–July 1, 1977, Springer Berlin Heidelberg, Berlin, Heidelberg, chap A fast algorithm for nonlinearly constrained optimization calculations, pp 144–157

Rennen G (2008) Subset selection from large datasets for kriging modeling. Structural and Multidisciplinary Optimization 38(6):545–569

Rojas R (2013) Neural networks: a systematic introduction. Springer Science & Business Media

Ryberg AB, Bckryd R, Nilsson L (2015) A metamodel-based multidisciplinary design optimization process for automotive structures. Engineering with Computers 31(4):711–728

Sacks J, Welch WJ, Mitchell TJ, Wynn HP (1989) Design and analysis of computer experiments. Statistical Science 4(4):409–423

Schölkopf B, Smola A (2002) Support Vector Machines and Kernel Algorithms. The Handbook of Brain Theory and Neural Networks, MIT Press, Cambridge, UK

Shanno DF (1970) Conditioning of quasi-newton methods for function minimization. Mathematics of computation 24(111):647–656

Sobieszczanski-Sobieski J, Haftka R (1997) Multidisciplinary aerospace design optimization: survey of recent developments. Structural optimization 14(1):1–23

Sobieszczanski-Sobieski J, Kodiyalam S, Yang R (2001) Optimization of car body under constraints of noise, vibration, and harshness (NVH), and crash. Structural and Multidisciplinary Optimization 22(4):295–306

Sobol' I (1967) On the distribution of points in a cube and the approximate evaluation of integrals. USSR Computational Mathematics and Mathematical Physics 7(4):86 – 112

Toal DJJ, Bressloff NW, Keane AJ, Holden CME (2011) The development of a hybridized particle swarm for krign hyperparameter tuning. Engineering Optimization 43(6):1–28

Toropov V (1989) Simulation approach to structural optimization. Structural Optimization 1(1):37–46

Toropov V (1992) Multipoint approximation method in optimization problems with expensive function values. In: Computational systems analysis 1992: Proceedings of the 4th International Symposium on Systems Analysis and Simulation. v. 4, Elsevier

Toropov V, Filatov A, Polynkin A (1993) Multiparameter structural optimization using FEM and multipoint explicit approximations. Structural optimization 6(1):7–14

Toropov V, van Keulen F, Markine V, de Boer H (1996) Refinements in the multipoint approximation method to reduce the effects of noisy structural responses. 6th Symposium on Multidisciplinary Analysis and Optimization, September 4-6, 1996, Bellevue, WA

Toropov V, Markine V, Holden C (1999) Use of mid-range approximations for optimization problems with functions of domain-dependent calculability. 3rd

ISSMO/UBCAD/UB/AIAA World Congress of Structural and Multidisciplinary Optimization, Buffalo, NY, USA, May 17-21, 1999

Tu J, Jones D (2003) Variable screening in metamodel design by cross-validated moving least squares method. 44th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics, and Materials Conference DOI 10.2514/6.2003-1669

Vanderplaats GN (1973) Conmin - a fortran program for constrained function minimization - user's manual. Technical Memorandum TM X-62,282, NASA

Vapnik VN, Vapnik V (1998) Statistical learning theory, vol 1. Wiley New York

Viana FA, Gogu C, Haftka RT (2010) Making the most out of surrogate models: tricks of the trade. In: ASME 2010 International Design Engineering Technical Conferences and Computers and Information in Engineering Conference, American Society of Mechanical Engineers, pp 587–598

Viana FAC, Simpson TW, Balabanov V, Toropov V (2014) Metamodeling in multidisciplinary design optimization: How far have we really come? Progress in Aerospace Sciences 52(4):670 – 690

Wang GG, Shan S (2007) Review of metamodeling techniques in support of engineering design optimization. Journal of Mechanical design 129(4):370–380

Xiong Y, Chen W, Apley D, Ding X (2007) A non-stationary covariance-based kriging method for metamodelling in engineering design. International Journal for Numerical Methods in Engineering 71:733756

Zimmermann R (2013) On the maximum likelihood training of gradient-enhanced spatial gaussian processes. SIAM Journal on Scientific Computing 35(6):2554–2574

Zimmermann R (2015) On the condition number anomaly of gaussian correlation

matrices. Linear Algebra and its Applications 466:512 – 526

Zoutendijk G (1960) Methods of feasible directions. a study in linear and non-linear programming.