Solving optimisation problems in metal forming using FEM: A metamodel based optimisation algorithm

M.H.A. Bonte

9th August 2005

Preface

This document is part of NIMR project MC1.03162, Optimisation of Forming Processes [96]. The aim of this report is to describe an initial concept for an optimisation algorithm and its theoretical background for solving optimisation problems in metal forming processes using expensive nonlinear Finite Element calculations. The applicability of the optimisation algorithm is demonstrated by means of optimising a simple hydroforming example and two more complex 3D forging processes.
The Appendices A and B of the report contain fairly extensive descriptions of two metamodelling techniques, Response Surface Methodology (RSM) and Kriging (DACE). Not everything presented in these appendices is necessary material for the reader or the user of OPTFORM, the MATLAB® software containing the optimisation algorithm described in this report. It presents, however, a very useful overview for those who would like to know more about the theory underlying the algorithm or about RSM, Kriging or Design Of Experiments (DOE) in general. RSM and DOE can also be of high interest to people involved in physical experimentation.
The extension of the optimisation algorithm with effective sequential improvement strategies (Section 4.2.6) and its application to forging (Chapter 6) were performed during an internship at the Ecole Nationale Supérieure des Mines de Paris (ENSMP), department Centre de Mise en Forme des Matériaux (CEMEF) in Sophia-Antipolis, France. I would like to thank profs. Huétink (UT) and Chenot (CEMEF) for supporting my internship and dr. Lionel Fourment and Tien Tho Do for the very useful and interesting cooperation during my stay. The financial support from the Institute of Mechanics, Processes And Control-Twente, IMPACT, is gratefully acknowledged.
During the writing of this report, two versions of the MATLAB® software OPTFORM were released and made available for the NIMR industrial partners. OPTFORMv1.2 contains the optimisation algorithm described in this report without the possibility for sequential improvement. Metamodel validation is only based on cross validation. OPTFORMv1.3 contains the full implementation of the optimisation algorithm.
The report is partly based on a previous NIMR internal report [12]. Until the report's publication date, two scientific papers related to its contents have been published [13, 14].

Sophia-Antipolis, 31 July 2005,

Martijn Bonte

Summary

During the last decades, Finite Element (FEM) simulations of metal forming processes have become important tools for designing feasible production processes. In more recent years, several authors recognised the potential of coupling FEM simulations to mathematical optimisation algorithms to design optimal metal forming processes instead of only feasible ones.

This report describes the selection, development and implementation of an optimisation algorithm for solving optimisation problems for metal forming processes using time consuming FEM simulations. A Sequential Approximate Optimisation algorithm is proposed, which incorporates metamodelling techniques and sequential improvement strategies for enhancing the efficiency of the algorithm. The algorithm has been implemented in MATLAB® and can be used in combination with any Finite Element code for simulating metal forming processes.

The good applicability of the proposed optimisation algorithm within the field of metal forming has been demonstrated by applying it to optimise the internal pressure and axial feeding load paths for manufacturing a simple hydroformed product. Resulting was a constantly distributed wall thickness throughout the final product. Subsequently, the algorithm was compared to other optimisation algorithms for optimising metal forming by applying it to two more complicated forging examples. In both cases, the geometry of the preform was optimised. For one forging application, the algorithm managed to solve a folding defect. For the other application both the folding susceptibility and the energy consumption required for forging the part were reduced by 10% w.r.t. the forging process proposed by the forging company. The algorithm proposed in this report yielded better results than the optimisation algorithms it was compared to.

Contents

Chapter 1

Introduction

Maximising product quality and profit (or minimising costs) have always been major goals of industrial companies. This is no different in the metal forming business, where e.g. steel plates are rolled, aluminium beverage cans are deep drawn and profiles are extruded.

The first concern of each metal forming company is producing a product, which fulfills all demands set by its customer. Until the 1980s, this was primarily done by experimental trial-and-error. An operator produced an initial die shape, decided on the initial process settings (process forces, feeding, etc.) and an initial material composition based on his experience. One can easily imagine that a first trial containing this initial process design often resulted in a final product that did not meet the demands. Subsequently, the operator changed the die shape, the process settings or the material and performed another test run. This time consuming and expensive iterative trial-and-error process was repeated until the final product met the demands set by the customer.

In the late 1980s and 1990s, the increasing speed of computers allowed researchers to develop methods for simulating metal forming processes using the computer. This software based on the Finite Element Method (FEM) allowed the production engineer to play around with the process variables until the computer simulation predicted a product satisfying the customer's demands. Subsequently, the obtained settings were implemented in the production plant resulting in a satisfactory product. However, the iterative procedure of trial-and-error remained, only now with computer simulations. This iterative procedure finally resulted in a sufficiently good or feasible product, whereas competitive industrial companies became more interested in an optimised product instead of just a sufficient one.

This is where the field of mathematical optimisation can assist. By coupling optimisation algorithms to FEM simulations, product or process optimality comes into sight. The industrial requests for optimisation were picked up by scientific researchers from the year 2000 onwards and is the next challenge in the simulation of forming processes.

This report is a part of the NIMR project *Optimisation of Forming Processes*, which aims at

developing an optimisation strategy for metal forming processes. It describes the selection and implementation of an optimisation algorithm suitable for optimising metal forming processes using time consuming FEM simulations. Chapter 2 starts with an overview of mathematical optimisation, followed by an introduction to metamodelling in Chapter 3. A metamodel based algorithm for the optimisation of forming processes is described in Chapter 4. In Chapter 5, the applicability of this algorithm within the field of metal forming is demonstrated by applying it to a simple hydroforming process. The algorithm is applied to more complicated 3D forging processes in Chapter 6. Chapter 7 finalises the report with some conclusions, recommendations and indicates areas for future work.

Chapter 2

Mathematical optimisation

This chapter starts with a short general introduction to mathematical optimisation in Section 2.1. Subsequently, the two major phases of mathematical optimisation, modelling and solving the optimisation problem, are described in further detail in the Sections 2.2 and 2.3. Section 2.4 contains some conclusions.

2.1 Introduction to mathematical optimisation

Mathematical optimisation has been subject of research during many decades and comprises a wide variety of algorithms for many applications. In this project, we limit ourselves to the optimisation of metal forming processes using numerical simulations based on the Finite Element Method (FEM).

Mathematical optimisation comprises four steps [102]:

- Selecting a set of variables to describe the design alternatives
- Defining an objective function/criterion
- Determining a set of constraints
- Determining the set of design variables for which the objective function is optimal taking into account the set of constraints

The relations between the four steps are presented in Figure 2.1.

The first three steps are denoted as the *modelling of the optimisation problem*. The light bulb in the figure denotes that this modelling should be done cleverly to prevent that the optimisation problem that was modelled can finally be solved efficiently by a suitably chosen optimisation algorithm. There is a strong interaction between modelling and solving an optimisation problem: if one first chooses to model an optimisation problem, one should be aware that the modelling also depends on the optimisation algorithm that is

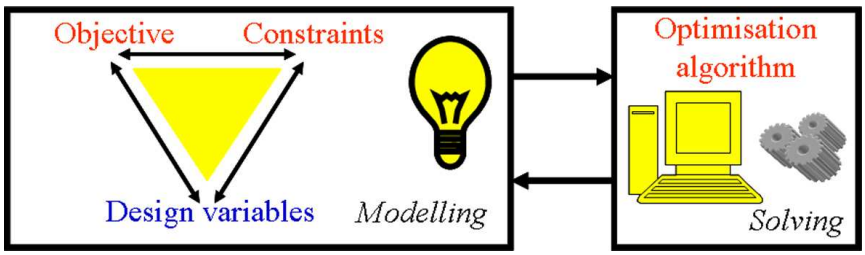


Figure 2.1: The four steps of mathematical optimisation

finally used to solve the modelled problem. On the other hand, if one selects an algorithm first, it is important to know something about the entity of the optimisation model that needs to be optimised. If the optimisation model and the algorithm are not compatible, it is likely that the problem is not solved efficiently or even at all. The fourth step of mathematical optimisation, *solving the optimisation problem*, is a matter of implementing a suitable optimisation algorithm and letting the computer do the work.

The next sections describe further details on the modelling and the solving of the optimisation problem.

2.2 Modelling the optimisation problem

Modelling the optimisation problem exists of three steps: selecting the design variables, determining the optimisation objective and determining the set of constraints. In case of optimisation using FEM, this modelling can be represented by the input - response model shown in Figure 2.2.

2.2.1 Design variables

Design variables, parameters and constants
An optimisation problem contains a number of *design variables*. The variables are subject to the optimisation algorithm in such a way that the result of the optimisation procedure provides the optimal values for the design variables only. The design variables should be chosen intelligently by the designer, who has the freedom to leave some other variables out. The latter are often denoted as *design parameters*: the designer can influence these parameters, but they are assumed to be constant for the current optimisation problem. Reasons for assuming some variables constant include the consideration that an optimisation problem with only a few variables requires a completely other optimisation algorithm than a problem containing hundreds of variables. Another group of system elements are fixed due to physical phenomena. These are denoted as *constants*. The designer cannot possibly change them or incorporate them as design variables. A typical example of a

constant is gravity.

Continuous and discrete variables
Design variables can be continuous or discrete [98]. Sometimes, these variables are referred to as *quantitative* and *qualitative* variables, respectively [90]. An example of the latter case is the amount of counters a supermarket needs to have to minimise the customer's waiting time. It would not make sense to install 11.4 counters, since the choice is to install or not to install an extra counter. The variables may only have integer values, which makes this a discrete optimisation problem.

Control, environmental and model variables
In the context of optimisation using computer simulations such as FEM, Santner et al. [114] distinguish between three types of variables. *Control variables* are comparable to the design variables mentioned earlier: these can be controlled entirely by a designer. *Environmental variables* are stochastic variables with a known or unknown distribution, e.g. the outdoor temperature. The last group, *model variables*, take into account modelling uncertainties within the simulation model.

2.2.2 Objective function

The second step contains choosing the optimisation goal, which is quantified by an objective function. This objective function has to be optimised, i.e. maximised or minimised. Maximisation problems can easily be transformed to minimisation problems by the relation:

$$\max(f) = \min(-f) \tag{2.1}$$

in which f denotes the objective function.

It is possible to include more than one objective function at the same time in an optimisation problem. This is the field of multicriterion or multiobjective optimisation. A well-known way of dealing with multiple objectives is by Pareto optimal points [102]. In this case, the objective functions are compared to each other. The solution of this problem of comparison is based on the preference to decrease one criterion without increasing the

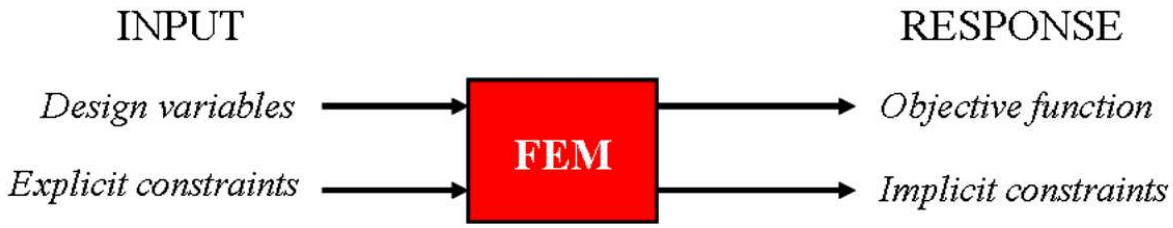


Figure 2.2: Input-response model for FEM

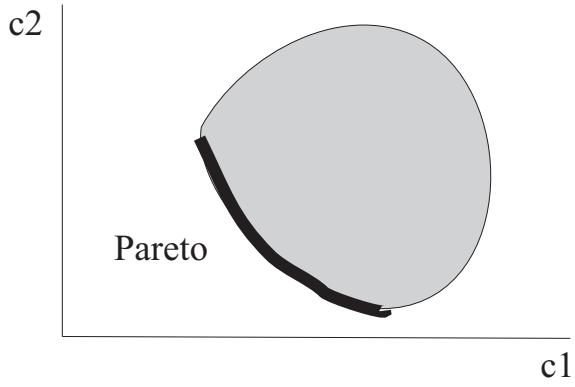


Figure 2.3: Pareto optimal points for two objective functions

others. The points in the optimal set are called the Pareto optimal points and these are depicted in Figure 2.3 for two objective functions c_1 and c_2. For a Pareto optimal point, c_1 can only be decreased by increasing c_2 and vice versa.

Alternatively, the presence of multiple objectives is often solved by denoting one as the single most important objective [106]. Subsequently, one requires that the other objectives should at least or at most have a specified value. Hence, these objectives are then modelled as constraints.

2.2.3 Constraints

Optimisation problems can be subdivided in two major groups with each their own optimisation algorithms for obtaining a solution: unconstrained and constrained optimisation. In the former case, the minimum (or maximum) of the objective function is determined without paying attention to further restrictions. The solution to the objective function may, however, be restricted to certain constraints. In this case we are dealing with constrained optimisation: the solution should be part of a feasible domain. Note that if constraints are present but the minimum of the objective function is located in the interior of the feasible domain, the problem is actually an unconstrained optimisation problem. Therefore, distinction is often made between optimisation containing interior and boundary optima instead of unconstrained and constrained optimisation problems [102].

Several different kinds of constraints exist, which are presented schematically in Figure 2.4.

Equality and inequality constraints
Constraints can be subdivided in equality and inequality constraints. The former prescribe that a relation should be on a specific constraint. Inequality constraints bound a domain in which the solution should be sought.

Linear and non-linear constraints
Constraints can be linear or non-linear. Depending on whether the constraints are linear or non-linear, optimisation problems are solved with different optimisation algorithms.

Explicit and implicit constraints
Sometimes, a constraint is known explicitly, i.e. the constraints depend on the design variables by pre-defined mathematical functions. However, this explicit relation between design variables and objective function and constraints is not known if the objective function and constraints result from e.g. a physical or a numerical experiment such as FEM. The difference between explicit and implicit constraints was illustrated in Figure 2.2.

Box constraints
A specific form of explicit constraints are *box constraints*, sometimes denoted as *bounds*. Box constraints simply limit the domain in which the design variables are allowed to vary by an upper and a lower bound.

The total optimisation problem can now be modelled as follows:

$$\begin{aligned} & \min f(\mathbf{x}) & & \\ \text{s.t.} \quad & \mathbf{h}(\mathbf{x}) & = & \mathbf{0} \\ & \mathbf{g}(\mathbf{x}) & \leq & \mathbf{0} \\ & \mathbf{lb} \leq \mathbf{x} & \leq & \mathbf{ub} \end{aligned} \tag{2.2}$$

where f is the objective function, $\mathbf{x}$ denotes the design variables and $\mathbf{h}$ and $\mathbf{g}$ are the equality and inequality constraints, respectively. Both the equality and inequality constraints

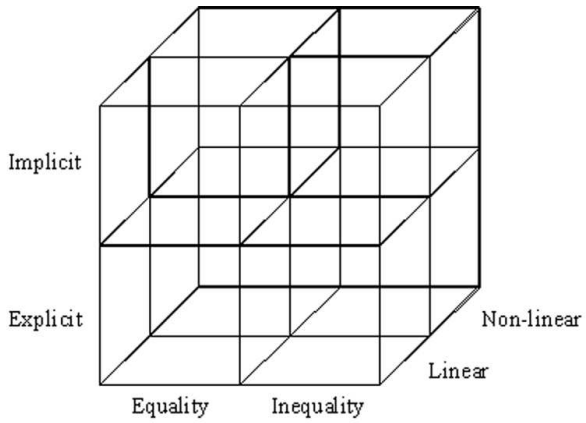


Figure 2.4: Possible constraints

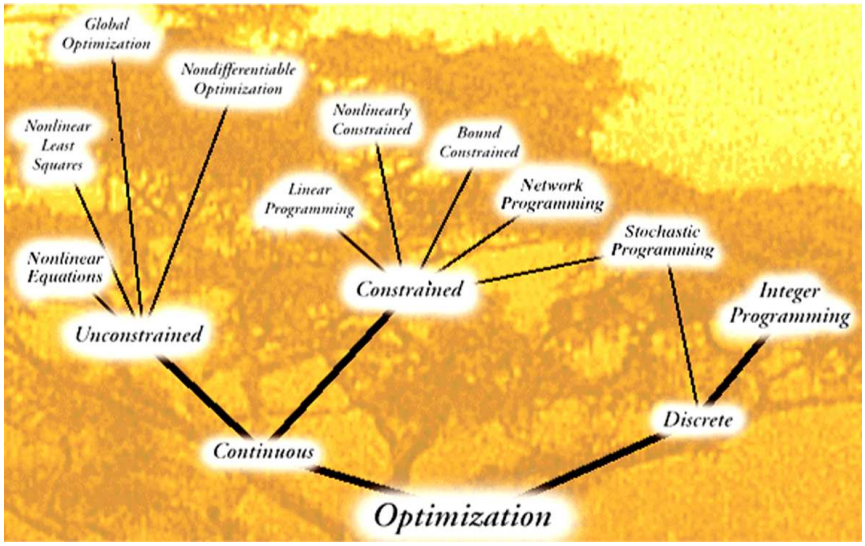


Figure 2.5: Optimisation tree [99]

can be divided further in linear or nonlinear and explicit or implicit constraints. Note that box constraints are modelled separately in Equation 2.2.

2.3 Solving the optimisation problem

When the design variables are chosen and the objective function and constraints are defined, the next step is to solve the optimisation problem with as a result the optimal values of the design variables and the minimum or maximum value of the objective function within the feasible domain. However, solving an optimisation problem is not straight-forward. The way the problem is modelled (selection of variables, objective function and constraints) is a critical factor that determines whether the problem can be solved efficiently or at all. Care during modelling may prevent many problems during solving. Next to good modelling, the decision for a specific algorithm is another critical factor. Many groups of algorithms have been developed for many different kinds of optimisation problems as indicated by Figure 2.5. The choice for one of the algorithms from one of the groups depends on many considerations as will be explained shortly below.

As can be seen from Figure 2.5 a first distinction is made between problems modelled with qualitative (discrete) or quantitative (continuous) variables. Another important indicator for which algorithm to use depends on the presence of constraints in the optimisation problem. Both require other optimisation algorithms for solving the problem effectively and efficiently. Since many textbooks have been written on all kinds of optimisation algorithms (see e.g. [25, 98, 102, 106, 131, 144]), details will not be given here. With respect to the remainder of this report, however, two aspects are especially important: the presence

of multiple local minima and the size of the optimisation problem.

A problem that can rise during solving is the presence of multiple optima within the objective function. Local and global optimisation methods exist. A risk of the former is that the achieved result may be a local optimum of the objective function rather than the global optimum. Global optimisation algorithms try to find the absolute optimum and generally search a larger domain than local optimisation methods. This makes them slower, which may form a problem, especially for large optimisation problems (many design variables).

The capability to solve optimisation problems containing computationally expensive function evaluations (e.g. the optimisation of Finite Element Models) may be limited by computing power (calculation time and memory). For this purpose, approximation methods were developed in addition to classical optimisation algorithms. Approximation methods do not result in the exact optimal solution in general, but approximate this. The lower accuracy is accepted at the benefit of a smaller computing time or less needed memory. In this way, a satisfactory solution can be obtained for quite large optimisation problems. However, approximate optimisation algorithms are known to suffer from the *curse of dimensionality*, i.e. they become exponentially more time consuming to solve when more design variables are included in the optimisation problem.

Thus, in case of expensive FEM calculations, it makes sense to limit the optimisation problem to only a few significant design variables and to choose an optimisation algorithm that is effective for optimisation problems with only a small number of design variables. Approximation methods depend on the fitting of a so-called metamodel. Chapter 3 will introduce metamodelling and several specific metamodelling techniques, which are very promising for optimisation using time consuming function evaluations.

2.4 Conclusions

This chapter introduced the four steps of mathematical optimisation. The first three steps (selecting the design variables and quantifying objective function and constraints) are part of the modelling of an optimisation problem, whereas the fourth step comprises the solving of this optimisation problem by an optimisation algorithm. The optimisation model and the optimisation algorithm should be compatible with each other to solve the optimisation problem both effectively and efficiently.

Several aspects of the four steps of mathematical optimisation with respect to the optimisation of metal forming processes using time consuming non-linear FEM calculations were discussed. In this case, it is recommendable to take only a small number of significant design variables into account in the optimisation problem to limit the time it takes to solve the problem. Approximate optimisation algorithms based on metamodelling techniques seem very promising for optimisation using time consuming function evaluations such as

FEM. The concept of metamodelling is introduced in the next chapter.

Chapter 3

Metamodelling

Metamodelling techniques are quite useful for the optimisation of phenomena for which function evaluations are time consuming or otherwise expensive. Examples of such expensive function evaluations are when physical experiments need to be performed, which is typically both very time consuming and financially demanding. Another example is when complicated computer simulations need to be performed for evaluating objective function and implicit constraints. In this context, computer simulations are sometimes referred to as numerical experiments. A specifically time consuming type of computer simulation is nonlinear FEM simulation of metal forming processes. This chapter will introduce the concept of metamodelling in Section 3.1. Before continuing to the application of metamodelling for the optimisation of metal forming processes, Section 3.2 will compare several of many metamodelling techniques. Section 3.3 formulates some conclusions.

3.1 Introduction to metamodelling

Since the evolution of computers it has become common within engineering to build computer simulation models of physical entities, which are capable of predicting (part of) reality. Within engineering, these simulation models avoid performing time consuming and expensive experimentation and lead to shorter product development times. However, for certain applications, it may be necessary to perform a large number of computer simulations. Optimisation is an example where it is not sufficient to run only one or a couple of simulations. This does not have to be a problem as long as the simulation time is limited. Unfortunately, this is often not the case. The nonlinear Finite Element simulations for metal forming we are considering in this report can take hours, sometimes days per calculation. If many simulations need to be performed, there is a strong need for a technique that limits the total simulation time.

Metamodelling is such a technique that limits the total amount of simulation time by allowing for parallel computing such that several calculations can be performed within the wall clock time of only one calculation. The term “metamodelling” refers to the Greek

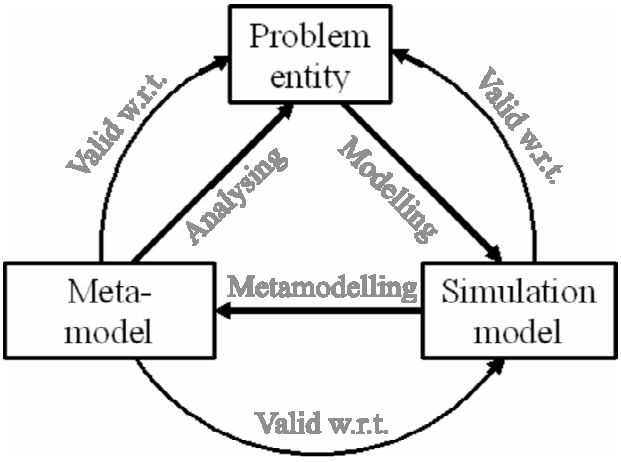


Figure 3.1: Metamodelling [64]

word "meta", which means higher, denoting that another, higher *model from a model* is made [126]. This easy to evaluate model from a model is called a metamodel. Metamodels are sometimes also referred to as *surrogate* or *response surface models*. One could picture the process of metamodelling as presented in Figure 3.1.

For the application that is considered in this report, the problem entity is the metal forming process itself, i.e. the physical c.q. mechanical phenomenon. The simulation model is the non-linear Finite Element Model (FEM) of the metal forming process, which should be valid with respect to the problem entity. The metamodel is subsequently based on the FE model and should thus be valid w.r.t. this FE model. Additionally, because the FE model is or should be a good representation of the metal forming process, the metamodel is also an easy to evaluate stand-in for the metal forming process itself. Note that the advantages of metamodelling come at the cost of a certain loss of accuracy, since:

1. The metamodel is an approximation of the FE model
2. The FE model is an approximation of the metal forming process

Because of this "double approximation", there is a large risk that the metamodel does not represent the problem entity very well. To minimise this risk, selecting a type of metamodel that is able to describe the problem entity, as well as obtaining and validating this metamodel, is very important.

We follow Kleijnen who proposes a 10 step methodology for the selection, fitting and validation of metamodels in simulation [64]:

1. Determine the goal of the metamodel

2. Identify the inputs and their characteristics
3. Specify the domain of applicability
4. Identify the outputs and their characteristics
5. Specify the required accuracy of the metamodel
6. Specify the metamodel's validity measures
7. Specify the type of metamodel
8. Specify a DOE strategy
9. Fit the metamodel
10. Validate the metamodel

These 10 steps are shortly described below.

Determine the goal of the metamodel
According to Kleijnen, metamodels can serve four general goals [64]:

1. Understanding the problem entity
2. Predicting values of the output or response variable
3. Optimisation
4. Verification and Validation of prior qualitative knowledge w.r.t. the simulation model or the problem entity

Different goals require different types of metamodels and different levels of accuracy.

Identify the inputs, the domain of applicability and the output variable
After having chosen the goal of the metamodel, it is important to select the design variables, their ranges and the response variables. In case of optimisation, the response variables are the objective function and implicit constraints. For optimisation, these steps are equal to the modelling of the optimisation problem as introduced in Section 2.2. At this point, it is also important to know whether the design variables and the responses are stochastic or deterministic, discrete or continuous, etc.

Specify the required accuracy and the metamodel's validity measures
Depending on the goal of the metamodel, the required accuracy of the metamodel and the way this accuracy is measured should be determined. A metamodel used for prediction should be very accurate, whereas for a metamodel used for understanding, it will be sufficient if only a trend is visible. The accuracy required for metamodels utilised for

optimisation purposes will lie somewhere in between. It is necessary to specify both the accuracy of the metamodel w.r.t. FE model and w.r.t. the metal forming process.

Specify the type of metamodel
Several types of metamodels exist. Each type of metamodel has its own advantages and disadvantages and since a metamodel should be suitable for the purpose it is used for, it should be selected with care. Section 3.2 compares several metamodelling techniques suitable for obtaining metamodels from computer simulations.

Specify a DOE strategy
Depending on the type of metamodel selected, a suitable Design Of Experiment (DOE) strategy should be chosen. DOE is sometimes referred to as "experimental design" or "DOX" [30]. A *Design of Experiment* strategy is a structured method for determining the relationship between factors affecting a process (design variables) and the output of that process (response) [50]. Since we are considering time consuming computer simulations, an important property of each DOE strategy is the potential to keep the number of simulations to be performed at a minimum while ensuring a certain required accuracy.

Fit the metamodel
This comprises constructing the FE model and performing the calculations with the design variable settings specified by the DOE strategy in the previous step. The values for the response are extracted from the simulation output and the metamodel is fitted.

Validate the metamodel
Now, a metamodel is present. The last question to answer is whether the metamodel satisfies the accuracy and validity measures specifies in the steps 5 and 6. Several validation techniques are available and the choice for a specific technique again depends on the type of metamodel selected. If the metamodel does not satisfy the accuracy demands, one may return to step 8 to add more experimental points and subsequently fit the metamodel with the additional information. This extra information will generally increase the accuracy of the metamodel.

Summarising, the process of constructing a metamodel incorporating the above steps is depicted in Figure 3.2. The modelling in Figure 3.2 comprises the first 7 steps mentioned above, i.e. determining the metamodel's goal, selecting the design variables and responses and determining the measures for the required accuracy and validity of the metamodel. After having chosen the type of metamodel, one first applies a suitable DOE strategy that provides the design variable settings for which a simulation should be run to determine the response variable values. Subsequently, the metamodel is fitted through the response values and validated. The validation technique compares the metamodel's accuracy to the required accuracy determined during the modelling phase. If the accuracy is not sufficient, additional numerical measurements may improve the metamodel until the required accuracy demands are met.

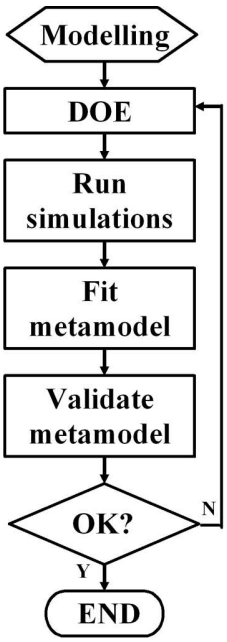


Figure 3.2: Flowchart of metamodelling

3.2 Comparison of metamodelling techniques

Before applying metamodelling techniques to the optimisation of metal forming processes, it is necessary to make a distinction between the many metamodelling techniques encountered in literature. This section will compare several metamodelling techniques to each other.

A well-known method for fitting a metamodel is *Response Surface Methodology* or *RSM* [38, 90]. Hence, the name "response surface" as an alternative for "metamodel". RSM uses least squares regression techniques to fit a lower order polynomial while allowing for a remaining random error as shown in Figure 3.3(a). RSM has officially been developed for constructing a metamodel or response surface from physical experiments, but many authors have applied it to numerical experiments (computer simulations) as well.

Although RSM has proved to be applicable in simulation practice, statisticians claim that for computer experiments, which are generally deterministic, the remaining error should formally be zero [114]. Thus, the metamodel should be interpolated through the objective function values as presented in Figure 3.3(b). They propose the field of *Design and Analysis of Computer Experiments* or *DACE* instead [111, 112]. DACE generally uses *Kriging*

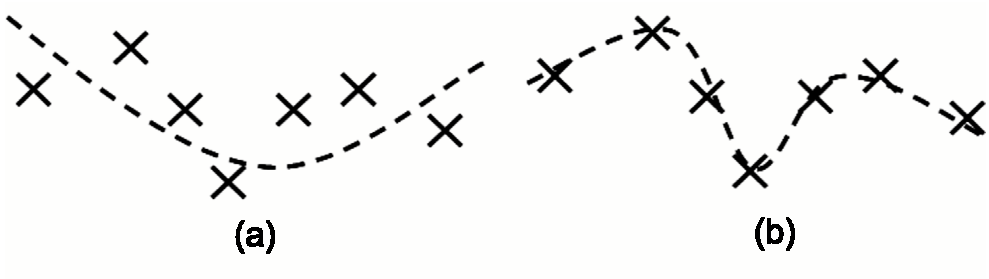


Figure 3.3: (a) RSM based metamodel; (b) DACE based metamodel

as interpolation technique. For both RSM and DACE, the user needs to make some preliminary assumptions for the shape of the metamodel. This shape is for DACE much more flexible than for RSM, which is limited to lower order polynomials. This larger flexibility generally implies that it will be necessary to perform more function evaluations in case of DACE than for RSM. On the other hand, DACE's flexibility has the potential to result in very accurate metamodels, even for complicated response shapes.

It is also possible to apply the machine learning technique *Artificial Neural Networks* for fitting metamodels. Using Neural Nets (NN), it is not needed to make any assumptions for the final shape, which comes at the cost of performing even more function evaluations, i.e. running even more expensive FEM simulations, for "training" the Neural Network. Without going into detail, yet other metamodelling techniques for computer simulations encountered in literature are *Inductive Learning (IL)*, *Multivariate Adaptive Regression Splines (MARS)*, *Radial Basis Functions (RBF)* and *Support Vector Regression (SVR)* [27, 55, 124–126].

Which metamodelling technique should be selected for which purpose is a difficult question. Several authors compare several metamodelling techniques [27, 55, 124–126] based on how accurate the metamodels are with respect to the nonlinearity c.q. noisy behaviour of approximated test functions, how many variables can be taken into account, the transparency of the metamodel, the computational costs to obtain the metamodel, the complexity of the method, etc. A summary of the comparison between the advantages and disadvantages of the several metamodelling techniques is presented in Table 3.1.

3.3 Conclusions

This chapter provided an introduction to the concept of metamodelling, which is a very useful technique when dealing with time consuming or costly function evaluations during optimisation. These expensive function evaluations often occur when using physical experiments, but can also occur for time consuming computer simulations like the non-

Metamodelling technique	*Advantages*	*Disadvantages*
RSM	Well-established Transparent Smooths numerical noise	Up to 2^{nd} order < 10 variables Stochastic
DACE	Flexible Deterministic	< 10 variables Time consuming to fit Sensitive to numerical noise Complexity of the method
NN	Flexible Many variables Deterministic	Many simulations needed Black box (not transparent)
IL	Discrete variables	Not transparent No continuous variables
MARS	Transparent Accurate for many simulations	Inaccurate for few simulations Relatively new in engineering
RBF	Not sensitive to numerical noise Accuracy?	Accuracy?
SVR	Transparent Accuracy	Recent development

Table 3.1: Advantages and disadvantages of several metamodelling techniques

linear Finite Element Method used for the simulation of metal forming processes. In the metamodelling context, running computer simulations can be referred to as performing numerical experiments in contrast to physical experiments.

Metamodelling is a way to obtain an easy to evaluate model from a model. The latter model is the computer model of some problem entity under investigation, the former is called the metamodel. In this way, the metamodel should be valid with respect to both the computer simulations and the problem entity. The procedure for obtaining a metamodel from computer simulations is as follows:

- Start with a modelling phase in which the problem is modelled (define the goal of the metamodel, the design variables, the response, validation measures, etc.)
- Apply a Design Of Experiments (DOE) strategy
- Run the computer simulations and obtain the response measurements
- Fit the metamodel
- Validate the metamodel

If a good metamodel is obtained, it can serve one of four goals:

1. Understanding the problem entity
2. Predicting values of the output or response variable
3. Optimisation
4. Verification and Validation of prior qualitative knowledge w.r.t. the simulation model or the problem entity

For the optimisation of metal forming processes, where we distinguished a "modelling" and a "solving" part in Chapter 2, we will primarily be interested in the first goal of metamodelling for the "modelling" phase and in the third goal during the "solving" part of the optimisation strategy. The next chapter will pay attention to a metamodel based optimisation algorithm for solving optimisation problems in metal forming. Therefore, this chapter was finalised with a short overview and comparison of several specific metamodelling techniques, from which a suitable technique can be selected for application to the optimisation of metal forming processes.

Chapter 4

An algorithm for the optimisation of metal forming processes

This chapter presents an optimisation algorithm for solving optimisation problems for metal forming processes using time consuming nonlinear FEM calculations. Section 4.1 covers a detailed literature review of optimisation algorithms applied in the field of metal forming. Based on this literature study, an algorithm based on metamodelling techniques was developed. This algorithm is presented in Section 4.2. Section 4.3 finalises this chapter with some conclusions.

4.1 Overview of optimisation algorithms used in metal forming

This section summarises the work, which has already been performed within the field of optimisation of metal forming processes. We will focus on literature incorporating the combination of mathematical optimisation algorithms with metal forming and FEM simulation, which is seen as a promising trend in the metal forming community [70]. Thus, process optimisation consisting of simply performing a number of FEM calculations and denoting the best results as the optimised results, but not including mathematical algorithms (see e.g. [44, 45, 79, 121, 146]) is not of interest and is excluded from the literature review. The literature including mathematical optimisation algorithms is summarised in the Sections 4.1.1 through 4.1.5. The publications are separated based on the type of algorithms used: classical iterative, approximate, adaptive, machine learning algorithms and algorithms comprising combinations of several of these different classes of optimisation algorithms.

4.1.1 Classical iterative optimisation algorithms

We refer to classical optimisation algorithms as a class of algorithms that are treated by many texts on optimisation, see e.g. [25, 38, 98, 102, 106, 131]. Examples are the Conjugate

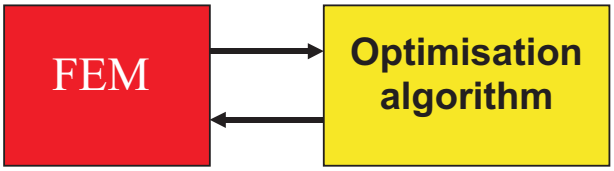


Figure 4.1: Iterative optimisation in combination with FEM

Gradient method, Quasi-Newton methods and Sequential Quadratic Programming techniques (SQP).

These classical optimisation methods often iterate towards a minimum. For the optimisation of metal forming processes using the Finite Element Method, it is necessary to run a FEM calculation for each iteration of the optimisation algorithm. This process of *iterative optimisation* using FEM is depicted in Figure 4.1.

A disadvantage of this iterative process is the necessity to run the time consuming FEM calculations sequentially: the settings of the design variables for which a function evaluation needs to be performed by a new FEM calculation, depend on the result of the previous iteration.

Another disadvantage is the local validity of most classical algorithms: they are only valid in a specific part of the feasible domain. Figure 4.2 shows that the optimisation of a function u, dependent on design variable x and initiated at x_0, will result in minimum A. This is clearly not the global minimum, since this is located in B.

Additionally, many classical algorithms require determination of the first and second order

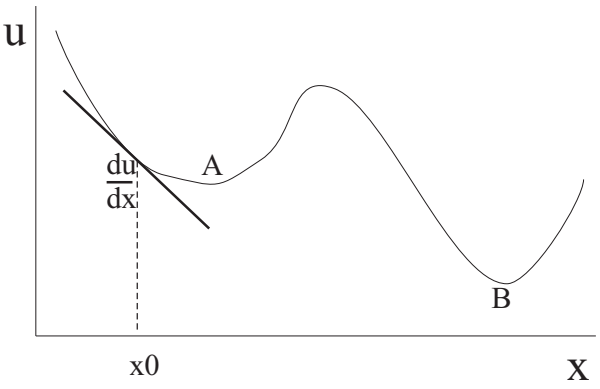


Figure 4.2: Local optimisation methods

derivatives or sensitivities of objective function and constraints with respect to the design variables. Several methods for calculating sensitivities exist [12, 38, 106]:

1. Finite Difference approximations
2. Discrete derivatives
3. Continual/variational derivatives

All these techniques have their difficulties when combined with time consuming Finite Element calculations. Finite Difference approximations are known to be relatively costly (many FEM calculations need to be performed) and can be quite inaccurate due to truncation and round-off errors. The method of discrete derivatives is more efficient, but access to the source code of the FEM software is necessary. This is clearly not always possible, especially when using commercial FE codes. A possibility to overcome this problem using discrete derivatives is applying the third method for determining the sensitivities. These continual or variational derivatives methods, however, require the analytical differentiation of the physical system described by the FEM model. For more complex situations, this is impossible. More information on sensitivity calculation in combination with FEM can be found in several texts [12, 38, 106].

Determining sensitivities for non-linear FEM calculations used for simulating metal forming is a field of ongoing research (see e.g. [26, 61, 63, 128, 145]). Nevertheless, it is concluded that it is still quite difficult to determine sensitivities for FEM calculations. Thus, optimisation algorithms requiring the determination of first and/or second order derivatives are facing serious challenges when combined with FEM.

Despite the disadvantages mentioned above, many authors in the field of metal forming have used classical optimisation algorithms, probably because the methods are well-known and widely available in many commercial software packages. Authors optimised many different processes using a variety of classical iterative optimisation algorithms. Table 4.1 gives an overview of a number of these publications.

In Table 4.1, it is presented which optimisation algorithm is applied. The classical optimisation algorithms are described in many works on optimisation [25, 38, 98, 102, 106, 131]. The interested reader is pointed to one of these documents for more information on the algorithms mentioned in Table 4.1. The table also presents which forming process is considered in the several articles, as well as which optimisation objective and which design variables are used.

Ref.	*Optimisation algorithm*	*Forming Process*	*Objective*	*Design variables*
[21]	Line search	Deep drawing	Reduce number of forming steps	Diameter
[29]	SQP	Deep drawing	Minimise thickness variations Optimise surface appearance	Addendum geometry
[61]	SQP	Deep drawing	Springback compensation	Die geometry
[66] [67] [103]	BFGS, Conj.grad.	Deep drawing Superplastic forming	Springback compensation Constant wall thickness	Tool shape geometry Initial thickness distribution
[95]	SQP, BFGS	Deep drawing	Prevent wrinkling/necking Optimal blankholder force	Blankholder force Drawbead geometry
[94]	BFGS	Deep drawing	Maximise distance FLC	Restraining force Material parameters
[77]	SQP	Deep drawing	Dissipated energy	Punch shape
[120]	Simplex	Deep drawing	Minimise risk ruptures and wrinkles	Tool geometry
[36] [37]	Non-linear Least Squares	Hydroforming	Constant thickness Geometrical errors	Axial force Internal pressure
[39]	Conj.grad.	Hydroforming	Uniform thickness Geometrical accuracy	Axial feeding Internal pressure
[56]	Iterative	Hydroforming	Minimise wall thickness variations	Axial feeding Internal pressure
[122]	SQP	Hydroforming	Constant wall thickness Minimum wrinkling	Axial feeding Internal pressure
[148]	SQP	Hydroforming	Minimise thickness variation	Axial displacement Internal pressure
[62]	Golden section	Superplastic forming	Uniform thickness Maximum deformation	Thickness parameters Thickness
[101]	Conj.grad., Golden section	Flexforming	Minimise springback	Blank diameters
[19]	Steepest descent, Golden section	Extrusion (cold, warm, hot)	Minimise extrusion force	Die shape geometry
[33]	SQP	Forging	Optimise die closure	Die geometry
[78]	SQP	Hot extrusion	Improve die life Minimum extrusion load/die stresses	Die radii
[151]	Iterative	Forging	Minimise shape deviations	Billet geometry

Table 4.1: Overview of literature on the optimisation of metal forming processes using iterative optimisation algorithms

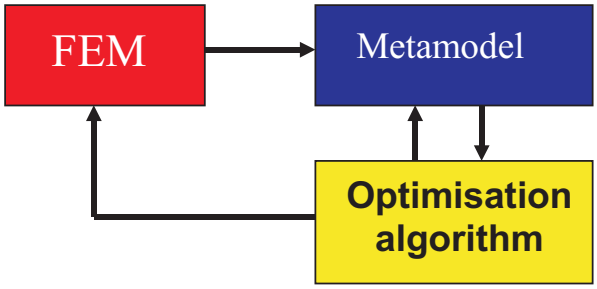


Figure 4.3: Approximate optimisation in combination with FEM

4.1.2 Approximate optimisation algorithms

In Section 4.1.1, it was mentioned that for iterative optimisation each iterate of the optimisation algorithm needs a numerically expensive FEM calculation. A total optimisation may exist of hundreds of sequential iterations, resulting in excessive calculation times.

For this reason, *approximate optimisation methods* or metamodel based techniques were developed in addition to classical methods. Metamodelling was introduced in detail in Chapter 3. For comparing approximate optimisation methods to the classical iterative algorithms mentioned in Section 4.1.1, let us presume Response Surface Methodology (RSM) as metamodelling technique. An illustrative example of RSM is shown in Figure 4.3.

As was presented in Chapter 3 the first step of RSM is to obtain a more or less simple model from the Finite Element model by regression [38]. For this, a number of Finite Element runs is performed resulting in a set of (numerical) experiments (the cross marks in Figure 4.4). Since this is a time consuming step, one would like to keep the number of runs as small as possible, i.e. to design optimal experiments. This is where the field

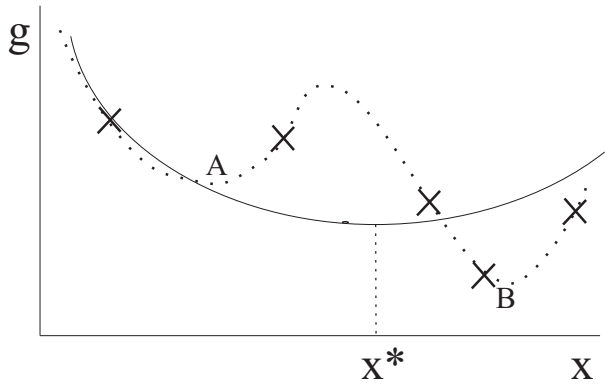


Figure 4.4: Global approximation

of *Design Of Experiments* or *DOE* is helping. After having run one FEM calculation for each design variable combination selected by the DOE strategy, the next step indicated in Figure 4.3 is to fit the *metamodel* through the obtained objective function values. Finally, the metamodel, being an explicit mathematical function that can be evaluated within a fraction of a second, may be optimised quickly using e.g. classical iterative algorithms.

Depending on the shape of the unknown objective function, the approximation obtained by RSM may be quite coarse. It is possible to perform another more detailed approximation within the area around the minimum x^* (see Figure 4.4) obtained after the first approximation. This approximation may be another metamodel approximation or a local method based on sensitivities. The continuing process of coarse approximation followed by more detailed approximations is denoted as *sequential approximate optimisation* [106].

A disadvantage of approximate optimisation algorithms is the replacement of an exact solution by a more or less accurate approximate solution. However, this disadvantage is compensated by five major advantages in case of expensive function evaluations like nonlinear FEM calculations.

The first advantage is the possibility for parallel computing. The settings obtained via a DOE strategy are independent of each other and can thus be run at the same time instead of sequentially. This means that many function evaluations can be performed at the wall clock time of only one.

A second advantage is the tendency of approximate optimisation algorithms to find the global optimum instead of only a local optimum. One can see in Figure 4.4 that the obtained optimum indeed approximates the global minimum B instead of the local optimum A. To find the global optimum, approximate optimisation algorithms generally search a larger domain than classical optimisation methods, which makes them slower for large optimisation problems (i.e. many design variables). Thus, it is of the utmost importance to limit the amount of design variables taken into account when using approximate optimisation algorithms, which can be a serious drawback.

As a third advantage, it is not necessary to determine sensitivities from FEM calculations, which are quite difficult to obtain as indicated in Section 4.1.1. The classical iterative optimisation algorithm optimises the metamodel. The metamodel ensures fast function and sensitivity evaluation since it is an explicit mathematical expression.

Yet another advantage is the black box approach, which is typical for approximate optimisation algorithms. The FEM calculations are seen as a black box: put something in and something will come out. Approximate optimisation algorithms only use this response (no sensitivities or other FE code dependent results), so every FE code that can deliver the response values, can be used in combination with approximate optimisation algorithms. This makes approximate algorithms unsensitive to which particular FE code the user wants

to incorporate in the optimisation strategy.

The fifth and final advantage is connected to the four goals of metamodelling, which were introduced in Section 3.1. Next to optimisation, visualisation of metamodels also allows for understanding the problem entity under investigation, which enhances the user's insight in the optimisation problem. Additionally, easy and cheap to evaluate accurate metamodels can replace time consuming FEM calculations for predicting the problem entity at untried design variable settings. Thus, the power of metamodelling is not restricted to optimisation only.

The advantages mentioned above have shown to be appealing to several authors in the field of metal forming. Publications using approximate optimisation algorithms are presented in Table 4.2. Again the table shows the kind of algorithm used (RSM, DACE, Neural Networks), as well as the forming process it is applied to and the objective function and design variables that are used.

Another class of publications worth mentioning is the application of approximate optimisation algorithms to non-linear FEM calculations for (automotive) crashworthiness problems. These FEM simulations are also based on non-linear mechanics and are extremely computationally expensive. They therefore show a large resemblance with FEM simulations for metal forming. Several works on approximate algorithms applied to crashworthiness problems were studied [5, 28, 48, 52, 54, 108, 147, 150].

Ref.	*Optimisation algorithm*	*Forming Process*	*Objective*	*Design variables*
[7] [8] [9]	RSM	Deep drawing	Minimise external work	Blankholder force
[16] [93]	RSM	Deep drawing	Minimise thickness variations Maximise distance to FLC	Blank geometry
[15]	RSM	Bending	Minimise springback	Tool geometry
[52]	RSM	Deep drawing	Blank holder force	Drawbead geometry
[53]	RSM	Deep drawing	Optimum draw-in	Drawbead geometry
[75]	RSM	Deep drawing	Minimise springback	Drawbead geometry
[76]	NN	Deep drawing	Minimise springback	Friction parameters
[92] [91]	RSM	Deep drawing	Minimise springback	Tool geometry
[68]	RSM	Hydroforming	Maximum protrusion height T-shape	Length, thickness, diameter
[110]	RSM NN	Hydroforming	Minimise wall thickness variation	Internal pressure Axial displacement
[113]	RSM	Flanging	Minimise springback	Sheet geometry Die geometry
[32] [41] [42]	DACE	Forging	Maximise forging quality Minimise forming energy Minimise folding potential	Preform geometry
[109]	RSM	Forging	Minimise forging energy	Preform geometry Forging load
[134]	RSM	Forging	Minimise strain variance	Preform geometry
[100]	RSM	Annealing	Constant wall thickness	Annealing temperature Annealing time

Table 4.2: Overview of literature on optimisation of metal forming processes using approximate optimisation algorithms

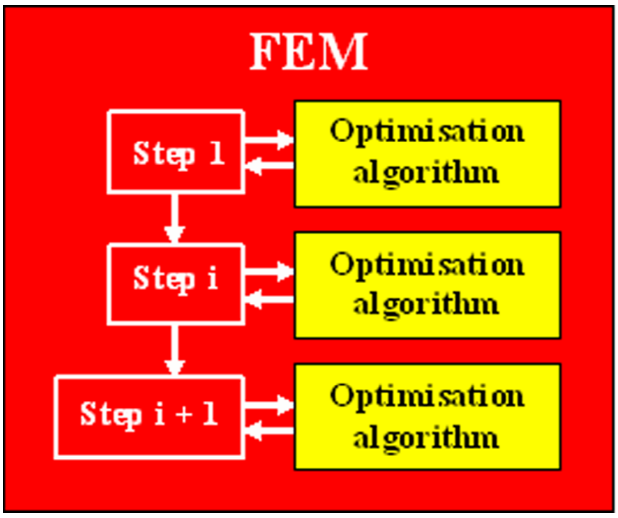


Figure 4.5: Optimisation in combination with FEM using an adaptive algorithm

4.1.3 Adaptive optimisation algorithms

If one has access to the source code of a Finite Element package, it is possible to implement a so-called adaptive optimisation algorithm to optimise a metal forming process. Using an adaptive algorithm, the design variables are continuously adapted for obtaining the best value of the objective function during each incremental time step of the nonlinear FEM calculation. The process of optimising metal forming using an adaptive algorithm is presented schematically in Figure 4.5.

The major advantage of using adaptive optimisation is that it is relatively computationally efficient. Since the optimisation model and algorithm are implemented within the FEM code, one only needs to run one FEM calculation which will immediately result in the optimised values for the design variables.

However, this efficiency comes at certain costs: it is unfortunately only possible to take into account time dependent design variables, since the design variables are optimised at each incremental time step. In case of metal forming processes, load paths are excellent examples of time dependent quantities that can be optimised using adaptive algorithms. In contradiction, it does not make sense to optimise time independent quantities like e.g. the initial blank thickness using this type of algorithm. Next to the limited possibilities for taking into account several groups of design variables, advanced accessibility to the FEM source code is necessary. This is not always possible, especially when commercial FEM packages are used.

Table 4.3 presents several publications that incorporate adaptive algorithms to optimise metal forming processes. One can indeed see that in all cases, the design variables are

Ref.	*Forming Process*	*Objective*	*Design variables*
[119]	Deep drawing	No wrinkling No fracture	Blank holder force
[130]	Deep drawing	No necking No wrinkling	Blank holder force Punch stroke
[73]	Hydromechanical deep drawing	Minimise damage	Pressure load path Punch force path
[71] [72]	Hydroforming	Minimise wall thickness variations	Pressure load path Axial displacement
[122]	Hydroforming	Minimum wrinkling	Pressure load path Axial feeding load path
[3]	Hydroforming	No necking (FLD) No wrinkling	Pressure load path Axial feeding load path
[31]	Hydroforming	No necking No wrinkling	Pressure load path Axial feeding load path
[57]	Hydroforming	Maximum formability	Pressure load path Axial feeding load path
[107]	Hydroforming	No necking No wrinkling	Pressure load path Axial feeding load path
[23] [22]	Superplastic forming	Optimum strain rate	Pressure load path

Table 4.3: Overview of literature on optimisation of metal forming processes using adaptive optimisation algorithms

time dependent load paths. However, in several articles (e.g. [3, 107]) geometrical or material parameters were taken into account by other optimisation algorithms, i.e. classical iterative or approximate optimisation algorithms. Optimisation strategies consisting of a combination of different groups of algorithms will be addressed in Section 4.1.5.

4.1.4 Optimisation algorithms based on machine learning

Next to classical iterative, approximate and adaptive algorithms, several authors use optimisation algorithms based on machine learning techniques for optimising metal forming processes. *Machine learning* is well-known in the field of Computer Sciences and can be described as the ability of a machine to improve its performance based on previous results.

Two well-known machine learning techniques are Neural Networks and Genetic Algorithms. Both techniques are encountered within the field of optimisation of metal forming processes. *Neural Networks* can be denoted as a metamodelling technique and was already treated shortly in Section 4.1.2. More detailed information on Neural Networks can be found in

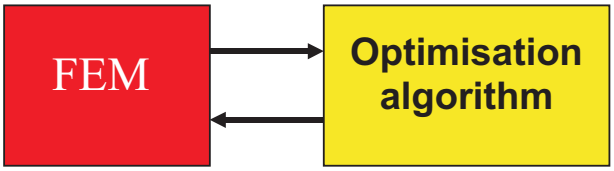


Figure 4.6: Optimisation using FEM and a genetic or evolutionary algorithm

one of the many dedicated books from the field of Computer Sciences or in MATLAB® , which has a Neural Networks toolbox [12,132]. Publications using Neural Networks for the optimisation of metal forming processes [76, 110] were mentioned in Table 4.2. Additionally, several articles present metamodels to serve as a computationally cheap prediction method for metal forming processes [60, 149]. The obtained metamodel is however not optimised.

Genetic Algorithms, or similarly *Evolutionary Algorithms*, are based on Darwin's theory "survival of the fittest". Parents reproduce offsprings in which only the strong characteristics (in case of optimisation: near optimal points) are passed on. Optimisation using FEM and a genetic algorithm can be interpreted schematically as presented in Figure 4.6. It is comparable to classical iterative algorithms treated in Section 4.1.1, since FEM simulations are evaluated directly by the algorithm. Genetic and evolutionary algorithms are known to require a large number of function evaluations [34]. Advantages of genetic and evolutionary algorithms are the possibility for parallel computing and the tendency to look for a global optimum.

Optimisation using genetic algorithms consists of a number of steps [2, 106, 116]:

1. *Initiation*: Select randomly an initial population. A population consists of a number of individuals. Each individual is a combination of design variables. Thus, the initial population is a number of initial settings x_{0i}, as depicted in Figure 4.7. The settings are presented as bits and zeros or ones are assigned to these bits.

2. *Function evaluation*: Perform the FEM calculation and obtain the values for objective and constraint functions (the cross marks in Figure 4.7)

3. *Reproduction*: select the settings with the best objective function values and/or eliminate the worst ones. In Figure 4.7 one would keep the design variable settings x_{02} and x_{03} and could perhaps eliminate x_{01} and x_{04}

4. *Crossover*: The remaining, better population is subjected to crossover, i.e. strings in the bits are paired off randomly between the several individuals. The result is a new

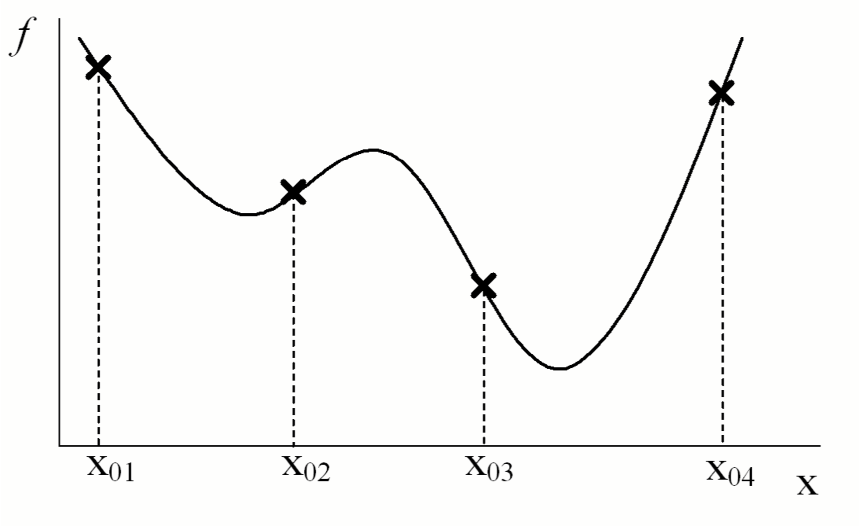


Figure 4.7: The functioning of a genetic algorithm

generation of offsprings

5. *Mutation*: Because of the elimination of non-fit settings during the reproduction phase, the generated offspring tends to obtain more and more equal genetic properties. Mutation overcomes this problem by randomly altering a bit string from an individual randomly chosen from the population. In this way, mutation makes sure that unexplored regions of the design space are also explored and guarantees the diversity of the population

6. *Return to item 2*: For the new population, FEM simulations are again performed and so on until a certain convergence criterion is satisfied

The final result of this process of *reproduction*, *crossover* and *mutation* is a number of optimal bit strings, i.e. design variable settings. Because of the random character of genetic algorithms, the global optimum is likely to be one of them and the global optimum is simply the design variable settings resulting in the lowest objective function value. Genetic algorithms are known to be robust and can be performed on parallel computers, which allows us to solve large problems or optimise several problems at the same time. Because of the binary character of the bit strings using zeros and ones, genetic algorithms are especially useful when qualitative (discrete) variables are involved [106].

Evolutionary algorithms are very similar to genetic algorithms. Slight differences are the real coded settings (non-binary) and mutation is the main genetic operator, whereas for genetic algorithms crossover is more important than mutation [4, 118]. Evolutionary algorithms tend to find a solution more quickly, whereas genetic algorithms may be more

Ref.	*Forming Process*	*Objective*	*Design variables*
[116]	Deep drawing	Maximise distance to FLC	Die geometry
[1]	Hydroforming	Maximum expansion	Pressure load path Axial displacement load path
[142]	Hydroforming	Minimise inertia variations No necking (thinning) No wrinkling (thickening)	Pressure load path Axial displacement load path
[105]	Forging	Minimum folding Minimum energy	Initial geometry
[32] [41]	Forging	Maximise forging quality Minimise forming energy Minimise folding potential	Preform geometry
[2] [24] [127]	Forging	Minimum energy Minimum shape deviation	Tool geometry

Table 4.4: Overview of literature on optimisation of metal forming processes using genetic algorithms

robust for determining the global optimum [32, 41].

Table 4.4 presents some publications from the field of metal forming in which genetic algorithms are used.

4.1.5 Combinations of different optimisation algorithms

The Sections 4.1.1 through 4.1.4 introduced several groups of optimisation algorithms, which are used for the optimisation of metal forming processes. The different groups each have advantages and disadvantages and the successful application of a specific algorithm depends heavily on the optimisation problem, which needs to be solved, and the way the optimisation problem is modelled. Several authors therefore try to combine the advantages of the different groups. Some attention to optimisation algorithms combining the different groups of algorithms presented in the Sections 4.1.1 through 4.1.4 is spent below.

The first remark needs to be made concerning the approximate and adaptive optimisation algorithm groups introduced above. They do not actually solve a problem themselves, but they necessarily incorporate a classical iterative or genetic algorithm presented in the Sections 4.1.1 to calculate an optimum. For the yellow “optimisation algorithm” blocks in the Figures 4.3 and 4.5, the user will need to choose an iterative or genetic algorithm for solving the metamodel and each incremental time step, respectively.

Within the metal forming community, most authors utilising approximate optimisation algorithms choose some sort of a classical iterative algorithm to minimise the metamodels. Some others were dealing with relatively noisy metamodels (i.e. many local optima), which are typically obtained from DACE and Neural Networks as metamodelling techniques, and chose a more global genetic algorithm [76, 110].

The same can be said for adaptive optimisation algorithms: some authors chose iterative algorithms, others genetic ones [73]. However, the most widely used strategy in adaptive simulation does not use optimisation algorithms to define an optimum load path, but some controller incorporating a collection of experience or theory based rules to ensure compression (e.g. wrinkling) and tension failures (e.g. necking) will not occur in metal forming [22, 23, 57, 119, 122]. In this context, several authors [3, 31, 107] use controllers based on fuzzy logic, which is a technique related to machine learning algorithms mentioned in Section 4.1.4. Fuzzy logic is a type of logic that recognises more than simple true and false values. With fuzzy logic, propositions can be represented with degrees of truthfulness and falsehood. For example, the statement, today is sunny, might be 100% true if there are no clouds, 80% true if there are a few clouds, 50% true if it is hazy and 0% true if it rains all day [141].

Next to the combination of approximate and adaptive algorithms to iterative and genetic algorithms, several authors deliberately attempt to combine the advantages of the different groups of optimisation algorithms. In Section 4.1.3 it was mentioned that adaptive optimisation algorithms can only take into account time dependent design variables like e.g. load paths. It is, however, possible to let an adaptive algorithm optimise the load paths and implement an iterative algorithm to determine the optimal settings for other, non time dependent design variable groups, i.e. geometrical and material parameters [3, 107].

Above it was emphasised that approximate optimisation algorithms implicitly need iterative or genetic algorithms to optimise the metamodel. The other way around, it is also possible to enhance an evolutionary machine learning algorithm with information provided by a metamodel based approximate algorithm to make it more efficient and overcome the large number of function evaluations that is generally required by genetic and evolutionary algorithms. Such a method is presented in [34] and applied to optimise the tool geometry in forging by Fourment and Do [32, 41, 42].

A last note on literature concerning the optimisation of metal forming is an indication to the presence of software packages in which many different optimisation (iterative, approximate and machine learning) algorithms are readily available. Concerning the application to metal forming using expensive FEM calculations, the "Institut für Bildsame Formgebung" of the University of Aachen, Germany, offers the optimisation software CAOT incorporating a variety of optimisation algorithms [6, 49, 135]. The user can select one of the algorithms and apply it to any nonlinear FEM code for metal forming to obtain the solution of the optimisation problem modelled by the user.

4.2 A metamodel based optimisation algorithm for metal forming processes

In this section, an optimisation algorithm for solving optimisation problems in metal forming using time consuming FEM simulations is proposed. Section 4.2.1 introduces the proposed algorithm by presenting an overview. The several steps of which the algorithm consists are discussed in detail in the Sections 4.2.2 through 4.2.6.

4.2.1 Overview of the optimisation algorithm

In Section 4.1, many publications were discussed in which mathematical optimisation algorithms were combined with FEM metal forming simulations. Many different algorithms are used for many different metal forming models, which indicates that academic research has not converged to one or two best optimisation algorithms yet. This makes the choice for a suitable algorithm for application in metal forming a difficult one. In this section a metamodel based sequential approximate optimisation algorithm is proposed for the optimisation of sheet and tube metal forming processes. This choice is based on the following considerations:

- Classical iterative algorithms can be computationally expensive because of the necessity to run FEM calculations sequentially. Moreover, sensitivities will need to be determined, which is quite a challenge using FEM as described in Section 4.1.1 and these algorithms are likely to be trapped in a local optimum

- Adaptive optimisation algorithms can only take into account time dependent design variables and generally need access to the source code of the FEM software. To keep the algorithm widely applicable, no preliminary restrictions are allowed concerning both the choice of design variables or a specific FEM package. The algorithm should be able to be combined with any FEM code, also commercial ones for which access to the source code is not available

- Genetic and evolutionary algorithms look promising because of their tendency to find the global optimum and the possibility for parallel computing. However, the rather large number of function evaluations that is expected to be necessary using genetic algorithms is regarded as a serious disadvantage

- Approximate optimisation algorithms are designed to be effective in case of expensive function evaluations like physical and time consuming numerical experiments. They do not require the calculation of sensitivities, allow for parallel computing and tend to find the global optimum. Additionally, their black box approach assures the optimisation strategy will be unsensitive w.r.t. different FEM codes and the metamodels can also be used for other goals than optimisation only. Unfortunately, these advantages come at the cost of having only an approximate optimum, but this disadvantage can

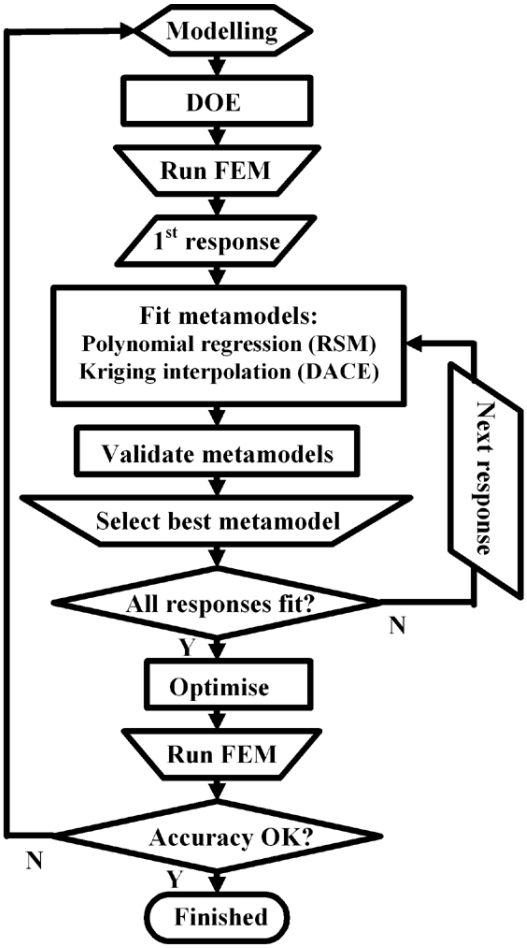


Figure 4.8: Sequential approximate optimisation algorithm for the optimisation of metal forming processes

be weakened by applying a sequential approximate optimisation algorithm, which allows for improving the accuracy of metamodels and thus the approximated optimum, sequentially

An overview of the optimisation algorithm implemented for the optimisation of metal forming processes using time consuming FEM calculations is presented in Figure 4.8. After the modelling of an optimisation problem, the metamodel based sequential approximate algorithm consists of several steps:

- A Design Of Experiments (DOE) strategy
- Running the FEM simulations
- Fitting the metamodels for each response (objective function and/or implicit constraints)

- Metamodel validation
- Metamodel optimisation
- Validation of the approximate optimum by running a final FEM calculation with the optimised settings
- Sequential improvement. If the user is not satisfied with the accuracy of the approximate optimum, the optimisation problem can be remodelled and the above steps can be executed again. In such a way, the optimisation algorithm is applied sequentially until the user is satisfied with the approximate optimum.

The different phases of the algorithm are elaborated in the next sections.

Before we do this, however, a choice will be made for a specific metamodelling technique from all the techniques introduced in Section 3.2. Both RSM and DACE are believed to be best applicable for the optimisation of metal forming processes using time consuming nonlinear FEM simulations. This choice is based on the following considerations:

- The training of a Neural Network (NN) needs too many expensive computer simulations and the lack of transparency of the metamodel prevents gaining knowledge on the metal forming process [27, 55, 125, 126]
- Inductive Learning (IL) is not ideal because of the presence of continuous variables within metal forming and the lack of a transparent metamodel [125, 126]
- The nonlinear FEM calculations used for simulating metal forming processes can be extremely time consuming, which will force the user to keep the number of simulations limited. The lack of many simulations will deteriorate the accuracy of Multivariate Adaptive Regression Splines (MARS) based metamodels [55]
- Radial Basis Functions (RBF) seem very promising as a substitute for DACE: fitting is performed very quickly and the metamodel is reported to be more accurate, especially in the presence of numerical noise [55, 124]. However, other authors report RBF to be quite inaccurate for other test functions [27], which seems to make this metamodelling technique sensitive to the type of problem it is applied to. Because of this uncertainty and the larger flexibility and popularity of DACE, the latter method is adopted for the optimisation of metal forming
- Support Vector Regression (SVR) is only recently proposed as an alternative for NN [27] and needs to be developed further before adopting this metamodelling technique
- RSM is well-established and often applied as approximate optimisation technique within the field of metal forming (see Section 4.1.2)

Response type	*Response shape*	
	Linear or parabolic	*Highly nonlinear*
Stochastic	RSM	RSM/DACE
Deterministic	DACE/RSM	DACE

Table 4.5: Criteria for using RSM or DACE

- The limited complexity of shapes of RSM metamodels (up to 2^{nd} order polynomials) can be compensated by the flexibility of DACE. On the contrary, the sensitivity of DACE to numerical noise, which may be present can be overcome by the smoothening behaviour of RSM

- The major disadvantage of both RSM and DACE is the "curse of dimensionality", which limits the number of design variables that can be taken into account. To overcome this problem, effort needs to be made for developing a methodology for indicating only the most significant variables (< 10). Such a methodology in itself would already give a lot of process knowledge

A final choice between RSM and DACE depends on the response function's type and the shape as presented in Table 4.5. RSM was originally developed for stochastic physical phenomena, which can be described by lower order polynomial metamodel shapes. The latter assumption is mostly satisfied when a relatively small design space is investigated: the metamodel is trusted to be either linear or at the most quadratic (in the neighbourhood of an optimum). Therefore, RSM is traditionally performed in a sequential procedure, where a small sub-design space or trust region is moved through the total design space, where the response can generally be highly nonlinear. This sequential procedure can be compared to iterative optimisation algorithms discussed in Section 4.1.1 and can end up in a local optimum, which is, to the opinion of the author, a large disadvantage of sequential RSM.

DACE's flexibility offers the possibility to fit highly nonlinear functions at once and its interpolative character makes it more suitable in case of deterministic phenomena. When applied to a stochastic phenomenon, however, DACE will fit the metamodel right through the measurements, thereby introducing phenomena, which are not physically present. In general, DACE is also applicable to linear or nearly linear responses. It will, however, be less efficient than RSM for these applications.

Let us now review whether we are willing to select either RSM or DACE for our application, the optimisation of metal forming processes using expensive nonlinear FEM simulations. It is quite impossible to make a solid assumption on the shape of the responses. It is still under investigation, which phenomena should be used as objective function and implicit constraints. Consequently, it is even more unclear how these phenomena are quantified using which design variables and any guess on the final shape of the metamodels of these

responses would be speculative. Thus, in this phase it is wise to reckon with the worst case scenario, i.e. highly nonlinear responses.

Regarding the response type, let us assume the nonlinear Finite Element calculations used for metal forming are deterministic: running a FEM calculation twice with the same settings will generally result in the same answer. We should notice, however, that slightly different settings can result in a significantly different outcome. This can be a consequence of the physical phenomenon, in which case there is no problem and the assumption that computer experiments are deterministic still holds. On the other hand, this difference can also be due to the adaptation of numerical parameters within the computer code. In case of FEM, one could think of adaptive mesh refinement, automatic step size adjustment, etc. Thus, the large difference in outcome is not physical and could be compared to the random measurement noise assumed to be present when using RSM. This kind of noise generated by computer experiments is referred to as *numerical noise*. The presence of noise could indicate that we are actually dealing with a stochastic problem, which seems to be in favour of using RSM. However, it is questionable if numerical noise caused by nonlinear FEM simulations results in the zero mean random errors assumed for RSM [125]. Additionally, running the same FEM simulations twice with exactly the same settings will generally result in exactly the same answer. Hence, DACE appears to be better applicable with respect to the response type, too.

The observations above lead us to have a small favour for applying DACE for the optimisation of metal forming processes. Nonetheless, it is well possible that it turns out that we are dealing with particularly smooth response shapes for which numerical noise is present. In that case, it would have been better to use RSM instead. Noticing this and the popularity of RSM as a metamodelling technique within the field of metal forming, however, pleads for incorporating both RSM and DACE within the optimisation algorithm proposed in Figure 4.8.

Broad overviews of Response Surface Methodology (RSM) and Design and Analysis of Computer Experiments (DACE) are attached in the Appendices A and B, respectively. Attention is given to the way metamodels are being fitted, assumptions underlying the use of RSM and DACE, the validation of the metamodels and Design Of Experiments (DOE) strategies suitable for RSM or DACE.

Let us now describe the different phases, which the metamodel based optimisation algorithm introduced in Figure 4.8 comprises. We will introduce the phases in chronological order: that is, the phases are described in the same order as they are performed by the optimisation algorithm.

4.2.2 Design Of Experiments strategy

After having modelled an optimisation problem, the first step of the proposed sequential approximate optimisation algorithm is to apply a Design Of Experiments (DOE) strategy. In Section 4.2.1, a choice was made for both RSM and DACE as metamodelling techniques. In the Appendices A.4 and B.4, suitable DOE strategies for RSM and DACE are presented, respectively. Since good DOE strategies for RSM and DACE are generally quite different, it is necessary for selecting a DOE strategy to make a preliminary assumption on the type of metamodel, which is to be fitted. In Section 4.2.1, DACE appeared to have a small advantage over RSM for the optimisation of metal forming processes. Therefore, we will focus on experimental designs suitable for DACE.

An important property for DOE strategies suitable for DACE is that the design is *space-filling*, i.e. the design points are nicely spread over the design space. In Appendix B.4, two groups of DOE strategies for DACE are distinguished: those that are based on geometrical criteria and those that are based on some statistical criterion. Among the former group are (stratified) Monte Carlo (random) sampling, Latin Hypercubes Sampling/Design (LHS/LHD), orthogonal arrays, lattice sampling and distance based designs (minimax and maximin designs), see e.g. [46, 69, 114]. The latter group statistical criterion based DOE strategies comprises DOE strategies that make assumptions on the DACE metamodel that is to be fitted and use this statistical information to achieve an optimal experimental design. This group has large similarities w.r.t. the computer generated optimal designs known for RSM [114], of which the D-optimal design is the most well-known. Computer generated designs for RSM are described in Appendix A.4.4. Examples of statistical criterion based DOE strategies for DACE are the maximum entropy design and designs based on the Integrated and Maximum Mean Squared Error (IMSE and MMSE), see e.g. [46, 69, 114] and Appendix B.4.3.

Which of the above DOE strategies is best applicable for the optimisation algorithm presented in the previous section? Santner et al. [114] notice that geometry based designs are used more frequently than designs based on a statistical criterion, because the latter are more difficult to implement and a lack of readily available software makes them less attractive to practitioners. Additionally, Lehman [74, 114] conducted an extensive study and concluded that the geometry based Latin Hypercubes Design (LHD) provided more accurate metamodels than the statistical criterion based D-optimal design. Regarding the different geometry based designs, Monte Carlo sampling and LHD appear to be intuitively attractive, easy to implement and comparably accurate [114]. LHD is in favour of a small advantage because of its good projection properties: if one design variable appears to be not significant, an LHD collapses uniformly onto the remaining dimensions. This uniform projection ensures no valuable information is lost.

For the proposed optimisation algorithm for metal forming processes, an LHD design is used. Latin Hypercubes Designs were originally proposed by McKay et al. [86] and apply-

ing them comprises dividing the design space into a number of strata and choosing one measurement randomly in one stratum. Each column or row of the design comprises only one measurement. An example of an LHD is presented in Figure 4.9(a). This design is, however, NOT spacefilling since no measurements are present in the upper left and lower right corner of the design space. As was already pointed out above, spacefillingness is essential for a good DOE for DACE. Santner et al. [114] recommend to modify an LHD with the maximin criterion, which maximises the minimum distance between the design points. The combination results in a *spacefilling Latin Hypercubes Design* as shown in Figure 4.9(b), which is a very powerful DOE strategy for DACE [114].

When a metamodel is used for optimisation, it is important that the metamodel gives accurate results in the neighbourhood of the optimum. Often, this optimum will be constrained, i.e. lies on the boundary of the design space. Therefore, an accurate prediction is needed on the boundary, which implies performing measurements on the boundary. An LHD will generally provide design points in the interior of the design space and hardly any on the boundary. To compensate for this lack of points on the boundary, the LHD is combined with a full factorial design (see Appendix A.4.2), which places DOE points right in the corners of the design space. This method was also proposed by Van Beers et al. [139] and Kleijnen et al. [65]. Figure 4.10(a) presents the LHD modified with a full factorial design for a two dimensional rectangular design space.

Unfortunately, the design space will not be rectangular when explicit constraints are present. In this case, the proposed algorithm will:

1. check which points of the LHD + full factorial design are non-feasible

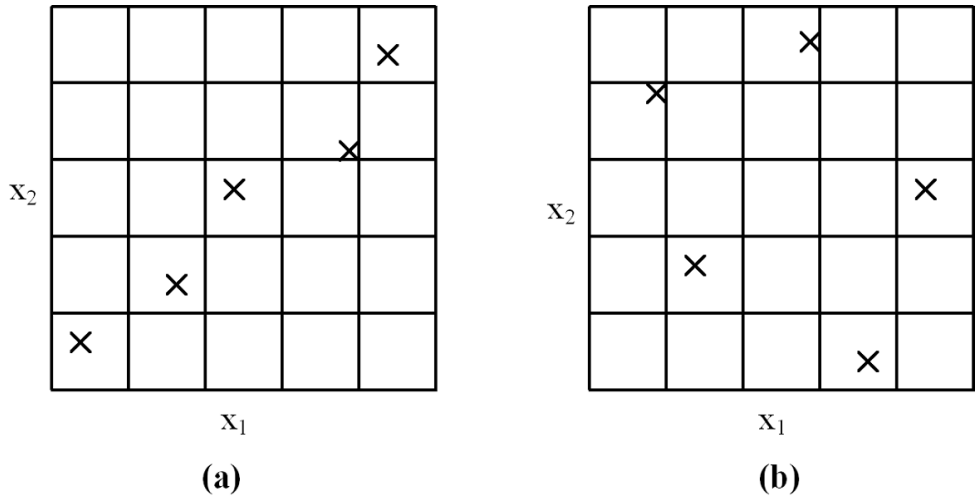


Figure 4.9: (a) A Latin Hypercubes Design; (b) A spacefilling maximin Latin Hypercubes Design

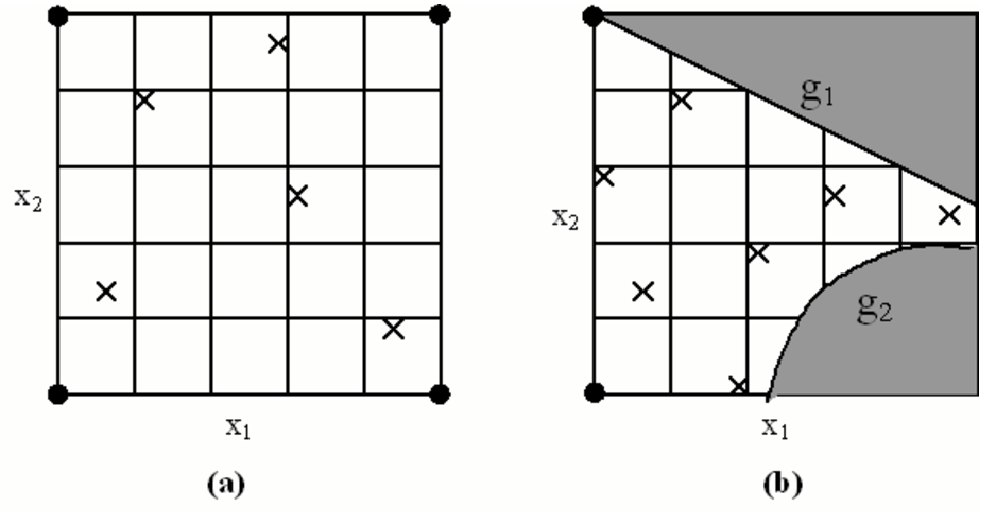


Figure 4.10: (a) LHD + full factorial design; (b) LHD + full factorial design including explicit constraints

2. skip the non-feasible points
3. replace the non-feasible points with new points
4. repeat the above procedure until all points are feasible

Replacing the non-feasible points is done by selecting a number of sets of additional design points. The new set of points is the one for which the minimum point to point distance is maximised. This maximin criterion is used for both the initial DOE and for the case when the user wants to generate additional experimental design points, for example for improving the accuracy of the metamodels. The DOE strategy incorporated in the proposed optimisation algorithm is presented in Figure 4.10(b).

4.2.3 Obtaining the results and fitting the metamodels

The next step of the optimisation algorithm is to run the Finite Element calculations. The input for each calculation are the settings of the design variables generated by the DOE strategy introduced in the previous section. The total number of simulations equals the number of design points. The calculations are mutually independent and may be run in parallel. The objective function and the implicit constraints for every DOE point are the results of the analyses.

Now review the data of the first response as indicated in the flowchart of the proposed optimisation algorithm presented in Figure 4.8. This is generally the collection of response points of the objective function values for each calculation. Thus, for example 16 calculations will result in 16 response points for each response. We will now fit a metamodel

through these 16 measurement points. Both RSM and DACE are incorporated in the optimisation algorithm as metamodelling techniques. The Appendices A.1 and B.1 describe thoroughly how to fit a metamodel using RSM and DACE, respectively. This section is a short summary of these appendices.

RSM

Starting with RSM, the response points $\mathbf{y}$ are presented as the sum of a lower order polynomial metamodel and a random error term $\boldsymbol{\varepsilon}$ [90]:

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{4.1}$$

where $\mathbf{X}$ is the design matrix containing the experimental design points and $\boldsymbol{\beta}$ are the regression coefficients obtained by least squares regression:

$$\hat{\boldsymbol{\beta}} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \tag{4.2}$$

The metamodel is

$$\hat{\mathbf{y}} = \mathbf{X}\hat{\boldsymbol{\beta}} \tag{4.3}$$

and can obtain four possible shapes, which are in ascending complexity:

- linear
- linear + interaction
- pure quadratic or elliptic
- (full) quadratic

After the metamodel is fitted, it can be used to predict the value of an unknown point y_0 at the location defined by the design vector $\mathbf{x}_0$, which is entirely defined by the design variable settings for that location and an assumption for one of the four final shapes of the metamodel:

$$\hat{y}_0 = \mathbf{x}_0^{\mathrm{T}}\hat{\boldsymbol{\beta}} \tag{4.4}$$

The predicted variance of y_0 at that location is [90]:

$$\mathrm{var}(\hat{y}_0) = \sigma^2\mathbf{x}_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{x}_0 \tag{4.5}$$

where σ^2 is the error variance, which can be estimated by the Mean Squared Error (MSE):

$$\mathrm{MSE} = \frac{\mathrm{SSE}}{n-p} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-p} \tag{4.6}$$

SSE is the Sum of Squared Errors, which equals the square of the difference between the measured response points and the response points predicted by the polynomial metamodel. n is the total number of measurements, whereas p is the number of regression coefficients.

DACE

DACE was proposed by Sacks et al. [111, 112] to fit metamodels using deterministic computer experiments. *Kriging*, which was originally developed for locating possible spots to find gold, is used to interpolate between the response measurements. Using Kriging, the random error term $\boldsymbol{\varepsilon}$ in RSM (Equation 4.1) is replaced by a Gaussian stochastic process $Z(\mathbf{x})$, which forces the metamodel to go through the measurement points:

$$\hat{\mathbf{y}} = \mathbf{F}^{\mathrm{T}}\boldsymbol{\beta} + Z(\mathbf{x}) \tag{4.7}$$

Note that the design matrix $\mathbf{X}$ is replaced by $\mathbf{F}^{\mathrm{T}}$, which is a standard notation in Kriging literature. We will adopt this formulation for DACE, whereas we will stick to the classical formulation using $\mathbf{X}$ for RSM. The first part of Equation 4.7 covers the global trend of the metamodel. The Gaussian stochastic process Z, which accounts for the local deviation of the data from the linear regression metamodel, has zero mean, variance σ_z^2 and covariance

$$\operatorname{cov}\left(Z(x_1), Z(x_2)\right) = \sigma_z^2 R\left(x_1 - x_2\right) \tag{4.8}$$

where R is the correlation function and x_1 and x_2 are two locations, which are determined by the design variable settings at these locations. For the proposed algorithm, a Gaussian exponential correlation function is adopted:

$$R(\vartheta, x_1, x_2) = \exp^{-\vartheta(x_1 - x_2)^2} \tag{4.9}$$

As opposed to other possibilities for the correlation function like e.g. cubic splines and ordinary exponential functions, see e.g. [69, 80, 81, 114], Gaussian exponential functions are intuitively attractive because they are infinitely differentiable. Moreover, Gaussian exponential functions are frequently used in literature [114] and have been found to give accurate results [80].

Equation 4.7 is now a function of the three parameters $\boldsymbol{\beta}$, σ_z^2 and R. The latter solely depends on ϑ. All these parameters are unknown. Several ways for estimating these unknown parameters can be found in literature, but Maximum Likelihood Estimation (MLE) is generally considered to be the best way [83, 114]. Using MLE, it is shown in Appendix B.1 that the estimated parameters are:

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^{\mathrm{T}}\hat{\mathbf{R}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\hat{\mathbf{R}}^{-1}\mathbf{y} \tag{4.10}$$

$$\hat{\sigma}_z^2 = \frac{1}{n}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^{\mathrm{T}}\hat{\mathbf{R}}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{4.11}$$

$$\hat{\boldsymbol{\vartheta}} = \max_{\boldsymbol{\vartheta}} n \ln \hat{\sigma}_z^2(\boldsymbol{\vartheta}) + \ln|\hat{\mathbf{R}}(\boldsymbol{\vartheta})| \tag{4.12}$$

In these equations, $\mathbf{F}$ is the design matrix, $\mathbf{y}$ the response measurements, n the number of response measurements and $|\hat{\mathbf{R}}(\boldsymbol{\vartheta})|$ denotes the determinant of $\hat{\mathbf{R}}$, the correlation matrix incorporating the estimated values of $\boldsymbol{\vartheta}$. Note that Equation 4.10 presents the Generalised Least Squares estimate of $\boldsymbol{\beta}$: if there are no correlations between the measurement points, $\mathbf{R}$ reduces to the unity matrix, in which case Equation 4.10 equals the ordinary least squares estimate of $\boldsymbol{\beta}$ presented in Equation 4.2. A second important note is that Equation 4.12 is an implicit relation. $\hat{\boldsymbol{\vartheta}}$ itself depends $\boldsymbol{\vartheta}$. This problem can only be solved by an optimisation procedure, which can be time consuming for larger problems [69].

Depending on how $\boldsymbol{\beta}$ is estimated, three types of Kriging are encountered in literature [82]:

- Ordinary Kriging
- Universal Kriging
- Detrended Kriging

Ordinary Kriging assumes a constant mean over the domain and the 0^{th} order polynomial is used as a regression model. *Universal* and *Detrended Kriging* both fit higher order linear regression models to account for non-stationary means. The difference between Universal and Detrended Kriging lies in the way the regression coefficients $\boldsymbol{\beta}$ are estimated. Using Detrended Kriging the data points are assumed to be uncorrelated, i.e. $\mathbf{R} = \mathbf{I}$, whereas Universal Kriging incorporates correlations between the data. It can be stated that Detrended Kriging combines both linear regression and Ordinary Kriging, whereas Universal Kriging will result in slightly different regression coefficients than linear regression [82].

Just as was the case for RSM, DACE may be used to predict the unknown value y_0 at a certain location x_0, see e.g. [69, 81, 82, 114]:

$$\hat{y}_0 = \mathbf{f}^{\mathrm{T}}\hat{\boldsymbol{\beta}} + \mathbf{r}^{\mathrm{T}}\hat{\mathbf{R}}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{4.13}$$

where $\mathbf{f}$ is the design vector comprising the location x_0 ($\mathbf{f}$ was denoted as $\mathbf{x}_0$ in RSM) and $\mathbf{r}$ is a vector with the correlation between x_0 and the known locations where measurements were performed.

The estimation of a value for y_0 by Equation 4.13 comes with uncertainty. The prediction of an unknown point y_0, which is close to a known measurement, will intuitively be more accurate than one that lies far from a known measurement. The expected Mean Squared Error (MSE) at y_0 is expressed by (see e.g. [82]):

$$\mathrm{MSE}\left(\hat{y}_0\right) = \hat{\sigma}_z^2\left(1 - \begin{bmatrix} \mathbf{f}^{\mathrm{T}} & \mathbf{r}^{\mathrm{T}} \end{bmatrix}\begin{bmatrix} \mathbf{0} & \mathbf{F}^{\mathrm{T}} \\ \mathbf{F} & \mathbf{R} \end{bmatrix}\begin{bmatrix} \mathbf{f} \\ \mathbf{r} \end{bmatrix}\right) \tag{4.14}$$

It can be shown that Equation 4.14 equals 0 in case y_0 equals a known measurement point, which demonstrates that Kriging interpolates between the measurement data. Equation

4.14 is comparable to Equation 4.5 for RSM, except that a good RSM metamodel will converge to the error variance σ^2, whereas a good Kriging metamodel will converge to 0.

If more, say k design variables are present, the total correlation function R is assumed to depend on the k one-dimensional correlation functions R_j as follows [111]:

$$R\left(x_1 - x_2\right) = \prod_{j=1}^{k} R_j\left(x_{1j} - x_{2j}\right) \tag{4.15}$$

That is, one assumes there is no relation between the different dimensions. Adopting the Gaussian correlation function introduced in Equation 4.9, the total correlation function becomes:

$$R\left(x_1 - x_2\right) = \prod_{j=1}^{k} \exp^{-\vartheta_j (x_{1j} - x_{2j})^2} \tag{4.16}$$

Thus, one ϑ is present for each design variable (each dimension).

Implementation issues

The implemented metamodel based optimisation algorithm for metal forming processes fits for each response (objective function, implicit constraints) four RSM metamodels and three DACE metamodels:

- linear
- linear + interaction
- pure quadratic or elliptic
- (full) quadratic
- Kriging with a 0^{th} order polynomial as a trend function
- Kriging with a 1^{st} order polynomial as a trend function
- Kriging with a 2^{nd} order polynomial as a trend function

The optimisation algorithm incorporates "DACE, a MATLAB Kriging Toolbox" implemented by Lophaven, Nielsen and Søndergaard [80, 81] of the Technical University of Denmark, Lyngby, which is downloadable from the internet [97].

The optimisation algorithm visualises the metamodel by evaluating both the predicted response values and the variance for RSM and the Mean Squared Error for DACE for a grid of locations. That is, for RSM, Equations 4.4 and 4.5 are used and for DACE Equations

4.13 and 4.14 are evaluated for this grid of locations.

However, visualisation becomes difficult when the optimisation problem contains more than three dimensions where one dimension is reserved for the response variable. Thus, the metamodel can only be visualised depending on two design variables. The optimisation algorithm can take care of this problem in two separate ways:

1. By fixing the other design variables at a constant value
2. By averaging over the other dimensions, i.e. design variables

Sacks et al. propose to fix the design variables that are not visualised, at the midrange [111, 112]. Schonlau [117] proposes averaging the dependence of the response on these design variables by integration. Suppose $\mathbf{x}_{\text{visualised}}$ is the one or two design variables, which are to be visualised. Then the average values of the responses w.r.t. the removed variables $\mathbf{x}_{\text{removed}}$ can be calculated by:

$$\boldsymbol{\mu}(\mathbf{x}_{\text{visualised}}) = \frac{1}{V} \int f(\mathbf{x}_{\text{removed}}) \mathrm{d}\mathbf{x}_{\text{removed}} \tag{4.17}$$

in which V is the total (hyper)volume of the removed design variables and f are the response values for the removed variables. Doing this for every variable excluding the visualised variables will provide the average responses $\boldsymbol{\mu}$, which only depend on the one or two design variables that will be visualised. Numerically, Equation 4.17 becomes [117]:

$$\boldsymbol{\mu}(\mathbf{x}_{\text{visualised}}) = \frac{1}{m} \sum_{i=1}^{m} f(\mathbf{x}_{\text{visualised}}, \mathbf{x}^i_{\text{removed}}) \tag{4.18}$$

with m the total number of removed response values. Note that the averaging method implies implementation problems for non-rectangular design spaces.

The implemented optimisation software provides both visualisation methods. If the metamodel is already optimised, the removed design variables will not be fixed at their midrange but at the calculated optimal value for that design variable. The optimisation of the metamodels is introduced in Section 4.2.5. The averaging method can be applied without optimising the metamodel first. Some specific trends, e.g. due to interaction between several design variables, however, may be averaged out [117].

4.2.4 Metamodel validation

After the seven metamodels have been fitted for a certain response, they need to be validated. Again a distinction is made between RSM and DACE. Validation occurs with respect to metamodel accuracy as well as the assumptions underlying both RSM and

DACE.

RSM

Appendix A.3 thoroughly describes several metamodel validation techniques for both accuracy and assumption assessment. In this section we will shortly indicate the validation measures implemented in the optimisation algorithm for metal forming processes.

In case of RSM, the metamodel accuracy is generally assessed by *ANalysis Of VAriance (ANOVA)* [90]. ANOVA is thoroughly discussed in Appendix A.3. Its output is summarised by an ANOVA table, of which an example is presented in Table A.2 in the appendix, and by mentioning the R^2 and R^2_{adj} values. R^2 values that approach a value of 1 indicate accurate metamodels.

Additionally, the $(1-\alpha)\%$ confidence intervals of the regression coefficients $\boldsymbol{\beta}$ can also provide useful validation information: if a confidence interval includes 0, the corresponding design variable is likely to be not significant with respect to the response. Hence, if all regression coefficients corresponding to the quadratic terms in an elliptic metamodel are not significant, a linear metamodel is sufficiently accurate to describe the response.

A final accuracy measure implemented for RSM is the PRediction Error Sum of Squares or PRESS statistic [90]. It is based on Cross Validation (CV), a well-known validation assessment for DACE, which is introduced in Appendix B.3. It is, however, also useful for RSM and produces a predicted R^2 value, R^2_{pred}, and a predicted adjusted R^2 value, $R^2_{\text{pred,adj}}$, which can provide information on how well an unknown point will be predicted. A metamodel having an ordinary R^2 and a predicted R^2_{pred}, which are both close to 1, is very likely to provide a good fit to the response data [90].

Next to accuracy assessment it is also important to gain a feeling whether RSM is rightfully applied or not, especially since we are applying a method originally developed for stochastic measurements to deterministic computer experiments. The assumptions underlying the use of RSM are mentioned extensively in Appendix A.2 and ways for validating these assumptions are introduced in Appendix A.3. For the optimisation of metal forming processes, we propose to validate the assumptions by investigating scatter plots, residual plots, normal probability plots and autocorrelation plots as displayed in the Figures A.2, A.3 and A.4, respectively. The residual plots should not reveal any trend and indicate a mean of 0. The same can be said for the autocorrelation plot and the normal probability plot should indicate a normal distribution.

DACE

DACE is typically validated by Cross Validation (CV), which means visualising a CV plot as presented in Figure B.3. The metamodel is accurate if the measurements lie exactly

on a straight line. The Cross Validation Root Mean Squared Error ($\mathrm{RMSE_{CV}}$) given in Equation B.19 is a quantitative measure for how accurate the metamodel is: an $\mathrm{RMSE_{CV}}$ of 0 indicates the perfect metamodel. However, if this error is not 0, the question remains whether the metamodel is accurate enough. This question is answered by Schonlau [117] who proposes to plot a residual plot with the standardised CV error given by Equation B.20. If the standardised CV errors stay within the domain [-2,2] or [-3,3] in case of many measurements, the metamodel is predicted to be accurate. The resemblance with the PRESS statistic mentioned for RSM is emphasised, which means that it is also possible for DACE to calculate an R^2_{pred} and $R^2_{\mathrm{pred,adj}}$. A value close to 1 indicates an accurate metamodel [82, 84].

One of the basic assumptions of DACE is that the stochastic process Z is a zero mean Gaussian stochastic process, which means it is stationary by definition and is multivariate normally distributed [140]. Following Schonlau [117], it is proposed to validate this assumption by investigating if the standardised CV error plot reveals any trend and if the mean is 0, and if the Q-Q plot presented in Figure B.4 indicates normality.

By the validation measures for RSM and DACE summarised above, the user is offered a number of assessment techniques, which may help him or her to determine the best metamodel, either RSM or DACE, for each response. If none of the metamodels is satisfactory, additional measurements may be performed by selecting an additional number of experimental design points based on the maximin criterion introduced in Section 4.2.2. The new simulations need to be performed and new metamodels are fitted and validated as described previously. This procedure can continue until a satisfactory, accurate metamodel is obtained.

However, obtaining an accurate metamodel is not always necessary if the main objective of the metamodelling is optimisation (compare the different goals of metamodelling mentioned in Section 3.1). Another approach is to choose the best metamodel from the seven RSM and DACE metamodels fitted by the optimisation algorithm based on the above validation measures. In this case, the metamodel is not accurate at all and the optimisation of the metamodel will result in a very rough approximation of the real optimum. After the optimisation, it is still possible to select additional DOE points based on the maximin criterion, but now, one can also incorporate the rough knowledge on the location of the optimum resulting from the optimisation of the metamodel. We believe that this method will be far more efficient for obtaining an approximate optimum fast, whereas the former method will be more accurate, though much more expensive. Since the optimisation using expensive nonlinear FEM calculations will already be time consuming, every additional saving of time is welcome for keeping the algorithm applicable for industrial practice. If one has time, however, the algorithm also offers the option to choose the time consuming approach.

4.2.5 Metamodel optimisation

After (accurate or less accurate) metamodels have been fitted for each response (objective function, implicit constraints), the optimisation problem is complete. It comprises the design variables and explicit constraints, which were already known, and the selected metamodels provide explicit mathematical functions of the objective function and implicit constraints depending on the design variables only. The next step presented in Figure 4.8 is optimising the metamodel of the objective function subjected to the explicit constraints and the metamodels of the implicit constraints. Any classical iterative or machine learning optimisation algorithm can be used for this. The number of function evaluations is not that important anymore since the metamodels can be evaluated many times within a fraction of a second. A smart choice for a specific algorithm depends on the shape of the objective function and possible constraints.

For the optimisation of metal forming processes, it is wise to reckon with rather nonlinear responses and thus complex metamodels. Complex metamodels imply local optima, thus, as a metamodel optimisation algorithm, genetic or evolutionary algorithms that are capable of finding a global optimum seem to be a proper choice. These algorithms were introduced shortly in Section 4.1.4 and were amongst others utilised to optimise metamodels by Büche [17, 18]. However, genetic and evolutionary algorithms are less widely available than classical iterative optimisation algorithms. We chose to implement an iterative Sequential Quadratic Programming (SQP) algorithm available in MATLAB®'s Optimization Toolbox [133]. Since iterative algorithms tend to get stuck in local optima, a multistart approach was implemented: the SQP algorithm will be started at more than one location, in this case all DOE points. Such a *multistart* approach is also reported in [58, 117, 129]. Compared to a genetic algorithm more function evaluations will be necessary, but this is not a problem since evaluating a metamodel is very time efficient. For more information on SQP algorithms, see one of the many texts on optimisation, e.g. [12, 25, 98, 102, 106, 131]. The specific algorithm implemented in MATLAB® is called "fmincon" and is described in MATLAB® Help [133].

The optimisation using the metamodels results in the optimal settings of the design variables, which are subsequently used as an input for a final nonlinear Finite Element calculation to evaluate how accurate the obtained approximate optimum is. The difference between the optimum predicted by the metamodel and the real value of the response for the same design variable settings is a measure for the accuracy of the obtained results. The combination of accurate metamodels with a small deviation of the approximate optimum w.r.t. the real response with the optimised design variable settings, indicates the global optimum is accurately determined. If either the metamodels are inaccurate or the deviation is large, it is recommended to sequentially improve the results.

4.2.6 Sequential improvement

If the results after optimisation are not accurate, it is possible to improve these results sequentially. The user can be dissatisfied with the outcome after the first step of the optimisation algorithm due to two reasons:

1. The metamodels are not accurate
2. The approximate optimum deviates significantly from the real value of the objective function obtained by running a FEM calculation for the optimised design variable settings

The second item generally implies the first one: accurate metamodels will produce an accurate approximation of the optimum, thus, an inaccurate optimum is likely to be caused by inaccurate metamodels. However, the other way around does not have to be the case essentially. The first item does not imply the second one: inaccurate metamodels may coincidentally result in a small deviation between the approximate response and the true response at a certain approximate optimum. Although one of the criteria is met, the obtained approximate optimum will in this case probably not present the real global optimum. Hence, it is advised to continue improving the results sequentially until both criteria are satisfied.

Two things can be improved in this context:

- The metamodel(s)
- The prediction of the global optimum

For the metamodel based optimisation algorithm, four different sequential improvement strategies are considered:

1. Metamodel improvement without zooming
2. Metamodel improvement with Zooming and Resampling (Z+R)
3. Rapid zooming near the global optimum by Minimising a Merit Function (MMF)
4. Rapid zooming near the global optimum based on Maximum Expected Improvement (MEI)

The first two methods are aiming at improving the metamodels, whereas the third and fourth methods are mainly focussing on finding the global optimum as fast as possible, regardless whether the metamodels used are accurate or not.

Metamodel improvement without zooming

The first approach is an expensive one, needing relatively many measurements. It is especially useful if the goal of metamodelling was to develop a metamodel for replacing the time consuming FEM calculations, i.e. one is interested in accurately predicting the response from certain design variable settings (the second goal in Section 3.1). For this method, the original design space is kept and new DOE points are added based on the maximin criterion described in Section 4.2.2. New simulations are run for the new experimental design points, metamodels are fitted, validated and the best ones are selected again. This procedure is repeated until the metamodels for all responses are predicted to be accurate. Optimisation using the metamodels is not necessary within this sequential approach: it is not until the metamodels are predicted to be accurate before optimisation of the metamodels is performed. However, optimisation after each batch allows for evaluating the approximate optimum after this batch. This may enhance the efficiency of this sequential improvement strategy since an inaccurately predicted optimum is found to be inaccurate immediately in the next batch. Metamodel improvement without zooming is schematically presented in Figure 4.11(a). The crosses are part of the initial DOE, whereas the circles denote the DOE points that are added by the sequential improvement strategy.

Metamodel improvement with Zooming and Resampling (Z+R)

The second approach seems intuitively computationally cheaper and is hence perhaps more useful when the premium goal is to use metamodelling for optimisation (the third goal in Section 3.1). After each sequential step of the optimisation algorithm, the best metamodels for each response are selected, although they may be quite inaccurate, and optimised. This will result in an approximate optimum, which will be more or less inaccurate, depending

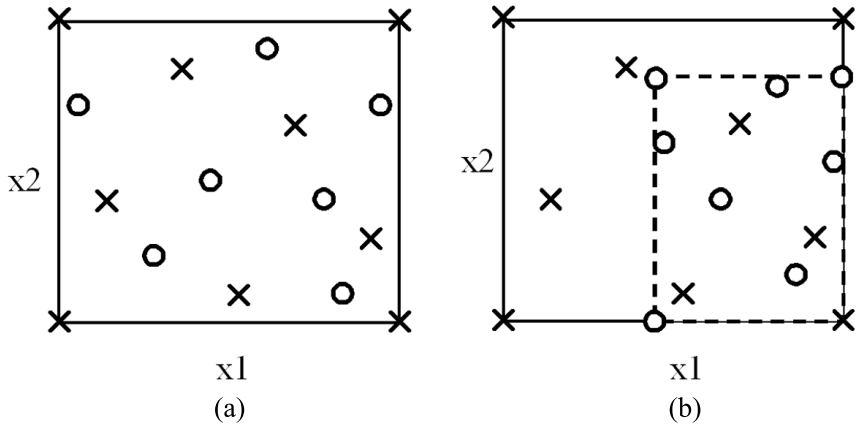


Figure 4.11: (a) Metamodel improvement without zooming; (b) Metamodel improvement with Zooming and Resampling

on how accurate the metamodels were at that stage of the sequential optimisation procedure. In contradiction to the first approach for sequential improvement, this optimisation using inaccurate metamodels will provide more or less coarse information on where the real optimum is likely to be located. The next step is to use this information to reduce or shift the design space to a region closer to the expected optimum. During the subsequent step, a smaller design space will be present, which still incorporates the expected optimum. Since a smaller design space implies that less response data are needed to fit an accurate metamodel, less time consuming FEM calculations need to be run to obtain two things: an accurate metamodel in the neighbourhood of the optimum, and hence a small deviation between the approximate optimum and the real objective function value obtained from the optimised design variable settings. Thus, the second approach will probably converge faster to the global optimum than the method without zooming. The method is presented schematically in Figure 4.11(b). The dashed line denotes the zoomed-in design space.

For the second approach, the choice how to zoom in near the optimum is critical. In [35], the design space is simply decreased by a factor 1/2. Santner [114] reports on a method that omits less significant design variables, which zooms by decreasing the design space's dimension. In Chapter 5 of this report and [13, 14], we use a rather ad hoc way of zooming based on visualisation of the metamodels and subjective user judgment.

Constraining ourselves to a minimisation problem, another more structured way for zooming is to exclude all points from the design space for which the predicted objective function value minus the predicted metamodel error exceeds the predicted optimum plus the predicted error of the optimum. I.e. for RSM:

$$\hat{y}_0 - w\sqrt{\mathrm{var}(\hat{y}_0)} \leq \hat{y}^* + w\sqrt{\mathrm{var}(\hat{y}^*)} \tag{4.19}$$

$\hat{y}_0$, $\hat{y}^*$, $\mathrm{var}\,(\hat{y}_0)$ and $\mathrm{var}\,(\hat{y}^*)$ follow from the Equations 4.4 and 4.5. w is a weight function that can be used to tune the relative importance between the objective function values and the predicted errors. A value of $w = 0$ will only take into account the current global optimum of the objective function, whereas an increasing value of w will widen the new design space more and more. A similar formula can be applied for DACE:

$$\hat{y}_0 - w\sqrt{\mathrm{MSE}(\hat{y}_0)} \leq \hat{y}^* + w\sqrt{\mathrm{MSE}(\hat{y}^*)} \tag{4.20}$$

where $\hat{y}_0$, $\hat{y}^*$, $\mathrm{MSE}\,(\hat{y}_0)$ and $\mathrm{MSE}\,(\hat{y}^*)$ can be calculated by the Equations 4.13 and 4.14.

Note that problems will arise when multiple optima are present. In this case, two options of applying the Z+R method are:

1. Include all optima separately

2. Include all optima at once

The first alternative implies all optima have to be dealt with sequentially. Using the second approach, it is not possible to zoom in further than to a design space that includes all optima, which can be generally quite large. Both approaches are quite inefficient in case multiple optima are present.

Zooming by Minimising a Merit Function (MMF)

Both sequential improvement strategies treated above were mainly focussing on improving the metamodels. Of course, an increasing accuracy of the metamodel will generally also imply an improved accuracy of the optimum. This section and the next one introduce two methods that are not focussed on fitting accurate metamodels as a whole or in a subspace of the feasible domain. They are designed to find the global optimum directly rather than to fit an accurate metamodel first.

The first of these methods selects the new points at which function evaluations need to be performed by minimising a so-called merit function. The merit function that is proposed is the one already seen in the previous section for the Z+R method:

$$f_{\text{merit}} = \hat{y} - w\text{RMSE}(\hat{y}) \tag{4.21}$$

Emmerich et al. [34, 35] use the same merit function for making evolutionary algorithms more efficient. They propose a "Metamodel-Assisted Evolution Strategy", which is applied to forging processes by Fourment, Do et al. [32,41,42] and to the Bridgman Casting process by Jakumeit et al. [35, 51]. The basic concept is displayed in Figure 4.12(a). Suppose the evolutionary strategy indicates it wants to obtain the real values of the objective function at the circles. In a previous iteration, an initial Kriging metamodel was already fitted through the data denoted by the crosses. This implies that the RMSE can also be predicted throughout the design space using Equation 4.14. Suppose we are now interested in running only two calculations instead of four. Then the metamodel can be used to select those two points at which the merit function value is the lowest in case of a minimisation problem. Hence, simulations will be run for the two lowest circles in Figure 4.12(a). In this way, one can let the evolutionary strategy zoom efficiently near the global optimum.

Alternatively, Torczon and Trosset [136] propose to minimise a merit function instead of only using it to select the best of a couple of alternative design points. Instead of the RMSE in Equation 4.21, they use the distance between a possible new candidate point and an already evaluated point as a measure for the error of the metamodel. Based on this approach, Büche [17, 18] uses the RMSE of Kriging-like Gaussian Processes as formulated by McKay [85] instead of the distance. Büche determines the minimisation of the merit function by an evolutionary strategy. His application is the optimisation of the design of gasturbine blades including as many as 19 design variables.

For the metamodel based optimisation algorithm for metal forming processes, it is proposed to use a similar method that minimises the merit function given in Equation 4.21. The procedure is as follows:

1. Start with a DOE of approximately 10 x the number of design variables. This rule of thumb is recommended by Kriging specialists to fit at least a reasonably accurate metamodel [117]. Run the FEM simulations
2. Fit the seven metamodels (RSM and Kriging)
3. Optimise the best metamodels using the multistart SQP algorithm
4. Minimise the merit function (Equation 4.21) using the multistart SQP algorithm
5. Select only the minima of the merit function that satisfy $\hat{y} - w\text{RMSE}(\hat{y}) \leq \hat{y}^* + w\text{RMSE}(\hat{y}^*)$ as the new DOE points
6. Run the calculations and return to step 2

The method is presented in Figure 4.12(b). After having obtained an initial metamodel from the calculations that resulted in the crosses, the merit function shown as the dotted line is minimised. This results in a number of DOE points shown as the circles. However, to avoid wasting expensive calculation time for locating local optima, only those DOE points are selected that are predicted to be underneath the level $\hat{y}^* + w\text{RMSE}(\hat{y}^*)$ as shown in Figure 4.12(b), in which $\hat{y}^*$ denotes the global optimum predicted by the metamodel of the objective function. The FEM calculations for the remaining DOE points can be run in parallel.

Method of Maximum Expected Improvement (MEI)

With respect to sequential improvement towards the global optimum, Schonlau [117] proposes a method making use of Kriging and the assumption that an untried point is normally

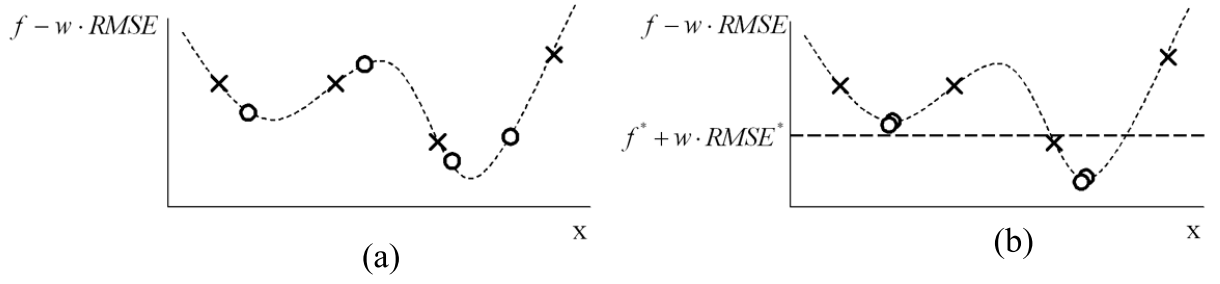


Figure 4.12: (a) Metamodel-Assisted Evolutionary Strategy; (b) Metamodel improvement by Minimising a Merit Function

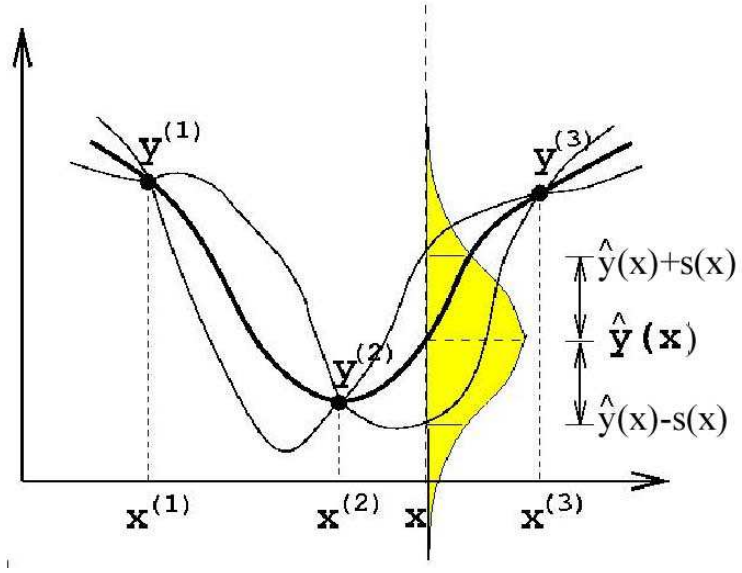


Figure 4.13: The fitting of a Kriging metamodel for MEI

distributed as shown in Figure 4.13. This method is also reported in [58, 59, 114, 115] and is similar to the concept of minimising a merit function.

The method starts by defining *Improvement* as [117]:

$$\begin{aligned} I &= f_{\min} - y \qquad \text{if} \quad y < f_{\min} \\ I &= 0 \qquad \text{otherwise} \end{aligned} \tag{4.22}$$

where $f_{\min}$ is the lowest objective function value obtained by the initial DOE and y is a possible new outcome of a function evaluation. Clearly, if $y < f_{\min}$, the situation has improved. Otherwise not and I is set to zero. In general, the expected value of a stochastic variable X is defined as [104]:

$$E(X) = \int_{-\infty}^{\infty} xp(x)\mathrm{d}x \tag{4.23}$$

in which x is a possible value of X and $p(x)$ is the probability that X actually obtains this value. $p(x)$ is the probability density function. Combination of Equations 4.22 and 4.23 yields the *Expected Improvement*:

$$E(I) = \int_{-\infty}^{f_{\min}} (f_{\min} - y)\phi(y)\mathrm{d}y \tag{4.24}$$

where $\phi(y)$ is used for the probability density function. Now y can be replaced by the Kriging predictor $\hat{y}$ and Equation 4.24 may be rewritten to [59, 117]:

$$\begin{aligned} E(I) &= (f_{\min} - \hat{y})\,\Phi\left(\frac{f_{\min} - \hat{y}}{s}\right) + s\phi\left(\frac{f_{\min} - \hat{y}}{s}\right) \qquad \text{if} \quad s > 0 \qquad (4.25) \\ E(I) &= 0 \qquad \text{if} \quad s = 0 \end{aligned}$$

where $\hat{y}$ is the Kriging metamodel and s its standard deviation as depicted in Figure 4.13. s equals the RMSE value mentioned before and is calculated as the square root of Equation 4.14. Since it is assumed that an untried point is normally distributed, ϕ and Φ denote the probability density and the cumulative distribution functions of the standard normal distribution, which are shown in Figure 4.14. Several things can be obtained from these figures:

1. If the improvement $I = f_{\min} - \hat{y}$ is large and positive and s is small, then $\Phi \approx 1$ and $\phi \approx 0$. Thus, the left part of Equation 4.25 will be large and the right part will be approximately 0: $E(I)$ is large and this point is a good candidate for evaluation

2. If the improvement $I = f_{\min} - \hat{y}$ is large and negative and s is small, then $\Phi \approx 0$ and $\phi \approx 0$. Thus, both parts of Equation 4.25 will be approximately 0: $E(I)$ is close to 0 and this point is NOT good a candidate for evaluation

3. If the improvement $I = f_{\min} - \hat{y}$ is close to 0 and s is large, then $\Phi \approx 0.5$ and $\phi \approx 0.4$. Thus, the left part of Equation 4.25 will be approximately 0 and the right part will be large (since s is large): $E(I)$ is large and this point is a good candidate for evaluation

4. For the other cases, $E(I)$ depends on the relative proportion of $f_{\min} - \hat{y}$ w.r.t. s

Schonlau [117] proposes to maximise $E(I)$ in Equation 4.25 to yield the point promising the Maximum Expected Improvement (MEI). Only this point is subsequently evaluated.

For the metamodel based optimisation algorithm for metal forming processes, it is proposed to use a similar method that maximises Equation 4.25. The procedure is as follows:

1. Start with a DOE of approximately 10 x the number of design variables. This rule of thumb is recommended by Kriging specialists to fit at least a reasonably accurate metamodel [117]. Run the FEM simulations

2. Fit the seven metamodels (RSM and Kriging)

3. Optimise the best metamodels using the multistart SQP algorithm

4. Maximise the Expected Improvement using the multistart SQP algorithm. Two remarks:

   - Since the SQP algorithm is a minimisation algorithm, the negative of Equation 4.25 is minimised

- Starting at the DOE points yields 0 Expected Improvement. Additionally, Equation 4.25 has quite local optima, which are difficult to locate [59]. To overcome these problems, the SQP algorithm is initialised at a new set of N x k points generated by a maximin Latin Hypercubes Design. k is the number of design variables, N the number of DOE points. The design for determining the Maximum Expected Improvement is generally a larger design than the 10 x k design used for step 3. Hence, there is a larger chance that the quite local optima are obtained

5. To exploit the possibility for parallel computing all optima of Equation 4.25 are selected for function evaluation instead of only one point as proposed by Schonlau [117]

6. Run the calculations and return to step 2

The method of MEI is similar to the method of MMF introduced in the previous section. It will select new DOE points in the neighbourhood of the global optimum or on locations where a high RMSE value is predicted. An advantage over MMF is that no choice needs to be made for the weight factor w. On the other hand, if the metamodels are not indicating the right region of the global optimum, MEI is known to look exhaustively near the erroneously predicted global optimum, before it finally decides to look at other locations. This is indicated by Jones et al. [58], who provide a very interesting comparison between global optimisation methods based on metamodels.

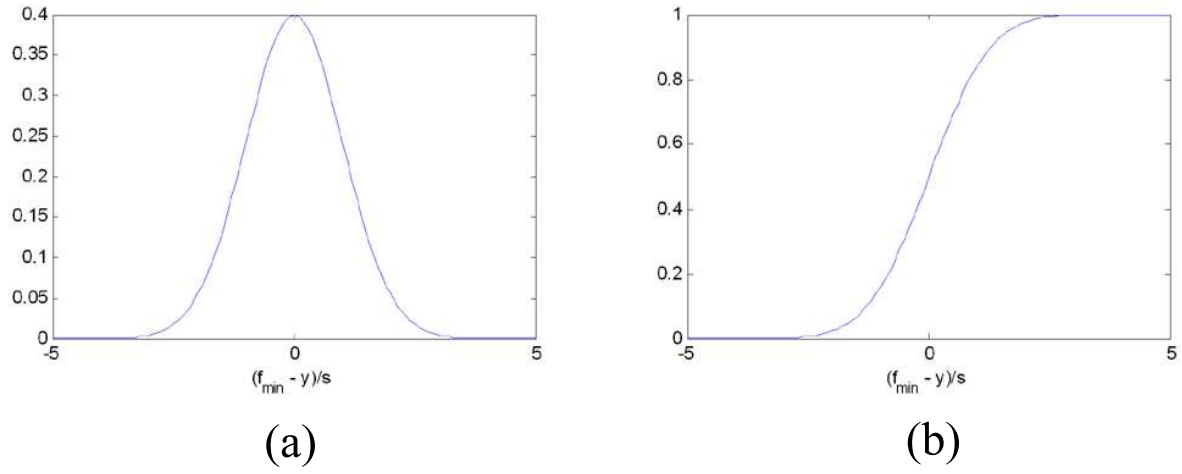


Figure 4.14: Standard normal distribution: (a) Probability density function; (b) Cumulative distribution function

A comparison between sequential improvement strategies

After having proposed four possible sequential improvement strategies, the question is which one is most recommendable. For answering this question, and studying the effect of the weight factor w in the Z+R and MMF strategies, the optimisation algorithm including the four sequential improvement strategies was subjected to two analytical test functions in Appendix C. Here, the conclusions and rules of thumb obtained in this appendix are repeated.

It can be concluded that:

1. The Zooming and Resampling (Z+R) strategy is less efficient than sequential improvement without zooming except for very low values of the weight factor w
2. The Z+R strategy tends to get stuck in local optima for low values of w
3. The Minimising a Merit Function (MMF) strategy is more efficient than sequential improvement without zooming
4. The accuracy of the MMF strategy seems to be insensitive to the value of the weight factor w
5. If the initial metamodel is able to locate the region of the global optimum, a low value of the weight factor w is the most efficient
6. MEI showed to be the most efficient sequential improvement strategy, although it approximated the optima slightly less accurately than the other methods

These conclusions lead to the following rules of thumb:

1. Fit an initial metamodel from about 10 times the number of design variables as initial points
2. Use the MEI or MMF method as sequential improvement strategy
3. When using MMF, set the value of the weight factor to 1

4.3 Conclusions

This chapter presented both an overview of the field optimisation of metal forming processes and a proposal for a metamodel based optimisation algorithm, which can be applied to metal forming processes using time consuming nonlinear FEM calculations.

Five types of optimisation algorithms are encountered within metal forming literature:

- Classical iterative algorithms
- Approximate algorithms
- Adaptive algorithms
- Algorithms based on machine learning
- Combinations of the above

Approximate optimisation algorithms, which use metamodelling techniques discussed in Chapter 3, are preferred for the optimisation of metal forming processes using time consuming FEM calculations for five reasons:

- They are time efficient since they allow for parallel computing
- They try to locate the global optimum, instead of just a local one
- It is not necessary to calculate sensitivities from a Finite Element model
- They offer a black box approach: it is not needed to have access to the FE software source code and any package may be used
- The visualisation of the metamodels offers insight in metal forming phenomena

A metamodel based sequential approximate optimisation algorithm for metal forming processes was proposed in this chapter. The algorithm combines Response Surface Methodology (RSM) and Design and Analysis of Computer Experiments (DACE) as metamodelling techniques. The optimisation algorithm comprises tools for selecting design variable settings for which FEM simulations need to be run by offering an advanced Design Of Experiments (DOE) strategy, constructing the RSM and DACE metamodels, validating the metamodels and finally optimising these metamodels to result in an approximate optimum. This approximate optimum may be validated by running a final FEM calculation using the optimised design variable settings and interpreting the deviation between the real response value and the approximated optimum response value. If either this deviation or the metamodels are found to be inaccurate, a sequential procedure may be followed to improve the results by repeating the above procedure until the user is satisfied with the results. Four methods for sequential improvement have been suggested and investigated. It is recommended to use either the method of Minimising a Merit Function (MMF) or the method of Maximum Expected Improvement (MEI) since these appear to be most efficient. Both methods focus on finding the global optimum as soon as possible rather than obtaining accurate metamodels.

Chapter 5

Demonstration of Concept: Application to hydroforming

The Demonstration of Concept (DoC) is a simple hydroforming process to demonstrate that metamodel based optimisation algorithms are indeed applicable to metal forming processes. The hydroforming process and the optimisation problem to be optimised are presented in Section 5.1. The results of the optimisation of the DoC are presented in Section 5.2. The optimisation algorithm uses metamodels and the final result of the optimisation depends on the accuracy of these metamodels. Therefore, Section 5.3 compares the metamodel used for optimisation to 1000 real objective function values to assess the metamodel's accuracy. It is possible to formulate the objective function to be introduced in Section 5.2 in an alternative way that may reduce the number of expensive FEM calculations needed to obtain the same accuracy of the metamodels. Section 5.4 describes this alternative approach. It also compares the results of this approach to the results of the original objective function and to the real objective function values obtained by running the 1000 FEM calculations. Conclusions are being formulated in Section 5.5.

5.1 A simple hydroforming process

The hydroforming product used for the Demonstration of Concept (DoC) is shown in Figure 5.1(a). The product is axisymmetric and can be modelled in 2D as presented in Figure 5.1(b). Figure 5.1(b) also presents the dimensions of the product.

For the DoC, we are interested in optimising the time variation of the internal pressure p and axial feeding u. These are process parameters for the hydroforming process. A typical time dependent load path for hydroforming is shown in Figure 5.1(c). Assuming a strain rate independent material (α is irrelevant), three design variables are remaining: the time when axial feeding starts t_1, the time when axial feeding stops t_2 and the total amount of axial feeding $u_{\max}$.

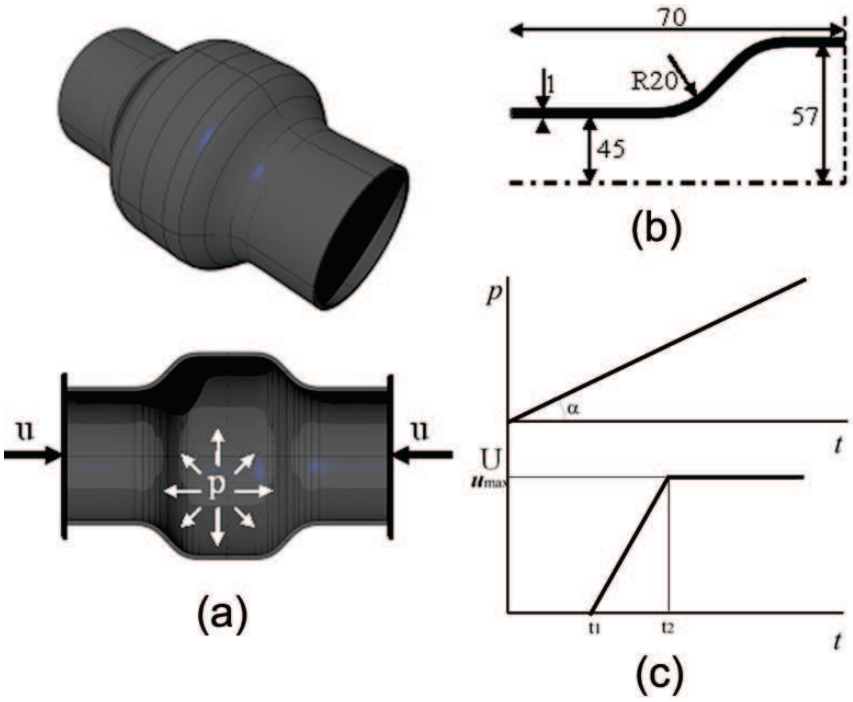


Figure 5.1: (a) Hydroformed product that is used as DoC; (b) 2D axisymmetric FE model and dimensions; (c) Typical load paths for hydroforming

As an optimisation objective, it was chosen to minimise variations in the wall thickness of the final product with respect to the initial tube thickness. One implicit and two explicit constraints were formulated. The implicit constraint ensures that the final product fills out the die nicely, one explicit constraint makes sure that the time when axial feeding stops is larger than the time when it starts and the last constraint was formulated to overcome convergence problems of the FEM calculations when t_2 approaches t_1 and the amount of axial feeding is high (large $u_{\max}$). Methods to handle non converged simulations are lacking and is a field of open research. The total optimisation problem is modelled as follows:

$$\begin{aligned} \min f\left(t_1, t_2, u_{\max}\right) &= \left\| \frac{h - h_0}{h_0} \right\|_2 \\ \text{s.t.} \qquad g_{\text{impl}} &= V \leq 0 \\ g_{\text{expl1}} &= t_1 - t_2 \leq 0 \\ g_{\text{expl2}} &= u_{\max} - 9\left(t_1 - t_2\right) \leq 0 \\ 0\,\text{s} &\leq t_1 \leq 5\,\text{s} \\ 2.5\,\text{s} &\leq t_2 \leq 10\,\text{s} \\ 0\,\text{mm} &\leq u_{\max} \leq 9\,\text{mm} \end{aligned} \tag{5.1}$$

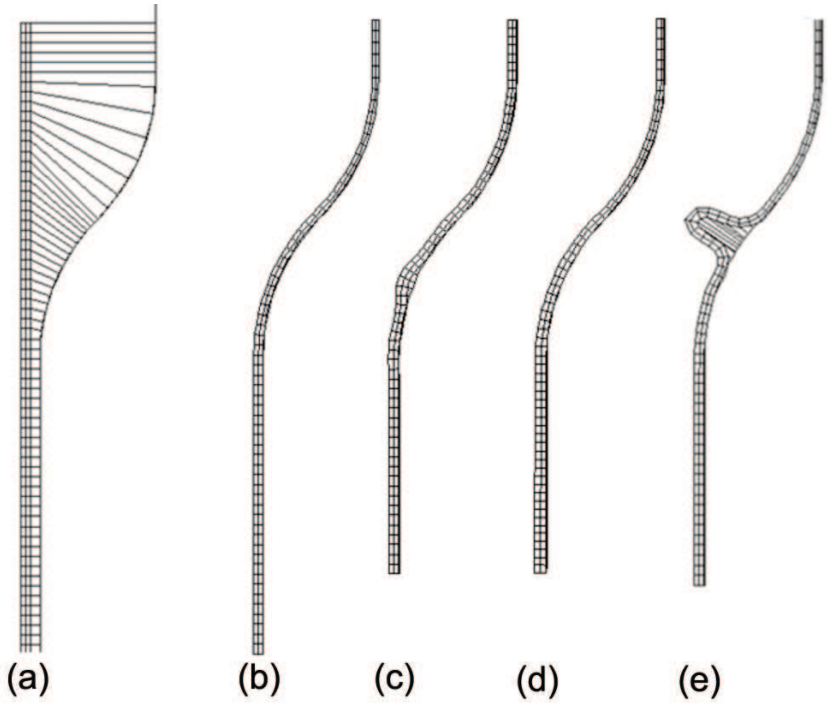


Figure 5.2: (a) FE model of the initial tube; (b-e) Final product formed with several arbitrarily selected load paths

where h is the final wall thickness at a certain location in the hydroformed product, h_0 is the wall thickness of the initial tube and V is the volume between the final product and the die. If this volume is larger than zero, there is a gap between the final product and the die and the final shape of the product is not satisfactory.

To get a feeling for the problem, let us look how the product deforms for several settings of the design variables. The 2D axisymmetric FE model of the initial product is shown in Figure 5.2(a). Note that the model is rotated 90° with respect to Figure 5.1(b). The

Product	t_1(s)	t_2(s)	$u_{\max}$(mm)	f	g_{impl}
(a)	–	–	–	0	–
(b)	0	0	0	1.39	-0.29
(c)	0	3	9	0.52	1.79
(d)	0	10	9	1.42	-0.34
(e)	4.8	6.2	7.7	1.37	32.64

Table 5.1: Design variable settings and response values

contact between the product and the die is modelled by contact elements, which have no stiffness when there is no contact and a very high stiffness when contact between the product and the die is established. The FEM calculations were run using the FE code DiekA.

Figures 5.2(b) through (e) present some final products deformed with arbitrary load paths. The design variable settings for t_1, t_2 and $u_{\max}$ and the response values for the objective function f and the implicit constraint g_{impl} are presented in Table 5.1. Note that product (a) is the initial undeformed product, which is seen as the product with the perfect wall thickness distribution by the objective function quantified in Equation 5.1. For the perfect product, the objective function equals 0. Also note that products (c) and (e) do not satisfy the implicit constraint g_{impl}, which can also clearly be seen from Figures 5.2(c) and (e).

5.2 Optimisation of hydroforming

After having modelled the optimisation problem, it is now solved by the optimisation algorithm presented in Chapter 4. The DOE strategy introduced in Section 4.2.2 was applied to

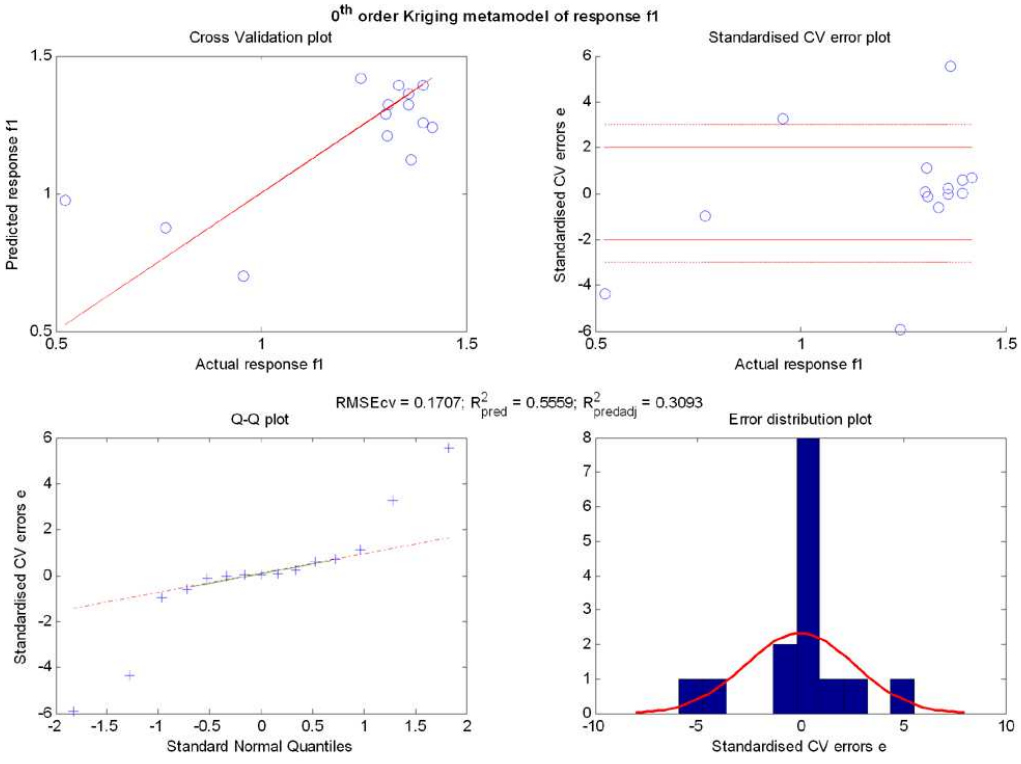


Figure 5.3: Metamodel validation of the 0^{th} order Kriging metamodel for the objective function after batch 1

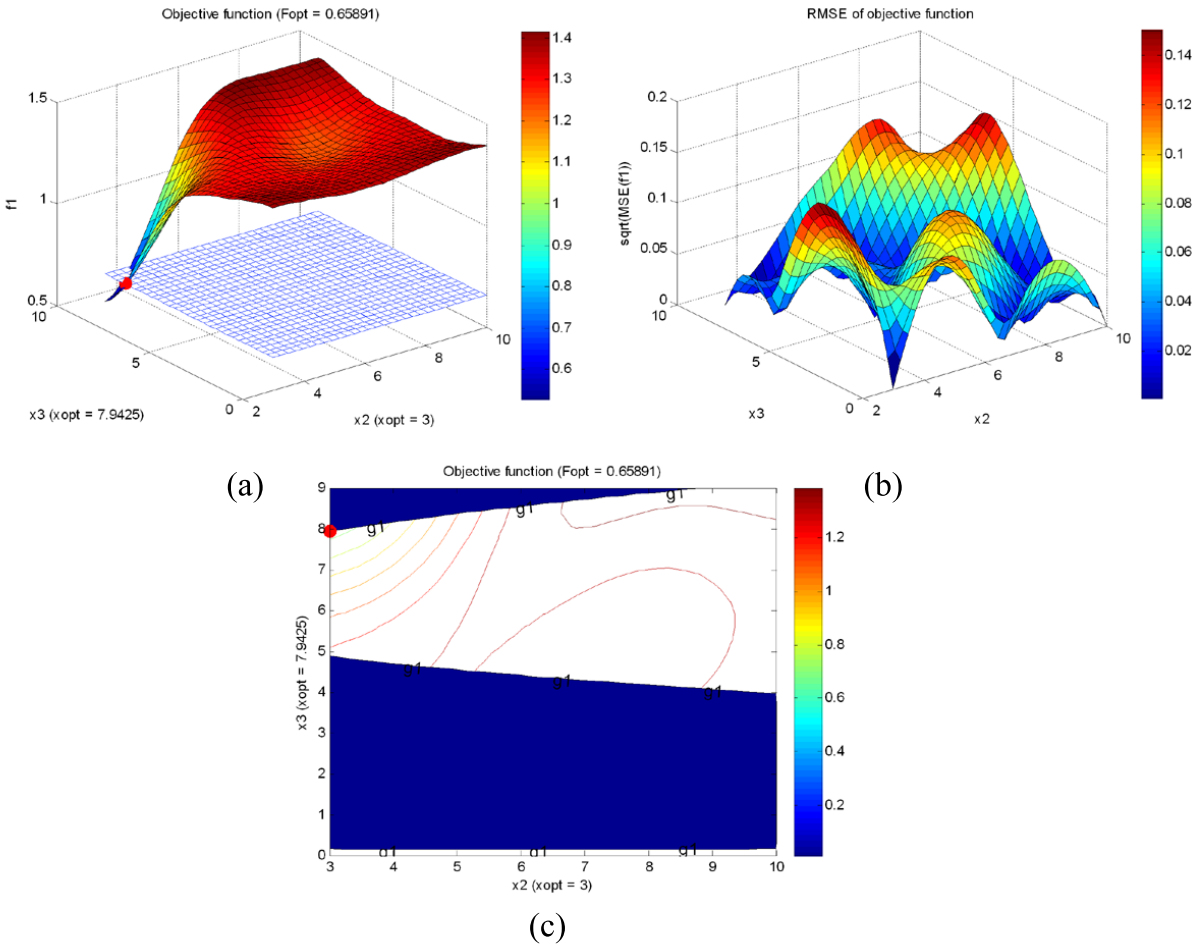


Figure 5.4: (a) Metamodel of the objective function after batch 1; (b) Metamodel of the RMSE of the objective function after batch 1; (c) Contour plot of the objective function after batch 1

generate 16 initial design variable settings for which 16 FEM calculations were performed with the FE code DiekA. The calculations were performed on 16 parallel processors, which limited the total time to the run time of one calculation, i.e. a couple of minutes for the 2D FE model we are considering. Subsequently, the four Response Surfaces and three Kriging metamodels were fitted for both responses (the objective function and the implicit constraint).

Figure 5.3 presents the metamodel validation of the 0^{th} order Kriging metamodel for the objective function. Although the several plots and the RMSE_{CV} and predicted R^2-values indicate that the metamodel is not accurate, it was found to be better than the other six

# FEM	t_1(s)	t_2(s)	$u_{\max}$	f_{opt}	f_{actual}
16	0	3.0	7.9	0.66	0.47
32	0	1.5	7.9	0.37	—
48	0.8	2.5	9.0	-0.06	0.55
64	0	3.0	7.7	0.31	0.50
80	0	3.1	8.2	0.37	0.49
96	0	2.5	8.3	0.30	0.37

Table 5.2: Optima of the DoC after the 6 batches of 16 FEM calculations each

metamodels. Thus, this 0^{th} order Kriging metamodel was included in the optimisation model. In a similar way, a 1^{st} order Kriging metamodel was identified to be most accurate for the implicit constraint and was included in the optimisation problem.

The optimisation model was subsequently optimised using the multistart SQP algorithm described in Section 4.2.5. Several local optima were observed; the global optimum was located at $(t_1, t_2, u_{\max}) = (0, 3, 7.9)$ and the corresponding objective function value was observed to be 0.66. Figure 5.4(a) shows the approximate optimum located on the metamodel of the objective function. The metamodel is depicted dependent on $x_2 = t_2$ and $x_3 = u_{\max}$ at the constant, optimal level of $t_1 = 0$. Figure 5.4(b) presents the RMSE of the objective function and Figure 5.4(c) shows a contour plot of the objective function and the constraints, where one can easily observe that the optimum is constrained by the implicit constraint and the box constraint $t_2 \geq 3$.

To validate the optimum, a FEM calculation was performed with the optimal design variable settings. The actual objective function value was found to be 0.47. Both the large difference between the approximate and the actual objective function values at the approximate optimum and the inaccurate metamodels of the objective function and the implicit constraint motivate to sequentially improve the metamodels. A preliminary form of the Z+R zooming algorithm introduced in Section 4.2.6 was used. The algorithm zooms in near the optimum to fit a more accurate metamodel in the direct neighbourhood of the global optimum. However, the structured way of zooming using merit functions was at the time not present. This may have reduced the efficiency or the robustness of the algorithm. The algorithm used is nevertheless still capable of evaluating the potential of metamodel based optimisation algorithms in a general context.

In total, six batches of each 16 FEM calculations were performed. Each time, the design space was reduced and/or shifted with the aim to construct accurate metamodels in the vicinity of the optimum. The results of all batches are presented in Table 5.2.

Figure 5.5 shows the validation of the 1^{st} order Kriging metamodel that best represented

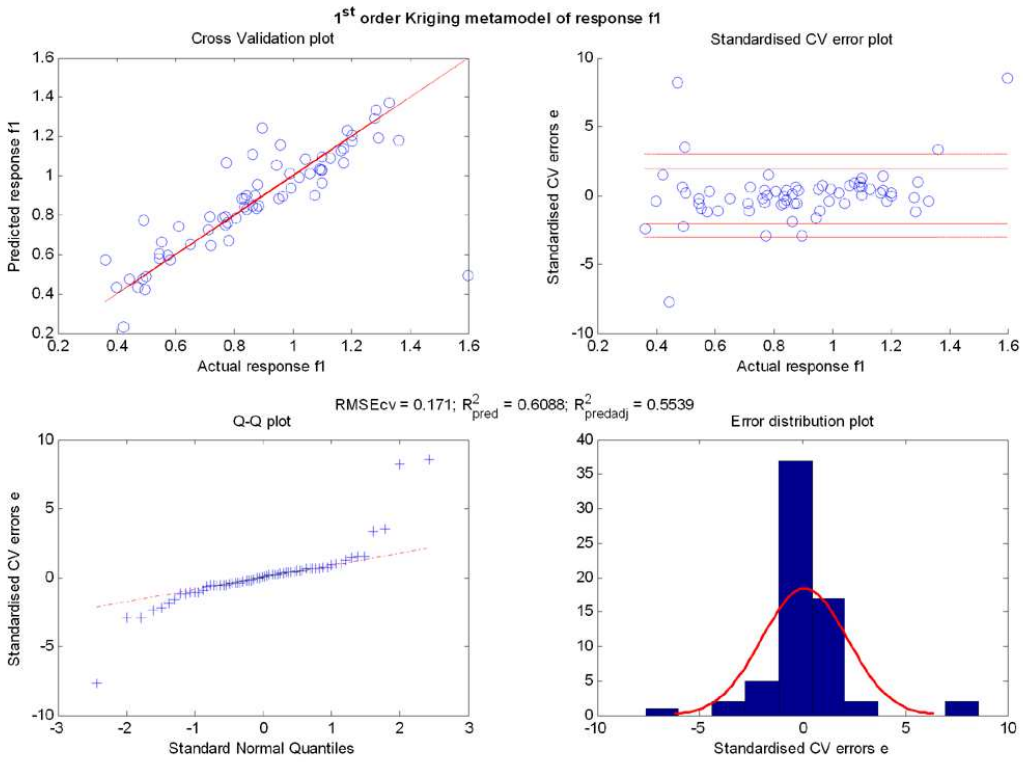


Figure 5.5: Metamodel validation of the 1st order Kriging metamodel for the objective function after batch 6

the data after the 6th batch of 16 FEM calculations. Note that the predicted R^2-values are higher than the ones after the first batch of calculations as reported in Figure 5.3, which implies a more accurate metamodel.

Figure 5.6 presents the metamodel of the objective function and its RMSE values, as well as a contour plot of the objective function. The obtained optimum $(t_1, t_2, u_{\max}) = (0, 2.5, 8.3)$ is constrained by the box constraint $t_2 \geq 2.5$.

A final 97th FEM calculation was performed with the optimised design variable settings. This calculation resulted in a real objective function value of 0.37, which is fairly close to the approximate objective function value of 0.30. Although the metamodels are still not perfect as indicated by Figure 5.5 and there is still a small difference between the approximate and the actual value of the objective function, it was decided to be satisfied with the approximate optimum obtained by this metamodel based optimisation algorithm.

Remaining is the question how the metal forming problem benefits from the exercise performed above. Figures 5.7(b) through (e) again show the final products deformed with the

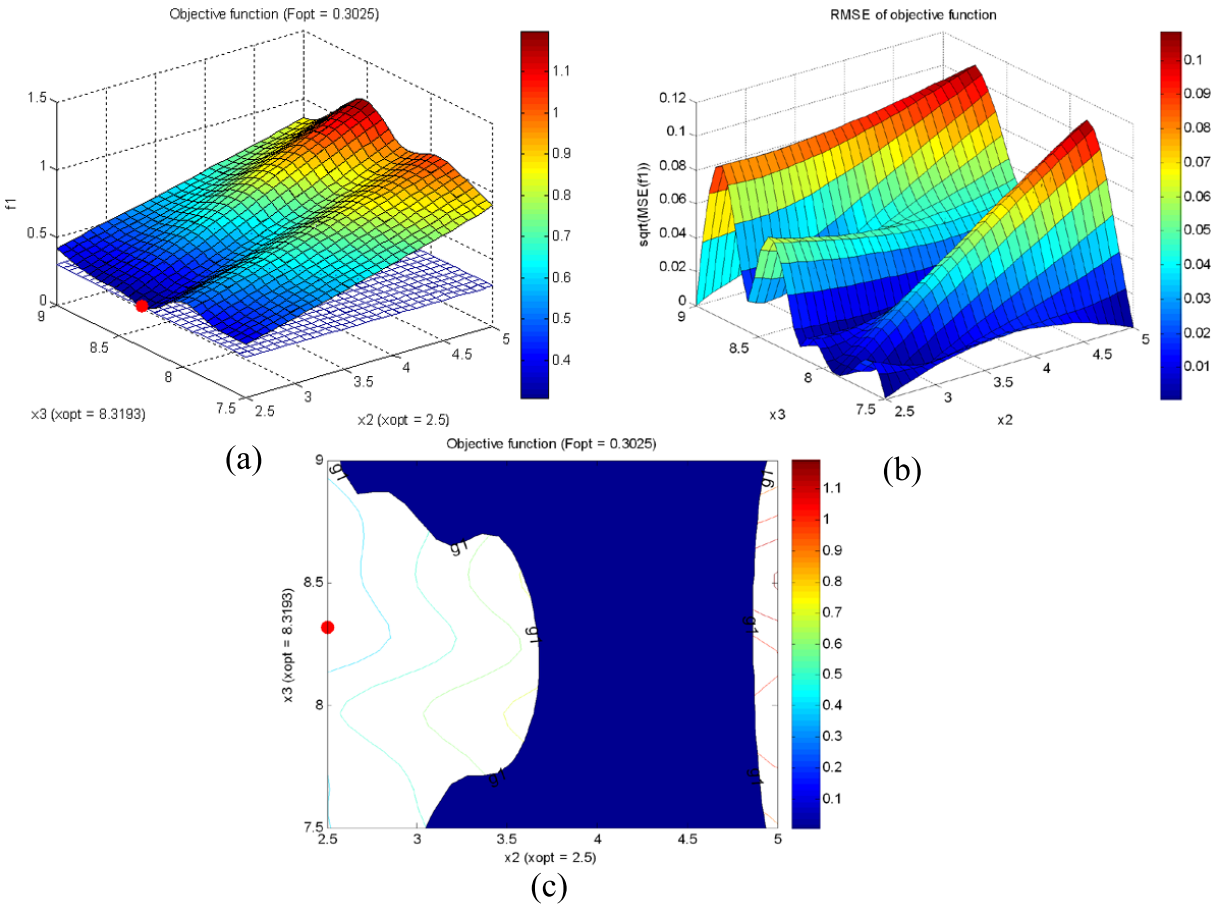


Figure 5.6: (a) Metamodel of the objective function after batch 6; (b) Metamodel of the RMSE of the objective function after batch 6; (c) Contour plot of the objective function after batch 6

arbitrary load paths introduced in Table 5.1. In Figure 5.7(f), the product hydroformed with the optimal load paths is added. Note the constant wall thickness distribution of the product formed with the optimised load paths.

Figure 5.8 shows the wall thickness throughout the final product for all load paths. The wall thickness distribution of the undeformed product (a) is seen as the perfect wall thickness distribution. The product deformed with the optimised load paths deviates significantly less from this perfect wall thickness distribution than the other products.

It can be concluded from the Figures 5.7 and 5.8 that the product deformed with the optimised load paths outperforms the other products formed with arbitrary settings, which demonstrates the good applicability of metamodel based optimisation algorithms to metal forming.

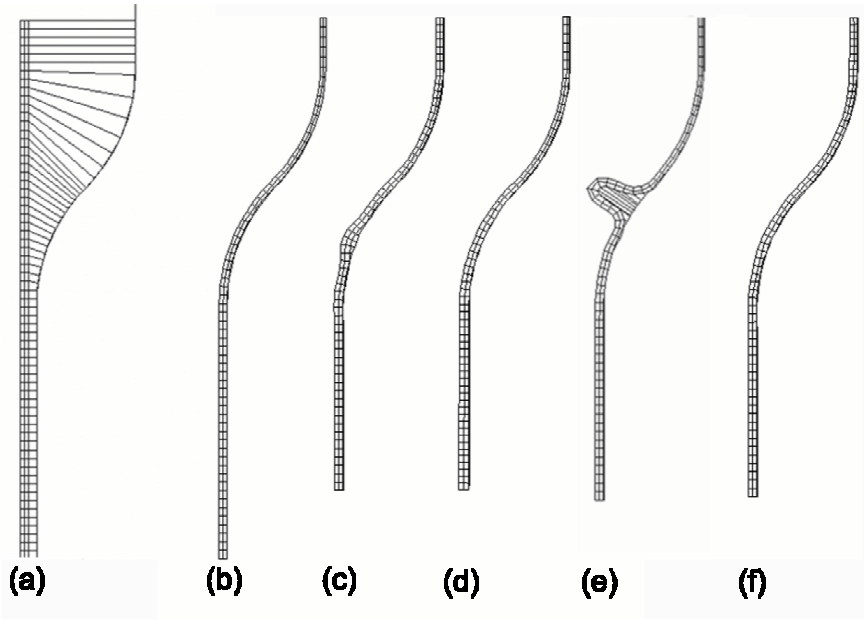


Figure 5.7: (a) FE model of the initial tube; (b-e) Final product formed with several arbitrary selected load paths; (f) Final product formed with optimised load paths

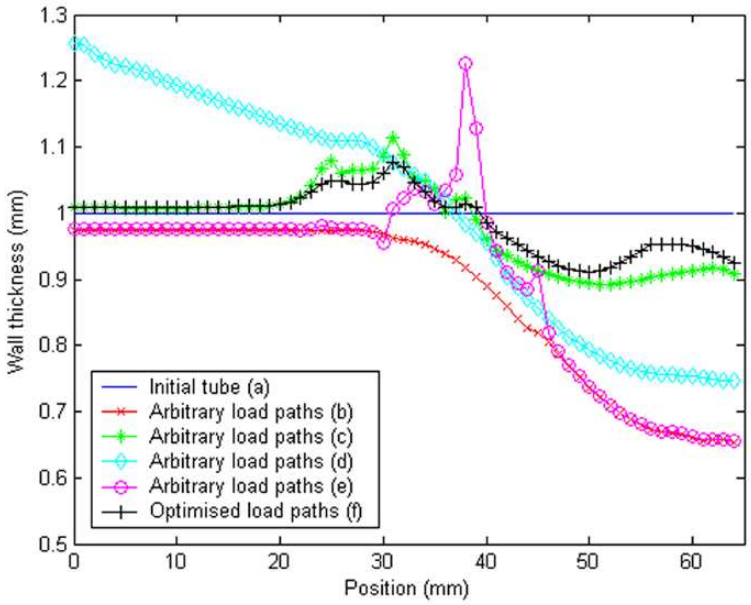
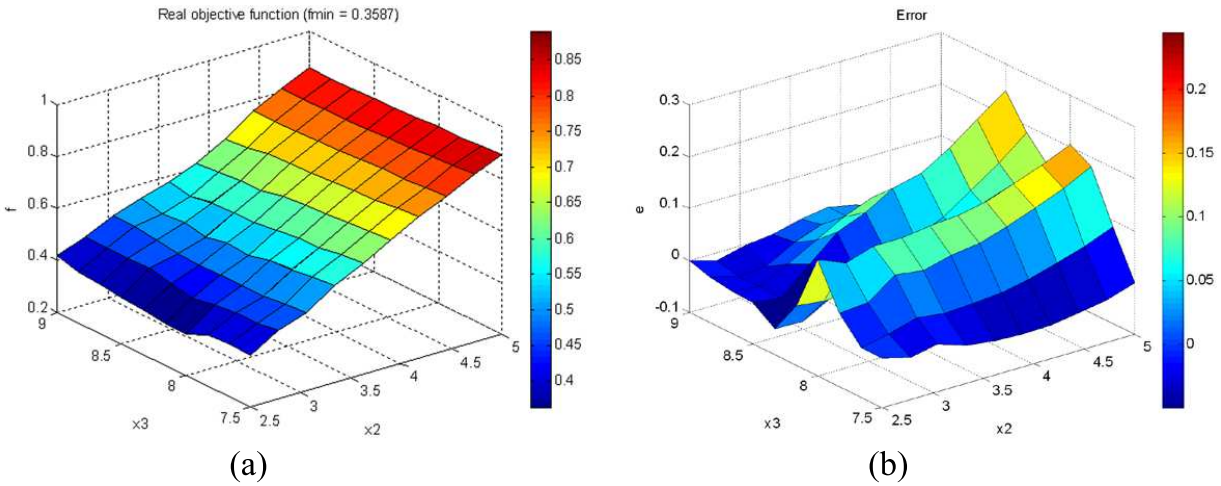


Figure 5.8: Wall thickness distribution of several hydroformed products



(a) (b)

Figure 5.9: (a) The real objective function; (b) The difference between the metamodel and the real objective function

5.3 Comparison with the true objective function

Since the optimisation algorithm is based on metamodels, which approximate the real objective function and implicit constraints, it is useful to get an idea of the true shape of the objective function. Normally, this objective function shape is not known. For this first application, however, we will spend some time to investigate the true response to compare it with the results obtained by the metamodel based optimisation algorithm as presented in the the previous section.

For this purpose, a grid of 10 x 10 x 10 evenly spaced points was placed on the design space spanned by the three design variables t_1, t_2 and $u_{\max}$. We will focus on the direct neighbourhood of the global optimum only. Thus, the design space is the reduced design space used during the 6^{th} batch of FEM calculations. The 10 x 10 x 10 = 1000 FEM calculations were again performed with the FE software DiekA. 123 calculations did not converge, which leaves 877 values for the comparison between the metamodel and the real objective function.

	Mean	Standard deviation	Maximum	Minimum
Original objective function	0.0035	0.1408	1.0400	-0.1934
Alternative objective function	-0.0063	0.1140	0.5906	-0.3557

Table 5.3: Errors of the metamodel w.r.t. the real objective function values for the two different formulations of the objective function

Figure 5.9 shows the real objective function dependent on the design variables t_2 and $u_{\max}$ at a constant level of $t_1 = 0$. Note the fairly good resemblance between the metamodel of Figure 5.6(a) and the real objective function. Figure 5.9(b) presents the difference between the metamodel and the real objective function. The mean, standard deviation and the maximum and minimum error for the 877 converged simulations are summarised in Table 5.3.

5.4 An alternative formulation of the objective function

Although the metamodel represents the real objective function fairly well in the neighbourhood of the optimum, a large number of potentially expensive FEM simulations (97) were needed for achieving these good results. The question arises, whether this number of calculations can be reduced.

A possibility for doing this may be reformulating the objective function. Let us take a look first on how the objective function was formulated in Equation 5.1 and review the FE model of the Demonstration of Concept in Figure 5.10.

In Equation 5.1, the objective function was formulated as:

$$f\left(t_1, t_2, u_{\max}\right) = \left\| \frac{h - h_0}{h_0} \right\|_2 \tag{5.2}$$

which incorporates the 2-norm of $(h - h_0)/h_0$:

$$\left\| \frac{h - h_0}{h_0} \right\|_2 = \sqrt{\sum_{i=1}^{n} \left(\frac{h_i - h_0}{h_0} \right)^2} \tag{5.3}$$

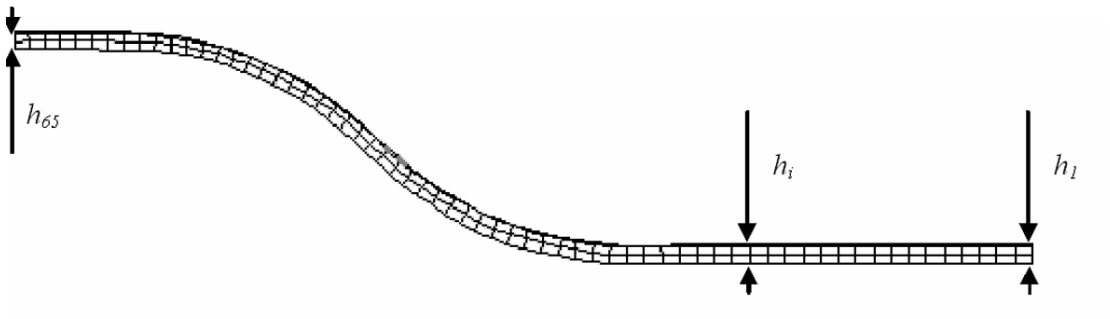


Figure 5.10: The FE model of the DoC

where h_0 is the initial wall thickness, h_i is the final wall thickness at position i and n is the total number of positions. One can obtain from Figure 5.10 that n equals the number of nodes in length direction (65 in this case). Equation 5.2 results in one value incorporating the total relation between the final wall thickness distribution in the entire product and the three design variables.

A way of reasoning is that this may be a lot of information to be contained by only one number. Consequently, this could result in a very complex objective function, which is very difficult to represent by a metamodel. At least not without running a very large number of function evaluations, i.e. FE calculations in this case. Hence, it would be beneficial to reformulate the objective function to contain less information since this will probably lower the number of FE calculations that need to be performed to acquire an accurate metamodel.

Following this approach, the objective function in Equation 5.2 can be divided into 65 "subresponses":

$$f_i = \left(\frac{h_i - h_0}{h_0} \right) \qquad i = 1, 2, ..., 65 \tag{5.4}$$

Each one of these responses contains less information than the total objective function and should therefore be easier to fit. This implies it is sufficient to run less FEM calculations for obtaining a metamodel of the same accuracy. Alternatively, using the same number of calculations, one can construct a more accurate metamodel than was the case for the total objective function. Finally, the 65 subresponses can be combined easily to the total objective function by taking the 2-norm:

$$f_{\text{tot}} = \left\| f_i \right\|_2 = \sqrt{\sum_{i=1}^{n} f_i^2} \tag{5.5}$$

This total objective function is an alternative formulation of the one in Equation 5.2.

Let us now investigate whether this alternative objective function indeed is more accurate than the one considered before. The same DOE points as used for the original metamodel in Section 5.2 were used for fitting the metamodel of the alternative objective function. Hence, following the reasoning above, an equal number of calculations would yield a more accurate metamodel.

The validation of four of the 65 submetamodels, f_1, f_{30}, f_{50} and f_{65}, is presented in the Figures 5.11(a-d). Most R^2-values are higher than those of the original metamodel shown in Figure 5.5. Thus, one can conclude that the submetamodels are in general more accurate and can be fitted more easily.

Figure 5.12(a) presents the total metamodel that follows from Equation 5.5. One can compare this alternative objective function to the original objective function that was visualised

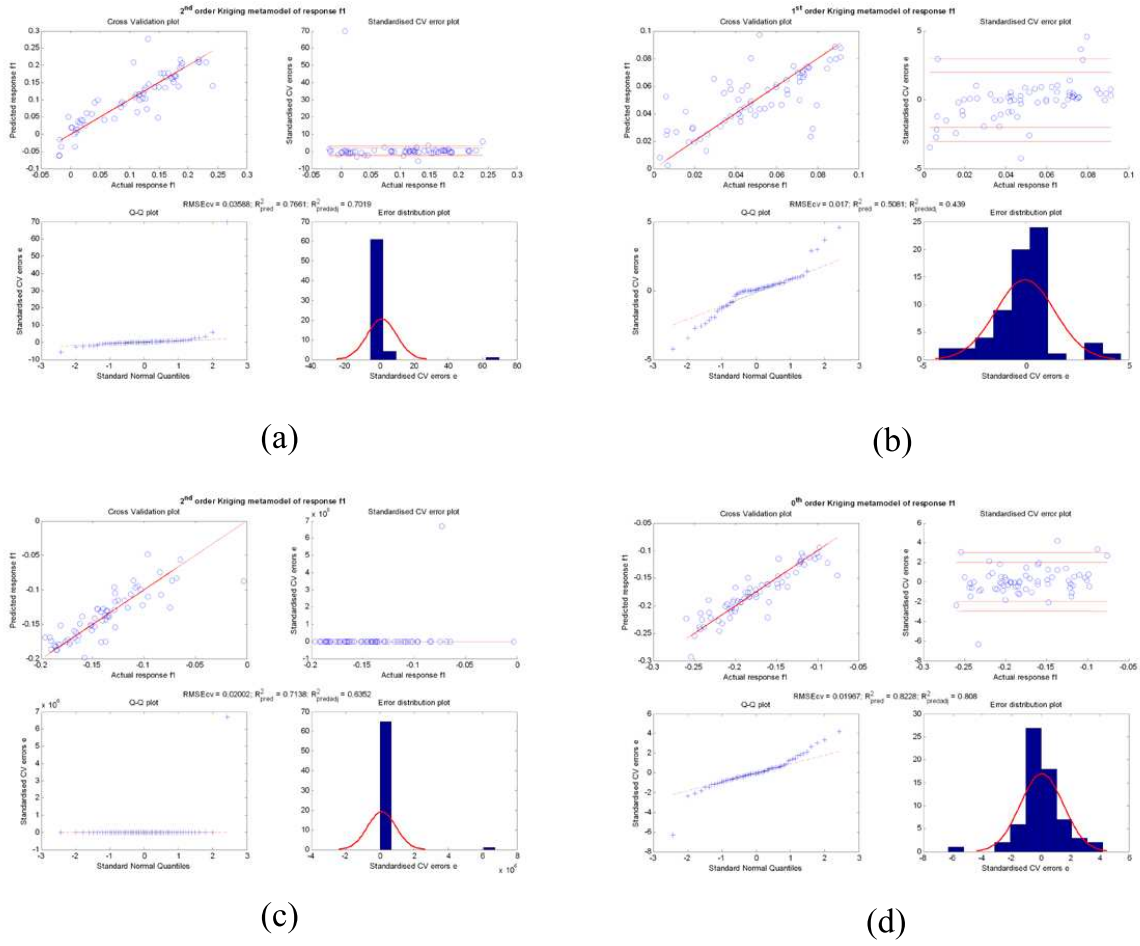
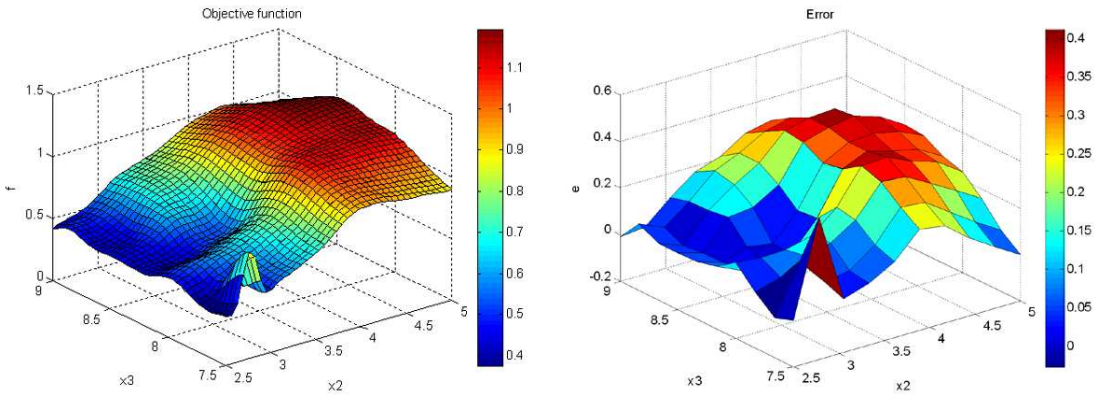


Figure 5.11: Metamodel validation of 4 submetamodels: (a) f_1; (b) f_{30}; (c) f_{50}; (d) f_{65}

in Figure 5.6(a) and to the real objective function values depicted in Figure 5.9(a). The metamodel of the original objective function appears to best represent the real objective function. This can also be concluded from comparing the errors $f_{\text{original}} - f_{\text{real}}$ visualised in Figure 5.9(b) and the errors $f_{\text{alternative}} - f_{\text{real}}$ that are presented in Figure 5.12(b). This result is surprising since it was expected that the alternative formulation would be more accurate. A possible explanation for this is that the submetamodels are better, however, not perfect. Therefore, an error is introduced, which is subsequently subjected to Equation 5.5. In this way, the error is propagated to result in a larger final error.

Nevertheless, this conclusion seems to be based on a local phenomenon: if we calculate the total error between the metamodel of the alternative objective function and all the 877 evaluations of the real objective function, the results are much closer to another. Table 5.3 shows that the standard deviation and the maximum error are lower for the



(a) (b)

Figure 5.12: (a) The metamodel of the alternative objective function; (b) The error between the metamodel of the alternative objective function and the real objective function

alternative formulation than for the original objective function. The mean and minimum error stay, however, better for the original formulation. Hence, it is not convincingly demonstrated that the alternative formulation results in a more accurate metamodel for the same number of FEM calculations. This implies that it is also not sure whether the alternative formulation yields an equally accurate metamodel using less FEM calculations.

5.5 Conclusions

For showing that the metamodel based optimisation algorithm introduced in Chapter 4 is indeed applicable to metal forming processes, it was applied to a hydroforming example in this chapter. For demonstration purposes, an axisymmetric product was selected of which the 2D Finite Element model took only a couple of minutes to execute. The fairly complete optimisation model included the objective function, three design variables, an implicit and two explicit constraints. The objective function was formulated to yield a constant wall thickness throughout the final product and the design variables described the internal pressure and axial feeding load paths.

It can be concluded that applying the optimisation algorithm provides a more constant wall thickness than arbitrary selected load paths. This demonstrates the good applicability of the metamodel based optimisation algorithm to metal forming processes. For validating the accuracy of the metamodels obtained by the algorithm, the metamodel of the objective function was compared to the results of 1000 validation simulations. The resemblance was fairly good, though far from perfect.

To improve the accuracy of the metamodels, it was attempted to reformulate the objective

function. This indeed showed slightly more accurate metamodels than for the original formulation, although these metamodels were also not identical to the real objective function values obtained by the 1000 validation simulations. Nevertheless, the optimisation algorithm yielded a much better wall thickness distribution than obtained using arbitrary load paths and is hence shown to be very useful for optimising metal forming processes using time consuming FEM simulations.

Chapter 6

Application to forging

This chapter describes the application of the metamodel based optimisation algorithm to more complex 3D forging processes. At the same time, it compares the algorithm to the Metamodel Assisted Evolutionary Strategy (MAES) mentioned in Section 4.2.6. This algorithm was developed by Emmerich et al. [34, 35] and applied to the Bridgman casting process by Jakumeit et al. [35, 51]. Fourment, Do et al. [32, 41, 42] applied the algorithm successfully to forging. The results of the latter authors will be used as a benchmark for the metamodel based optimisation algorithm, which will be referred to in this chapter as Sequential Approximate Optimisation (SAO). For comparing the two optimisation algorithms, they are applied to two forging process optimisation cases in the Sections 6.1 and 6.2. Section 6.3 concludes the comparison by discussing the results.

6.1 Triaxe

The first example case is the so-called Triaxe, a spindle in English. It is shown in Figure 6.1(b). The optimisation problem is modelled in Section 6.1.1 and solved by the optimisation algorithms in Section 6.1.2.

6.1.1 Modelling the optimisation problem

The Triaxe is forged in two subsequent steps: first a preform is made by upsetting, which is subsequently forged to result in the final product. These two production steps for forging the Triaxe are presented in the Figures 6.1(a) and (b), respectively. Note that only the upper half of the product is depicted. The lower half is omitted since the Triaxe is symmetric.

To evaluate whether the final product can be made in the factory, a Finite Element (FEM) calculation was performed using the FE code Forge3®. Figures 6.2(a) and (b) show the FE models after the two forging steps. Use has been made of symmetry. One full simulation of the Triaxe takes about four hours to run on an Intel Pentium IV desktop computer with 2GB memory. In Figure 6.2(c), one can observe that a folding defect occurs, which dete-

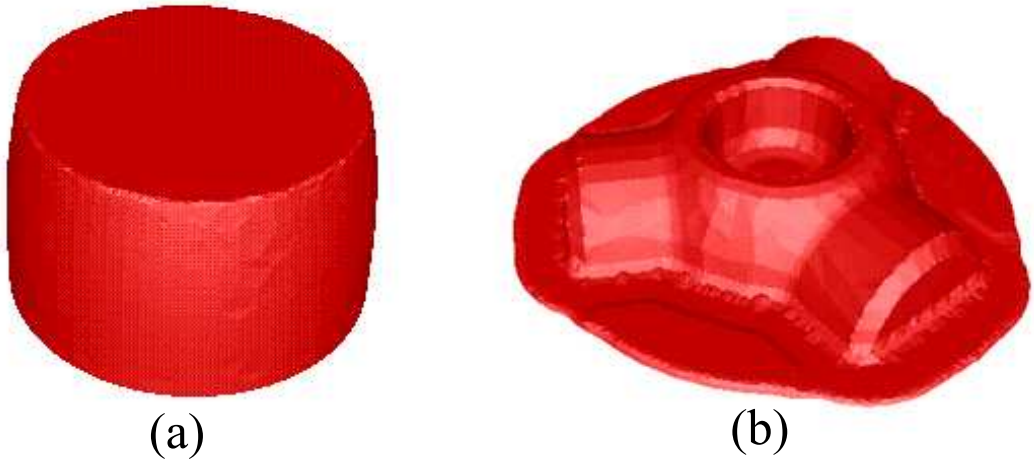


Figure 6.1: The Triaxe: (a) The preform; (b) The final product

riorates the final product's quality. The depicted quantity is the equivalent plastic strain rate at the free surface of the product, i.e. where no contact exists between the product and the die.

To overcome the folding defect, it is proposed to optimise the geometry of the preform with as an optimisation goal to minimise the equivalent plastic strain rate at the free surface during forging. The objective function is formulated as follows:

$$\Phi_{\text{fold}} = \frac{1}{t_{\text{end}} - t_0} \int_{t=t_0}^{t_{\text{end}}} \left(\frac{1}{\Omega_{\text{ft,ref}}} \int_{\Omega_{\text{ft}}} \left(\frac{\dot{\varepsilon}_{\text{eq}}}{\dot{\varepsilon}_{\text{eq,ref}}} \right)^{\alpha} \mathrm{d}s \right)^{\frac{1}{\alpha}} \mathrm{d}t \tag{6.1}$$

t denotes the time, Ω_{ft} is the free surface of the discretised domain at time t and $\Omega_{\text{ft,ref}}$ the reference free surface at time $t = t_0$. $\dot{\varepsilon}_{\text{eq}}$ and $\dot{\varepsilon}_{\text{eq,ref}}$ are the equivalent strain rate and a reference equivalent strain rate and α is an amplification factor, which is selected to be equal to 10.

The geometry of the preform is modelled by the B-spline shown in Figure 6.3 as the thin red line. The B-spline is controlled by the six control points $C_1 ... C_6$. The three design variables are presented in Figure 6.3 as $P_1 ... P_3$. All design variables are allowed to vary between -10 and 20 mm. Besides these box constraints on the design variables, other constraints are not present.

The total optimisation problem can now be modelled as follows:

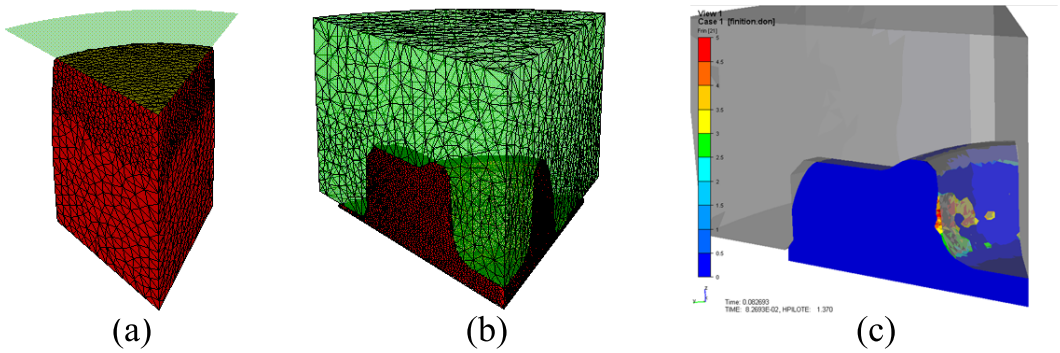


Figure 6.2: (a) The FE model after upsetting; (b) The FE model after forging; (c) Folding

$$\begin{aligned} &\min \Phi_{\text{fold}}(P_1, P_2, P_3) \\ &\text{s.t.} \quad -10\,\text{mm} \leq P_1 \leq 20\,\text{mm} \\ &\qquad -10\,\text{mm} \leq P_2 \leq 20\,\text{mm} \\ &\qquad -10\,\text{mm} \leq P_3 \leq 20\,\text{mm} \end{aligned} \tag{6.2}$$

6.1.2 Applying the optimisation algorithms

The optimisation algorithms are:

1. A BFGS algorithm belonging to the group of classical iterative algorithms introduced in Section 4.1.1. It can be used since Forge3® allows for the calculation of sensitivities
2. The Metamodel Assisted Evolutionary Strategy (MAES) mentioned in Section 4.2.6

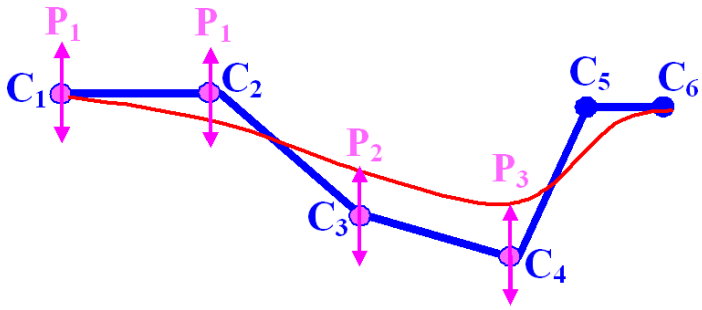


Figure 6.3: The B-spline describing the geometry of the preform of the Triaxe

	$N_{\text{opt}}/N_{\text{tot}}$	P_1 [mm]	P_2 [mm]	P_3 [mm]	Φ_{fold}	OK?
Reference	–	0	0	0	10.49	No
BFGS	3/12	7.02	2.92	6.55	10.25	No
MAES	30/48	-7.26	-3.84	-10	8.35	Yes
SAO	36/50	-1.69	-5.95	8.33	9.20	No
SAO-MMF	38/48	-10	-6.50	-9.82	8.06	Yes
SAO-MEI	22/50	-8.52	-6.40	-10	7.97	Yes

Table 6.1: Results of optimising the Triaxe

3. The Sequential Approximate Optimisation (SAO) algorithm presented in Section 4.2 utilising three sequential improvement strategies as introduced in Section 4.2.6:

   (a) Without zooming. The calculations were run in batches of 20 simulations. After each batch, the optimum predicted by the metamodel was included in the next batch. This algorithm will be referred to as SAO

   (b) Improvement by Minimising a Merit Function. The initial batch are the same 20 calculations as those used for SAO. Subsequently, the sequential improvement strategy picks the size of the next batches itself. This algorithm will be referred to as SAO-MMF

   (c) Improvement by the method of Maximum Expected Improvement. The initial batch are the same 20 calculations as those used for SAO and SAO-MMF. Subsequently, the sequential improvement strategy picks the size of the next batches itself. This algorithm will be referred to as SAO-MEI

The fourth sequential improvement strategy (Zooming + Resampling) was not included due to the bad results obtained for the analytical test functions in Appendix C.

Table 6.1 presents the results. The table shows for all algorithms the number of the FEM calculation N_{opt}, which gave the optimal settings, as a fraction of the total number of simulations N_{tot} performed for a specific algorithm. Additionally, it presents the optimal design variable settings and corresponding objective function values and it answers the question whether the folding defect has been solved or not. The convergence of the optimisation algorithms is depicted in Figure 6.4. Figure 6.5 visualises the preforms of the reference and after optimisation using the different algorithms and Figure 6.6 shows the corresponding folding behaviour during forging.

According to Table 6.1 and Figure 6.4, all optimisation algorithms reduce folding w.r.t. the reference situation. All objective function values are lower than the reference value of 10.49.

The table and convergence plot present, however, that several optimisation algorithms perform better than others. The iterative BFGS algorithm is outperformed by the other

algorithms. A possible explanation is that BFGS is a local algorithm: it finds the optimum closest to the location where it is initialised and this is most of the times not the global optimum. Hence, a suboptimal result is obtained. The Metamodel Assisted Evolutionary Strategy and the Sequential Approximate Optimisation algorithms are global algorithms and all obtain lower objective function values than the BFGS algorithm. On the other hand, the less optimal results of BFGS come at significantly lower computational costs. Note that the BFGS algorithm obtained its optimum already after 3 calculations, whereas the other algorithms need many more simulations. Therefore, BFGS may still be a very good algorithm if fast improvement is desired, whereas one is less interested in finding a very low objective function value. In the case of the Triaxe, however, one can see in Figure 6.6(b) that the folding defect is still present after optimisation by the BFGS algorithm. Hence, this algorithm does not live up to the expectation that the folding problem can be solved by optimisation.

The same is the case for the Sequential Approximate Optimisation algorithm without zooming (SAO). The objective function value of 9.20 presented in Table 6.1 and Figure 6.4 is considerably lower than that of both the reference and the BFGS algorithm, but Figure 6.6(d) shows that there is still a small folding defect. Comparing this figure to Figures 6.6(a) and (b), however, one may conclude that the folding problem has already decreased.

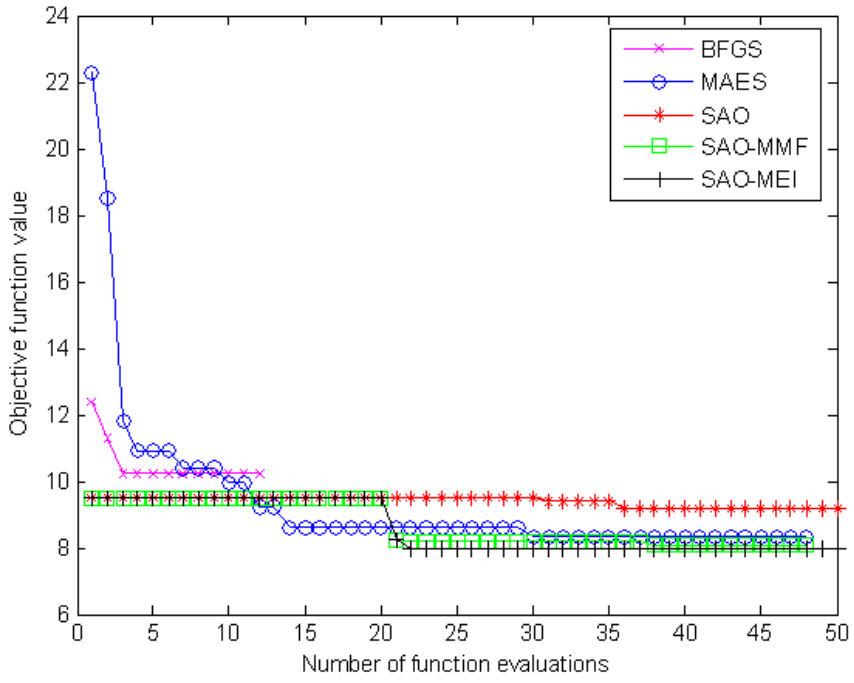


Figure 6.4: Convergence of the algorithms for optimising the Triaxe

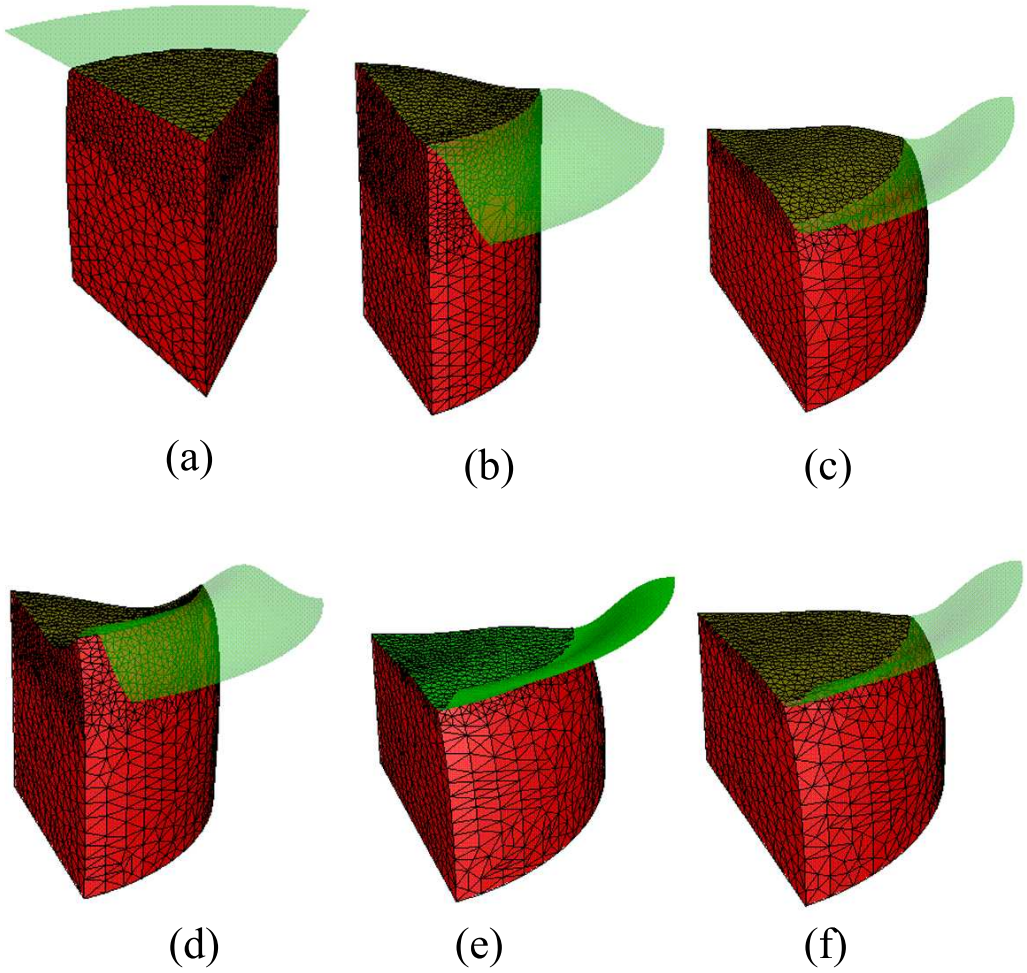


Figure 6.5: Preforms of the Triaxe: (a) Reference; (b) BFGS; (c) MAES; (d) SAO; (e) SAO-MMF; (f) SAO-MEI

The other optimisation algorithms, MAES, SAO-MMF and SAO-MEI, do solve the folding defect as one can see in the Figures 6.6(c), (e) and (f), respectively. The performance of these three algorithms is very similar: it can be seen in Table 6.1 and Figures 6.5(c), (e) and (f) that the resulting preform shapes are almost identical. The same can be said for the objective function values. Strictly speaking, SAO-MEI performs the best, followed closely by SAO-MMF and MAES. Note from Table 6.1 and Figure 6.4 that SAO-MEI not only obtains the lowest objective function value; it also obtains this value already after 22 FEM calculations, which is considerably faster than SAO-MMF and MAES (38 and 30 calculations, respectively). As a matter of fact, after the initial batch of 20 calculations, SAO-MEI needs only two iterations to obtain its optimum. Note from Figure 6.4 that the first 20 calculations of SAO, SAO-MMF and SAO-MEI are the same.

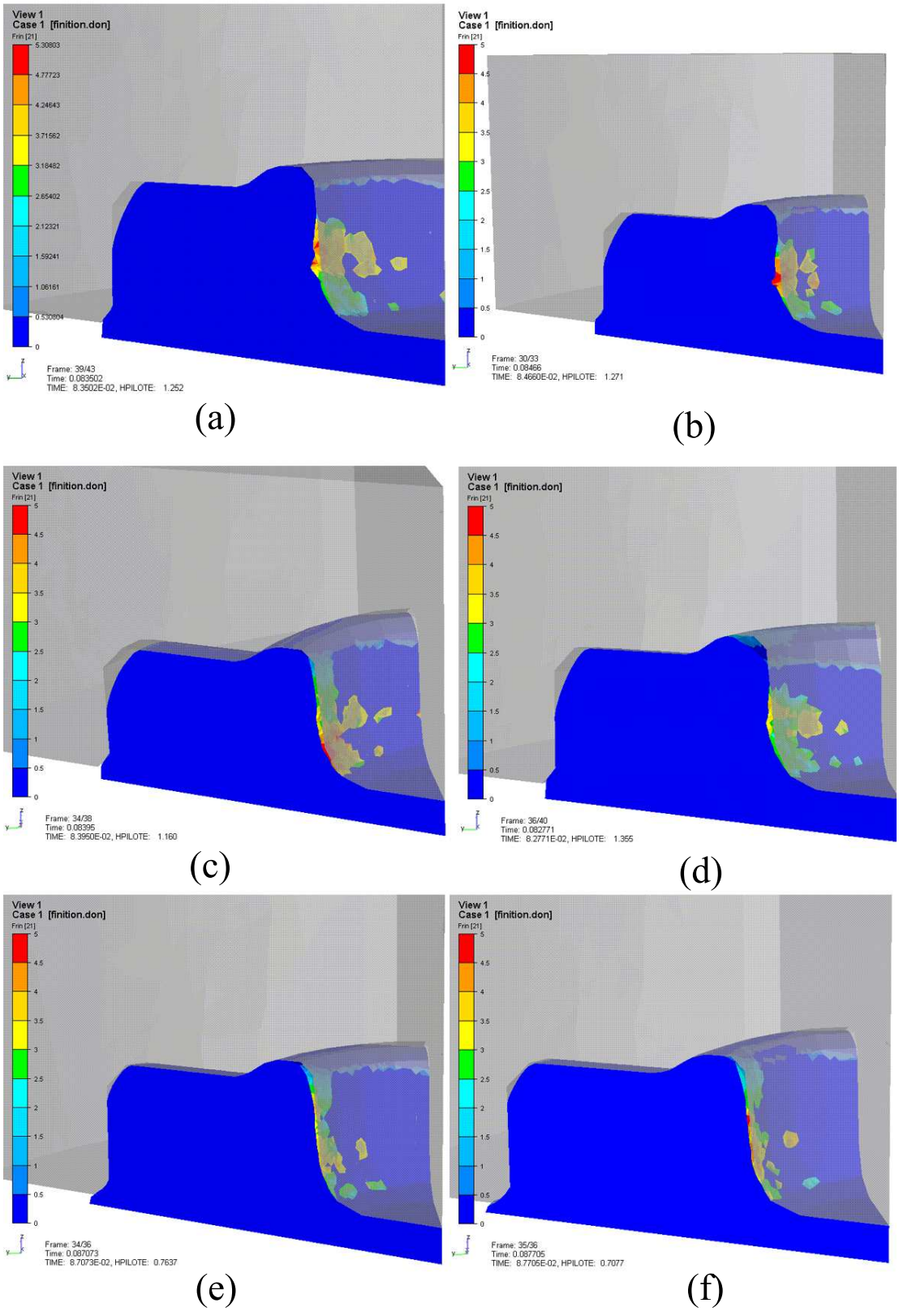


Figure 6.6: Folding behaviour of the Triaxe: (a) Reference; (b) BFGS; (c) MAES; (d) SAO; (e) SAO-MMF; (f) SAO-MEI

6.2 Engrenage

The second forging example is a gear, which will be referred to as the Engrenage. The part is shown in Figure 6.7(b). The optimisation problem is modelled in Section 6.2.1 and in Section 6.2.2, the optimisation algorithms are applied to solve the optimisation problem.

6.2.1 Modelling the optimisation problem

Just as was the case for the Triaxe, forming the Engrenage is a two step process. First, a preform is established. Subsequently, the preform is forged to obtain the final product. The resulting parts after both production steps are shown in Figures 6.7(a) and (b), respectively.

For optimising the Engrenage, a Finite Element model needs to be made. Figure 6.8(a) shows the FE model of the Engrenage after producing the preform. Note that use has been of the product's symmetry. Forge3® is used as FE code. One full simulation of the Engrenage takes about one hour on an Intel Pentium IV desktop computer with 2GB memory.

To limit the production costs and environmental pollution, it is beneficial to minimise the total energy required for forging the Engrenage. An objective function is formulated as [32, 41, 42]:

$$\Phi_{\text{ene}} = \int_{t=t_0}^{t_{\text{end}}} \left(\int_{\Omega_t} \boldsymbol{\sigma} : \dot{\boldsymbol{\varepsilon}} \mathrm{d}w + \int_{\Omega_{\text{ct}}} \boldsymbol{\tau} \cdot \mathbf{v} \mathrm{d}s \right) \mathrm{d}t \tag{6.3}$$

t denotes the time, Ω_t is the discretised domain at time t, $\boldsymbol{\sigma}$ and $\dot{\boldsymbol{\varepsilon}}$ are the stress and strain rate tensors. In the second part of the equation, Ω_{ct} is the contact surface and $\boldsymbol{\tau}$ and $\mathbf{v}$ are the shear stress and the velocity field, respectively. However, to assure a final part without folding defects, it is also recommendable to include folding in the optimisation problem.

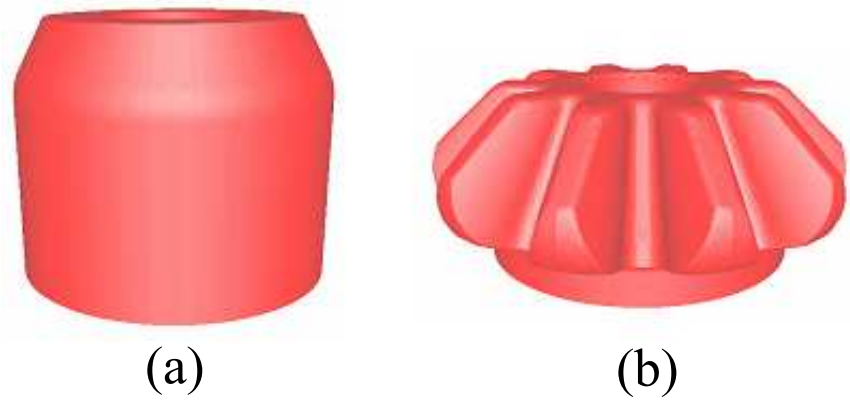


Figure 6.7: The Engrenage: (a) The preform; (b) The final product

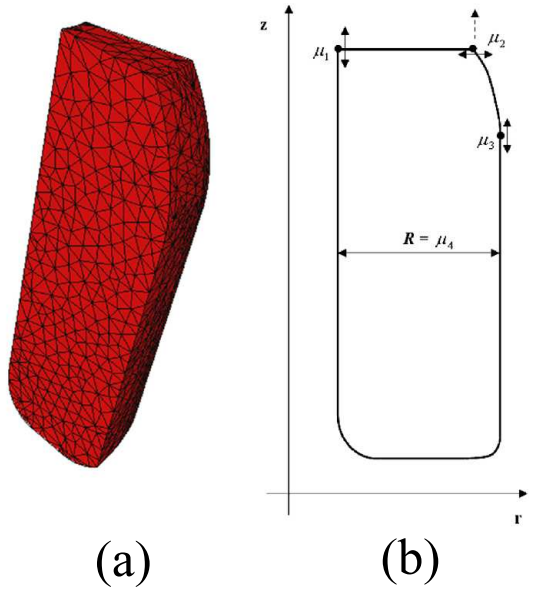


Figure 6.8: (a) The B-spline describing the geometry of the preform of the Engrenage; (b) The 3D FE model of the Engrenage

A second objective function is the same as already formulated for the Triaxe in Equation 6.1.

The two objective functions from Equations 6.1 and 6.3 are combined to one total objective function:

$$\Phi_{\text{tot}} = a\frac{\Phi_{\text{fold}}}{\Phi_{\text{fold,opt}}} + (1-a)\frac{\Phi_{\text{ene}}}{\Phi_{\text{ene,opt}}} \tag{6.4}$$

where the weight factor a is chosen equal to 0.5. To make the two objective functions comparable to each other, they are normalised by $\Phi_{\text{fold,opt}}$ and $\Phi_{\text{ene,opt}}$. These values have been determined by first optimising the Engrenage for the folding and energy objective functions separately. Subsequently, $\Phi_{\text{fold,opt}}$ and $\Phi_{\text{ene,opt}}$ are known values and can be used within the multi-objective optimisation problem of which the objective function was given by Equation 6.4.

The design variables are presented in Figure 6.8(b). Just as was the case for the Triaxe, the design variables describe the preform geometry. The geometry is described by the five variables $\mu_1, \mu_{2\text{r}}, \mu_{2\text{z}}, \mu_3$ and μ_4. To reduce the number of design variables to three, the z-coordinate of μ_2 is connected to μ_1. Additionally, demanding volume conservation will result in three design variables since the fourth one can be expressed as a function of the other three:

$$\mu_4 = f(\mu_1, \mu_2, \mu_3) \tag{6.5}$$

where the subscript r is omitted for μ_2. Thus, remaining is a set of three design variables μ_1, μ_2 and μ_3. No constraints have been formulated except the box constraints bounding the design variables. These box constraints are included in the following optimisation model that is used for optimising the Engrenage:

$$\begin{aligned} &\min \Phi_{\text{tot}}(\mu_1, \mu_2, \mu_3) \\ &\text{s.t.} \quad 40\,\text{mm} \leq \mu_1 \leq 46\,\text{mm} \\ &\qquad 18\,\text{mm} \leq \mu_2 \leq 22\,\text{mm} \\ &\qquad 30\,\text{mm} \leq \mu_3 \leq 34\,\text{mm} \end{aligned} \tag{6.6}$$

6.2.2 Applying the optimisation algorithms

The preform suggested by experts from the company Ascoforge is taken as a reference. Their settings, $\mu_1 = 44.60\,\text{mm}, \mu_2 = 21.65\,\text{mm}$ and $\mu_3 = 32.33\,\text{mm}$ result in an objective function value Φ_{tot} of 1.19. The amount of energy (Φ_{ene}) and folding (Φ_{fold}) obtained for these design variable values are set to 100%.

Now, different optimisation algorithms are compared to this reference preform and to each other by applying them to the optimisation problem modelled in Equation 6.6. Again, the optimisation algorithms are:

1. A BFGS algorithm belonging to the group of classical iterative algorithms introduced in Section 4.1.1. It can be used since Forge3® allows for the calculation of sensitivities
2. The Metamodel Assisted Evolutionary Strategy (MAES) mentioned in Section 4.2.6
3. The Sequential Approximate Optimisation (SAO) algorithm presented in Section 4.2 utilising three sequential improvement strategies as introduced in Section 4.2.6:

	$N_{\text{opt}}/N_{\text{tot}}$	μ_1 [mm]	μ_2 [mm]	μ_3 [mm]	Φ_{ene}	Φ_{fold}	Φ_{tot}
Reference	–	44.60	21.65	32.33	100%	100%	1.19
BFGS	3/7	40.91	17.17	30.11	–	–	1.157
MAES	42/49	45.67	18.36	30.25	-9.7%	-7.6%	1.079
SAO	51/51	45.25	18.00	30.58	-8.7%	-6.6%	1.091
SAO-MMF	31/49	45.25	18.18	30.65	-8.2%	-9.8%	1.076
SAO-MEI	49/54	45.35	18.00	30.41	-9.4%	-9.8%	1.068

Table 6.2: Results of optimising the Engrenage

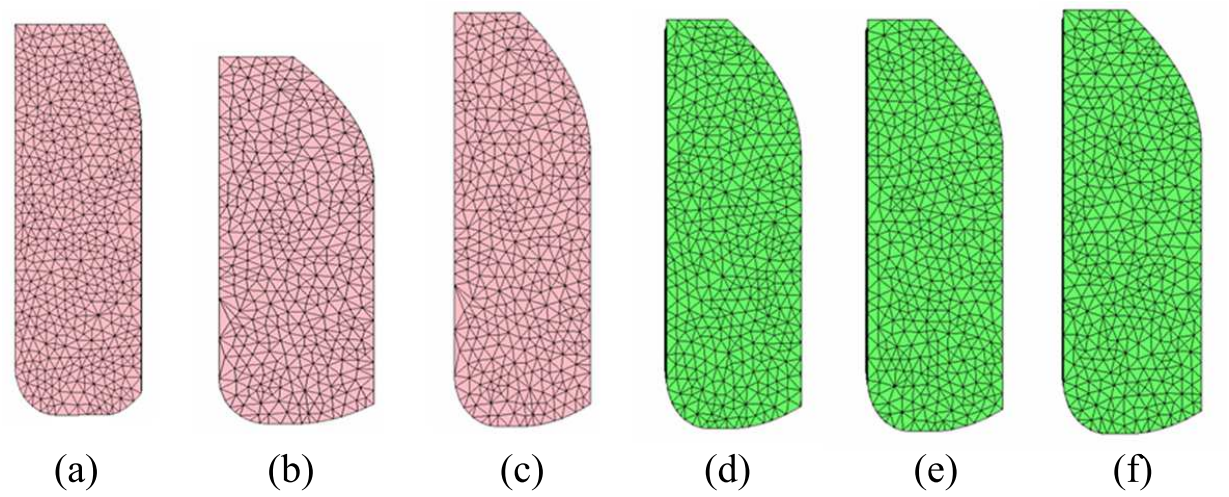


Figure 6.9: Preforms of the Engrenage: (a) Reference; (b) BFGS; (c) MAES; (d) SAO; (e) SAO-MMF; (f) SAO-MEI

(a) Without zooming. The calculations were run in batches of 20 simulations. After each batch, the optimum predicted by the metamodel was included in the next batch. This algorithm will be referred to as SAO

(b) Improvement by Minimising a Merit Function. The initial batch are the same 20 calculations as those used for SAO. Subsequently, the sequential improvement strategy picks the size of the next batches itself. This algorithm will be referred to as SAO-MMF

(c) Improvement by the method of Maximum Expected Improvement. The initial batch are the same 20 calculations as those used for SAO and SAO-MMF. Subsequently, the sequential improvement strategy picks the size of the next batches itself. This algorithm will be referred to as SAO-MEI

Table 6.2 presents the results. The table again shows for all algorithms the number of the FEM calculation N_{opt}, which gave the optimal settings, as a fraction of the total number of simulations N_{tot} performed for a specific algorithm. Additionally, it presents the optimal design variable settings and corresponding objective function values as well as the reduction in energy and folding potential obtained by optimisation. Figure 6.9 visualises the preforms of the reference and after optimisation using the different algorithms. The convergence of the algorithms is depicted in Figure 6.10.

A first remark concerns the optimisation model to which the BFGS algorithm was applied. This model deviated slightly from Equation 6.6, which explains why the optimal value of μ_2 is outside the box constraints. Values of Φ_{ene} and Φ_{fold} were not available for this optimisation problem and thus, no improvement with respect to the reference preform could

be calculated.

A first conclusion that can be drawn from Table 6.2 is that all optimisation algorithms yield better results than the reference settings proposed by Ascoforge. Both the energy consumption and the susceptibility to folding can be reduced up to 10%. A second important conclusion, which is nicely depicted by the convergence plot of Figure 6.10, is that the global algorithms outperform the local, iterative BFGS algorithm. Because BFGS is local, its results depend heavily on the design variable settings where the algorithm is initialised. As was the case for the Triaxe, however, BFGS obtains its optimum after only 3 calculations, whereas the other algorithms need many more simulations.

Turning to the Sequential Approximate Optimisation algorithms now, it can be concluded that utilising an effective sequential improvement strategy is essential. Although better than BFGS, SAO without zooming performs significantly worse than SAO-MMF and SAO-MEI. Hence, it turns out to be recommendable to apply a sequential improvement strategy that aims at focussing near the optimum rather than to improve the metamodel of the objective function in a more global sense (see Section 4.2.6). SAO-MMF and SAO-MEI yield more or less equal results, although the latter algorithm performs slightly better for the Engrenage. Note from Figure 6.10 that the first batch of 20 FEM simulations was

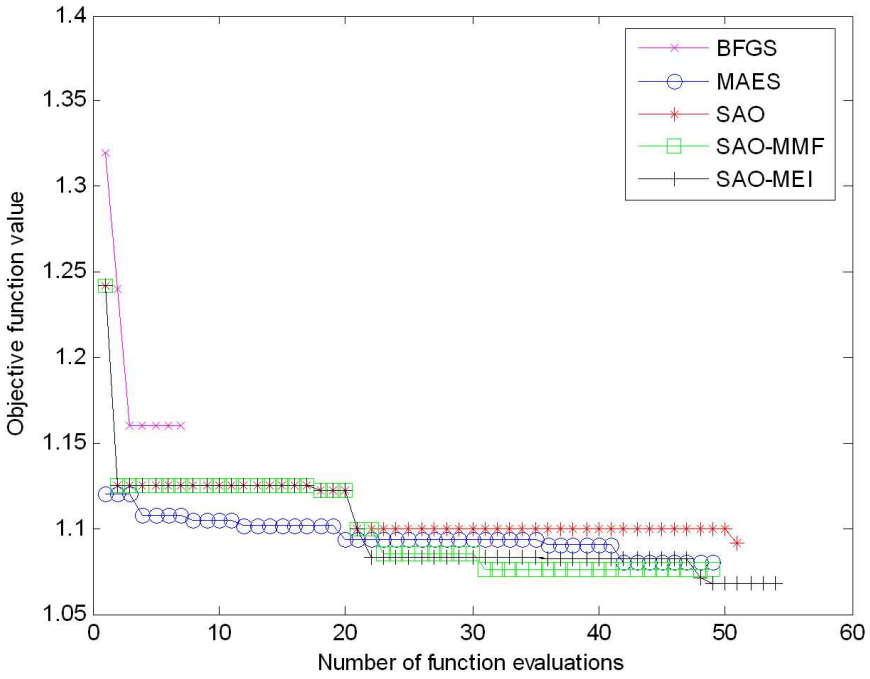


Figure 6.10: Convergence of the algorithms for optimising the Engrenage

identical for all three SAO algorithms.

The results of the Metamodel Assisted Evolutionary Strategy are comparable to those obtained by SAO-MMF. From Table 6.2, it can be seen that the optimum of MAES maximally reduces the energy consumption, whereas SAO-MMF yields a lower susceptibility to folding defects. SAO-MEI clearly performs the best for this forging example, combining a very high reduction in both energy consumption and folding potential. Nevertheless, the differences between MAES, SAO-MMF and SAO-MEI are quite small. This can also be concluded from Figure 6.9, where one can barely see a difference in the optimal preforms indicated by MAES and all three SAO algorithms.

6.3 Conclusions

The optimisation algorithm described in Section 4.2 has been applied to two forging cases and was compared to the Metamodel Assisted Evolutionary Strategy (MAES) mentioned in Section 4.2.6, to a classical iterative BFGS algorithm and to a reference case. For the Triaxe, this reference case was a product that suffered from a problem: a folding defect. It was tried to solve this problem by applying optimisation techniques. The second example, the Engrenage, had no problem: it was merely tried to utilise the optimisation algorithms to improve the forging process w.r.t. the reference situation, which in this case was the forging process proposed by the forging company. It was attempted to improve the forging of the Engrenage on two fields: to decrease its susceptibility to folding defects and to decrease the amount of energy required for forging the part.

Concluding this comparison between optimisation algorithms, let us summarise the findings. For both forging cases, the results were very similar. All algorithms yielded better results than the reference situation. For the Triaxe, it was shown that the folding defect can be solved using optimisation. For the Engrenage, both the folding susceptibility and the energy consumption could be decreased by 10% with respect to the forging process proposed by the company.

Not all algorithms performed equally well. The global algorithms introduced outperform the local iterative BFGS algorithm. The latter is e.g. not capable of solving the folding defect for the Triaxe, whereas most global algorithms are. The BFGS algorithm, on the other hand, requires significantly less time consuming FEM calculations for obtaining its optimum than the other algorithms.

Regarding the Sequential Approximate Optimisation (SAO) algorithms proposed in Section 4.2 of this report, sequential improvement strategies that focus near the optimum (SAO-MMF and SAO-MEI) are more effective than SAO without zooming, which is primarily aiming at obtaining an accurate metamodel. For the Triaxe, SAO without zooming was not capable of fully removing the folding defect, whereas SAO-MMF and SAO-MEI

did solve the folding problem convincingly. For both the Triaxe and the Engrenage, SAO-MEI performed slightly better than SAO-MMF. This conclusion for SAO, SAO-MMF and SAO-MEI was also drawn from the application of the algorithms to analytical test functions in Appendix C, which provides a nice confirmation of the findings.

The Metamodel Assisted Evolutionary Strategy (MAES) performed approximately equally well as SAO-MMF and SAO-MEI. For both the Engrenage and the Triaxe, MAES yielded slightly less good results, but the difference is so small that it can be neglected.

In the end, it can be concluded that MAES, SAO-MMF and SAO-MEI all are very promising optimisation algorithms for the optimisation of metal forming processes. They have been shown to eliminate the folding defect for the Triaxe and have reduced both the folding susceptibility of the Engrenage and the energy consumption needed for forging this part by approximately 10%.

Chapter 7

Conclusions, recommendations & future work

This chapter draws some conclusions from the topics covered by this report. Additionally, recommendations are formulated and the direction for future work is indicated.

7.1 Conclusions

It can be concluded that:

- Mathematical optimisation exists of modelling and solving an optimisation problem. Modelling consists of formulating an objective function and constraints, and selecting the design variables. Solving is done by a suitably chosen optimisation algorithm
- This report focussed on selecting and implementing a suitable optimisation algorithm for solving optimisation problems in metal forming using time consuming nonlinear FEM simulations. After an extensive literature review, a Sequential Approximate Optimisation algorithm based on metamodelling was selected since it provides the global optimum instead of only a local one, allows for parallel computing, does not require the calculation of sensitivities from FEM, offers a black box approach and apart from optimisation, the visualisation of metamodels offers a lot of insight in the metal forming process
- Response Surface Methodology (RSM) and Design and Analysis of Computer Experiments (DACE) or Kriging have been selected to be the most applicable metamodelling techniques
- The sequential improvement part of the optimisation algorithm is crucial for its efficiency. Four sequential improvement strategies have been investigated. The most promising strategies are the methods of Minimising a Merit Function (MMF) and Maximum Expected Improvement (MEI)

- The algorithm is implemented in MATLAB® and can be used in combination with any Finite Element code

- The optimisation algorithm has been applied to optimising the internal pressure and axial feeding load paths of a simple 2D hydroforming process. The results were satisfactory, which demonstrates the good applicability of the proposed algorithm to metal forming processes

- The optimisation algorithm was compared to other optimisation algorithms for metal forming by applying it to optimise the preform geometries of two more complex 3D forging products. The proposed Sequential Approximate Optimisation (SAO) algorithm with MMF and MEI sequential improvement strategies appeared to perform better than the other algorithms taken into consideration. Moreover, it managed to remove a folding defect for one forging process and it decreased the energy consumption and folding susceptibility of another forging process by 10% w.r.t. the forging process proposed by experts of the forging company

7.2 Recommendations

Several recommendations are formulated based on the findings described in this report:

- It was found that fitting accurate metamodels is quite a hard job. Therefore it is recommended to acquire more knowledge on the responses (objective function and implicit constraints) that are used within models of optimisation problems in metal forming. Questions waiting for an answer are amongst others: Are responses highly nonlinear? Is numerical noise important? Are RSM and DACE sufficient to approximate the responses or are other metamodelling techniques better applicable? Is the Gaussian correlation function within DACE a good choice? When is a metamodel accurate enough for optimisation? Answering these questions may result in a decision whether to use RSM or DACE, or maybe another metamodelling technique. Additionally, these answers will yield more efficient and more accurate metamodel based optimisation algorithms

- It is also recommended to acquire more experience with the algorithm by applying it to more metal forming processes, other objective functions and a different number of design variables. This helps answering the questions above and in this way, perhaps a final decision can be made whether to use MEI or MMF as sequential improvement strategy

- A final recommendation is to extend the algorithm to qualitative (discrete) variables such that also different materials or production processes and the number of process steps can be taken into account as design variables

7.3 Future work

The recommendations presented in the previous section are important directions for future work. In the near future, however, we will shift from the solving part of mathematical optimisation to the modelling part. This comprises formulating objective function and constraints and developing a method for selecting design variables effectively.

Objective function and constraints will be modelled to optimise for robust metal forming processes, since robustness is an essential condition for cost-effective production. Also, attention will be paid to the relation between designing a part and manufacturing the part w.r.t. the optimisation of forming processes.

Next to formulating objective function and constraints, effort will be put in developing methodologies for selecting the design variables that should be taken into account in the optimisation problem. Approximate optimisation algorithms suffer from the "curse of dimensionality", i.e. it is prohibitively expensive to take more than a couple of design variables into account. On the other hand, industrial problems include a large number of possible design variables. Thus, it is of the utmost importance to select the few design variables taken into account cleverly. It will be tried to apply design variable screening techniques to obtain only those say 5% of the variables that determine 80% of the variation in the objective function.

After having developed methods for modelling and solving optimisation problems, the total optimisation strategy will be applied to industrial hydroforming and deep drawing applications.

Appendix A

Response Surface Methodology (RSM)

Response Surface Methodology, which was originally developed for analysing physical experiments, is a well-known metamodelling technique. Section A.1 describes how the metamodel is fitted using RSM. Special attention is given to the assumptions underlying the use of RSM in Section A.2. These assumptions and the accuracy of the metamodel need to be validated as described in Section A.3. Finally, Section A.4 describes several DOE strategies suitable for RSM.

A.1 Fitting the metamodel

Response Surface Methodology is a linear regression technique described in many textbooks [38,89,90]. Let us look at the measurements of a response variable y, which depends on the $n = 6$ settings of one design variable x as presented in Table A.1. The measurements are also depicted in Figure A.1.

Using least squares regression, the true response y is predicted by a metamodel $\hat{y}$, which is a linear and additive combination of all k design variables (hence the term "linear"

x	y
0	4.5378
2.0000	7.2705
4.0000	8.2637
6.0000	12.5248
8.0000	13.0060
10.0000	14.3263

Table A.1: Some example measurements

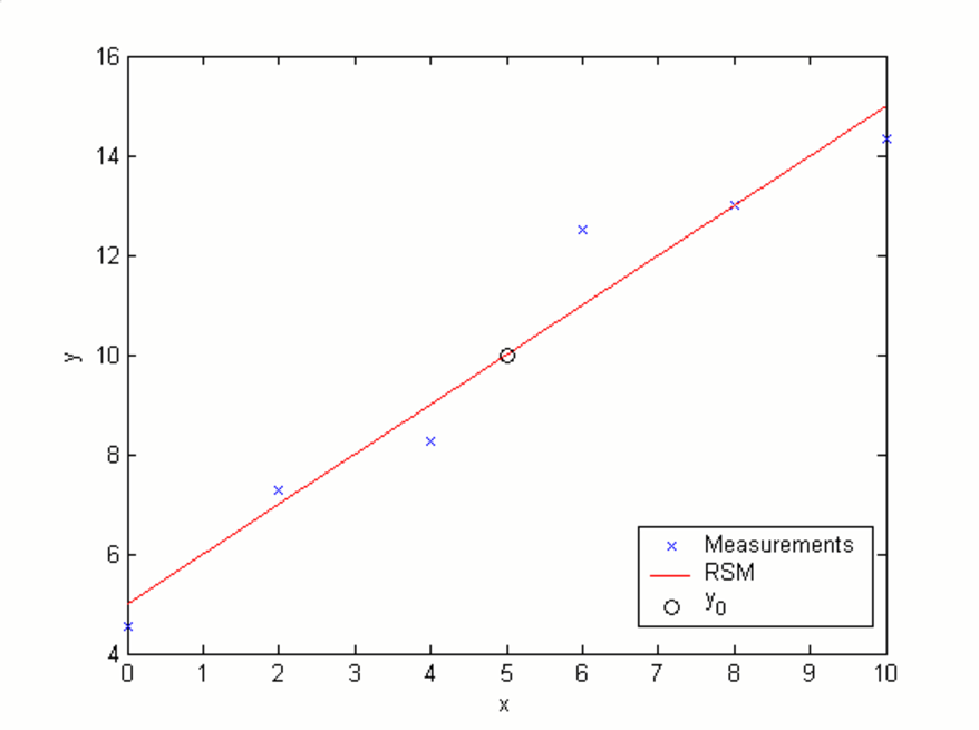


Figure A.1: Least squares regression

regression):

$$\hat{y} = \beta_0 + \beta_1 x_1 + ... + \beta_j x_j + ... + \beta_k x_k \tag{A.1}$$

in which β are the regression coefficients. However, the metamodel is only an approximation. The actual response values y_i can be determined from the metamodel and a remaining error:

$$y_i = \beta_0 + \sum_{j=1}^{k} \beta_j x_{ij} + \varepsilon_i \tag{A.2}$$

ε is the error. In matrix notation, Equation A.2 can be written as:

$$\begin{pmatrix} y_1 \\ \vdots \\ y_i \\ \vdots \\ y_n \end{pmatrix} = \begin{bmatrix} 1 & x_{11} & \cdots & x_{1j} & \cdots & x_{1k} \\ \vdots & \vdots & \ddots & & & \vdots \\ 1 & x_{i1} & & \ddots & & \vdots \\ \vdots & \vdots & & & \ddots & \vdots \\ 1 & x_{n1} & \cdots & \cdots & \cdots & x_{nk} \end{bmatrix} \begin{pmatrix} \beta_0 \\ \vdots \\ \beta_j \\ \vdots \\ \beta_k \end{pmatrix} + \begin{pmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_i \\ \vdots \\ \varepsilon_n \end{pmatrix} \tag{A.3}$$

$$\Rightarrow \mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{A.4}$$

The regression coefficients $\boldsymbol{\beta}$ are unknown. Using RSM, these coefficients are determined by least squares regression, which aims at minimising the quadratic error:

$$\min_{\boldsymbol{\beta}} L = \sum_{i=1}^{n} \varepsilon_i^2 = \boldsymbol{\varepsilon}^{\mathrm{T}} \boldsymbol{\varepsilon} \tag{A.5}$$

This can be accomplished by substituting Equation A.3 in Equation A.5, differentiating w.r.t. the regression coefficients $\boldsymbol{\beta}$ and putting the result equal to 0:

$$\left.\frac{\partial L}{\partial \boldsymbol{\beta}}\right|_{\mathbf{b}} = -2\mathbf{X}^{\mathrm{T}}\mathbf{y} + 2\mathbf{X}^{\mathrm{T}}\mathbf{X}\mathbf{b} = \mathbf{0} \Rightarrow \mathbf{b} = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\mathbf{y} \tag{A.6}$$

in which $\mathbf{b}$ are the least squares estimators of $\boldsymbol{\beta}$. Thus, the solution for $\mathbf{b}$ is the vector containing the values for the regression coefficients $\boldsymbol{\beta}$ that minimise the quadratic error and the metamodel $\hat{\mathbf{y}} = \mathbf{X}\mathbf{b}$ provides the best linear function for the measured data.

What can we do with this metamodel obtained by RSM? Well, we can use it to predict the value y_0, which is the response of an untried combination of the design variables, which are represented in the design vector $\mathbf{x_0}$:

$$\hat{y}_0 = \mathbf{x}_0^{\mathrm{T}}\mathbf{b} \tag{A.7}$$

y_0 is also shown in Figure A.1. Additionally, one can also determine the variance of $\hat{y}_0$ at that location [90]:

$$\mathrm{var}(\hat{y_0}) = \sigma^2 \mathbf{x}_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{x_0} \tag{A.8}$$

where σ^2 is the error variance, which can be estimated by the Mean Squared Error (MSE):

$$\mathrm{MSE} = \frac{\mathrm{SSE}}{n-p} = \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{n-p} \tag{A.9}$$

SSE is the Sum of Squared Errors, which equals the square of the difference between the measured response points and the response points predicted by the polynomial metamodel. n is the total number of measurements, whereas p is the number of regression coefficients.

Above, we introduced the concept of linear regression. However, in many cases, the relation between the design variables and the response will not be linear, but quadratic, with or without interaction effects, or of even higher order. In this case, one can simply substitute some linear variables by other quadratic variables. For example, the quadratic metamodel in the expression

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_{12} x_1 x_2 + \beta_{11} x_1^2 + \beta_{22} x_2^2 + \varepsilon \tag{A.10}$$

equals the one in

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \beta_3 x_3 + \beta_4 x_4 + \beta_5 x_5 + \varepsilon \tag{A.11}$$

where $\beta_3 = \beta_{12}, \beta_4 = \beta_{11}$ and $\beta_5 = \beta_{22}$ and the design variables x_3, x_4 and x_5 equal $x_1 x_2, x_1^2$ and x_2^2, respectively. Thus, using linear regression one can also fit higher order metamodels. In fact, as long as the metamodels stay linear w.r.t. the regression coefficients $\boldsymbol{\beta}$, linear regression can still be applied [90]. In RSM practice, metamodels of a higher order than two are seldom encountered, since 2$^{\text{nd}}$ order methods can take a variety of shapes and are sufficient in many practical cases. Additionally, the number of measurements necessary for fitting higher order metamodels increases exponentially.

The choice for polynomials in RSM is based on Taylor series expansion. Traditional RSM fits these polynomials through a number of obtained measurement points. However, if one has the opportunity to determine the first, second, third, etc. derivatives of the response with respect to all design variables, one may construct a Taylor polynomial from only one physical or numerical measurement as is done within the Finite Element code ANSYS [20]. The method is, however, not applicable to non-linear FEM simulations as is the case for metal forming.

A.2 Assumptions for RSM

It is important to be aware of the underlying assumptions when selecting a specific type of metamodel for certain application, especially when the metamodelling technique has not been developed for the application one is using it for. As already mentioned in Section 3.2, RSM was originally developed for obtaining easy to fit models from physical experiments. Physical experiments are stochastic, i.e. they typically show a specific amount of variation, which is nicely reflected by the random error term in Equation A.2. However, when applying RSM to numerical experiments like computer simulations, one should take notice of the assumptions underlying RSM presented in this section.

Two assumptions were already indicated in the previous section: the response should depend on the design variables in a linear (at least w.r.t. the regression coefficients) and additive way. If this is not the case, RSM will not provide a good metamodel. Additionally, Tunali et al. [137] point out the following assumptions for applying least squares regression to simulation models:

1. Observations are random samples from the population about which inferences are desired. This demand comes from physical or other stochastic processes, where recognised and unrecognised nuisance variables are present. This is why experiments for RSM need to be *randomised* for preventing systematic influence of unrecognised nuisance variables and *blocked* for eliminating the influence of recognised nuisance variables. A third technique called *replication* (performing experiments with the

same settings more than once) is important for finding the variance of the stochastic response

2. The metamodel is structurally adequate: the expected error equals 0 or $E(\varepsilon) = 0$
3. The errors are normally distributed
4. The error variance is homogenous, i.e. the variance of the error does not vary for different design variable settings: $\text{var}(\boldsymbol{\varepsilon}) = \sigma^2\mathbf{I}$. If this is the case anyway, *heteroscedasticity* is said to occur
5. The errors are statistically independent: $\text{cov}(\varepsilon_i, \varepsilon_j) = 0$ for $i \neq j$. This is known as autocorrelation of the variances

Other authors from the field of Social Sciences, in which linear regression techniques are used frequently, mention two additional assumptions [10, 11, 43]:

6. Collinearity should not be present, i.e. two design variables should not be linearly dependent on each other. If this is the case, the solution of the least squares regression coefficients will not be defined uniquely. If more than two design variables are linearly dependent, this is referred to as multicollinearity [47]
7. Each design variable is uncorrelated with the error term: $\text{cov}(x_i, \varepsilon) = 0$

When applying RSM, it is important to bare these assumptions in mind and to validate the metamodel for violations of the assumptions mentioned above.

A.3 Metamodel validation

The metamodel obtained by RSM should be validated on two fields. It should be tested for:

- the accuracy of the metamodel with respect to the measured data
- the assumptions presented in Section A.2

A.3.1 Testing for accuracy

Several tests can indicate the accuracy of the metamodel obtained by Response Surface Methodology [90]:

1. Test for significance of regression
2. Tests on individual regression coefficients

3. Test for lack of fit

The first test is to test if any design variable contributes to (the variability of) the response. If this is not the case, the metamodel does not reflect the measurements. This test can be done by the hypotheses [90]:

$$\begin{aligned} H_0 &: \beta_1 = \beta_2 = \cdots = \beta_k = 0 \\ H_1 &: \beta_j \neq 0 \text{ for at least one } j \end{aligned} \tag{A.12}$$

Let us again take a look at Figure A.1 and Equation A.2. We can now define the sum of squares (SS) of the total variability of the measurements (SST), the variability of the metamodel or regression model (SSR) and the variability of the remaining error (SSE) [38, 90]:

$$\text{SST} = \sum_{i=1}^{n} (y_i - \bar{y})^2 \tag{A.13}$$

$$\text{SSR} = \sum_{i=1}^{n} (\hat{y}_i - \bar{y})^2 \tag{A.14}$$

$$\text{SSE} = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 \tag{A.15}$$

$$\tag{A.16}$$

One can easily obtain that the following relation holds:

$$\text{SST} = \text{SSR} + \text{SSE} \tag{A.17}$$

Equation A.12 can now be tested by using *ANalysis Of VAriance* or *ANOVA* and the F-statistic:

$$F_0 = \frac{\text{SSR}/k}{\text{SSE}/(n-k-1)} = \frac{\text{MSR}}{\text{MSE}} \tag{A.18}$$

in which n is the number of measurements and k is the number of design variables. H_0 is rejected if:

- $F_0 \geq F_{\alpha,k,n-k-1}$
- $p \leq \alpha$

in which α is related to the confidence level and p is the p-value for the test statistic F_0. A measure for how well the metamodel explains the total variability in the measurements is the R^2-value defined as:

$$R^2 = \frac{\text{SSR}}{\text{SST}} = 1 - \frac{\text{SSE}}{\text{SST}} \tag{A.19}$$

A larger R^2 indicates a better metamodel, whereas the metamodel is perfect when $R^2 = 1$. Since R^2 increases with every design variable, also non-significant design variables will cause an increase of the R^2-value. To investigate if a design variable is significant, one can use the adjusted R^2-value, which is defined as [90]:

$$R^2_{\text{adj}} = 1 - \frac{n-1}{n-k-1}\left(1 - R^2\right) = 1 - \frac{\text{MSE}}{\text{MST}} \tag{A.20}$$

R^2_{adj} decreases when not significant variables are added. The F-statistic, p, R^2 and R^2_{adj}-values are all important measures for the significance of the metamodel w.r.t. the measured data.

The second test is also to determine if design variables are significant or not. The following hypotheses are tested:

$$\begin{aligned} H_0 &: \beta_j = 0 \\ H_1 &: \beta_j \neq 0 \end{aligned} \tag{A.21}$$

H_0 is rejected when the test statistic $t_0 \geq t_{\alpha/2,n-k-1}$ where t denotes the Student distribution and

$$t_0 = \frac{b_j}{\sqrt{\hat{\sigma}^2(\mathbf{X}^{\text{T}}\mathbf{X})^{-1}}} = \frac{b_j}{\text{se}(b_j)} \tag{A.22}$$

where b_j is the least squares estimator of the j^{th} regression coefficient, $\hat{\sigma}^2$ is the predicted response variance, $\mathbf{X}$ is the matrix given in Equation A.3 and $\text{se}(b_j)$ is short for the estimated standard error of the j^{th} regression coefficient. The significance of a specific regression coefficient can easily be seen from the confidence interval of this regression coefficient:

$$b_j - t_{\alpha/2,n-k-1}\text{se}(b_j) \leq \beta_j \leq b_j + t_{\alpha/2,n-k-1}\text{se}(b_j) \tag{A.23}$$

If the confidence interval includes the value 0, the considered design variable is not significant.

The third and last test is a test for lack of fit (LOF) of the metamodel and is only possible if replicate measurements of certain design variable settings are present [90]. Suppose we perform m measurements of the response y, replicate measurements not included, and $p = k+1$ regression coefficients need to be fitted. This leaves $n-m$ replicate measurement points for distinguishing between a Pure Error (PE) and a LOF Error part of the total SSE:

$$\text{SSE} = \text{SSPE} + \text{SSLOF} \tag{A.24}$$

Using ANOVA again, we can now test the F-statistic

$$F_0 = \frac{\text{SSLOF}/(m-p)}{\text{SSPE}/(n-m)} = \frac{\text{MSLOF}}{\text{MSPE}} \tag{A.25}$$

and conclude that there is no strong evidence for LOF if

- $F_0 \leq F_{\alpha,m-p,n-m}$
- $p \geq \alpha$

Thus, there is no evidence for suspecting the regression model to be of a higher order than assumed [90]. If on the contrary F_0 exceeds $F_{\alpha,m-p,n-m}$, LOF is expected to be present.

ANOVA is often presented in a table as shown in Table A.2.

Source of variation	*Sum of Squares*	*Degrees of Freedom*	Mean Square	F_0	p-value
Regression	SSR	k	MSR	$F_0 = \frac{\text{MSLOF}}{\text{MSPE}}$	p-value for regression
Residual	SSE	$n-k-1$	MSE		
Lack Of Fit	SSLOF	$m-k-1$	MSLOF	$F_0 = \frac{\text{MSLOF}}{\text{MSPE}}$	p-value for LOF
Pure Error	SSPE	$n-m$	MSPE		
Total	SST	$n-1$			

Table A.2: ANalysis Of VAriance

Next to the tests displayed above, another measure for the accuracy of the metamodel is the additional visualisation of the variance of the predicted response vs. the design variables. In Section A.1, it was indicated that one may predict the response at any design variable setting $\mathbf{x_0}$ by the polynomial metamodel:

$$\hat{y}(\mathbf{x}_0) = \mathbf{x}_0^{\mathrm{T}}\mathbf{b} \tag{A.26}$$

At this location, the prediction variance is defined as [90]:

$$\text{var}(\hat{y}(\mathbf{x}_0)) = \sigma^2 \mathbf{x}_0^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{x}_0 \tag{A.27}$$

The square root $\sqrt{\text{var}(\hat{y}(\mathbf{x}))}$ of this variance in relation to the metamodel is a measure for the accuracy of the metamodel. σ^2 may be estimated by the MSE from the ANOVA table.

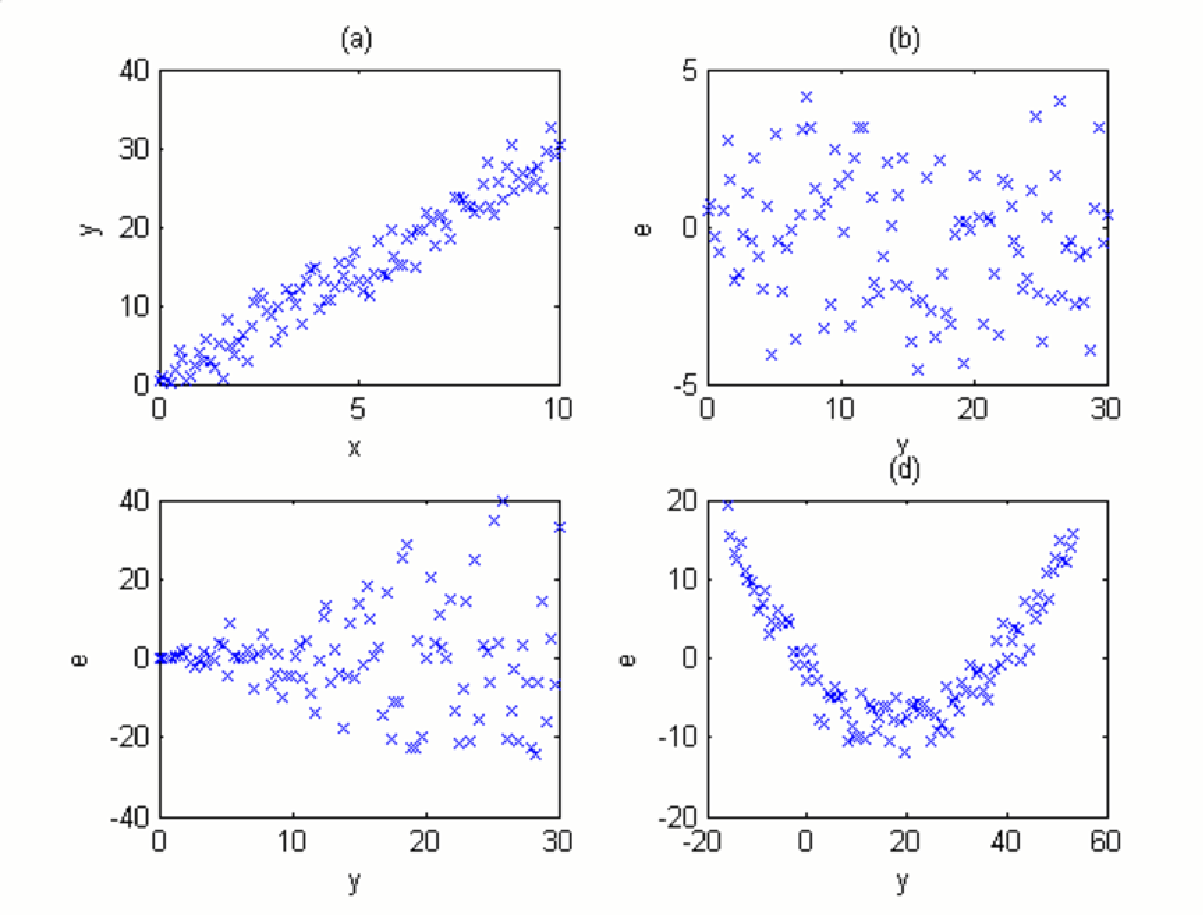


Figure A.2: (a) Partial regression plot; (b) Residual plot; (c) Residual plot showing heteroscedasticity; (d) Residual plot showing nonlinearity

A.3.2 Testing the assumptions

The first assumption of Section A.2 cannot be validated, but one should take randomisation, blocking and replication into account during sampling the measurements, i.e. while selecting a suitable DOE strategy (refer to Section A.4).

Linearity can easily be observed when plotting the response measurements vs. a design variable as presented in Figure A.2(a) [47]. This so-called *partial regression plot* works nicely for linear metamodels, but its use diminishes if interaction or quadratic metamodels are fitted using RSM. A better way is to take a look at the residuals (errors), i.e. the difference between the observed responses and the metamodel. This *Residual Analysis* is a well-known method, which is also capable of validating several error related assumptions from Section A.2. Residual analysis comprises analysing a number of residual plots. The first residual plot is a plot of the residual e vs. the response predicted by the metamodel

$\hat{y}$. Sometimes, the residual is standardised (or similar, studentised) w.r.t. the response variance. The standardised residual is defined as [90]:

$$d_i = \frac{e_i}{\hat{\sigma}} \tag{A.28}$$

in which $\hat{\sigma}$ is standard deviation, mostly defined as the square root of the Mean Sum of Squares of the Errors (MSE):

$$\hat{\sigma} = \sqrt{\text{MSE}} \tag{A.29}$$

Figure A.2(b) implies a good metamodel is fit. There is no trend in the errors, so:

- $E(\varepsilon) = 0$
- $\text{var}(\boldsymbol{\varepsilon}) = \sigma^2\mathbf{I}$
- The response is linear

Figure A.2(c) and (d) show residual plots for responses where heteroscedasticity and non-linearity occur, respectively. In contradiction to the partial regression plot shown in Figure A.2(a), the latter plot is also applicable for testing nonlinearity if higher order metamodels are used.

The assumption that the error is normally distributed can be validated by a *Normal probability plot* or by displaying the error distribution directly [89, 90]. The latter is known to be less reliable when only few measurements have been performed. Both are presented in Figure A.3. If the normal probability plot follows the line $y = x$, the errors are distributed normally. Otherwise, the distribution may be higher, lower, skewed, etc. depending on the

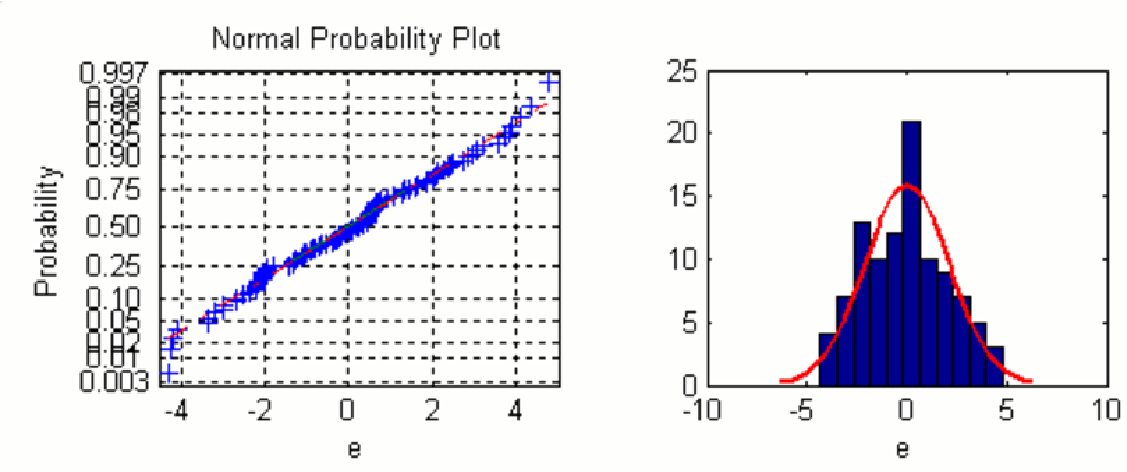


Figure A.3: Normal probability plot and error distribution

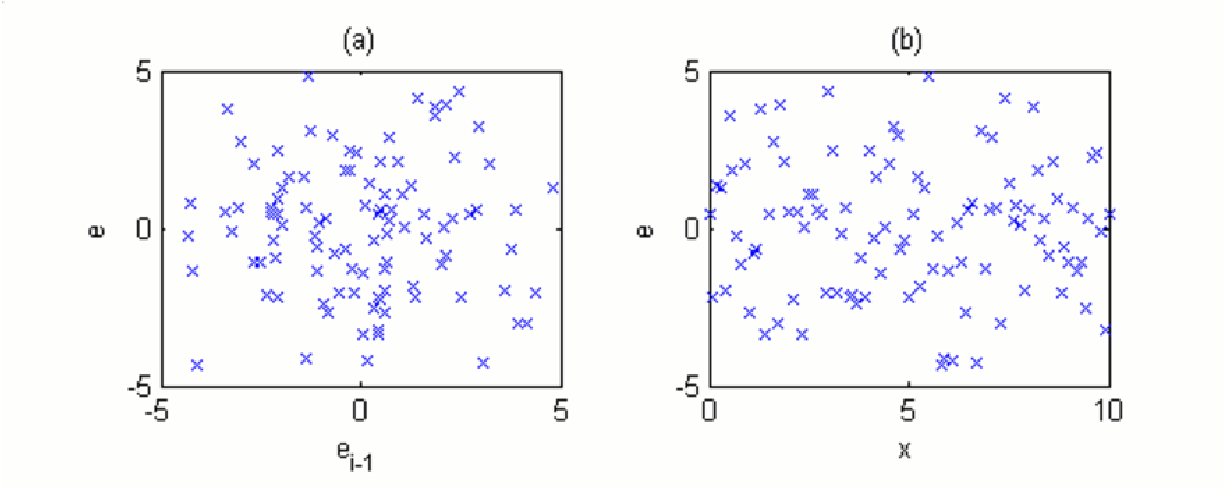


Figure A.4: (a) Autocorrelation plot; (b) Error - variable plot

shape of the deviation from the line $y = x$ [47].

A validation measure for the assumption about the autocorrelation of the errors is indicated by Tunali et al. [137]. It is a plot of the residuals e_i vs. the previous residual in run order e_{i-1}. If this plot shows any regularity, the errors are correlated. The example in Figure A.4(a) does not show any regularity, meaning the assumption is met:

$$\text{cov}\,(\varepsilon_i, \varepsilon_j) = 0 \;\text{ for }\; i \neq j \tag{A.30}$$

Figure A.4(b) shows a residual plot of the error vs. a design variable. Such a plot reveals if the design variable and error are correlated. If this is not the case as in Figure A.4(b), the following assumption is satisfied:

$$\text{cov}(x_i, \varepsilon) = 0 \tag{A.31}$$

Non-additivity and (multi-)collinearity are the last assumptions that need to validated. Tunali et al. [137] describe a test statistic and use Anscombe-Tukey plots for validating additivity. Multicollinearity is not a strict assumption, but may result in an erroneous solution of the regression variables, since the solution is not defined uniquely anymore. A way for validating multicollinearity is by applying RSM to each design variable w.r.t. the other design variables. The collinearity of each design variable can be estimated by the *tolerance* [47]:

$$\text{TOL} = 1 - R^2 \tag{A.32}$$

The lower the tolerance, the higher the collinearity.

A.4 Design Of Experiments (DOE) for RSM

A *Design of Experiment (DOE)* is a structured, organised method for determining the relationship between factors affecting a process and the output of that process [50]. "Classical" DOE strategies are often used to decide at which factor settings physical experiments should be performed in order to be able to characterise a phenomenon that is output of the process under investigation. After having performed the experiments indicated by the DOE strategy, this phenomenon can subsequently be fitted and analysed using Response Surface Methodology. Several good books from the field of Statistics are available on DOE (sometimes referred to as *experimental design* or *DOX*) [30,88,90]. Below, attention is paid to the desirable properties of DOE strategies used for RSM and several specific groups of DOE strategies are addressed.

A.4.1 Desirable properties of DOE strategies for RSM

Several desirable properties of DOE strategies for RSM are indicated by Myers and Montgomery [90]. The experimental design should:

- result in a good fit of the response
- test for Lack Of Fit and estimate the Pure Error
- be cost-effective

The first property seems trivial, but becomes less trivial when one realises it is often not known beforehand how the response behaves (linearly, quadratic, etc.). Two measures of how good a metamodel based on RSM is, are the variance of the regression coefficients and the prediction variance.

DOE strategies, which minimise the variance of the predicted regression coefficients are called *orthogonal* designs. Clearly, a lower variance of the regression coefficients, which are the unknown variables for fitting an RSM metamodel, results in a better metamodel. In a context of DOE, orthogonality is defined as:

$$\min \frac{\mathrm{var}(b)}{\sigma^2} \tag{A.33}$$

An orthogonal design satisfies the following relation:

$$\mathbf{X}^{\mathrm{T}}\mathbf{X} \sim \mathbf{I} \tag{A.34}$$

where $\mathbf{X}$ is the design matrix containing the settings of the design variables, which form the experimental design. Orthogonality ensures that the design variables are assessed independently w.r.t. each other. Orthogonality is especially important for fitting linear metamodels, which are often used for screening, i.e. trying to understand the response

entity and the important variables influencing this response, or for the initial scanning of a large design space for an optimum (see the four goals of metamodelling in Section 3.1). An additional property of orthogonal designs is that there is no correlation between the input settings. Thus, (multi-)collinearity cannot occur [114].

For second order metamodels, prediction variance is a much more important measure than the variance of the regression coefficients. The prediction variance ν is defined as:

$$\nu(\mathbf{x}) = \frac{\text{var}(\hat{y}(\mathbf{x}))}{\sigma^2} \tag{A.35}$$

For second order metamodels, prediction variance optimality is reflected by a property called *rotatability*. A rotatable experimental design is a design for which the prediction variance is equally large for any two points who are located at the same distance from the design center. The purpose of rotatability is to keep the prediction variance stable throughout the design space, which is a desirable property if one is interested in the metamodel as a prediction method or during the final, accurate optimisation of the response (again compare the four goals of metamodelling in Section 3.1). Second order response surfaces are often used for these goals.

For resulting in a good fit, the random error present in stochastic phenomena (ε in Equation A.3) will generally influence the estimation of the regression parameters. This influence is minimal if the experimental design points are located at the boundaries of the design space [90]. For explaining this, let us take a look at Figure A.5(a) [46], which presents two responses from DOE points that lie in the interior of the design space. Presume that the dashed line describes the real but unknown relation $y(x)$. Note that the linear metamodel (the solid line) provides a poor estimation, which is caused by the random error and the

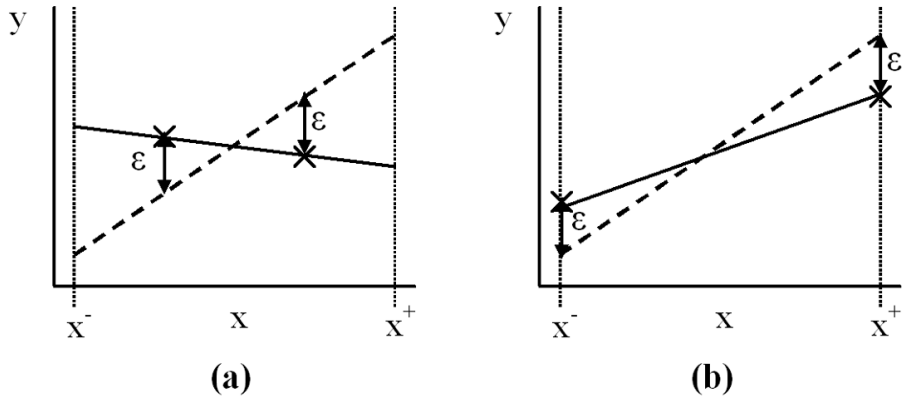


Figure A.5: (a) A linear metamodel constructed from DOE points in the interior of the design space; (b) A linear metamodel constructed from DOE points on the boundary of the design space

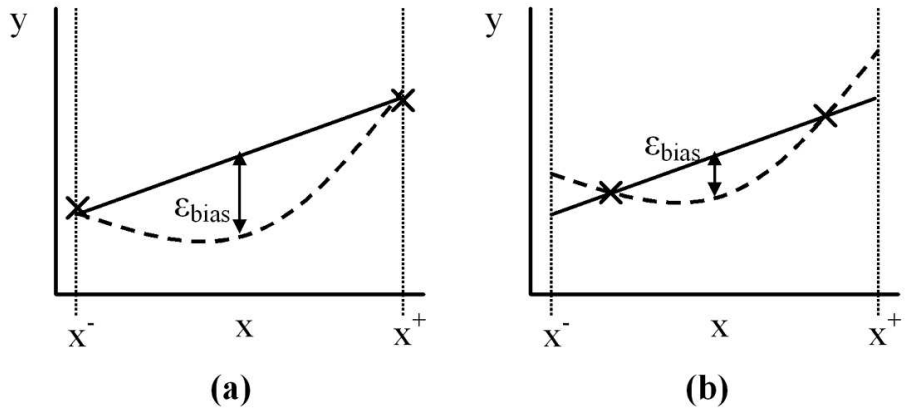


Figure A.6: (a) A linear metamodel constructed from DOE points on the boundary of the design space approximating a quadratic true response (b) A linear metamodel approximating constructed from DOE points in the interior of the design space a quadratic true response

small distance between the DOE points. Now, compare this result with the metamodel fitted through DOE points that lie on the boundary of the design space as presented in Figure A.5(b). The metamodel is much better and in general it can be concluded that good DOE strategies for RSM are those experimental designs for which the DOE points are forced to the boundaries of the design space [46, 90].

Whereas DOE points on the boundary of the design space increase the metamodel accuracy by minimising the variance of the regression coefficients, they are known to enhance the bias error. A *bias error* evolves when the true response is of a different shape than the presumed metamodel [46, 90]. For example, suppose a true response y obeys a quadratic relationship w.r.t. the design variable x as shown by the dashed line in Figure A.6(a). If we erroneously presume a linear polynomial metamodel and use DOE points on the boundary of the design space to fit this metamodel, we can distinguish a relatively large bias error $\varepsilon_{\text{bias}}$ from Figure A.6(a). If we, however, shift the DOE points to the interior of the design space, the bias error reduces significantly as presented in Figure A.6(b). Although minimising the bias error is an important feature of DOE strategies, it is assumed for RSM that the presumed metamodel shape matches the true response behaviour, which is in favour of placing the DOE points on the boundaries of the design space.

Summarising, whether a certain experimental design is a good DOE strategy w.r.t. the first desirable property, i.e. providing an accurate metamodel, is strongly related to the type of metamodel one is planning to fit. A design for fitting a first order polynomial should be (nearly) orthogonal, a suitable DOE strategy for a second order polynomial metamodel benefits more from good rotatability properties. For both orders of polynomials, experi-

mental designs with DOE points on the boundaries will provide more accurate estimates of the true phenomenon.

The second desirable property mentioned above is the possibility to test for Lack Of Fit (LOF) and estimation of the Pure Error (PE) of the response. As indicated in Section A.3, this can be done if one has the availability of *replicate* runs, i.e. running a number of experiments with the same design variable settings. The DOE strategy should allow for this. Since known and unknown nuisance variables can be present in case of realtime experimentation, a suitable DOE strategy should also include the concepts of blocking and randomisation. Often it is impossible to perform all experiments under the same homogeneous conditions, e.g. it is impossible to perform all experiments on one single day and a response may depend on the day of experimentation. The influence of this nuisance variable can be taken into account by *blocking*, i.e. grouping the measurements into several blocks. The extra variation caused by the nuisance variables is taken into account by spreading the replicate runs over the blocks [90]. In addition to known nuisance variables, many unknown nuisance variables may unsuspectedly influence the response. Performing the experiments in a random order, i.e. *randomisation*, will minimise the chance that the response gets structurally influenced by these unknown nuisance variables. If blocks are present, the DOE settings within the blocks are randomised [90]. A suitable DOE strategy for a certain problem will allow for all three concepts of replication, blocking and randomisation to be included.

The third and last property is the cost-effectiveness of the experimental design. Since (physical) experiments are quite time consuming and costly, the DOE strategy will have to attempt to keep the number of experiments to be performed at a minimum while ensuring that the phenomenon under investigation stays observable. *Saturated* designs play an important role for cost-effective experimentation. To explain this, let us review the minimum number of experiments required to fit a linear and a quadratic metamodel using RSM, respectively. The metamodel is dependent of k design variables:

$$\begin{aligned} p_{\text{lin}} &= 1 + k \\ p_{\text{quad}} &= 1 + 2k + k\frac{k-1}{2} \end{aligned} \tag{A.36}$$

If the number of experiments n is exactly equal to this minimum number of measurements necessary to fit the linear metamodel p_{lin} (or p_{quad} for a quadratic one), the experimental design is called a saturated design. A saturated design will always result in a metamodel with an R^2-value of 1 and does not leave any Degrees Of Freedom for testing for LOF. A saturated or nearly saturated DOE strategy is not recommendable w.r.t. the accuracy of the metamodel. However, saturated designs are the most efficient designs from a time and costs point of view.

Another property, which may prove useful if one does not know the order of the metamodel that is to be fitted, is the possibility for sequential experimentation. The procedure when fitting a metamodel using sequential experimentation starts with assuming the most simple, i.e. a linear, metamodel. Since Equation A.36 implies that a simple metamodel needs less experiments to fit, an appropriate DOE strategy for a linear metamodel will be relatively cost-efficient. If it turns out after metamodel validation that the response is indeed linear, the result is obtained without performing an extensive amount of experiments. If on the other hand, a linear metamodel appears to be insufficient, the DOE strategy should allow for the addition of only a few design variable settings to result in an experimental design for second order metamodels containing the desirable properties introduced earlier. The procedure for sequential experimentation guarantees a limited amount of required experiments for any shape of the response quantity.

After having introduced several desirable properties of DOE strategies for RSM, let us now turn to the description of several well-known designs.

A.4.2 Factorial designs

The most well-known group of experimental designs are the factorial designs. n^k *Full factorial designs* choose n levels for each of the k design variables, which results in a DOE strategy for performing a total of n^k experiments (note that each design variable denotes one dimensional direction). Figures A.7(a) and (b) present a 2^3 and a 3^2 full factorial design, respectively. A 2^k full factorial is suitable for fitting linear and interaction metamodels, whereas at least 3 levels are necessary for fitting second order metamodels.

Full factorial designs are relatively expensive, since the number of experiments increases exponentially with each extra level. A solution to limit the number of experiments is to use *fractional factorial designs*. The applicability of fractional factorial designs is based on three key ideas [90]:

- The Sparsity-of-Effects Principle, which states that it is likely that a system or process is primarily driven by the main effects and lower order interactions rather than higher order effects and interactions
- The Projection Property: fractional factorial designs can be projected into stronger designs in a subset of (fewer) significant factors
- Sequential Experimentation: it is possible to combine two or more fractional factorial designs to assemble a stronger design sequentially to estimate higher order effects and interactions as already discussed earlier

An n^{k-p} fractional factorial design is the $(1/n)^p$ fraction of an n^k full factorial design. Figure A.7(c) presents a 2^{3-1} fractional factorial design. Note that the number of experiments

that needs to be performed is actually $2^{3-1} = 2^2 = 4$. This reduction of the total number of experiments comes at the cost of being able to determine only $(1/2)^p$ part of all regression coefficients uniquely (group 1). The remaining regression coefficients (group 2) are *aliased* with group 1. For determining which of the regression coefficients is actually significant, the Sparsity-of-Effects Principle states that the lowest order main or interaction effect is significant and that the highest order terms can be neglected. The *resolution* of an experimental design indicates which effects from group 2 are aliased with which effects from group 1 [90]:

- Resolution III: main effects are not aliased with each other, but with two-factor interactions and two-factor interactions are aliased with each other
- Resolution IV: main effects are not aliased with each other nor with two-factor interactions, but two-factor interactions are aliased with each other
- Resolution V: main effects are not aliased with each other nor with two-factor interactions, two-factor interactions are not aliased with each other either, but they are aliased with three-factor interactions

The 2^{3-1} fractional factorial design presented in Figure A.7(c) is of resolution III and is a 2^{3-1}_{III} design: the four experiments only allow for the estimation of the four main effects $\beta_0, \beta_1, \beta_2$ and β_3. The two-factor interactions ($\beta_{12}, \beta_{13}, \beta_{23}$) and the three-factor interaction β_{123} are aliased with the main factors.

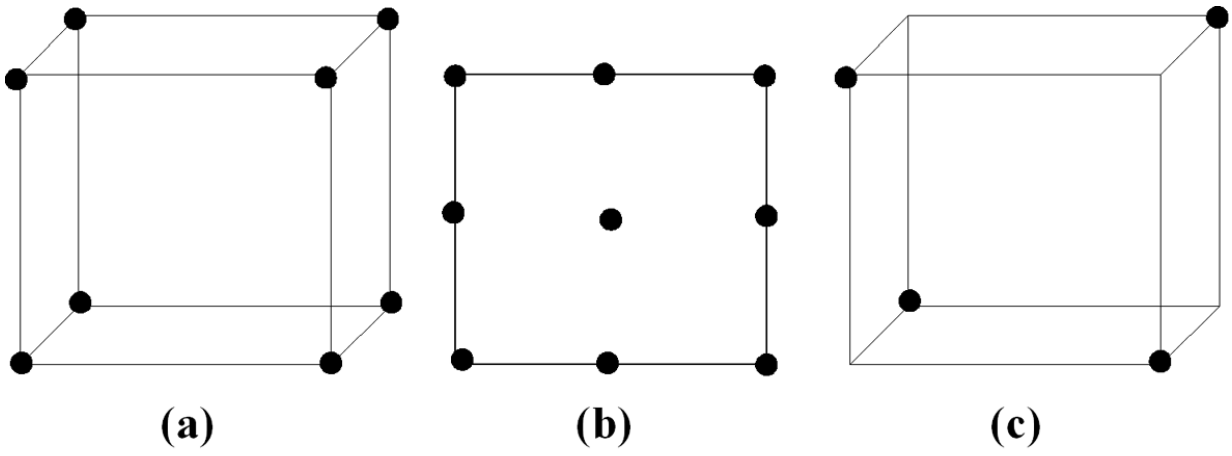


Figure A.7: (a) 2^3 full factorial design; (b) 3^2 full factorial design; (c) 2^{3-1} fractional factorial design

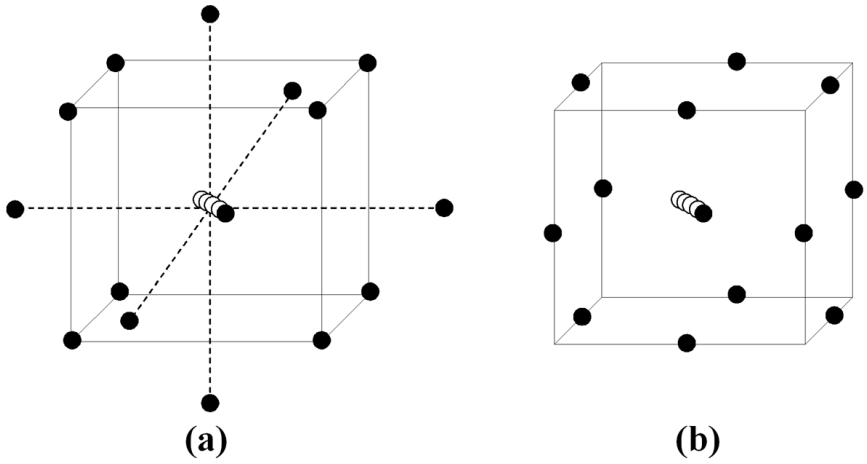


Figure A.8: (a) 3 factor Central Composite Design; (b) 3 factor Box-Behnken Design

A.4.3 Response Surface designs

Next to factorial designs, a second important group of DOE strategies for RSM are special Response Surface designs. Two important designs are the Central Composite Design and Box-Behnken design, which are presented for three factors in the Figures A.8(a) and (b), respectively.

The *Central Composite Design* (CCD) is a well-known and very attractive DOE strategy for RSM because it satisfies many of the desirable properties introduced in Section A.4.1:

- It allows for Sequential Experimentation, since it is a combination of a 2^k full factorial design with $2k$ axial or star points and n_c center points. The factorial design can initially be used for the fitting of a linear metamodel, whereas the other points can be added if the linear metamodel is found to be insufficient
- The factorial design is variance optimal and of Resolution V, meaning that the main and interaction effects can be determined uniquely (are not aliased w.r.t. each other)
- The axial points contribute to the estimation of second order effects in a major way, since they are spread over five levels
- The axial points are chosen in such a way that the CCD is rotatable
- The n_c center runs contribute to the estimation of the quadratic effects and provide an estimate of the Pure Error

Note that the CCD covers a (hyper)spherical design space. In many cases, however, one is bounded to a rectangular design space and the CCD will penetrate into the non-feasible

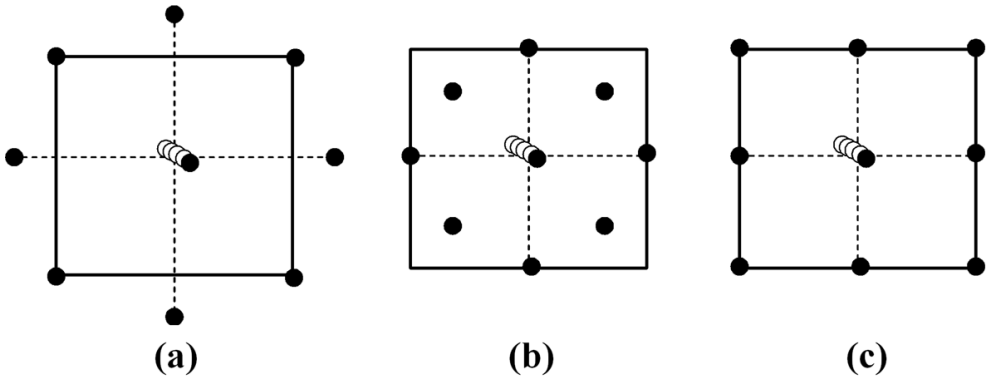


Figure A.9: (a) 2 factor (Circumscribed) CCD (b) 2 factor Inscribed CCD (c) 2 factor Face Centered CCD

domain. To overcome this problem, one may scale the original *Circumscribed CCD* into an *Inscribed CCD* reducing the radius of the design as presented in Figure A.9(b). A disadvantage of doing this is the lack of measurements in the corners of the design space. If one is interested in an accurate metamodel at the boundaries of the design space, for example if one uses the metamodel for constrained optimisation, this design is not recommendable. In this case, one is advised to consider a *Face Centered CCD* as presented in Figure A.9(c). Note that a 2 factor Face Centered CCD is similar to a 3^2 full factorial design. This is not the case for 3 or more factors.

The second Response Surface design is the Box-Behnken design (BBD) (see Figure A.8(b)). This is a design of three levels, whereas the CCD comprised five levels. It is recommended for fitting second order metamodels using RSM if the experimenter is somehow limited to performing experiments on three levels only. A BBD is nearly rotatable and the center runs allow for fitting second order effects and estimating Pure Error. Note, however, that the BBD is an inscribed spherical design meaning that the design, just as the inscribed CCD, is not recommended if one is interested in highly accurate results at the boundaries of the design space [90].

A.4.4 Computer generated designs

A major problem for applying Factorial or Response Surface designs in practice is their assumption of (hyper-)circular or (hyper-)rectangular design spaces. If explicit constraints are present for metamodel based optimisation for example, the design space will not be rectangular anymore, which rules out classical DOE strategies. This is where in case of RSM computer generated designs should be applied [90]. Computer generated designs are useful for:

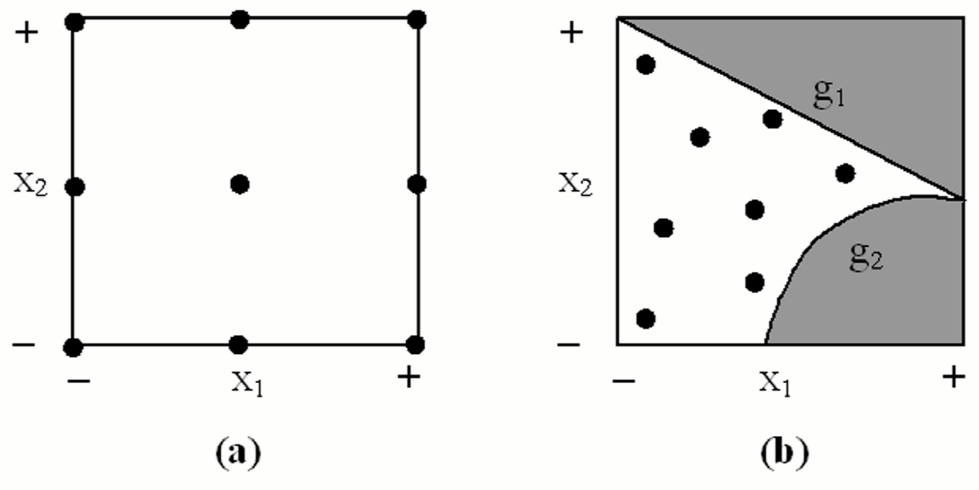


Figure A.10: (a) Face Centred Central Composite Design (b) Computer generated design

1. non-spherical or non-rectangular design spaces
2. augmenting a design or replacing a point
3. designing an experiment from scratch, without selecting a specific type of DOE strategy beforehand

The first use is presented in Figure A.10. The rectangular design space from Figure A.10(a) is described by a Face Centered CCD and the design space restricted by the explicit constraints g_1 and g_2 as presented in Figure A.10(b) is filled by a computer generated design. No classical DOE strategy is suitable for selecting experiments in such an irregular design space. The second and third advantages of applying computer generated designs mainly find their application in the possibility for Sequential Experimentation in which an initial design, e.g. a factorial design, is augmented with additional experiments to be able to fit higher order metamodels.

Several computer generated designs exist and are named after letters from the alphabet. A distinction is made between computer generated designs based on design optimality w.r.t. the variance of the regression coefficients and ones based on optimality w.r.t. prediction variance (see orthogonality and rotatability in Section A.4.1).

Computer generated designs based on the variance of the regression coefficients
The most well-known computer generated design is a D-optimal design. A DOE strategy is D-optimal when:

$$\max_{\mathbf{x}} \left| \frac{\mathbf{X}^{\mathrm{T}}\mathbf{X}}{N^p} \right| \tag{A.37}$$

where $\mathbf{X}$ is the design matrix containing the experimental design, N is the number of design points and p the number of parameters in the model. $|.|$ denotes the determinant. A D-optimal design minimises the volume of the confidence intervals on the RSM regression coefficients. A second computer generated design, which is optimal w.r.t. the variance of the regression coefficients is A-optimality:

$$\min_{\mathbf{x}} \operatorname{tr} \left(\frac{\mathbf{X}^{\mathrm{T}}\mathbf{X}}{N^p} \right)^{-1} \tag{A.38}$$

For A-optimality only the variance of the regression coefficients is taken into account, whereas D-optimality also includes the covariances.

Computer generated designs based on the prediction variance
Next to the D- and A-optimal experimental designs, DOE strategies exist that are G-, V- or Q-optimal. These are, however, not based on minimising the variance of the regression coefficients, but are based on the prediction variance ν given by Equation A.35 in Section A.4.1. A G-optimal design minimises the maximum prediction variance in the design space:

$$\min_{\mathbf{x}} \left(\max \nu(\mathbf{x}) \right) \tag{A.39}$$

Another criterion based on prediction variance is V-optimality, which minimises the average prediction variance over a set of predefined points in the design space. A computer generated design related to V-optimality is Q-optimality, which averages the prediction variance over some region of interest R by integration over this region of interest and division by the volume of this region of interest V_R:

$$\min_{\mathbf{x}} \frac{1}{V_R} \int_R \nu(\mathbf{x}) d\mathbf{x} \tag{A.40}$$

Both minimising the variance of the regression coefficients and the prediction variance are important properties of a good DOE strategy. Although D-optimality is most widely used, it should be emphasised that this criterion sometimes lacks the competence of minimising the prediction variance. As a consequence, D-optimality sometimes results in quite large prediction variances.

Appendix B

Design and Analysis of Computer Experiments (DACE)

Design and Analysis of Computer Experiments or DACE was introduced by Sacks, Welch et al. [111, 112] as an alternative for using Response Surface Methodology for computer simulations. It is based on the assumption that computer simulations are deterministic, i.e. will result in the same response for two or more replicate runs. This is in contradiction to the stochastic nature of physical experiments for which RSM was developed. Sacks et al. claim that in case of deterministic responses, the use of RSM is statistically incorrect since the random error term is absent. They propose to interpolate the metamodel through the response measurements and refer to this metamodelling technique as Design and Analysis of Computer Experiments (DACE). Section B.1 describes how the metamodel is fitted using DACE. Some attention is given to assumptions underlying DACE in Section B.2. These assumptions and the accuracy of the metamodel need to be validated as described in Section B.3. Finally, Section B.4 describes several DOE strategies suitable for DACE.

B.1 Fitting the metamodel

For RSM it was presented in Appendix A that a response y is predicted by a polynomial metamodel $\hat{y} = \mathbf{X}\boldsymbol{\beta}$ allowing for a remaining random error term:

$$y = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\varepsilon} \tag{B.1}$$

Since computer experiments are said to be deterministic, the remaining error at a response measurement should equal 0 indicating that interpolated metamodels are better applicable to computer simulations than RSM. Sacks et al. [111, 112] have proposed to use *Kriging* to achieve this. Using Kriging, the random error term in RSM $\boldsymbol{\varepsilon}$ is replaced by a Gaussian stochastic process $Z(\mathbf{x})$, which forces the metamodel to go through the measurement points. The DACE metamodel now becomes:

$$\hat{y} = \mathbf{X}\boldsymbol{\beta} + Z(\mathbf{x}) \tag{B.2}$$

The first part of Equation B.2 covers the global trend of the metamodel. The Gaussian stochastic process Z, which accounts for the local deviation of the data from the linear regression metamodel, has zero mean, variance σ_z^2 and covariance

$$\operatorname{cov}\left(Z(x_1), Z(x_2)\right) = \sigma_z^2 R\left(x_1 - x_2\right) \tag{B.3}$$

where R is the correlation function and x_1 and x_2 are two design variable settings. Any correlation function for which $R\left(x_i, x_i\right) = 1$ would suffice, but a correlation function of the form $R\left(x_1 - x_2\right)$ makes the stochastic process stationary [69]. A further restriction is making the correlation function symmetric, i.e. only dependent on the magnitude of the distance between the two design variable settings: $R\left(|x_1 - x_2|\right)$.

Up to now, the exact shape of the correlation function has not been specified yet and this is the point where an a priori choice needs to be made for this exact shape. Several types of correlation functions exist that satisfy the restriction $R\left(|x_1 - x_2|\right)$ mentioned above. Literature [69, 81, 114] reports exponential functions of the form $\exp^{-\vartheta|x_1 - x_2|^p}$, the Matèrn correlation function, splines, linear and cubic functions. Although a correlation function should be chosen that fits the behaviour of the problem entity, the Gaussian exponential functions (exponential functions for which $p = 2$) are intuitively attractive because they are infinitely differentiable. Moreover, Gaussian exponential functions are frequently used in literature [114] and have been found to give accurate results [80]. Due to a lack of knowledge on the behaviour of the problem entity, this fact led us to use Gaussian exponential correlation functions in this report. This group of functions is defined as:

$$R(\vartheta, x_1, x_2) = \exp^{-\vartheta(x_1 - x_2)^2} \tag{B.4}$$

When p is assumed to be 2, the only remaining coefficient is ϑ. The influence of the value of ϑ on the metamodel fitted through several measurement points in one dimension is indicated in Figure B.1: the larger ϑ, the smaller the distance influenced by a certain measurement.

The remaining question is how to fit a Kriging metamodel given a number of measurement points $\mathbf{x}$ and their responses $\mathbf{y}$. We use the measurements presented in Table A.1 in Section A.1, which are again shown in Figure B.2. Suppose we would like to predict the response y_0 shown in the figure by applying Kriging. Recall from Equations B.2 and B.3 that the y_0 depends on $\boldsymbol{\beta}$ and the Gaussian stochastic process Z, which is completely dependent on the process variance σ_z^2 and the correlation function R. Adopting the Gaussian exponential correlation function, R is solely dependent on the parameter ϑ. We do not know either of them, thus if we would like to predict y_0, we should estimate $\boldsymbol{\beta}$, σ_z^2 and ϑ [114].

The basic idea of estimating $\boldsymbol{\beta}$, σ_z^2 and ϑ is to minimise the *Mean Squared Prediction Error* [114]:

$$\text{MSPE} = E\left(\hat{y}_0 - y_0\right)^2 \tag{B.5}$$

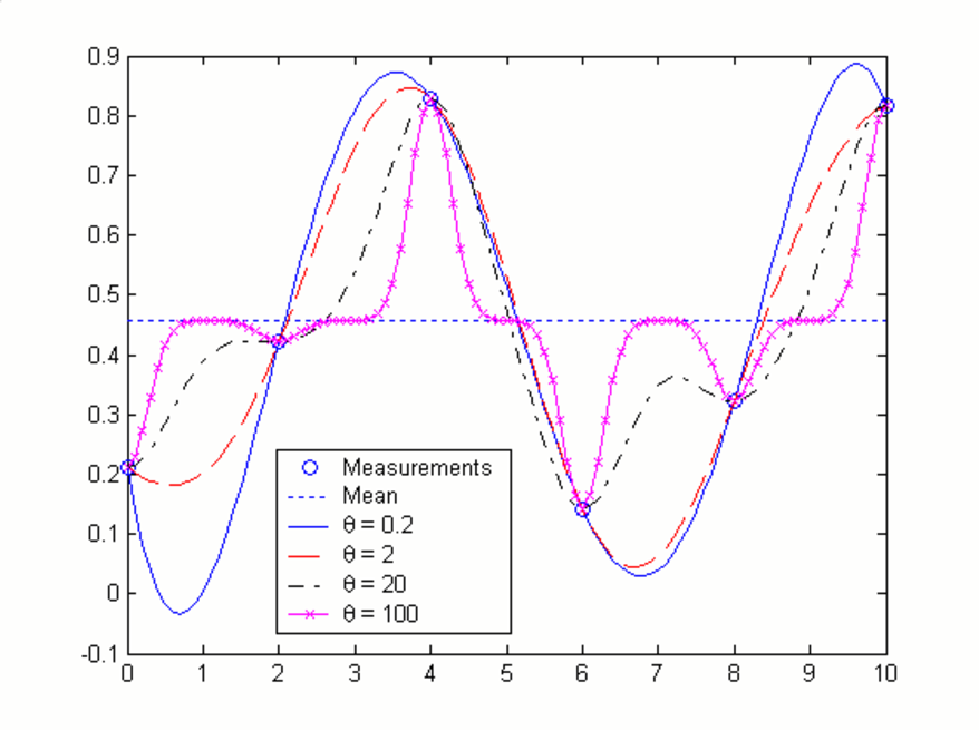


Figure B.1: The influence of ϑ on the DACE metamodel

Note that this idea is basically the same as minimising the Least Squares Error in RSM. A well-known method for estimating the unknown parameters is *Maximum Likelihood Estimation* (MLE), see e.g. [104]. MLE relies on the assumption that the observed measurements are a result of a Gaussian stochastic process, which is the basic assumption of Kriging anyway. Maximising the likelihood function, or equivalently the *log likelihood*, with respect to all unknown parameters $\boldsymbol{\beta}$, σ_z^2 and ϑ will result in the following *Best Linear Unbiased Predictor (BLUP)* for the response y_0 as a function of the design variable settings x_0 is (see e.g. [69, 81, 82, 114]):

$$\hat{y}_0 = \mathbf{f}^{\mathrm{T}}(x_0)\hat{\boldsymbol{\beta}} + \mathbf{r}^{\mathrm{T}}(x_0)\hat{\mathbf{R}}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{B.6}$$

in which $\mathbf{f}$ is a vector of regression functions, i.e. the design matrix for x_0:

$$\mathbf{f} = \begin{pmatrix} 1 \\ x_0 \end{pmatrix} \tag{B.7}$$

$\mathbf{r}$ is a vector containing the correlation between the the unknown point (x_0, y_0) and the known measurements (x_i, y_i) for $i = 1...6$:

$$\mathbf{r} = \begin{pmatrix} R(x_0, x_1) & R(x_0, x_2) & R(x_0, x_3) & R(x_0, x_4) & R(x_0, x_5) & R(x_0, x_6) \end{pmatrix}^{\mathrm{T}} \tag{B.8}$$

$\mathbf{F}$ is the design matrix at each known measurement location:

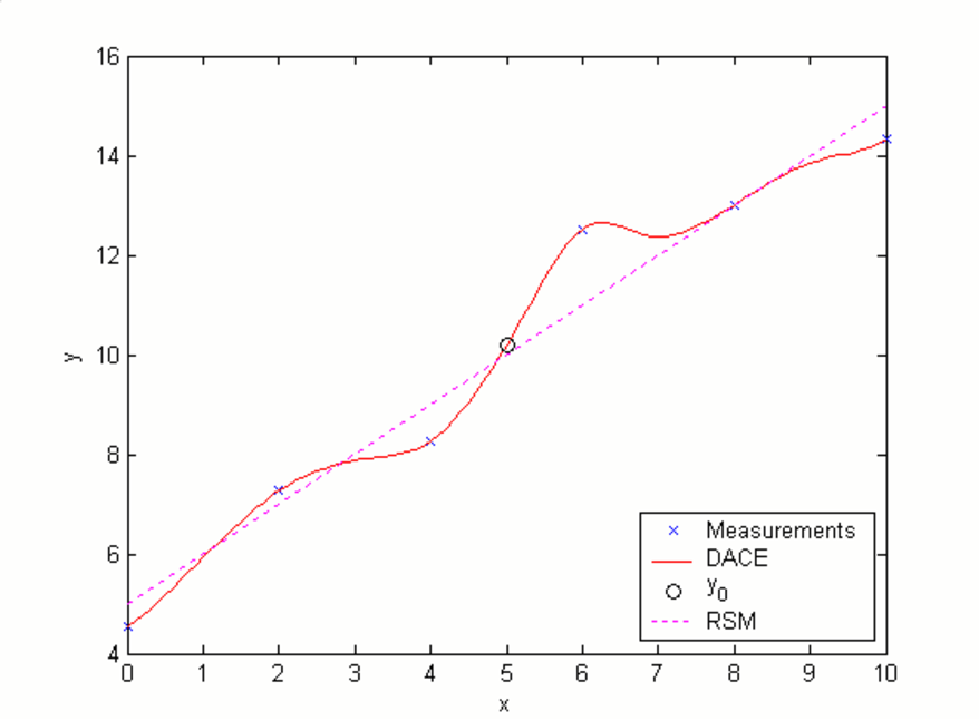


Figure B.2: Fitting a Kriging metamodel

$$\mathbf{F} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 \\ x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{bmatrix} \tag{B.9}$$

$\mathbf{y}$ is a vector with known response measurements:

$$\mathbf{y} = \begin{pmatrix} y_1 & y_2 & y_3 & y_4 & y_5 & y_6 \end{pmatrix}^{\mathrm{T}} \tag{B.10}$$

Note that the regression coefficients $\boldsymbol{\beta}$ and the correlation function R, which only depends on ϑ, are replaced by their estimators $\hat{\boldsymbol{\beta}}$ and $\hat{R}$. It can be shown (see e.g. [69, 81, 82, 114]) that $\hat{\boldsymbol{\beta}}$ is the *Generalised Least Squares estimate* of the regression coefficients $\boldsymbol{\beta}$:

$$\hat{\boldsymbol{\beta}} = (\mathbf{F}^{\mathrm{T}}\hat{\mathbf{R}}\mathbf{F})^{-1}\mathbf{F}^{\mathrm{T}}\hat{\mathbf{R}}^{-1}\mathbf{y} \tag{B.11}$$

The term "Generalised Least Squares" refers to the extension of the (Ordinary) Least Squares estimate of the regression coefficients within linear regression (RSM) as presented in Equation A.6 of Section A.1. Note that if the measurement points are uncorrelated, i.e. $\hat{\mathbf{R}} = \mathbf{I}$, Equation B.11 reduces to Equation A.6 (since $\mathbf{X}$ and $\mathbf{F}$ are both symbols for the same design matrix).

At this point, it is convenient to explain the difference between three types of Kriging encountered in literature: Ordinary, Universal and Detrended Kriging [82]. *Ordinary Kriging*

assumes a constant mean over the domain and the 0^{th} order polynomial is used as a regression model. *Universal* and *Detrended Kriging* both fit higher order linear regression models to account for non-stationary means. The difference between Universal and Detrended Kriging lies in the way the regression coefficients $\boldsymbol{\beta}$ are estimated. Using Detrended Kriging the data points are assumed to be uncorrelated, i.e. $\mathbf{R} = \mathbf{I}$, whereas Universal Kriging incorporates correlations between the data. It can be stated that Detrended Kriging combines both linear regression and Ordinary Kriging, whereas Universal Kriging will result in slightly different regression coefficients than linear regression [82].

Similarly to the estimation of $\boldsymbol{\beta}$, we can also use MLE to estimate σ_z^2. It turns out that:

$$\hat{\sigma}_z^2 = \frac{1}{n}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right)^{\mathrm{T}} \hat{\mathbf{R}}^{-1}\left(\mathbf{y} - \mathbf{F}\hat{\boldsymbol{\beta}}\right) \tag{B.12}$$

in which n is the number of known measurements.

Now $\boldsymbol{\beta}$ and σ_z^2 have been estimated by MLE, let us shift to the estimation of R, which is also necessary for estimating $\boldsymbol{\beta}$ and σ_z^2 as one can see in Equation B.11 and B.12. As already mentioned above, R is solely dependent on the parameter ϑ in case of a Gaussian exponential correlation function. The MLE estimate of ϑ can be shown to be the ϑ that maximises [80, 114]:

$$\hat{\vartheta} = \max_{\vartheta} n \ln \hat{\sigma}_z^2(\vartheta) + \ln |\mathbf{R}(\vartheta)| \tag{B.13}$$

This is equivalent to the ϑ that minimises [80, 81]:

$$\hat{\vartheta} = \min_{\vartheta} |\mathbf{R}(\vartheta)|^{\frac{1}{n}} \hat{\sigma}_z^2(\vartheta) \tag{B.14}$$

in which $|\mathbf{R}|$ denotes the determinant of $\mathbf{R}$. A nonlinear optimisation algorithm is applied to maximise Equation B.13 or to minimise Equation B.14. For larger problems, this can become a time consuming operation. As soon as $\hat{\vartheta}$ is known, $\hat{\boldsymbol{\beta}}$ and $\hat{\sigma}_z^2$ can be calculated easily using Equations B.11 and B.12 and the BLUP of y_0 is consequently also known from Equation B.6.

The estimation of a value for y_0 comes with uncertainty. The prediction of an unknown point y_0, which is close to a known measurement, will intuitively be more accurate than one that lies far from a known measurement. This is taken into account by the *Mean Squared Error (MSE)* at y_0, which is expressed by (see e.g. [82]):

$$\mathrm{MSE}\,(y_0) = \sigma_z^2\left(1 - \begin{bmatrix} \mathbf{f}^{\mathrm{T}} & \mathbf{r}^{\mathrm{T}} \end{bmatrix} \begin{bmatrix} \mathbf{0} & \mathbf{F}^{\mathrm{T}} \\ \mathbf{F} & \mathbf{R} \end{bmatrix} \begin{bmatrix} \mathbf{f} \\ \mathbf{r} \end{bmatrix}\right) \tag{B.15}$$

It can be shown that Equation B.15 equals 0 in case y_0 equals a known measurement point, which demonstrates that Kriging interpolates between the measurement data.

Above, it was shown how to predict one unknown value y_0 and its expected MSE given a number of measurement points, which depend on one single design variable. The procedure can readily be extended to the prediction of many unknown values $\mathbf{y}_0$ dependent on one design variable. Predicting a number of points and connecting them will visualise the Kriging metamodel as shown in Figure B.2. Note that the metamodel indeed interpolates the known measurement points $y_1 ... y_6$.

If more, say k design variables are present, the total correlation function R is assumed to depend on the k one-dimensional correlation functions R_j as follows [111], i.e. one assumes there is no relation between the different dimensions:

$$R\left(x_1 - x_2\right) = \prod_{j=1}^{k} R_j\left(x_{1j} - x_{2j}\right) \tag{B.16}$$

Adopting the Gaussian correlation function introduced in Equation B.4, the total correlation function becomes:

$$R\left(x_1 - x_2\right) = \prod_{j=1}^{k} \exp^{-\vartheta_j (x_{1j} - x_{2j})^2} \tag{B.17}$$

Above, we used MLE for estimating the unknown parameters. Several other ways for estimation are known. One of them is based on minimising the *Cross Validation (CV)* error. Cross Validation will be addressed in detail in Section B.3. Here, we restrict ourselves to mentioning that MLE is preferred above CV and other estimation techniques [83, 114].

B.2 Assumptions for DACE

The main assumption for DACE is indicated by Martin and Simpson [82]: one assumes a zero mean Gaussian stochastic process for $Z(x)$ in Equation B.2. A Gaussian stochastic process is a stochastic process for which all stochastic variables have the multivariate normal distribution, the process is stationary by definition [140] and its variance is bounded and constant over its domain [82].

The above demands coincide with a number of the assumptions underlying RSM described in Section A.2. Assumptions 1 (stochastic process), 5 ($\mathrm{cov}(\varepsilon_i, \varepsilon_j) = 0$), 6 (multicollinearity) and 7 ($\mathrm{cov}(x_i, \varepsilon_j) = 0$) can be dropped. Testing the remaining assumptions 2, 3 and 4 from Section A.2 is still useful for DACE. Replacing the errors ε for RSM by the Gaussian stochastic process $Z(x)$ for DACE, the assumptions may be formulated as follows:

1. The mean of the Gaussian stochastic process $Z(x)$ is 0 and constant: $E(Z(x)) = 0$

2. The stochastic variables determined by $Z(x)$ have a multivariate normal distribution

3. The variance of the Gaussian stochastic process $Z(x)$ is bounded and constant: $\text{var}\left(Z(x_i), Z(x_j)\right) = \sigma_z^2 \mathbf{I}$

Next to the Gaussian stochastic process assumption, a second assumption lies within the choice of a Gaussian exponential correlation function. This is the most popular correlation function and has shown to perform well for many applications [80, 117], but it is not certain whatsoever that it is also suitable for the optimisation of metal forming processes. Additionally, the use of Equation B.16 implies that there is no relation between correlation functions in different dimensions.

The last fundamental assumption of DACE is that the measurements are deterministic. Computer experiments are thought to be deterministic, i.e. two calculations with the same design variable settings result in exactly the same answer. However, for Finite Element analyses, numerical parameters like element mesh refinement, step size adjustment, etc. are known to cause so-called *numerical noise*. Caution is necessary when dealing with this phenomenon.

B.3 Metamodel validation

As was the case for RSM, it is again important to validate both the accuracy of the DACE metamodel and the assumptions underlying the use of DACE.

B.3.1 Testing for accuracy

When a metamodel is fitted using Kriging, the most straight-forward way of validation is to run a batch of validation calculations. The results of these validation calculations can be compared to the metamodel, which was constructed using a number of previous computer experiments. This demands, however, a number of calculations that is much larger than the amount of experiments needed for constructing the metamodel. Given the fact that these additional computer experiments each take a considerable amount of time, makes this method an unattractive one.

Cross validation (CV) is a cheap way of metamodel validation and a more attractive alternative in case of expensive function evaluations (see e.g. [114, 117, 123]). Using CV, one leaves out a number of the n result measurements. Regularly, one result is left out, whereas some literature reports that, for Kriging, leaving $0.1n$ or $\sqrt{n}$ measurements out is more reliable [87]. We will leave only one result out, i.e. *leave-1-out cross validation*.

After having removed a single, say the i^{th}, result y_i, the metamodel is fitted through the remaining $\mathbf{y_{-i}}$ measurements and the squared error $(y_i - \hat{y}_{-i})^2$ is calculated. $\hat{y}_{-i}$ is the value of y_i predicted by the metamodel using only the remaining $\mathbf{y_{-i}}$ measurements. Repeating the same procedure for every one of the n measurements and summing the results gives the Sum of Squared cross validation errors:

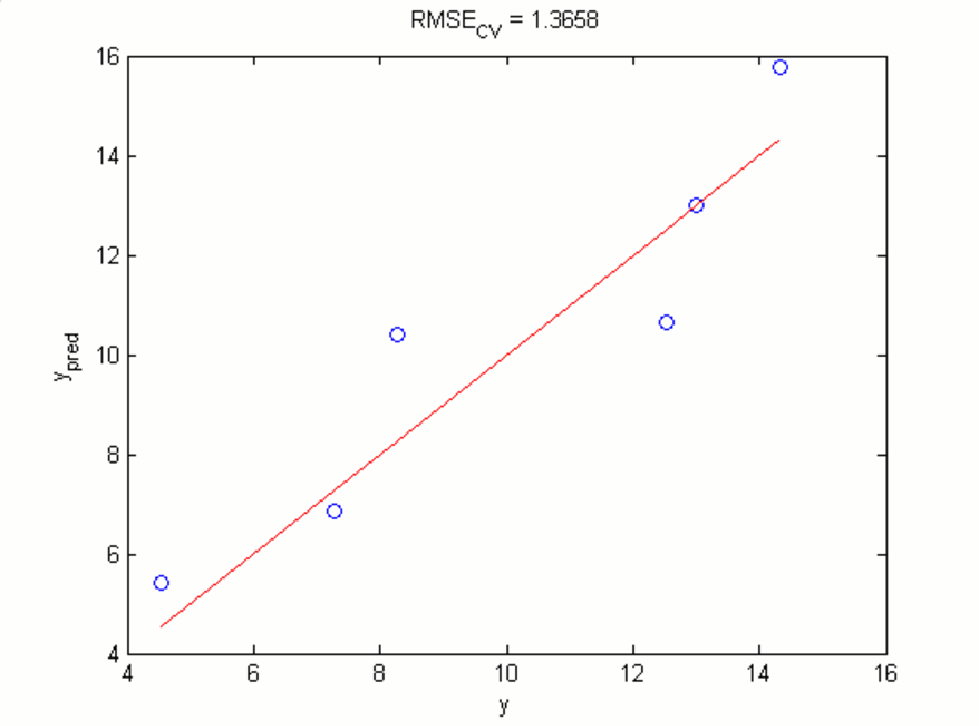


Figure B.3: Cross validation plot

$$\mathrm{SSE}_{\mathrm{CV}} = \sum_{i=1}^{n} \left(y_i - \hat{y}_{-i}\right)^2 \tag{B.18}$$

Dividing by the number of measurements n yields the cross validation MSE and taking the square root results in the cross validation Root Mean Squared Error ($\mathrm{RMSE}_{\mathrm{CV}}$):

$$\mathrm{RMSE}_{\mathrm{CV}} = \sqrt{\sum_{i=1}^{n} \frac{\left(y_i - \hat{y}_{-i}\right)^2}{n}} \tag{B.19}$$

As $\mathrm{RMSE}_{\mathrm{CV}}$ approaches 0, the metamodel becomes more and more accurate. Cross validation can also be visualised in a cross validation plot. An example of such a plot is presented in Figure B.3. If the measurements follow the line $y_{\mathrm{pred}} = y$, the metamodel fits the data well.

A disadvantage of calculating this measure for metamodel accuracy is that $\mathrm{RMSE}_{\mathrm{CV}}$ is scale dependent. Larger values of the measurements give larger errors. This is why some authors propose to standardise the $\mathrm{RMSE}_{\mathrm{CV}}$. Next to a cross validation plot, Schonlau [117] proposes to plot the standardised cross validated errors e_i versus the predictions $\hat{y}_{-i}$. e_i is defined as:

$$e_i = \frac{(y_i - \hat{y}_{-i})}{\hat{s}_{-i}} \tag{B.20}$$

where one may use the square root of the MSE defined in Equation B.15 at the location of $\hat{y}_i$ as an estimator of $\hat{s}_{-i}$, which is the cross validation standard deviation. Schonlau states that the standardised cross validation errors e_i should lie within the range [-2,2] or, if many measurements are available, within [-3,3]. If this is not the case, the metamodel is not accurate.

Martin and Simpson [83, 84] mention a cross validation R^2-value based on the PRediction Error Sum of Squares (PRESS) statistic known in Response Surface Methodology for determining influential result points. Recall from Equation A.19 that the R^2-value is defined as:

$$R^2 = 1 - \frac{\text{SSE}}{\text{SST}} \tag{B.21}$$

Now, replace SSE by PRESS. The PRESS residual equals the Sum of Squares cross validation Errors defined in Equation B.18 [90]. The result is the predicted R^2_{pred}-value:

$$R^2_{\text{pred}} = 1 - \frac{\text{PRESS}}{\text{SST}} \tag{B.22}$$

Analogously to the adjusted R^2-value introduced for RSM in Section A.3, an adjusted predicted R^2-value is defined as [83, 84]:

$$R^2_{\text{pred,adj}} = 1 - \frac{n-1}{n-q}(1 - R^2_{\text{pred}}) \tag{B.23}$$

in which n is the number of measurements and q is the number of Degrees Of Freedom present in a Kriging metamodel. Recall that the unknown parameters are the regression coefficients $\boldsymbol{\beta}$, the process variance σ_z^2 and $\boldsymbol{\vartheta}$. Consequently q equals the number of $\beta + 1$ (for σ_z^2) + the number of $\vartheta + 1$ [83]. Both the R^2_{pred} and $R^2_{\text{pred,adj}}$ can be used to interpret the accuracy of the DACE metamodel, although R^2_{pred} is found to be more reliable [84].

B.3.2 Testing the assumptions

In Section B.2, three assumptions underlying DACE were indicated. The only one that can be tested, is if the the Gaussian stochastic process assumption is satisfied.

The Gaussian stochastic assumptions were formulated in Section B.2 as follows:

1. The mean of the Gaussian stochastic process $Z(x)$ is 0 and constant: $E(Z(x)) = 0$

2. The stochastic variables determined by $Z(x)$ have a multivariate normal distribution

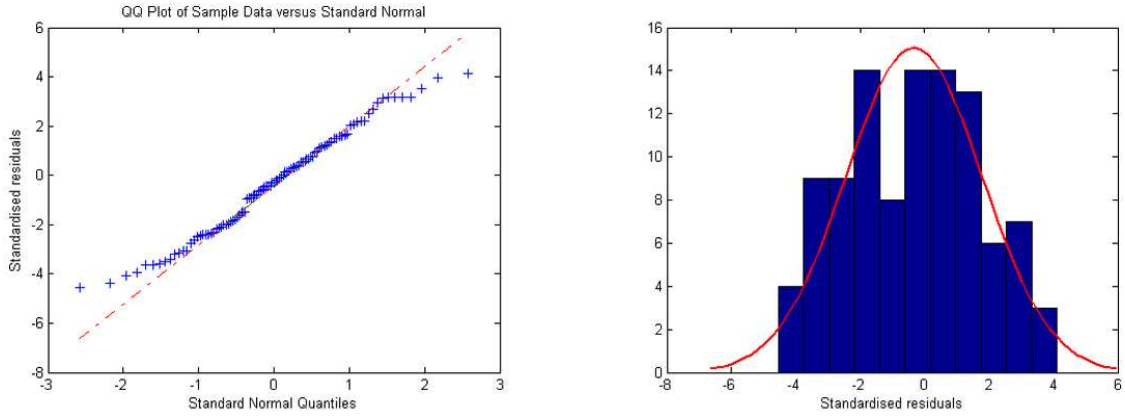


Figure B.4: Q-Q plot and histogram of the standardised cross validated residuals

3. The variance of the Gaussian stochastic process $Z(x)$ is bounded and constant: $\text{var}\left(Z(x_i), Z(x_j)\right) = \sigma_z^2 \mathbf{I}$

For validating the underlying assumptions of RSM, the above demands were investigated by visualising residual plots (see Figure A.2 of Section A.3). Residual plots are, however, not meaningful in case of interpolation, since the errors at the measurement points are 0 by definition.

One could, however, use the (standardised) cross validation errors introduced in the previous section instead of the actual errors at the measurement points. The standardised cross validation error plot proposed by Schonlau [117] can be used to test if the mean of the cross validated errors (and thus, of the Gaussian stochastic process) is indeed 0. Any trend in this standardised cross validation error plot as e.g. shown in Figures A.2(c) and (d) indicates a non-constant variance and a poor fit, respectively. A Gaussian stochastic process would present a standardised cross validation error plot similar to the plot shown in Figure A.2(b).

The final assumption to test is if the stochastic variables have the normal distribution. For this, Schonlau [117] proposes to visualise the standardised cross validation error presented in Equation B.20 versus the quantiles of the standard normal distribution, a so-called Q-Q plot. An example of such a Q-Q plot is similar to the normal probability plot presented in Figure A.3 of Section A.3 and is shown in Figure B.4. A straight line $x = y$ denotes the distribution of errors is normal, which indicates the normality assumption is met as can also be seen from the histogram presented in Figure B.4.

Another way of validation of the normality assumption can be found in Martin and Simpson [83], who use the Shapiro-Wilk's Test for Normality to investigate if a sampled distribution is Gaussian.

The combination of investigating standardised cross validation error plot e_i versus the cross validated predictions for trends and obtaining an idea of normality from the Q-Q plot are considered to be sufficient to conclude whether the measurements are likely to result from a Gaussian stochastic process or not.

B.4 Design Of Experiments (DOE) for DACE

In Section A.4, detailed attention was given to Design Of Experiments (DOE) strategies for RSM. Now, we are dealing with interpolating DACE metamodels, which require different desirable properties and, as a consequence, other DOE strategies. First the desirable properties of experimental design used for DACE metamodels are described. In the Sections B.4.2 and B.4.3, we will make a distinction between several specific DOE strategies for DACE that are purely based on geometrical criteria and those that are based on certain statistical criteria, respectively.

B.4.1 Desirable properties of DOE strategies for DACE

For RSM, a good experimental design should (see Section A.4):

- result in a good fit of the response
- test for Lack Of Fit and estimate the Pure Error
- be cost-effective

Since computer experiments are deterministic, the second property is not relevant in case of DACE. Furthermore, concepts like blocking, replication and randomisation mentioned in Section A.4 are also not meaningful. Thus, to prevent us from performing useless expensive computer experiments, DOE strategies for DACE should not have replicate runs.

The two other properties obviously stay relevant. Clearly, a good representation of the actual response is desirable. Note that for RSM, the shape of the response was known (a lower order polynomial) except for the exact values of $\boldsymbol{\beta}$. For RSM, we strived for a variance optimal (orthogonal) design with respect to the regression coefficients (for first order metamodels) or a rotatable DOE strategy, which was optimal w.r.t. to prediction variance, for second order metamodels. In addition, good DOE strategies contained DOE points at the boundaries of the design space. For DACE, these concepts are no longer applicable because:

1. Kriging functions are extremely flexible. Therefore, no assumption can be made for the final shape of the metamodel and it is important to gather information on the investigated phenomenon throughout the entire design space

2. The random error term forcing the design points to move towards the boundary of the design space is not present for deterministic computer experiments. The absence of the random error implies that, if an error is present, this is a bias error as introduced in Section A.4

Noticing these facts advocates the choice of a DOE strategy for which the experimental design points are evenly spread over the (interior of the) entire design space [46,114]. Such a design is called a *spacefilling* design.

The last desirable property, cost-effectiveness, is a controversial demand in case of Kriging. Of course, it is necessary to keep the number of expensive computer simulations limited, but, unfortunately, the large flexibility of Kriging metamodels generally demands significantly more measurement data to fit a good metamodel. Despite this fact, measures may be taken to minimise the number of expensive computer experiments. On the one hand, this can be accomplished by minding that no useless replicate points are present within the DOE strategy. On the other hand, use can be made of sequential experimentation, just as was the case with RSM. Applying such a sequential strategy, one starts with an initial number of simulations. Schonlau [117] reports that specialists advice to start with 10 experiments per design variable, which demonstrates the fairly large number of measurements used for fitting a good DACE metamodel. Subsequently, the metamodel is fitted and validated. If the quality of the metamodel is not satisfying, one or more design points are added, the metamodel is fitted and validated again, and so on until some quality criterion is met. This addition of DOE points is mostly done cleverly by some sort of an *expected metamodel improvement* measure (see e.g. [40,65,74,114,117,138]).

B.4.2 Geometry based DOE strategies

The first group of DOE strategies for DACE are based on geometrical criteria only. Noticing that the shape of the metamodel is not known in case of DACE, we will only take into account the shape of the design space and place DOE points in this design space in a spacefilling way.

The most straight-forward method for doing this is to fill the design space randomly with DOE points [114]. *Pseudo-Monte Carlo sampling* occurs by using a random number generator, which attempts to approximate a truly random natural process [46]. The term "Monte Carlo" is a reference to the famous casino in the city of Monte Carlo, which lies in the city state of Monaco, southern Europe. Its use of randomness and the repetitive nature of the process are analogous to the activities conducted at a casino [143].

A modification of pseudo-Monte Carlo sampling is stratified Monte Carlo sampling or stratified random number generation [46, 114]. Stratification finds its origin in the way ancient civilisations constructed their city roads or "strata". The roads were built perpendicular with respect to each other, thereby dividing the city into rectangular areas. As an illustration of this phenomenon, Figure B.5(a) contains a map of the ancient town of Herculaneum, near Naples, Italy; that is, before it was destroyed by the eruption of the volcano Vesuvius in the year 79 A.D. The concept of stratification of roads is still very popular today, as illustrated by the city map of New York City in Figure B.5(b).

Returning to the Design Of Experiments strategies for DACE, stratified random sampling comprises dividing the design space in several strata and picking one DOE point from each stratum randomly. An advantage of stratified random sampling above ordinary Pseudo-Monte Carlo sampling is that the DOE points are more uniformly distributed among each dimension [46]. Figures B.6(a) and (b) present 2 factor Pseudo-Monte Carlo sampling and stratified random sampling, respectively.

As can be seen from Figure B.6(b), stratified random sampling comprises n^k expensive computer experiments, where n is the number of strata and k the number of factors (design variables). Both the relatively large number of experiments to be performed and

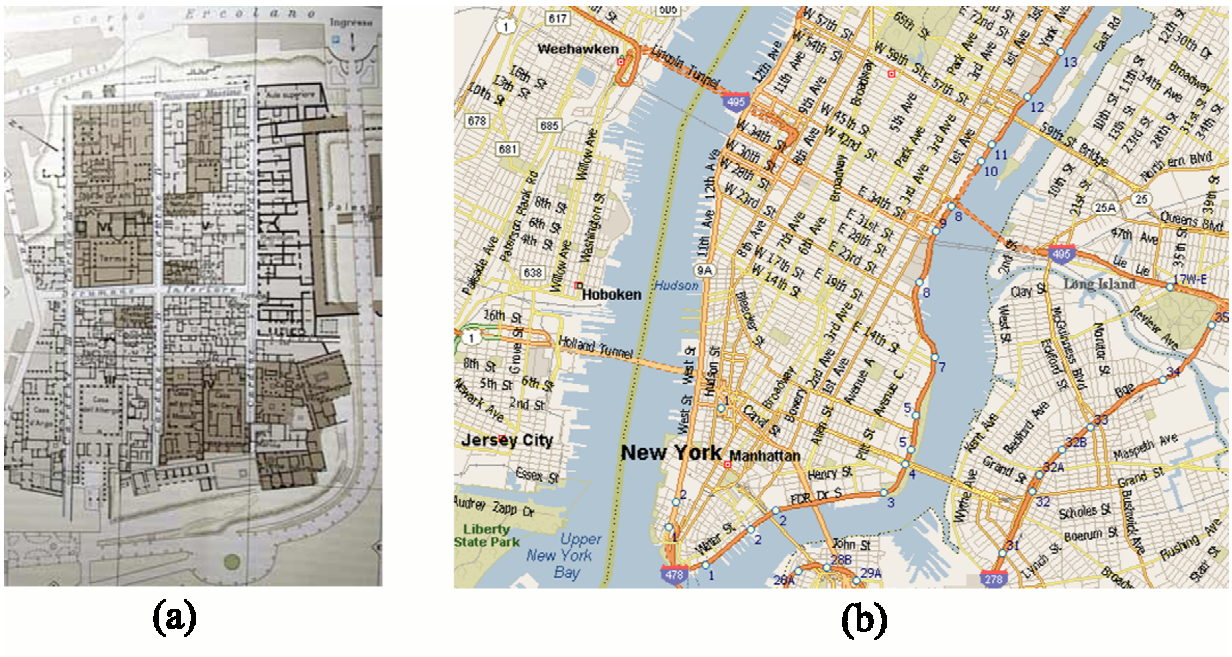


Figure B.5: (a) City map of the ancient town of Herculaneum presenting the concept of stratification; (b) City map and impression of New York City demonstrating the concept of stratification

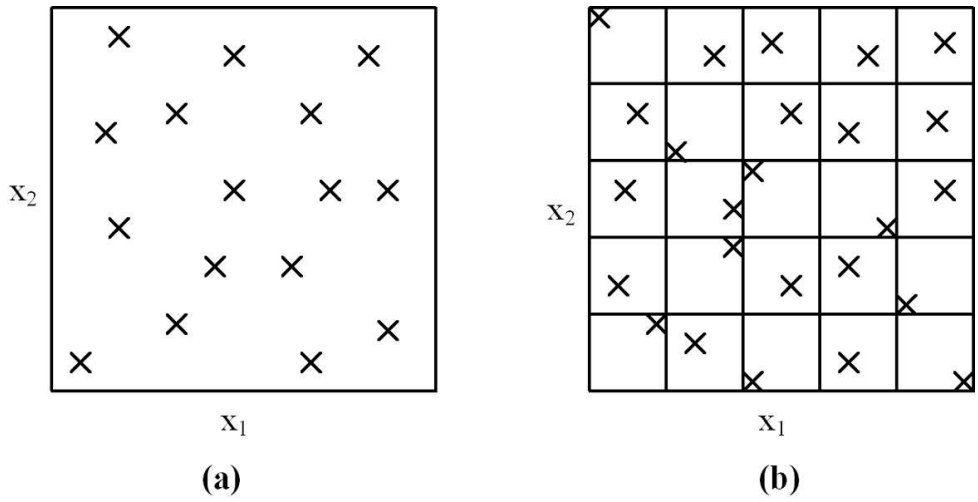


Figure B.6: (a) 2 factor Pseudo-Monte Carlo or random sampling; (b) 2 factor stratified random sampling

the lack of freedom to be able to select any arbitrary number of simulations (this number always equals a power of n) are disadvantages of stratified random sampling. In an attempt to overcome these disadvantages, McKay et al. [86] proposed Latin Hypercube Sampling (LHS) or Latin Hypercubes Designs (LHD). The concept equals stratified random sampling, except that only one DOE point occurs per stratum in each dimension. LHS combines the advantage of being uniformly distributed among each dimension with overcoming both disadvantages of stratified random sampling mentioned above. Further-

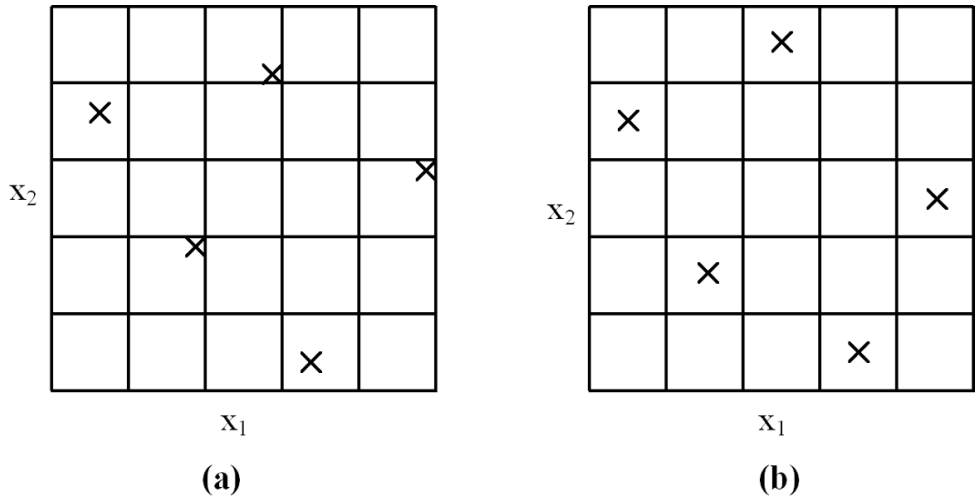


Figure B.7: (a) 2 factor Latin Hypercubes Design; (b) 2 factor lattice design

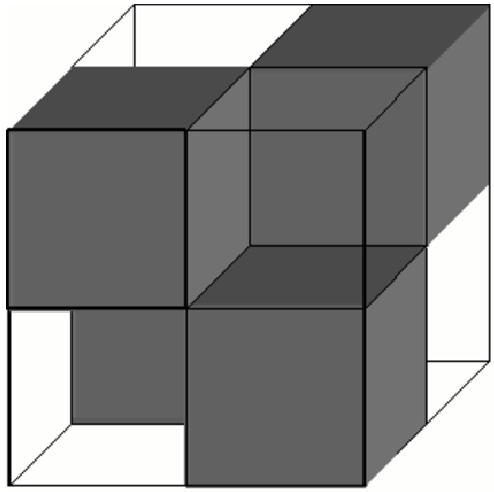


Figure B.8: Orthogonal Array with good projection properties in two dimensions

more, LHS provides excellent so-called projection properties. Suppose one of two design variables appears to be not significant, then all measurements in that dimension can be skipped. Using LHS, the expensive simulations that have already been performed by then, can be projected onto the other dimension resulting in a uniform distribution in this other dimension. Good projection properties are especially useful when it is not yet known whether all design variables are significant, i.e. LHS can well be used for screening purposes.

A specific form of LHS designs are *lattice* designs, for which the DOE points are not selected randomly from a stratum, but the DOE point is simply chosen to be the centre of the stratum [46, 69]. Figure B.7(a) presents a 2 factor LHS design, whereas Figure B.7(b) shows a lattice design.

Another LHS-like DOE strategy for DACE are Orthogonal Arrays (OA). It was indicated before that LHS has good projection properties w.r.t. any 1D dimension. *Orthogonal Arrays* have good projection properties onto a more dimensional subspace of the design space [46]. For example the DOE

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{bmatrix} \tag{B.24}$$

is an OA with uniform projection properties in two dimensions. Figure B.8 shows the visualisation of A. There is one DOE point in each shaded block. Notice that an OA differs from an orthogonal design presented in Section A.4.1. The Orthogonal Array A does not satisfy the critical property for orthogonal experimental designs mentioned in Equation A.34.

Figure B.9: (a) A pseudo-car; (b) A quasi-car

The above DOE strategies (stratified) Pseudo-Monte Carlo sampling, LHS, lattice and OA, are all based on sampling strategies to fill a given design space. A second intuitive way of spreading design points spacefilling over a feasible domain is by *distance based* geometrical criteria. Two methods are the so-called maximin and minimax designs. A *maximin* design maximises the minimum distance between two design points, ensuring that no two design points are too close to each other. On the other hand, a *minimax* design minimises the maximum distance between two DOE points making sure that no gaps exist in the experimental design. A more general way of interpreting the maximin and minimax distance based criteria is to maximise (or minimise) the average distance between all DOE points [114].

The property of uniformly distributing the design points over the design space is an important one, which is reflected by a third type of DOE strategies for DACE. These designs are referred to as *uniform designs* [114] or *Quasi-Monte Carlo sampling* [46]. It is emphasised that Quasi-Monte Carlo methods are not the same as (Pseudo-) Monte Carlo methods introduced above. The term "pseudo" reflects that Pseudo-Monte Carlo methods deceitfully pretend to generate truly random numbers, whereas the term "quasi" means that Quasi-Monte Carlo methods have some resemblance with, but are not entirely, a Monte Carlo sampling method. To make a link to mechanical engineering, Figure B.9(a) shows a pseudo-car: a vehicle that deceitfully pretends to be a car, but is actually a motorised bicycle. Figure B.9(b) on the other hand shows a quasi-car, an old Lada: it resembles a car, comes close, but is in the modest opinion of the author not quite it yet.

Anyway, Quasi-Monte Carlo DOE strategies attempt to deviate minimally from a uniform distribution of the experimental design points [46]. This deviation is called the *discrepancy*.

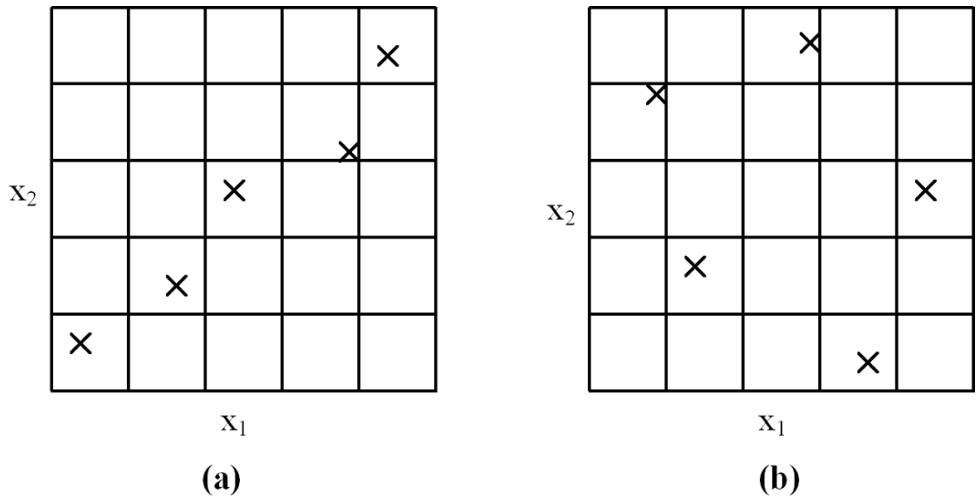


Figure B.10: (a) An arbitrary Latin Hypercubes design; (b) A spacefilling maximin Latin Hypercubes design

Quasi-Monte Carlo methods result in DOE points that are nicely uniformly spread over the design space. *Hammersley sampling* is a special case of Quasi-Monte Carlo sampling using Hammersley sequences [46].

All the above methods for sampling DOE strategies, each with its own advantages and disadvantages, are useful for DACE. However, Santner et al. [114] point out that it may be beneficial to not select just one of these methods, but to choose a combination of several methods. To illustrate this, let us take a look at the popular method Latin Hypercube Sampling (LHS). Adopting only LHS, one could end up with the experimental design presented in Figure B.10(a). The DOE points lie about the diagonal of the design space, which is NOT a spacefilling design. However, combining LHS with a maximin criterion will result in a *spacefilling Latin Hypercubes design* as shown in Figure B.10(b). In fact, Santner et al. recommend the latter design as a very powerful DOE strategy for DACE [114].

B.4.3 DOE strategies based on statistical criteria

In Section B.4.2, several DOE strategies for DACE were treated that assumed nothing is known about the metamodel that is fitted. Use was solely made of geometrical criteria for obtaining a spacefilling design. However, note that DOE strategies for RSM described in Section A.4 are always based on some statistical criterion, whereas it is necessary to assume the shape of the metamodel to be fitted (a lower order polynomial): minimising the variance of the regression parameters led to an orthogonal DOE, minimising the prediction variance resulted in rotatable designs. Just as was the case for RSM, we may be willing to assume some features of the metamodel, which allows us to derive DOE strategies for

DACE that are based on some statistical criterion. This section introduces several of these strategies without attempting to go into detail.

The first DOE strategy is a *maximum entropy design* and is based on the so-called *entropy* [69, 114]:

$$H(X) = -\int_{\mathcal{X}} f(x)\ln(f(x))\mathrm{d}x \tag{B.25}$$

If the stochastic variable X with distribution $\mathcal{X}$ is random, no information is available and the entropy takes its maximum value, whereas the entropy is minimal if X is deterministic. If we assume $\boldsymbol{\vartheta}$ to be the vector with unknown regression parameters, i.e. $\boldsymbol{\beta}$ in case of RSM, $\boldsymbol{\beta}$ and $\boldsymbol{\vartheta}$ in case of DACE, then the procedure is to link the distribution of $\boldsymbol{\vartheta}$ to the amount of information provided by an experimental design [114]. Typically, the information I is viewed as the negative of the entropy:

$$I = -H = \int_{\vartheta} \ln(\vartheta)\mathrm{d}\vartheta \tag{B.26}$$

A certain DOE $\mathcal{D}$ provides $I_{\mathcal{D}}$ information:

$$I_{\mathcal{D}} = \int \vartheta\ln(\vartheta)\mathrm{d}\vartheta \tag{B.27}$$

Thus, the change in information is $I - I_{\mathcal{D}}$. The best design is the one that minimises this change in information, i.e. maximises the entropy. Hence the name maximum entropy design.

Assuming a Gaussian stochastic process for the measurement data, it can be shown that the entropy reduces to [114]:

$$\det\left(\sigma_z^2\mathbf{R}\right) \tag{B.28}$$

where σ_z^2 is the process variance and $\mathbf{R}$ is the correlation matrix between the measurement data.

Next to maximum entropy designs, another group of DOEs for DACE based on a statistical criterion are Integrated Mean Squared Error (IMSE) and the Maximum Mean Squared Error (MMSE) designs. Recall that, assuming a Gaussian stochastic process, the Mean Squared Error can be predicted by Equation B.15. Suppose we are considering a new experimental design $\mathcal{D}$, then we can evaluate Equation B.15 at all these design points over the design space $\mathcal{X}$. Sacks et al. [111, 112] have proposed that the best design is the DOE that minimises the average, i.e. integrated, MSE [114]:

$$\mathrm{IMSE} = \int_{\mathcal{X}} \frac{\hat{\mathbf{Y}}_0(\mathbf{x})}{\sigma_z^2}\mathrm{d}\mathbf{x} \tag{B.29}$$

where $\hat{\mathbf{Y}}_0(\mathbf{x})$ are the predicted values at the design points $\mathbf{x}$. This is called the *Integrated Mean Squared Error (IMSE)* criterion and can be shown to be a generalisation of the A-optimal computer generated design for RSM introduced in Section A.4.4 [114].

Analogously, one could also minimise the Maximum MSE (MMSE) instead of the integrated MSE [114]:

$$\text{MMSE} = \max \frac{\hat{\mathbf{Y}}_0(\mathbf{x})}{\sigma_z^2} \tag{B.30}$$

This *Maximum Mean Squared Error (MMSE)* criterion is a generalisation of the G-optimal computer generated design for RSM, which was also treated in Section A.4.4.

Note that it is necessary for both the IMSE and MMSE criteria to have already fitted a metamodel before Equation B.15 can be evaluated. Thus, the procedure to use these criteria is to start with an initial set of experimental design points, for example generated by a Latin Hypercubes design, and then to improve the DOE by selecting additional points by the IMSE or MMSE criterion.

This sequential procedure is continued until the DACE metamodel fits "reality". However, in many cases the experimenter is not interested in a metamodel that is accurate over the entire design space, but solely in optimising the problem entity. For this, it is only necessary to construct an accurate metamodel in the vicinity of the optimum. These DOE strategies zoom in near the optimum using certain optimisation criteria. For the overview, it is sufficient to mention that several *optimisation criteria based DOE strategies* are encountered in literature, see e.g. [114].

Appendix C

Analytical test functions

This appendix applies the optimisation algorithm and its sequential improvement strategies proposed in Chapter 4 to two analytical test functions. The functions used are the Camelback and Branin functions and are treated in the Sections C.1 and C.2, respectively. Section C.3 summarises some conclusions and some rules of thumb with respect to the use of sequential improvement strategies.

C.1 Camelback

The Camelback function is given by:

$$\begin{aligned} f(x_1, x_2) = 4x_1^2 - 2.1x_1^4 \quad + \quad & \frac{1}{3}x_1^6 + x_1x_2 - 4x_2^2 + 4x_2^4 \\ -5 \leq \quad x_1, x_2 \quad & \leq 5 \end{aligned} \tag{C.1}$$

Figure C.1 shows the Camelback function. It has 2 global minima and 2 local minima, which are presented in Table C.1. The function provides a good test to observe whether the optimisation algorithm is able to obtain (one of) the global optima.

Let us now apply the metamodel based optimisation algorithm to optimise the Camelback function. The optimisation is initialised by running 10 times the number of design variables,

x_1	x_2	f
0.0899	-0.7127	-1.0316
-0.0899	0.7127	-1.0316
1.7036	-0.7961	-0.2155
-1.7036	0.7961	-0.2155

Table C.1: Minima of the Camelback function

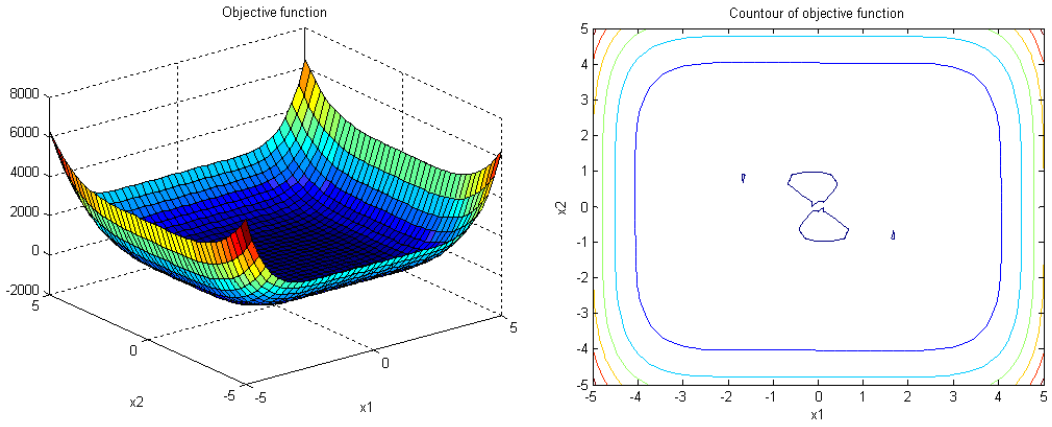


Figure C.1: The Camelback function

i.e. 20 calculations and fitting the metamodels. Subsequently, the metamodels are improved sequentially by the four methods introduced in Section 4.2.6:

- In batches of 20 calculations without zoom
- In batches of 20 calculations with Zoom and Resampling according to $f - w\text{RMSE} \leq f^* + w\text{RMSE}^*$ (Z+R)
- By adding the Minima of the Merit Function $f - w\text{RMSE}$ (MMF). The number of calculations is determined by the sequential improvement strategy. The calculations may be performed in parallel
- By adding the Maxima of the Expected Improvement defined in Equation 4.25 (MEI). The number of calculations is determined by the sequential improvement strategy. The calculations may be performed in parallel

For the second and third method, it is necessary to choose a weight factor w. A value of zero will omit the contribution of the RMSE and the new points will be chosen based on the shape of the metamodel of the objective function solely. An increasing value of w will put more weight on the RMSE part of the merit function. The latter generally implies exploring unexplored regions since the RMSE at a certain location will increase with increasing distance to locations where calculations were already performed.

Literature presents some guidelines for suitable values of w. Torczon and Trosset [136] leave the choice of w open to the user. Jones [58] emphasises that the RMSE is merely an estimate of the error, which can be deceiving. In this case, certain areas of the design space are likely to be excluded from the search. Keeping this in mind, one should not assume a too small value for w. In [34], Emmerich et al. use a value of 1 for fast convergence. The same authors avoid a clear choice in [35] where both $w = 0$ and $w = 2.5$

are used. Büche [17, 18] states that a value higher than 4 diminishes convergence speed and sets w to 0, 1, 2 and 4 while maximally exploiting the availability of parallel processors.

Since these guidelines are relatively poor, a small additional study to an acceptable value of w is performed. If the analytical function is known, one can obtain an estimate of w by plotting

$$\hat{w} = \frac{\hat{f} - f}{\text{RMSE}} \tag{C.2}$$

where $\hat{f} - f$ is the real error between the metamodel of the objective function $\hat{f}$ and the real function value f. $\hat{w}$ provides an indication of how many standard deviations the metamodel deviates from the real value of the objective function. The standard deviation is RMSE, which is the error estimated by RSM or Kriging, depending on which method is used. Figure C.2 presents such a plot for the Camelback function after the initial 20 calculations have been performed. One can see that the estimate of w does not exceed the value of 4 and at only few locations, a value of 3 is exceeded. As a matter of fact, a more extensive study for more analytical test functions reveals that, most of the time, a value of $\hat{w} = 3$ is seldom exceeded. Hence, it is decided to limit the values of w to the range between 0 and 3.

Eight optimisations were carried out: sequential improvement without zooming, two times three optimisations using Z+R and MMF and one optimisation using MEI. For Z+R and MMF, values of w of 1, 2 and 3 were taken into account. $w = 0$ is included automatically in each sequential improvement strategy since the approximate optimum is evaluated always at the end of a batch of calculations.

The results of the optimisation of the Camelback function for the four sequential improve-

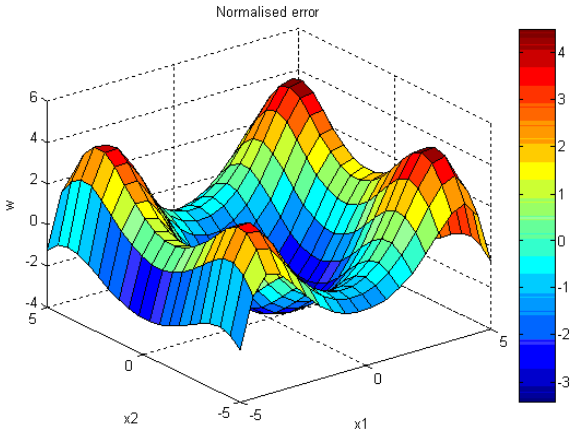


Figure C.2: Estimated weight factors of the Camelback function after 20 calculations

Method	# *calculations*	# *batches*	x_1	x_2	f
No zoom	80	4	0.0935	-0.7002	-1.0303
Z+R, w = 1	80	4	0.0871	-0.7186	-1.0313
			1.6667	-0.7855	-0.2025
Z+R, w = 2	100	5	0.0875	-0.7110	-1.0316
			-0.0718	0.7149	-1.0303
Z+R, w = 3	120	6	0.0774	-0.6715	-1.0184
MMF, w = 1	57	6	-0.0874	0.7213	-1.0310
			0.0878	-0.7247	-1.0304
MMF, w = 2	55	5	0.1022	-0.7117	-1.0310
			-0.0891	0.7272	-1.0298
MMF, w = 3	63	6	-0.0960	0.7176	-1.0313
			0.0976	-0.7004	-1.0301
MEI	43	4	0.1153	-0.7133	-1.0291
			-0.1243	0.7199	-1.0268

Table C.2: Results of the optimisation of the Camelback function

ment strategies are presented in Table C.2.

A surprising result is that Zooming and Resampling (Z+R) does not yield the increase in efficiency that would be expected from a method that focusses in a smaller region in the vicinity of an optimum. A possible explanation is that calculation results that lie outside the zoomed in design space are omitted. Hence, expensive and valuable information is discarded. Additional calculations are needed for replacing them, which in the end results in a higher number of function evaluations. For $w = 1$, this increase in calculations appears to be compensated by the decrease in the size of the design space. A negative side effect of this strong focussing is that there is a risk of getting trapped in a local optimum, as can also be seen in Table C.2. Choosing a value of 2 for w results in both global optima, however at higher costs than fitting an accurate metamodel without zooming. The same is true in an even stronger sense for $w = 3$.

Sequential improvement using MMF is much more efficient than no zooming. Regarding choosing the weight factor w, a more focussed search around the optimum is performed when $w = 1$ and a more global search when $w = 3$. This implies that one would expect more function evaluations for higher values of w. Table C.2 presents that this appears to be the case for the Camelback function for $w = 3$. If, on the other hand, it is not clear yet in which region the global optimum will be, choosing a higher weight factor will prevent zooming in near a global optimum and is hence more robust. In the Camelback case, however, the algorithm has focussed on the right region also with lower weight factor values and a small w appears to be both accurate and efficient.

The most efficient method for optimising the Camelback function is sequential improvement using MEI. The algorithm only needs 43 function evaluations divided over 4 batches only. It finds both global optima, although Table C.2 shows that the results are a little less accurate than for most other sequential improvement strategies.

C.2 Branin

The second analytical test function under consideration is the Branin function:

$$\begin{aligned} f(x_1, x_2) = \left(x_2 - \frac{5}{4\pi^2}x_1^2 + \frac{5}{\pi}x_1 - 6\right)^2 \quad &+ \quad 10\left(1\frac{1}{8\pi}\right)\cos(x_1) + 10 \\ -5 \leq \quad x_1 \quad &\leq 10 \\ 0 \leq \quad x_2 \quad &\leq 15 \end{aligned} \tag{C.3}$$

Figure C.3 shows the Branin function. It has three global minima at $(-\pi, 12.275)$, $(\pi, 2.275)$ and $(9.424, 2.275)$. The objective function value is at these points -0.3979.

The procedure of optimising the Branin function is equal to the way described in the previous section with one exception: the Zooming and Resampling method (Z+R) was not taken into account anymore because of the bad results this sequential improvement strategy showed for the Camelback function. Thus, it was started to construct an initial metamodel from 20 initial function evaluations and subsequently five optimisation runs were performed with several sequential improvement strategies: one without zoom, three using the MMF strategy with weight factors $w = 1, 2$ and 3 and one using the method of

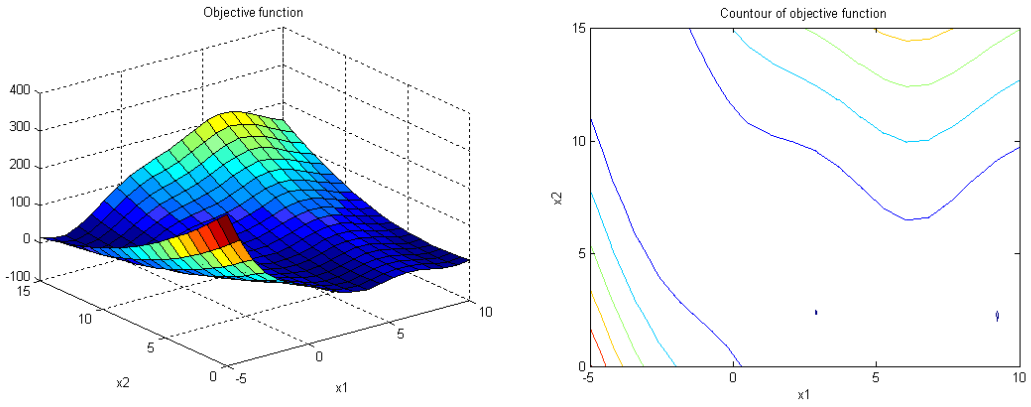


Figure C.3: The Branin function

Method	*# calculations*	*# batches*	x_1	x_2	f
No zoom	40	2	3.1293	2.2940	-0.3959
MMF, w = 1	30	3	3.1378	2.2360	-0.3975
			9.4222	2.2627	-0.3976
			-3 .1283	12.2108	-0.3969
MMF, w = 2	30	3	9.4218	2.2567	-0.3978
			-3.1376	12.2508	-0.3977
			3.1420	2.2511	-0.3979
MMF, w = 3	31	3	9.4186	2.2439	-0.3977
			-3.1420	12.2558	-0.3979
			3.1433	2.2480	-0.3979
MEI	28	4	9.4232	2.2692	-0.3975
			3.1174	2.2966	-0.3941
			-3.1551	12.2666	-0.3967

Table C.3: Results of the optimisation of the Branin function

MEI.

The results are summarised in Table C.3. The algorithm finds all global optima without trouble, regardless which sequential improvement strategy was used. The MMF method is again more efficient than the sequential improvement strategy without zoom. As was the case for the Camelback function, the accuracy of the results obtained using the MMF strategy are relatively insensitive to the value selected for w. A value of 3 uses one more calculation. Using only 28 calculations, MEI again outperformed the other methods on the field of efficiency. The accuracy, however, was slightly lower, as was also the case for the Camelback function.

C.3 Conclusions

Summarising the short investigation of several sequential improvement strategies using two analytical test functions, the following conclusions can be drawn:

1. The Zooming and Resampling (Z+R) strategy is less efficient than sequential improvement without zooming except for very low values of the weight factor w
2. The Z+R strategy tends to get stuck in local optima for low values of w
3. The Minimising a Merit Function (MMF) strategy is more efficient than sequential improvement without zooming
4. The accuracy of the MMF strategy seems to be insensitive to the value of the weight factor w

5. If the initial metamodel is able to locate the region of the global optimum, a low value of the weight factor w is the most efficient

6. MEI showed to be the most efficient sequential improvement strategy, although it approximated the optima slightly less accurately than the other methods

These conclusions lead to the following rules of thumb:

1. Fit an initial metamodel from about 10 times the number of design variables as initial points

2. Use the MEI or MMF method as sequential improvement strategy

3. When using MMF, set the value of the weight function to 1

The question is whether these recommendations also hold for the case when the initial metamodel indicates the wrong region for the global optimum, when constraints are present and when more than two design variables are considered.

If the algorithm zooms in near a local optimum, it will explore this region first. After it has unexpectedly found relatively high values of the objective function in this region, it will shift to other regions where the objective function is also predicted to be relatively low and where a high RMSE value reveals that the region is not explored fully yet. This counts for both MMF and MEI. Using high values of w for MMF will cause the algorithm to shift earlier to other regions, whereas low values imply that many calculations will be performed near the local optimum. MEI is also known to totally explore a region first before shifting to another promising region [58]. Thus, using MEI and MMF with $w = 1$, the algorithm will not be that efficient anymore, but it is highly likely that the global optimum will be retrieved. For MEI, this is even guaranteed [117]. In the case where it is not sure where the global optimum will be, it may be beneficial to choose MMF with a higher value of w. This will result in batches of more calculations each, but it will explore more regions at the same time. Hence, more parallel processors are welcome in this case.

In the presence of constraints, no problems are foreseen since the same multistart SQP optimisation algorithm that also optimises the metamodels is applied for both MMF and MEI. This algorithm takes into account explicit and implicit constraints. Hence, it is at a certain batch not possible to end up with a new DOE point that is infeasible at the previous batch. If the merit function is constrained by an implicit constraint, the new DOE points will be selected at this implicit constraint, which will increase the accuracy of the implicit constraint's metamodel.

In more than two dimensions, it becomes more and more difficult to fit an accurate metamodel. Thus, sequential improvement without zooming becomes prohibitively expensive. In this situation, the usefulness of zooming by Minimising a Merit Function or Maximum

Expected Improvement increases, since they tend to obtain optimal or near-optimal results very efficiently. However, also MMF and MEI depend on the accuracy of the initial and subsequent metamodels and the accuracy of the obtained optima decreases with decreasing metamodel accuracy. In more than two dimensions, it is expected that MMF and MEI will deliver significantly improved results very quickly. Guarantees that the obtained improvement is indeed the global optimum decreases with increasing number of design variables.

Bibliography

[1] Abedrabbo, N., Zafar, N., Averill, R., Pourboghrat, F., and Sidhu, R. Optimization of a tube hydroforming process. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[2] António, C., Castro, C., and Sousa, L. Optimization of metal forming processes. *Computers and Structures 82* (2004), 1425–1433.

[3] Aydemir, A., de Vree, J., Brekelmans, W., Geers, M., Sillekens, W., and Werkhoven, R. An adaptive simulation approach design for tube hydroforming processes. *Journal of Materials Processing Technology 159* (2005), 303–310.

[4] Bäck, T., and Schwefel, H. An overview of evolutionary algorithms for parameters optimization. *Evolutionary Computation 1* (1993), 1–23.

[5] Bahloul, R., Santo, P. D., and Potiron, A. Shape optimisation of automotive security parts regarding the influence of residual stresses and material damage. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[6] Becker, M. *Anwendung von höheren Optimierungsmethoden in der Umformtechnik*. PhD thesis, RWTH Aachen, 1991. (In German).

[7] Ben Ayed, L., Delamézière, A., Batoz, J., and Knopf-Lenoir, C. Optimization and control of the blankholder force in sheet metal stamping with application to deep drawing of a cylindrical cup. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[8] Ben Ayed, L., Delamézière, A., Batoz, J., and Knopf-Lenoir, C. Optimization of the blankholder force with application to the numisheet'02 deep drawing benchmark test b1. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[9] Ben Ayed, L., Delamézière, A., Batoz, J., and Knopf-Lenoir, C. Optimization of the blankholder force distribution in deep drawing. In *Proceedings of APOMAT* (Morschach, Switzerland, 2005).

[10] Berry, W. *Understanding Regression Assumptions*. Sage Publications, Inc., Newbury Park CA, USA, 1993.

[11] Berry, W., and Feldman, S. *Multiple Regression in Practice*. Sage Publications, Inc., Newbury Park CA, USA, 1985.

[12] Bonte, M. H. A. Optimisation: An introduction. Internal report P04.1.047, Netherlands Institute for Metals Research (NIMR), Delft, Netherlands, 2004.

[13] Bonte, M. H. A., van den Boogaard, A. H., and Huétink, J. Metamodelling techniques for the optimisation of metal forming processes. In *Proceedings of ESAFORM* (Cluj-Napoca, Romania, 2005), pp. 155–158.

[14] Bonte, M. H. A., van den Boogaard, A. H., and Huétink, J. Solving optimisation problems in metal forming using finite element simulation and metamodelling techniques. In *Proceedings of APOMAT* (Morschach, Switzerland, 2005), pp. 242–251.

[15] Breitkopf, P., Naceur, H., Rassineux, A., and Villon, P. Optimizing metal forming processes using response surface approximation and inverse approach surrogate model. In *Proceedings of ESAFORM* (Trondheim, Norway, 2004).

[16] Breitkopf, P., Naceur, H., Rassineux, A., and Villon, P. Moving least squares response surface approximation: Formulation and metal forming applications. *Computers and Structures* (submitted).

[17] Büche, D. *Multi-objective evolutionary optimization of gas turbine components*. PhD thesis, ETH Zürich, Zürich, Switzerland, 2004.

[18] Büche, D., Schraudolph, N., and Koumoutsakos, P. Accelerating evolutionary algorithms with gaussian process fitness function models. *IEEE Transactions on Systems, Man, and Cybernetics* (submitted).

[19] Byon, S., and Hwang, S. FEM-based process optimal design in steady-state metal forming considering strain-hardening. *Computers and Structures 79*, 14 (2001), 1363–1375.

[20] CADOE-ANSYS, Inc. Mechanical design synthesis: New processes for innovative product development. http://www.cadoe.com/WhitePaper-CADOE.pdf, 2002.

[21] Cao, J., Li, S., Xia, Z., and Tang, S. Analysis of an axisymmetric deep-drawn part forming using reduced forming steps. *Journal of Materials Processing Technology 117*, 1-2 (2001), 193–200.

[22] Carrino, L., Giuliano, G., and Napolitano, G. A posteriori optimisation of the forming pressure in superplastic forming processes by the finite element method. *Finite Elements in Analysis and Design 39* (2003), 1083–1093.

[23] Carrino, L., Giuliano, G., and Palmieri, C. On the optimisation of superplastic forming processes by the finite-element method. *Journal of Materials Processing Technology 143-144*, 1 (2003), 373–377.

[24] Castro, C., António, C., and Sousa, L. Optimisation of shape and process parameters in metal forging using genetic algorithms. *International Journal of Materials Processing Technology 146* (2004), 356–364.

[25] Chong, E., and Zak, S. *An introduction to optimization*. John Wiley and Sons, Inc., New York, USA, 1996. ISBN 0-471-08949-4.

[26] Chung, S., Fourment, L., Chenot, J. L., and Hwang, S. Adjoint state method for shape sensitivity analysis in non-steady forming applications. *International Journal for Numerical Methods in Engineering 57* (2003), 1431–1444.

[27] Clarke, S., Griebsch, J., and Simpson, T. Analysis of support vector regression for approximation of complex engineering analyses. In *Proceedings of the ASME Design Engineering Technical Conferences DETC* (2003).

[28] Craig, K., Stander, N., Dooge, D., and Varadappa, S. Automotive crashworthiness design using response surface-based variable screening and optimization. *Engineering Computations 22* (2005), 38–61.

[29] Debray, K., Sun, Z., Radjai, R., Guo, Y., Dai, L., and Gu, Y. Optimum design of addendum surfaces in sheet metal forming process. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[30] Del Vecchio, R. *Understanding Design Of Experiments*. Hanser Verlag, München, Germany, 1997. ISBN 3-446-18657-3.

[31] Di Lorenzo, R., Filice, L., Umbrello, D., and Micari, F. An integrated approach to the design of tube hydroforming processes: artificial intelligence, numerical analysis and experimental investigation. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[32] Do, T., Fourment, L., and Laroussi, M. Sensitivity analysis and optimization algorithms for 3D forging process design. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[33] Duan, X., and Sheppard, T. Shape optimisation using FEA software: A V-shaped anvil as an example. *Journal of Materials Processing Technology 120*, 1-3 (2002), 426–431.

[34] Emmerich, M., Giotis, A., Özdemir, M., Bäck, T., and Giannakoglou, K. Metamodel-assisted evolution strategies. In *Proceedings of the International Conference on Parallel Problem Solving from Nature* (2002).

[35] Emmerich, M., and Jakumeit, J. Metamodel-assisted optimisation with constraints: A case study in material process design. In *Proceedings of the International Congress on Evolutionary Methods for Design, Optimization an Control with Applications to Industrial Problems EUROGEN* (Barcelona, Spain, 2003).

[36] Endelt, B. *Least Square Optimization Techniques Applied on Sheet Metal Forming.* PhD thesis, Aalborg University, Aalborg, Denmark, 2003.

[37] Endelt, B., and Nielsen, K. Least-square formulation of the object function, applied on hydro mechanical tube forming. In *Proceedings of SheMet* (2001).

[38] Engineering Mechanics Graduate School. Engineering Optimization course. Course readings, 2003.

[39] Fann, K., and Hsiao, P. Optimization of loading conditions for tube hydroforming. *Journal of Materials Processing Technology 140*, 1-3 (2003), 520–524.

[40] Farhang-Mehr, A., and Azarm, S. Bayesian meta-modelling of engineering design simulations: a sequential approach with adaptation to irregularities in the response behaviour. *International Journal for Numerical Methods in Engineering 62* (2005), 2104–2126.

[41] Fourment, L., Do, T., Habbal, A., and Bouzaiane, M. Gradient, non-gradient and hybrid algorithms for optimizing 2D and 3D forging sequences. In *Proceedings of ESAFORM* (Cluj-Napoca, Romania, 2005).

[42] Fourment, L., Do, T., Habbal, A., and Bouzaiane, M. Optimization of forging sequences using gradient, non-gradient and hybrid algorithms. In *Proceedings of APOMAT* (Morschach, Switzerland, 2005).

[43] Fox, J. *Regression Diagnostics.* Sage Publications, Inc., Newbury Park CA, USA, 1991.

[44] Gantar, G., Kuzman, K., and Makarovic, M. An industrial example of FE supported development of a new product. *Journal of Materials Processing Technology 150* (2004), 163–169.

[45] Gantar, G., Pepelnjak, T., and Kuzman, K. Optimization of sheet metal forming processes by the use of numerical simulations. *Journal of Materials Processing Technology 130-131* (2002), 54–59.

[46] Giunta, A., Wojtkiewicz, S., and Eldred, M. Overview of modern design of experiments methods for computational simulations. In 41st *AIAA Aerospace Sciences Meeting and Exhibit* (Reno NV, USA, 2003).

[47] Hair, J., Anderson, R., Tatham, R., and Black, W. *Multivariate Data Analysis: with readings*, 3rd ed. MacMillan Publishing Company, New York NY, USA, 1992. ISBN 0-02-946564-8.

[48] Han, J., Yamazaki, K., and Nishiyama, S. Optimization of the crushing characteristics of triangulated aluminum beverage cans. *Structural and Multidisciplinary Optimization 28* (2004), 47–54.

[49] Institut für Bildsame Formgebung. Computer Aided Optimization Tool (CAOT). http://www.ibf.rwth-aachen.de/Ww/forsch/sim/caot/caot.htm.

[50] Isixsigma. Definition of Design Of Experiments. http://www.isixsigma.com/dictionary/ Design_of_Experiments_-_DOE-41.htm.

[51] Jakumeit, J. Numerical optimization of the Bridgman casting process for stationary gas turbine blades. In *Proceedings of APOMAT* (Morschach, Switzerland, 2005).

[52] Jansson, T. *Optimization of Sheet Metal Forming Processes*. Licentiate thesis, Universitet Linköping, Linköping, Sweden, 2002.

[53] Jansson, T., Andersson, A., and Nilsson, L. Optimization of draw-in for an automotive sheet metal part – an evaluation using surrogate models and response surfaces. *Journal of Materials Processing Technology 159* (2005), 426–234.

[54] Jansson, T., Nilsson, L., and Redhe, M. Using surrogate models and response surfaces in structural optimization - with application to crashworthiness design and sheet metal forming. *Structural and Multidisciplinary Optimization 25*, 2 (2003), 129–140.

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

[56] Jirathearanat, S., and Altan, T. Optimization of loading paths for tube hydroforming. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[57] Johnson, K., Nguyen, B., Dawies, R., Grant, G., and Khaleel, M. A numerical process control method for circular-tube hydroforming prediction. *International Journal of Plasticity 20* (2004), 1111–1137.

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

[59] Jones, D., Schonlau, M., and Welch, W. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization 13* (1998), 455–492.

[60] Karboub, M., Elkholy, A., and Al-Hawaj, O. Modelling deformation of hydroformed circular plates using neural networks. *International Journal of Advanced Manufacturing Technology 20* (2002), 871–882.

[61] Kim, N., Choi, K., and Chen, J. Die shape design optimization of sheet metal stamping process using meshfree method. *International Journal for Numerical Methods in Engineering 51* (2001), 1385–1405.

[62] Kim, Y., Lee, J., and Hong, S. Optimal design of superplastic forming processes. *Journal of Materials Processing Technology 112*, 2-3 (2001), 166–173.

[63] Kleiber, M., Hien, T., Antúnez, H., and Kowalczyk, P. Parameter sensitivity of elastoplastic response. *Engineering Computations 12* (1995), 263–280.

[64] Kleijnen, J., and Sargent, R. A methodology for fitting and validating metamodels in simulation. *European Journal of Operational Research 120* (2000), 14–29.

[65] Kleijnen, J., and van Beers, W. Application-driven sequential designs for simulation experiments: Kriging metamodelling. *European Journal of Operational Research* (submitted).

[66] Kleinermann, J. *Identification paramétrique et optimisation des procédés de mise à forme par problèmes inverses.* PhD thesis, Université de Liège, Liège, Belgium, 2001. In French.

[67] Kleinermann, J., and Ponthot, J. Parameter identification and shape/process optimization in metal forming simulation. *Journal of Materials Processing Technology 139*, 1-3 (2003), 521–526.

[68] Koc, M., Allen, T., Jiratheranat, S., and Altan, T. Use of FEA and design of experiments to establish design guidelines for simple hydroformed parts. *International Journal of Machine Tools and Manufacture 40*, 15 (2000), 2249–2266.

[69] Koehler, J., and Owen, A. *Handbook of Statistics.* Elsevier Science, New York, USA, 1996, ch. Computer Experiments, pp. 261–308.

[70] Kopp, R. Some current development trends in metal-forming technology. *Journal of Materials Processing Technology 60* (1996), 1–9.

[71] Labergère, C., and Gelin, J. Comparison between optimization and control strategies to determine command laws in hydroforming processes. In *Proceedings of ESAFORM* (Trondheim, Norway, 2004).

[72] Labergère, C., and Gelin, J. New strategies for optimal control of command laws for tube hydroforming processes. In *Proceedings of NUMIFORM* (Columbus, OH, USA, 2004).

[73] Labergère, C., Lejeune, A., and Gelin, J. Control of damage in flanges hydroforming. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[74] Lehman, J. *Sequential Design of Computer Experiments for Robust Parameter Design*. PhD thesis, Ohio State University, Columbus OH, USA, 2002.

[75] Lenoir, H., and Boudeau, N. An optimization procedure for springback compensation. In *Proceedings of ESAFORM* (Salerno, Italy, 2003).

[76] Liew, K., Tan, H., Ray, T., and Tan, M. Optimal process design of sheet metal forming for minimum springback via an integrated neural network evolutionary algorithm. *Structural and Multidisciplinary Optimization 26* (2004), 284–294.

[77] Lin, Z., Juchen, X., Xinyun, W., and Guoan, H. Sensitivity based optimization of sheet metal forming tools. *Journal of Materials Processing Technology 124*, 3 (2002), 319–328.

[78] Lin, Z., Juchen, X., Xinyun, W., and Guoan, H. Optimization of die profile for improving die life in the hot extrusion process. *Journal of Materials Processing Technology 142*, 3 (2003), 659–664.

[79] Liu, Y., Peng, X., and Qin, Y. FE simulation for concurrent design and manufacture of automotive sheet-metal parts. *Journal of Materials Processing Technology 150* (2004), 145–150.

[80] Lophaven, S., Nielsen, H., and J. Søndergaard. Aspects of the MATLAB Toolbox DACE. Technical Report IMM-REP-2002-13, Technical University of Denmark - Department of Informatics and Mathematical Modelling, Lyngby, Denmark, 2002.

[81] Lophaven, S., Nielsen, H., and Søndergaard, J. DACE - A MATLAB Kriging Toolbox. Technical Report IMM-TR-2002-12, Technical University of Denmark - Department of Informatics and Mathematical Modelling, Lyngby, Denmark, 2002.

[82] Martin, J., and Simpson, T. A study on the use of Kriging models to approximate deterministic computer models. In *Proceedings of the ASME Design Engineering Technical Conferences DETC* (2003).

[83] Martin, J., and Simpson, T. On the use of Kriging models to approximate deterministic computer models. In *Proceedings of the ASME Design Engineering Technical Conferences DETC* (2004).

[84] Martin, J., and Simpson, T. Use of Kriging models to approximate deterministic computer models. *AIAA Journal 43* (2005), 853–863.

[85] McKay, D. Introduction to Gaussian processes. http://wol.ra.phy.cam.ac.uk/mackay/gpB.pdf.

[86] McKay, M., Beckman, R., and Conover, W. A comparison of three methods for selecting values of input variables in the analysis of output from a computer code. *Technometrics 21* (1979), 239–245.

[87] Meckesheimer, M., and Booker, A. Computationally inexpensive metamodel assessment strategies. *AIAA Journal 40* (2002), 2053–2060.

[88] Montgomery, D. *Design and Analysis of Experiments*, 5th ed. John Wiley and Sons, Inc., New York, USA, 2001. ISBN 0-471-31469-0.

[89] Moore, D., and McCabe, G. *Introduction to the Practice of Statistics*, 3rd ed. W.H. Freeman and Company, New York, USA, 1999. ISBN 90-395-1420-8.

[90] Myers, R., and Montgomery, D. *Response Surface Methodology: Process and Product Optimization Using Designed Experiments*, 2nd ed. John Wiley and Sons, Inc., New York, USA, 2002. ISBN 0-471-41255-4.

[91] Naceur, H., Ben-Elechi, S., and Batoz, J. The inverse approach for the design of sheet metal forming parameters to control springback effects. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[92] Naceur, H., Ben-Elechi, S., Knopf-Lenoir, C., and Batoz, J. Response surface methodology for the design of sheet metal forming parameters to control springback effects using the inverse approach. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[93] Naceur, H., Breitkopf, P., Knopf-Lenoir, C., Rassineux, A., and Villon, P. Response surface methods for the optimal design of the initial blank in sheet metal forming process. In *Proceedings of ESAFORM* (Salerno, Italy, 2003).

[94] Naceur, H., Delamézière, A., Batoz, J., Guo, Y., and Knopf-Lenoir, C. Optimization of drawbead restraining forces and drawbead design in sheet metal forming process. *Journal of Materials Processing Technology 146* (2004), 250–262.

[95] Naceur, H., Guo, Y., Batoz, J., and Knopf-Lenoir, C. Optimization of drawbead restraining forces and drawbead design in sheet metal forming process. *International Journal of Mechanical Sciences 43*, 10 (2001), 2407–2434.

[96] Netherlands Institute for Metals Research (NIMR). Project proposal NIMR project MC1.03162. https://metnet1.nimr.nl.

[97] Nielsen, H. DACE, A MATLAB Kriging toolbox. http://www.imm.dtu.dk/ hbn/dace/.

[98] Nocedal, J., and Wright, S. *Numerical optimization.* Springer-Verlag, New York, USA, 1999. ISBN 0-387-98793-2.

[99] Northwestern University Optimization Technology Center (OTC). Optimization tree. http://www-fp.mcs.anl.gov/otc.

[100] Ohata, T., Nakamura, Y., Katayama, T., and Nakamachi, E. Development of optimum process design system for sheet fabrication using response surface method. *Journal of Materials Processing Technology 143-144*, 1 (2003), 667–672.

[101] Palaniswamy, H., Ngaile, G., and Altan, T. Optimization of blank dimensions to reduce springback in the flexforming process. *Journal of Materials Processing Technology 146* (2004), 28–34.

[102] Papalambros, P., and Wilde, D. *Principles of optimal design.* Cambridge University Press, New York, USA, 2000. ISBN 0-521-62727.

[103] Ponthot, J., and Kleinermann, J. Optimisation methods for initial/tool shape optimisation in metal forming processes. *International Journal of Vehicle Design* (submitted).

[104] Poortema, K. Statistiek en stochastiek; deel 1. Lecture notes Universiteit Twente, Enschede, Netherlands, 1999. (In Dutch).

[105] Poursina, M., António, C., Parvizian, J., Sousa, L., and Castro, C. Eliminating folding defect in forging parts using a genetic algorithm. In *Proceedings of ESAFORM* (Salerno, Italy, 2003).

[106] R. Haftka and Z. Gürdal. *Elements of structural optimization*, 3rd ed. Kluwer academic publishers, Dordrecht, Netherlands, 1992. ISBN 0-7923-1504-9.

[107] Ray, P., and Mac Donald, B. Determination of the optimal load path for tube hydroforming processes using a fuzzy load control algorithm and finite element analysis. *Finite Elements in Analysis and Design 41* (2004), 173–192.

[108] Redhe, M., Forsberg, J., Jansson, T., Marklund, P., and Nilsson, L. Using the response surface methodology and the d-optimality criterion in crashworthiness related problems – an analysis of the surface approximation error versus the number of function evaluations. *Structural and Multidisciplinary Optimization 24*, 3 (2002), 185–194.

[109] Repalle, J., and Grandhi, R. Optimum design of forging process parameters and preform shape under uncertainties. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[110] Revuelta, A., and Larkiola, J. Applying neural networks in the final thickness optimisation of a thin hydroformed part. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[111] Sacks, J., Schiller, S., and Welch, W. Design for computer experiments. *Technometrics 31* (1989), 41–47.

[112] Sacks, J., Welch, W., Mitchell, T., and Wynn, H. Design and analysis of computer experiments. *Statistical Science 4* (1989), 409–423.

[113] Sahai, A., Schramm, U., Buranathiti, T., Chen, W., Cao, J., and Xia, C. Sequential optimization and reliability assessment method for metal forming processes. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[114] Santner, T., Williams, B., and Notz, W. *The Design and Analysis of Computer Experiments*. Springer-Verlag, New York, USA, 2003. ISBN 0-387-95420-1.

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

[116] Schenk, O., and Hillmann, M. Optimal design of metal forming die surfaces with evolution strategies. *Computers and Structures 82* (2004), 1695–1705.

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

[118] Schwefel, H. *Evolution and Optimum Seeking*, 6th ed. John Wiley and Sons, New York, USA, 1995. ISBN 0-471-57148-2.

[119] Sheng, Z., Jirathearanat, S., and Altan, T. Adaptive FEM simulation for prediction of variable blank holder force in conical cup drawing. *International Journal of Machine Tools and Manufacture 44* (2004), 487–494.

[120] Shi, X., Chen, J., Peng, Y., and Ruan, X. A new approach of die shape optimization for sheet metal forming processes. *Journal of Materials Processing Technology 152* (2004), 35–42.

[121] Shirgoakar, M., Cho, H., Ngaile, G., Altan, T., Yu, J., Balconi, J., Rentfrow, R., and Worrell, W. Optimization of mechanical crimping to assemble tubular components. *Journal of Materials Processing Technology 146* (2004), 35–43.

[122] Sillekens, W., and Werkhoven, R. Hydroforming processes for tubular parts: Optimisation by means of adaptive and iterative FEM simulation. *International Journal of Forming Processes 4*, 3-4 (2001), 377–393.

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

[124] Simpson, T., Lin, D., and Chen, W. Sampling strategies for computer experiments: Design and analysis. *International Journal of Reliability and Applications 2* (2001), 209–240.

[125] Simpson, T., Peplinski, J., Koch, P., and Allen, J. On the use of statistics in design and the implications for deterministic computer experiments. In *Proceedings of the ASME Design Engineering Technical Conferences DETC* (1997).

[126] Simpson, T., Peplinski, J., Koch, P., and Allen, J. Metamodels for computer-based engineering design: Survey and recommendations. *Engineering with Computers 17* (2001), 129–150.

[127] Sousa, L., António, C., and Castro, C. Optimisation of multi-step hot forging operations. In *Proceedings of ESAFORM* (Salerno, Italy, 2003).

[128] Srikanth, A., and Zabaras, N. An updated Lagrangian finite element sensitivity analysis of large deformations using quadrilateral elements. *International Journal for Numerical Methods in Engineering 52*, 10 (2001), 1131–1163.

[129] Stinstra, E., Driessen, L., and Stehouwer, P. Design optimisation: Some pitfalls and their remedies. In 3^{rd} *ASMO UK/ISSMO Conference on Engineering Design Optimization* (Harrogate, UK, 2001).

[130] Strano, M., and Carrino, L. Adaptive selection of loads during FEM simulation of sheet forming processes. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[131] Sundaram, R. *A first course in optimization theory.* Cambridge University Press, New York, USA, 1996. ISBN 0-521-49770-1.

[132] The Mathworks, Inc. MATLAB Neural Network Toolbox User Guide.

[133] The Mathworks, Inc. MATLAB Optimization Toolbox User Guide.

[134] Thiyagarajan, N., and Grandhi, R. 3-D preform shape optimization in metal forming. In *Proceedings of NUMIFORM* (Columbus OH, USA, 2004).

[135] Thomas, R. *Simulationsgestützte Optimierung mehrstufiger Umformprozesse auf Parallelrechnern.* PhD thesis, RWTH Aachen, 1998. (In German).

[136] Torczon, V., and Trosset, M. Using approximations to accelerate engineering design optimization. ICASE 98-33, NASA, Hampton VI, USA, 1998.

[137] Tunali, S., and Batmaz, I. Dealing with the least squares regression assumptions in simulation metamodeling. *Computers and Industrial Engineering 38* (2000), 307–320.

[138] van Beers, W., and Kleijnen, J. Customized sequential designs for random simulation experiments: Kriging metamodeling and bootstrapping. Working Paper WP7, Center for Economic Research (CentER), University of Tilburg, Tilburg, The Netherlands, 2004.

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

[140] van Doorn, E. Statistiek en stochastiek; deel 2. Lecture notes Universiteit Twente, Enschede, Netherlands, 2000. (In Dutch).

[141] Webopedia. Definition of Fuzzy Logic. http://www.webopedia.com/TERM/f/fuzzy_logic.html.

[142] Weyler, R., Neamtu, L., and Duffett, G. An approach to the computer simulation of metal forming processes. In *Proceedings of ECCOMAS* (Jyväskylä, Finland, 2004).

[143] Wikipedia Encyclopedia. Definition Monte Carlo method. http://en.wikipedia.org/wiki/Monte_Carlo_method.

[144] Winston, W. *Operations Research: Applications and Algorithms*, 3rd ed. Duxbury Press, Belmont, CA, USA, 1993. ISBN 0-534-20971-8.

[145] Wisniewski, K., Kowalczyk, P., and Turska, E. On the computation of design derivatives for Huber–Mises plasticity with non-linear hardening. *International Journal for Numerical Methods in Engineering 57* (2003), 271–300.

[146] Wu, H., and Altan, T. Process optimization in stamping – a case study for flanging a clutch hub from steel plate. *Journal of Materials Processing Technology 146* (2004), 8–19.

[147] Yamazaki, K., and Han, J. Maximization of the crushing energy absorption of cylindrical shells. *Advances in Engineering Software 31* (2000), 425–434.

[148] Yang, J., Jeon, B., and Oh, S. Design sensitivity analysis and optimization of the hydroforming process. *Journal of Materials Processing Technology 113* (2001), 666–672.

[149] Yang, Y., Linkens, D., and Talamantes-Silva, J. Roll load prediction – data collection, analysis and neural network modelling. *Journal of Materials Processing Technology 152* (2004), 304–315.

[150] Youn, B. D., and Choi, K. K. A new response surface methodology for reliability-based design optimization. *Computers and Structures 82* (2004), 241–256.

[151] Zhao, G., Ma, X., Zhao, X., and Grandhi, R. Studies on optimization of metal forming processes using sensitivity analysis methods. *Journal of Materials Processing Technology 147* (2004), 217–228.