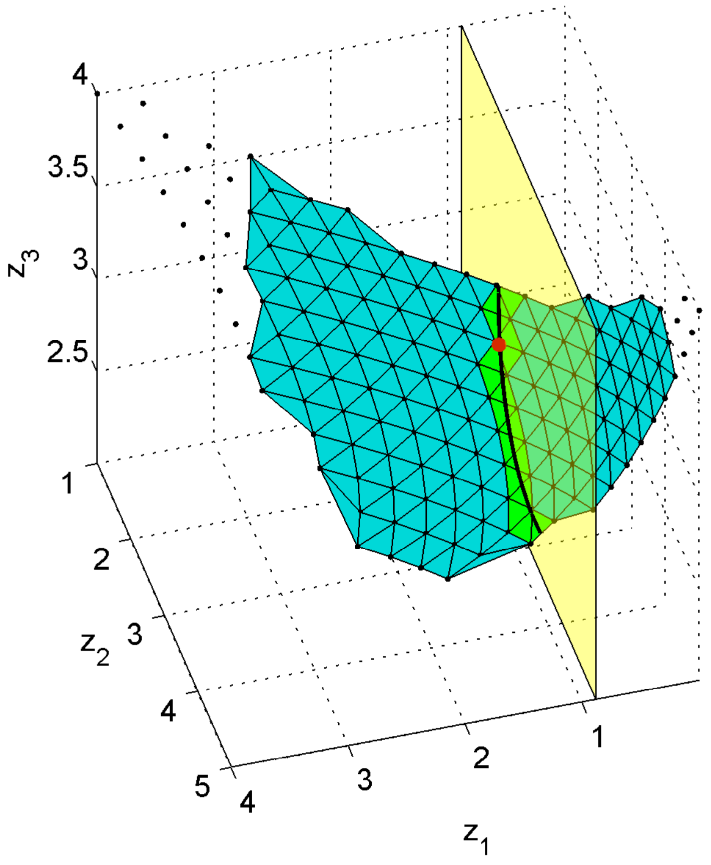


Interactive Multiobjective optimization

with application to hot rolling mills

Linder, Jonas
Lindkvist, Simon

Department of Signals & Systems
Chalmers University of Technology

Gothenburg, Sweden 2011
Master's Thesis EX043/2011

Abstract

In many real-world optimization applications there are often a number of conflicting objective functions that are all important to optimize. For example when producing metal sheets there can be a desire to minimize energy consumption of the process, maximize process speed and maximize the strength of the product at the same time. The purpose of multi-objective optimization is to give the decision maker an understanding of how these functions are conflicting and the possibility to choose an appropriate trade-off between them.

First, the basics of multi-objective optimization are described with concepts such as Pareto frontier, ideal point and Pareto optimality.

A two step generic framework for interactive multiobjective optimization was developed. The first step includes the methods for an offline tool for obtaining the Pareto set that can handle non-convex problems with up to four conflicting objectives. The second step includes methods for an online tool which can be used to navigate continuously, smoothly and in real-time on the obtained Pareto set.

The methods were implemented in a prototype tool to verify the claim of the theory. The framework was applied to a set of small nonlinear and non-convex example problems, a pendulum-cart-problem with variable pendulum length and a large scale hot rolling mill application with three objective functions and up to 100 000 variables. The resulting tools performed well and the interactive online navigation works in real-time.

The influence of different parameters in the hot rolling mill model were investigated and the combinatorial behavior was especially analyzed. The results confirms that the objectives are conflicting and the presumed dependencies between different objectives and/or parameters are as expected. When the combinatorial behavior of the model is included in the optimization the obtained Pareto set is non-convex and holes emerges in the Pareto frontier.

Acknowledgements

We would like to show our deepest gratitude to our supervisor, Johan Sjöberg, at ABB Corporate Research for all his input during the work with this thesis. His feedback and guidance during the thesis have been invaluable.

We would like to thank our examiner, Bo Egardt, for his interest and questions during the work with the thesis.

We would also like to thank the project leader of the Roll.LABB project, Anders Daneryd, for his insight of the hot rolling problem and input during the thesis work. At last we want to thank our co-workers and fellow thesis writers at ABB Corporate Research for making the thesis period an enjoyable experience.

The authors, Västerås, 2011-06-03

Abbreviations

CLP	Coin-or Linear Programming
DSD	Directed Search Domain
GUI	Graphical User Interface
IPOPT	Interior Point OPTimizer
MOO	MultiObjective Optimization
NBI	Normal Boundary Intersection
NC	Normal Constraint
SOO	SingleObjective Optimization

Contents

1 Introduction

This is the report for the master thesis *Interactive Multiobjective optimization– with application to hot rolling mills* performed at the company ABB Corporate Research in Västerås.

1.1 Background

In many real-world optimization applications there are often a number of conflicting objective functions that are all important to optimize. For example when producing metal sheets there can be a desire to minimize energy consumption of the process, maximize process speed and maximize the strength of the product at the same time. The purpose of multi-objective optimization is to give an understanding of how these functions are conflicting and to give the user the possibility to choose an appropriate trade-off between them.

In hot rolling mills this problem occurs and there are a number of objective functions that are of interest, e.g. energy consumption and production speed. Several settings such as roll gaps, roll speeds and temperatures can be controlled to change the production process and to find the appropriate settings to obtain an optimal process is difficult. If *singleobjective optimization* (SOO) is used, which only considers one objective at a time, it is difficult for the decision maker to achieve a proper balance between the different objectives and understand the solution space. One solution to this is to use *multiobjective optimization* (MOO) where several objective functions are considered at once (Marler and Arora, 2004).

The Roll.LABB project has developed a model detailed enough to simulate and control a hot rolling mill. The Roll.LABB model is a large scale problem with 10 000 - 200 000 variables and where both the constraints and the objective functions can be nonlinear. This makes the problem very computationally heavy and hence finding solutions to the problem is time consuming and can not be done in real-time. It would be useful for the decision maker to be able to change its desired balance of the objectives and at the same time get a quick visual feedback of the suggested solution in an interactive way.

1.2 Purpose

This master thesis work is supposed to evaluate and implement a generic framework for interactive MOO that should be applied to a hot rolling mill application. The framework and knowledge gained can be used as a foundation for further development.

1.3 Project goals

To develop a framework for generic MOO problems and specifically apply it to the Roll.LABB hot rolling mill model.

1.3.1 Sub goals

- Form several simple example problems that mimics important properties of the Roll.LABB model.
- Implement a method that computes a discrete Pareto set for 3-4 objective functions.

- Design a method which the decision maker can use to navigate in the discrete Pareto set smoothly (near continuous) and get a fast visual feedback.
- Create a framework consisting of the methods implemented in the Maple/IPOPT-tool developed at ABB Corporate Research and a *graphical user interface* (GUI) for visualization and user inputs.
- Evaluate the framework using a number of generic example problems.
- Evaluate the framework when applied to the Roll.LABB model.

1.3.2 Extended goals

Adapt the GUI specifically for the hot rolling mill application.

1.4 Exclusions

The hot rolling mill optimization model will be supplied by ABB Corporate Research and is not supposed to be developed in this project.

More than four objective functions should not be considered at the same time.

1.5 Method

At first a comprehensive theoretical study of MOO was performed. This study was the basis to choose and implement the appropriate methods used to create a generic framework for interactive MOO. This included choosing method for obtaining the discrete Pareto set, choosing an interpolation method and creating a GUI.

A small number of example problems were created to mimic different properties of the Roll.LABB model and to evaluate the methods. This was done to understand the optimization methods and test important features of the Roll.LABB model.

The framework was then applied to the example problems in order to thoroughly understand the functionality and performance of the implemented methods with respect to important properties of the Roll.LABB model. Finally, the framework was applied to the hot rolling application, an evaluation of its performance was done and the results were analyzed.

The whole process was iterative, e.g. further literature studies had to be done.

2 Introduction to multiobjective optimization

Multiobjective optimization (MOO) is optimization where several objective functions are considered simultaneously. The problem is to

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}}\ \mathbf{f}(\mathbf{x}) = \{f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x})\} \\ &\text{subject to } \mathbf{x} \in X \end{aligned} \tag{1}$$

where $\mathbf{f}(\mathbf{x})$ is a vector of nonlinear objective functions and the *feasible decision space* X might as well be described by nonlinear constraints.

Figure 1 shows the basic relations in the MOO problem. The function $\mathbf{f}(\mathbf{x})$ maps the *decision vector*, $\mathbf{x} \in \mathbb{R}^n$, to the *objective vector* $\mathbf{z} = \mathbf{f}(\mathbf{x}) \in \mathbb{R}^m$. X is the *feasible decision space* and is a subset of $\mathbb{R}^n$. $Z = \mathbf{f}(X)$ is the *feasible objective space* and is a subset of $\mathbb{R}^m$.

The right plot in Figure 1 shows a simple example of the objective space, $Z \subset \mathbb{R}^2$, which is marked with the dashed line. In general, the individual objective functions ($f_i(\mathbf{x}), i = 1, \ldots, m$) are conflicting, e.g. decreasing $f_1(\mathbf{x})$ will increase $f_2(\mathbf{x})$. Consider the objective point $\mathbf{z}$ marked with the black dot. If $z_2 = f_2(\mathbf{x})$ is to be decreased then at the same time $z_1 = f_1(\mathbf{x})$ has to be increased to stay in the feasible objective space. This means that the two objectives z_1 and z_2 are conflicting.

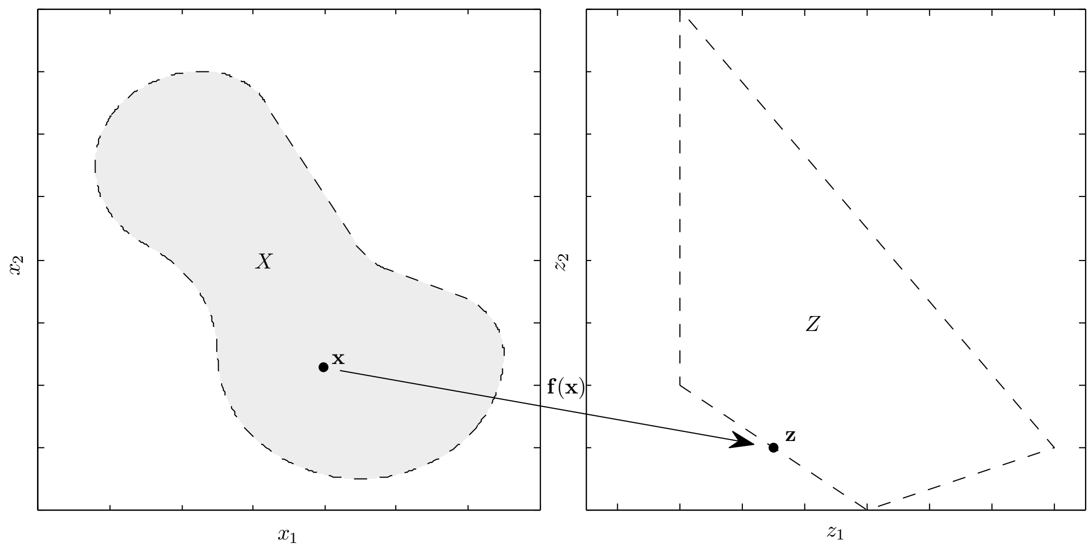


Figure 1: An example of the decision space (left plot) mapped to the objective space (right plot).

2.1 Pareto optimality

In general the objective functions are conflicting which means that there is no single optimal solution. A common way to describe optimality in MOO is Pareto optimality. The set of all Pareto optimal solutions are called the *Pareto frontier*.

Definition 1. *A point $\mathbf{x}^* \in X$, is Pareto optimal if and only if there does not exist another point, $\mathbf{x} \in X$, such that $\mathbf{f}(\mathbf{x}) \leq \mathbf{f}(\mathbf{x}^*)$, and $f_i(\mathbf{x}) < f_i(\mathbf{x}^*)$ for at least one function (Marler and Arora, 2004).*

This means that a point $\mathbf{z} = \mathbf{f}(\mathbf{x})$ is Pareto optimal if there is no way to improve one objective without deteriorating another. Figure 2 shows an example of the objective space with the Pareto frontier marked with a thick black line.

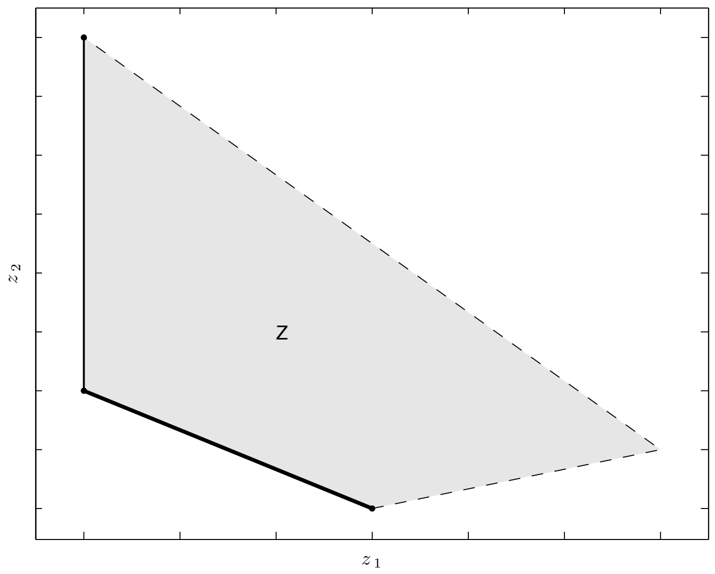


Figure 2: A simple example of a Pareto frontier which has Pareto optimal (thick solid line) and weakly Pareto optimal (thin solid line) sections.

Definition 2. *A point* $\mathbf{x}^* \in X$*, is weakly Pareto optimal if and only if there does not exist another point,* $\mathbf{x} \in X$*, such that* $\mathbf{f}(\mathbf{x}) \leq \mathbf{f}(\mathbf{x}^*)$ *(Marler and Arora, 2004).*

This means that there is no point $\mathbf{z} = \mathbf{f}(\mathbf{x})$ that improves all objectives at once. The solid line in Figure 2 parallel to the z_2-axis is a set of weakly Pareto optimal points.

It should be noted that every Pareto optimal point is also a weakly Pareto optimal point.

2.1.1 Local Pareto optimality

In practice, it is usually not possible to calculate global (weak) Pareto optimal points, as defined in Definition 1 and 2, but rather only local (weak) Pareto optimal points are available (Miettinen, 1998).

Definition 3. *A point* $\mathbf{x}^* \in X$*, is local Pareto optimal if it is Pareto optimal in a neighborhood of* $\mathbf{x}^*$ *(Messac et al., 2003).*

Definition 4. *A point* $\mathbf{x}^* \in X$*, is local weakly Pareto optimal if it is weakly Pareto optimal in a neighborhood of* $\mathbf{x}^*$ *(Messac et al., 2003).*

Note that a global Pareto optimal solution is also a local Pareto optimal solution but only in special cases this implication is reversible (Miettinen, 1998). Figure 3 shows a part of the boundary of the feasible set. The curves AB and DE are global Pareto optimal, the curve CD is local Pareto optimal and BC is neither global nor local Pareto optimal. To understand how the curve CD can be local Pareto optimal, cut the boundary such that the curve CE is the only thing left. The curve CE is now Pareto optimal, but if the curve segment AC is included it is possible to improve both z_1 and z_2, hence CD is local Pareto optimal.

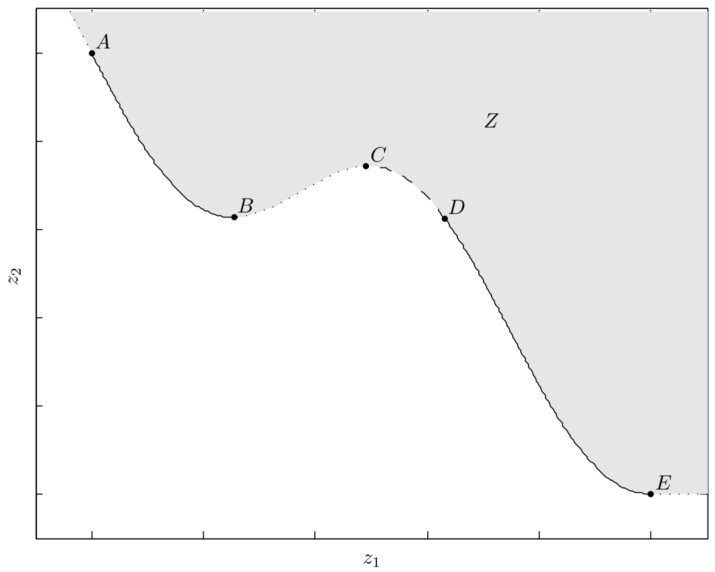


Figure 3: A Pareto frontier with non- (line segment BC), global- (line segment AB and DE) and local- (line segment CD) Pareto optimal sections. Note that the curve segments AB and DE also are local Pareto optimal since all global Pareto optimal points also are local Pareto optimal.

2.1.2 Efficiency & dominance

A more general approach to describe optimality in MOO is the concept of efficiency. The concept introduces a pointed convex cone called an ordering cone. This cone C is used to partially order two points. A vector $\mathbf{z}^2 \in \mathbb{R}^m$ is said to dominate $\mathbf{z}^1 \in \mathbb{R}^m$ if $\mathbf{0} \neq \mathbf{z}^1 \in \mathbf{z}^2 + C$ (Miettinen, 1998).

Figure 4 shows an illustration for $\mathbb{R}^2$ with the ordering cone $C = \mathbb{R}^2_+$ which is also called the *Pareto cone*. Note that $\mathbf{z}^2$ is in the cone $\mathbf{z}^1 - \mathbb{R}^2_+$ which means that $\mathbf{z}^1$ is dominated by $\mathbf{z}^2$. Also note that the Pareto set (fat solid line) dominates the entire feasible region Z.

Definition 5. *A point* $\mathbf{x}^* \in X$*, is* C*-efficient if there does not exist another point,* $\mathbf{x} \in X$*, such that* $\mathbf{0} \neq \mathbf{f}(\mathbf{x}^*) \in \mathbf{f}(\mathbf{x}) + C$ *(Miettinen, 1998).*

A point is efficient with respect to the ordering cone C if it is not dominated by any other point.

Definition 6. *A point* $\mathbf{x}^* \in X$*, is weakly* C*-efficient if there does not exist another point,* $\mathbf{x} \in X$*, such that* $\mathbf{0} \neq \mathbf{f}(\mathbf{x}^*) \in \mathbf{f}(\mathbf{x}) + interior(C)$ *(Miettinen, 1998).*

A point is weakly efficient with respect to the ordering cone C if a point dominating it, is not in the interior of the cone.

If the ordering cone is equal to $\mathbb{R}^m_+$ the concept of efficiency is equivalent with Pareto optimality.

Figure 4 shows three points in the feasible set Z. Note that $\mathbf{z}^1$ is not C-efficient since $\mathbf{z}^2$ dominates it. However, since $\mathbf{z}^2$ is on the boundary of the cone, $\mathbf{z}^1$ is weakly C-efficient (weakly Pareto optimal). $\mathbf{z}^2$ and $\mathbf{z}^3$ are C-efficient (Pareto optimal) since there is no other point dominating it.

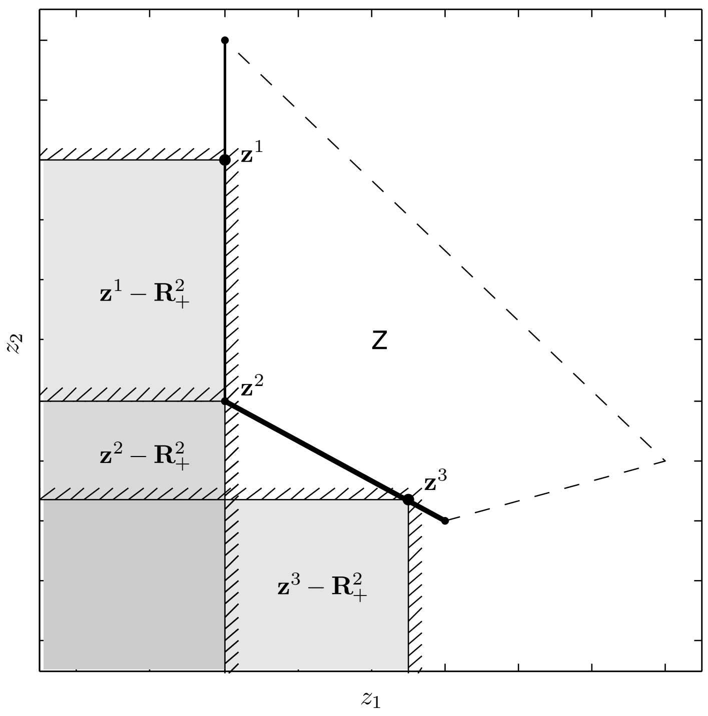

$\mathbf{z}^1$
$\mathbf{z}^1 - \mathbf{R}^2_+$
Z
$\mathbf{z}^2$
$\mathbf{z}^2 - \mathbf{R}^2_+$
$\mathbf{z}^3$
$\mathbf{z}^3 - \mathbf{R}^2_+$
z_2
z_1

Figure 4: The efficiency of three points on the Pareto frontier is checked with the negative ordering cone $-\mathbb{R}^2_+$. Both $\mathbf{z}^2$ and $\mathbf{z}^3$ are C-efficient while $\mathbf{z}^1$ is only weakly C-efficient. The gray areas are inside the cone of respective point.

2.2 Convexity

In MOO the convexity of a problem is an important property that makes a big difference in how the problem can be solved. One way of defining convexity was made by Miettinen (1998).

Definition 7. *A multi-objective optimization problem is convex if all objective functions are convex and the feasible region is convex.*

A MOO problem has both a design space and an objective space that can be either convex or non-convex individually. It is important to note that a MOO problem with a non-convex design space still can have a convex Pareto frontier (Deb, 2001).

2.3 Connectedness

The property of connectedness is important in MOO to show if it possible to continuously move between all points in the Pareto optimal set.

Definition 8. *A set is connected when it is not possible to partition it into two non-empty subsets such that each subset has no common point with the set closure of the other (Insall and Weisstein, 2011).*

Figure 3 show a Pareto optimal set (the curve segments AB and DE) which is an example of a disconnected set where two segments are separated with the curve segment BD. Connectedness can only be guaranteed for a few special cases, e.g. convex MOO problems (Warburton, 1983).

2.4 Ideal and nadir point

If all objective functions would be optimized individually, i.e. without regarding the other objective functions, the composition of these values represents the *ideal objective point*, $\mathbf{z}^I$. Mathematically it is described as

$$\begin{aligned}&\text{minimize } f_i(\mathbf{x}), \quad i = 1, \dots, m\\ &\text{subject to } \mathbf{x} \in X.\end{aligned} \tag{2}$$

This is a unique point that is the optimal solution of a MOO problem if it would be feasible. However, since MOO problems in general have conflicting objective functions the ideal point is almost never in the feasible objective space. Figure 5 shows an example in $\mathbb{R}^2$. The individual minimas of the objective functions $\mathbf{z}^{1*}$ and $\mathbf{z}^{2*}$ construct the ideal point $\mathbf{z}^I$. In this case the ideal point is not feasible since it is outside the feasible space Z. The ideal point describes the lower bounds of the Pareto optimal set. An extension to the ideal point is the utopian point, $\mathbf{z}^U$ which is a strictly better solution than the ideal point and is therefore never feasible.

The *nadir point*, $\mathbf{z}^N$, is instead the worst value of every objective function of the Pareto optimal set. This obtains the upper bounds of the set. Note that it should not be confused with the worst possible objective value of the complete objective space. An illustration of the nadir point can be seen in Figure 5.

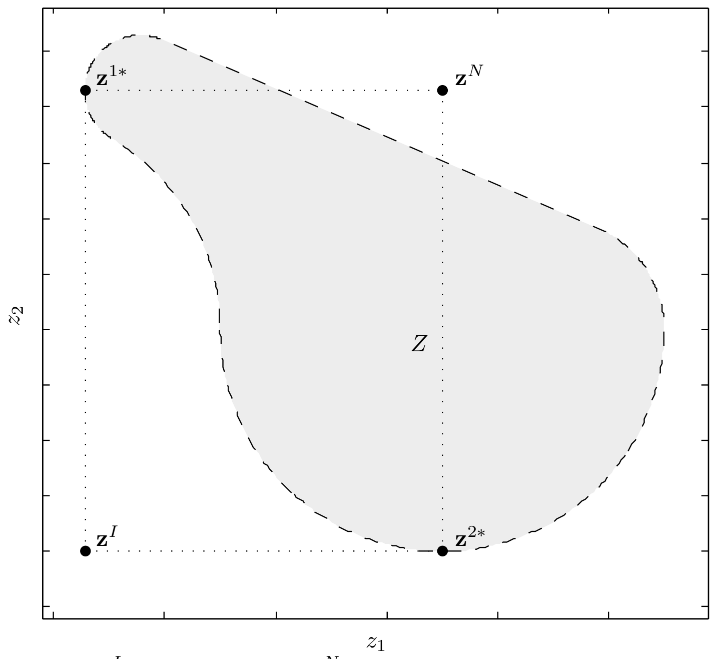


Figure 5: The ideal point $\mathbf{z}^I$ and nadir point $\mathbf{z}^N$ are shown together with the individual minima $\mathbf{z}^{1*}$ and $\mathbf{z}^{2*}$. The dotted line can be seen as the boundaries of the Pareto optimal set and all possible Pareto optimal solutions are inside the rectangle.

The boundaries of the Pareto optimal set are useful in several ways. In some optimization methods, the ideal point is used as a reference point which the method should optimize towards. In MOO problems with many objective functions it can be difficult to visualize the Pareto set and knowledge about the best and worse solutions, i.e. the range of solutions, is then important information for the decision maker.

Even though finding the ideal point involves m SOO problems it can still be a problematic task to solve, for instance when one or several of the objective functions are non-convex it is difficult to find the global minimum.

The search for the nadir point is even more problematic and in many cases only a bad approximation can be obtained. In non-convex problems there are no good methods to find the nadir point since it is a global optimization problem. Several approximating methods do exist, e.g. using evolutionary algorithms or payoff tables. In general, these methods are either computational heavy or give an approximation where the precision of the estimation is unknown (Miettinen, 1998; Monz, 2006).

The method of using the payoff table is simply performed by making a table of the objective vectors from the search for the ideal point, i.e. using the results from SOO. The payoff table or payoff matrix is

$$\Phi = \begin{bmatrix} f_1^{1*} & & f_1^{m*} \\ \vdots & \dots & \vdots \\ f_m^{1*} & & f_m^{m*} \end{bmatrix} \tag{3}$$

$$\mathbf{z}^I = \begin{bmatrix} \min & f_1 \\ & \vdots \\ \min & f_m \end{bmatrix} \tag{4}$$

The maximal values of each row in the payoff table can then be used as an estimation of the upper bounds of the Pareto set. This method is simple and fast since it uses the results from the search for the ideal point but it is however only a simple estimation where the actual nadir point can be much lower or higher than the estimation (Miettinen, 1998).

2.5 Scaling

In most cases it is advantageous that all objective functions have the same magnitudes to give them the same relative importance. There are several methods to scale the objective vector. Miettinen (1998) suggested one where the boundaries of the Pareto optimal set are used to normalize the objective functions as

$$\bar{f}_i(x) = \frac{f_i(\mathbf{x}) - z_i^I}{z_i^N - z_i^I}, \quad i = 1, \dots, m. \tag{5}$$

2.6 Method classification

There is a large number of different methods available to solve MOO problems. These methods all have different properties, e.g. how much that has to be known about the problem in advance and how much effort the decision maker has to put in, in order to control the solver. One important criteria when selecting method is how the solver interacts with the decision maker. To be able to select which method that is suitable for a certain case there are several classification schemes. Miettinen (1998) describes one widely used classification where the methods have been divided into four classes dependent on which role the decision maker has. Depending on when the decision maker specifies the optimization parameters and when the solution is chosen the method is classified into one or several of these classes:

No-preference methods No parameters can be controlled by the decision maker and the optimization returns only one optimal point.

A priori methods The decision maker sets the parameters before the optimization is performed and the optimization returns one optimal point.

A posteriori methods A range of parameters are automatically set, this creates a set of solutions. The decision maker then chooses one preferred solution from the set.

Interactive methods The decision maker can make changes of parameters both before and after optimization, e.g. in an iterative process where the decision maker change the parameters in each iteration and continue changing until a desired solution is obtained.

These classes only describes in which way a method can be used by the decision maker. This means that one method can be included in several classes. One example of this is the weighted-sum method which scalarizes (1) to the SOO problem

$$\begin{aligned} &\underset{\mathbf{x}}{\text{minimize}}\ w_1 f_1(\mathbf{x}) + w_2 f_2(\mathbf{x}) + \ldots + w_m f_m(\mathbf{x}) \\ &\text{subject to } \mathbf{x} \in X \end{aligned} \tag{6}$$

where $w_i \geq 0, \quad i = 1, \ldots, m$ and $\sum_{i=1}^{m} w_i = 1$. The weights describe how much every objective functions should be taken into consideration when optimizing. For example for a two dimensional problem where the weights are set to $w_1 = 0.5$ and $w_2 = 0.5$ then both objective functions are considered equally much.

If the decision maker sets the weights to specific values before optimizing and then accepts the obtained solution it is considered an *a priori* method. This means that the decision maker is only active before the actual optimization. If a range of weights are instead specified automatically, i.e. the Pareto frontier is sampled, and the decision maker chooses a preferred solution among the samples it is considered an *a posteriori* method. The decision maker is only active after the actual optimization. The third case is when the decision maker is active both before and after the optimization which is considered as an *interactive* method. An example of this is when the decision maker is not satisfied with the solution for a set of preset of weights and changes them to make a new optimization and continues doing so until a preferred solution is found.

3 Generic framework for interactive MOO

The aim of the generic interactive MOO framework is to aid the decision maker by showing the available optimal solutions of MOO problems, i.e. visualizing the Pareto frontier. There are several a posteriori methods available for sampling the Pareto frontier which the decision maker can use to simply pick the preferred solution. However, for large scale problems it might take several hours or even days to cover the complete frontier with enough samples to cover the entire Pareto frontier. For problems with more than three dimensions it can also be difficult to visualize the Pareto set. One solution to this is to use interactive methods which the decision maker can use to navigate on the Pareto frontier without solving the complete frontier. However, when the size and complexity of the problem grows, the time needed to find even one feasible solution might be minutes and it is even possible that the optimization routine fails to find a solution. This makes the work for the decision tedious and slow.

The approach used in this thesis is to first do an offline sparse sampling of the Pareto frontier and then do a linear combination of the resulting set of Pareto optimal points. An online navigation step is then used by the decision maker to continously move around on the approximated Pareto frontier in realtime. The basic assumption is that the Pareto frontier is sampled "dense enough“ for the linear approximation to be a good estimation of the original MOO problem. This means that the approximation is accurate enough to give the decision maker a better understanding of the problem, to show how it behaves and to select a final solution. The selected solution is then verified in the original MOO problem.

Figure 21 shows an overview of the framework and how different parts work together. It is divided into three blocks:

1. **Problem formulation** The main MOO problem (1) is formulated in a MAPLE / IPOPT-tool and this formulation is used in the offline optimization process and the verification in the online optimization process.
2. **Offline sampling** Creates a set of Pareto optimal points for the online navigation.
   (a) **Pre-processing of the problem** Creates data used to sample the Pareto frontier. There are three routines in this step.
      - **Estimation of ideal- & nadir point** These points are used to normalize the objective functions $(f_1(\mathbf{x}), f_2(\mathbf{x}), \ldots, f_m(\mathbf{x}))$ between zero and one.
      - **Reference point creation** Used to get an even distribution of samples over the Pareto frontier.
      - **Creation of initial values** A starting guess of optimal values for the optimization tool to use.

   (b) **Optimization** The MOO problem is reduced to a number of SOO problems with the direction method and these are solved to obtain the discrete Pareto set.

   (c) **Post-processing of solutions** The obtained set is filtered in order to guarantee that all points in the set are Pareto optimal solutions.

3. **Online navigation** Navigation of the linear approximation of the Pareto set. The interaction and optimization steps are iterated until the decision maker finds a suitable solution.
    - **Interaction** This step interacts with the decision maker who is able to select desired values, range of both objective functions and decision variables. The result is visualized to get a better understanding of the problem.
    - **Optimization** This step has two routines. The linear approximation optimization is used for ordinary navigation on the Pareto frontier and the verification routine is used to check if the linear approximation is feasible and/or make it feasible. Both routines use the direction method.

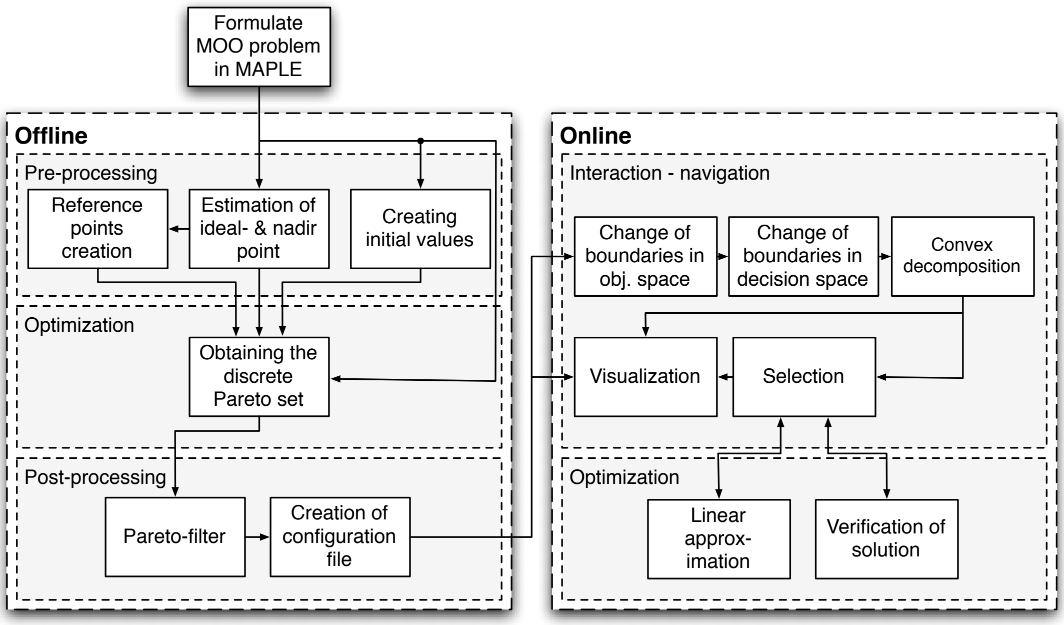

Formulate MOO problem in MAPLE
Offline
Pre-processing
Reference points creation
Estimation of ideal- & nadir point
Creating initial values
Optimization
Obtaining the discrete Pareto set
Post-processing
Pareto-filter
Creation of configuration file
Online
Interaction - navigation
Change of boundaries in obj. space
Change of boundaries in decision space
Convex decomposition
Visualization
Selection
Optimization
Linear approx-imation
Verification of solution

Figure 6: Overview of the framework. The offline block is explained in Section 3.3 and the Online block is explained in Section 3.4.

In the following sections the theoretical background of these parts will be described. First, previous work in interactive MOO is described. In Section 3.2, the direction method is explained, which is the foundation of both the offline and online blocks. Block two and three will be described thoroughly in Section 3.3 and Section 3.4. Also a short description of how these methods were implemented is made in Section 3.5.

3.1 Previous work

There are many methods to approximate a discrete set of the Pareto frontier. There are only a few available methods which can produce an even distribution of points, can generate all available Pareto points with an appropriate choice of variables and is relatively easy to apply (Messac et al., 2003).

The *normal boundary intersection* (NBI) method is based on the direction method. The method creates starting point inside the convex hull of the individual minimas and solves one SOO optimization problem for each point. The NBI method can create non-Pareto optimal solutions and potentially do not sample the entire Pareto frontier

(Das and Dennis, 1998).

The *normal constraint* (NC) method is a development of the NBI method. It uses the same idea to create reference points but solves the SOO problems with a method which is more robust. The NC method has the same problems as the NBI method but removes the Pareto optimal solutions with a Pareto filter (Messac et al., 2003). Messac and Mattson (2004) extends the original NC method and describes how to create a set of reference points which guarantees to cover the entire Pareto frontier.

The *directed search domain* (DSD) method is a development of the *physical programming* method. It creates reference points in the same way as the NBI method but changes the direction of the search to cover the entire Pareto frontier. The method minimizes the sum of all objective values and use the reference points together with a cone to restrict the possible solutions (Utyuzhnikov et al., 2009; Erfani and Utyuzhnikov, 2010).

Evolutionary algorithms are interesting, but they do not generate an even distribution or guarantee to cover the entire Pareto frontier and is thereby not interesting for this thesis (Utyuzhnikov et al., 2009).

The *Pareto race*- and the *Quadratic Pareto race* methods are two interactive navigation methods for MOO linear- and quadratic programming problems respectively. The MOO problem is solved in real-time and the decision maker chooses how to move on the Pareto frontier by changing the speed and direction of motion (Korhonen and Wallenius, 1988; Korhonen and Yu, 1997).

The *Pareto navigator* method is a development of the *Pareto race* method and is an extension to general nonlinear MOO problems. Instead of directly solving the optimization problem, it navigates on a polyhedral approximation of a set of sampled points. Even though the method is specified for nonlinear MOO problems, dealing with highly non-convex problems is a subject of research (Eskelinen et al., 2010).

Monz (2006); Monz et al. (2008) describes the *Pareto navigation* method which is an interactive navigation method where the decision maker navigates on an approximated Pareto frontier by choosing the desired levels for one objective at the time. The method includes a restriction-, selection- and discounting mechanism. The restriction mechanism is used to set upper boundaries on the objectives. The selection mechanism is the navigation tool which is based on a cone achievement scalarising function, e.g. the direction method by Pascoletti and Serafini (1984). The discounting mechanism is basically a way to limit how rapidly the solution is allowed to change in each objective. This method mainly deals with convex set but discuss non-convex sets in a conceptual manner.

Hartikainen et al. (2011) introduces an approximation approach which can deal with non-convex Pareto frontiers. The method use a pre-sampled set with no assumptions other than it being Pareto optimal. Delaunay triangulation is then performed on the sampled set. The set of polytopes created by the Delaunay triangulation is filtered to make sure that non of the created polytopes dominates the other. This is done by optimization and it is possible that $O(m^k)$ optimizations needs to be performed where m is the number of polytopes and k is the dimension of the problem. The filtering is computational heavy and more research is needed to implement the approximation approach.

3.2 Direction method

The direction method is the basis in both the sampling of the Pareto frontier and the online navigation, hence a brief description is necessary. It is a vector scalarization method which reduces the MOO problem (1) into the SOO problem,

$$\begin{aligned} &\underset{\mathbf{x},\alpha,\mathbf{q}}{\text{maximize}}\ \alpha \\ &\text{subject to } \mathbf{x} \in X \\ &\qquad\quad \mathbf{f}(\mathbf{x}) = \alpha\mathbf{d} + \mathbf{z}^R - \mathbf{q} \end{aligned} \tag{7}$$

where $\alpha \in \mathbb{R}$, $\mathbf{q} \in \mathbb{R}^m_+$ and $\mathbf{d}, \mathbf{z}^R \in \mathbb{R}^m$ (Pascoletti and Serafini, 1984). The vectors $\mathbf{d}, \mathbf{z}^R$ are called the *direction vector* and the *reference point*, respectively. The method pushes the negative *Pareto cone* in the direction $\mathbf{d}$ from the starting point $\mathbf{z}^R$ until the set $(\mathbf{z}^* - \mathbb{R}^m_+) \cap Z = \{\mathbf{z}^*\}$ where the optimal solution $\{\mathbf{x}^*, \mathbf{q}^*, \alpha^*\}$ gives $\mathbf{z}^* = \mathbf{f}(\mathbf{x}^*) = \mathbf{z}^R + \alpha^*\mathbf{d} - \mathbf{q}^*$ (Eichfelder, 2008). The vector $\mathbf{q}$ is called the *slack variable* and if it is removed from (7) the method reduces down to a line search in $\mathbb{R}^m$.

By varying $\mathbf{d}$ and $\mathbf{z}^R$ the entire weakly Pareto optimal set can be obtained (Klamroth and Tind, 2006; Pascoletti and Serafini, 1984). If an optimal solution is found when using the direction method, it is at least a weakly Pareto optimal solution, but note that a Pareto optimal solution is also a weak Pareto optimal solution as described in Section 2. This is however not true when no slack is used, since any feasible solution is on the search line. Figure 7 shows an example where the optimization would yield the solution $\mathbf{z}^*$ without slack but the point B with slack. Any line crossing only the curve segment BC will yield non-Pareto optimal solutions if slack is not used.

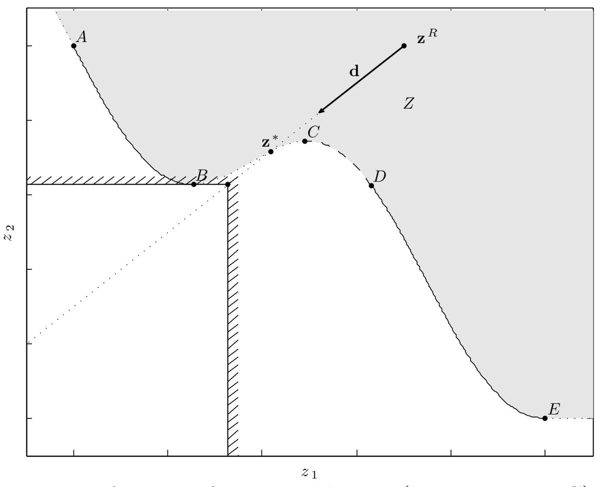


Figure 7: A Pareto frontier with non- (line segment BC), global- (line segment AB and DE) and local- (line segment CD) Pareto optimal sections.

The left plot in Figure 8 shows a situation with $m = 2$. $\mathbf{z}^1$ is not Pareto optimal since the intersection between Z (the feasible set) and $\mathbf{z}^1 - \mathbb{R}^2_+$ is not equal to $\mathbf{z}^1$. At the point $\mathbf{z}^*$ the intersection $(\mathbf{z}^* - \mathbb{R}^2_+) \cap Z = \mathbf{z}^*$ which makes $\mathbf{z}^*$ Pareto optimal since it is unique (Pascoletti and Serafini, 1984).

The right plot in Figure 8 shows a feasible set which is disconnected. $\mathbf{z}^*$ will be feasible and Pareto optimal since it is unique and the intersection $(\mathbf{z}^* - \mathbb{R}^2_+) \cap Z$ is $\mathbf{z}^*$ itself. If the slack variable $\mathbf{q}$ is discarded the problem will be infeasible since there is no solution on the line $\mathbf{z}^R + \alpha\mathbf{d}$.

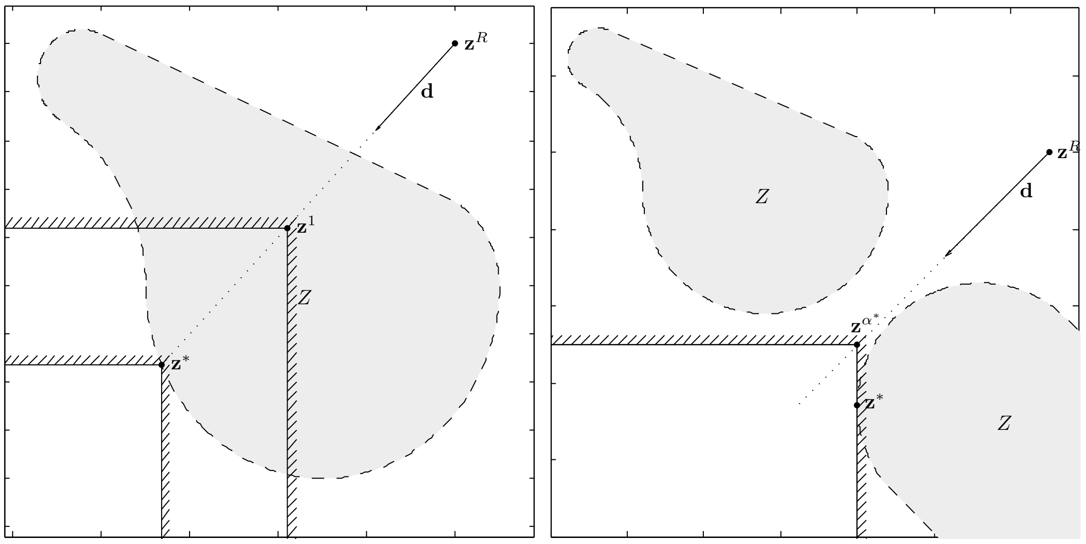


Figure 8: The direction method geometrically interpreted. The left plot shows the concept of the intersection between the feasible set Z and the negative Pareto cone. The right plot shows how the cone (slack) is used to find a solution which is not on the line.

3.3 Obtaining the discrete Pareto set

As mentioned in Section 3 the method for obtaining the discrete Pareto set is divided into three parts, pre-processing, sampling and post-processing.

The pre-processing stage creates a direction and a set of reference points that are later used with the direction method to get an even coverage of the entire Pareto frontier. Figure 9 shows an example in $\mathbb{R}^2$ with 11 reference points.

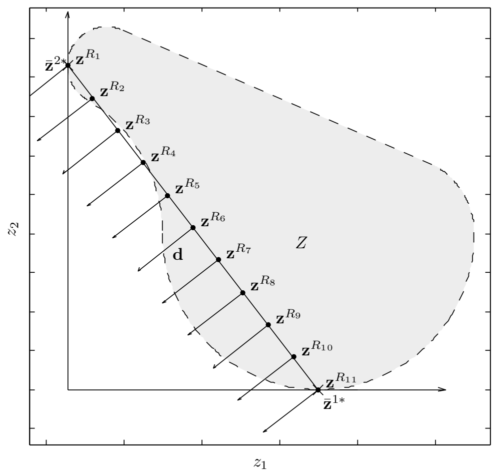


Figure 9: An example of the normalized feasible set with the set of 11 reference points, the reference hyperplane and the individual minima z^{1*} and z^{2*}

The reference points created in the pre-processing stage is used together with the direction method to sample the Pareto frontier. The original MOO problem is reduced to a number of SOO problems which can be solved with existing optimization routines.

The post-processing stage rejects non-Pareto optimal solutions with a *Pareto filter* and prepares it for the online navigation.

3.3.1 Reference point creation

To get a good approximation of the Pareto frontier it is important to have an even distribution of sampled points. This means that since the direction method is used to sample the Pareto frontier, the pattern of the created reference points is crucial in order to evenly distribute the sampled points. Hence, a new method to distribute the points is needed. To give all objectives the same relative importance, they are normalized according to Section 2.5. This means that all points are inside the hyper-cube $0 \leq f_i(\mathbf{x}) \leq 1,\ i = 1, \ldots, m$ in the objective space Z.

In the same way as the NBI method uses the convex hull of the individual minima to create reference points (Das and Dennis, 1998), the approach used in this thesis is to create points on the hyperplane spanned by the m individual minima vectors. All points on a hyperplane in $\mathbb{R}^m$ can be described as a sum of one point on the plane and $m - 1$ linearly independent vectors which lies in the plane.

In order to be able to spread points evenly, an orthonormal basis is chosen such that

$m-1$ of the vectors in the basis coincide with the hyperplane. The left plot in Figure 10 shows an example in $\mathbb{R}^2$. The vector $\mathbf{b}_2$ coincide with the hyperplane (the dotted line between $[0,1]^T$ and $[1,0]^T$) and the second basis $\mathbf{b}_1$ is chosen as the normal to the hyperplane. All points on the line can be described as the sum of a point on the line and the scalar multiple of $\mathbf{b}_2$, i.e. any point $\mathbf{p}$ on the line can be described as $\mathbf{p} = \mathbf{p}_0 + \alpha \mathbf{b}_2$.

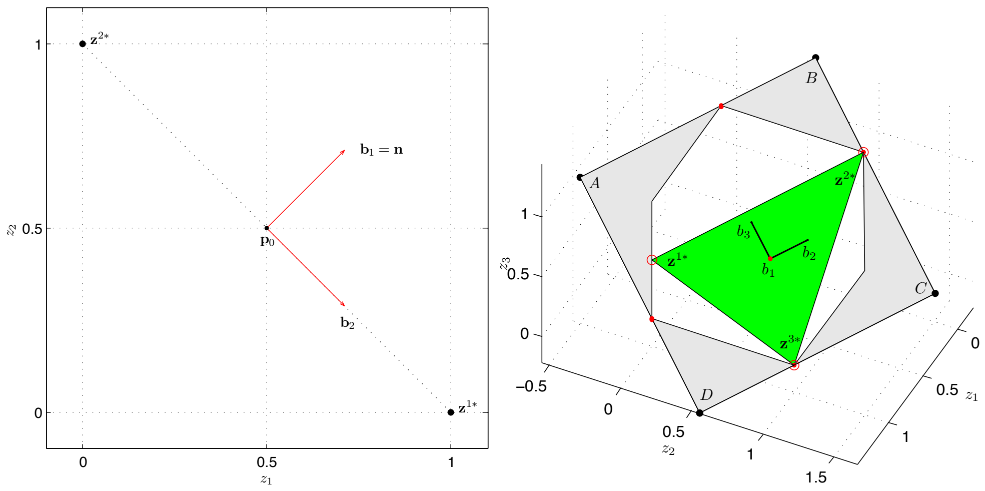


Figure 10: Two examples of orthonormal basis for hyperplanes in two and three dimensions. The right plot is shown in the normal direction to the plane. Note that the vector b_1 in the right plot is the normal to the plane and that b_2 and b_3 lies in the plane. The rectangle ABCD is a subset of the hyperplane. The gray areas are not useful because their projection will not be inside the hyper-cube which contains all Pareto optimal solutions.

For the m-dimensional case the choice of basis is straight forward, any point on the hyperplane can be desribed as

$$\mathbf{p} = \mathbf{p}_0 + \sum_{i=1}^{m} \alpha_i \mathbf{b}_i = \mathbf{p}_0 + \begin{bmatrix} \mathbf{b}_1 & \dots & \mathbf{b}_m \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \vdots \\ \alpha_m \end{bmatrix} \tag{8}$$

where $\mathbf{p}_0$ is a point on the plane and $\mathbf{b}_i, i = 2, \dots, m$, are linearly independent vectors lying in the plane and $\alpha_1 = 0$ since $\mathbf{b}_1$ is the normal to the hyperplane. Since the hyperplane is spanned by $\mathbf{b}_i, i = 2, \dots, m$ and the basis should be orthonormal this means that $\mathbf{b}_1$ must be normal to the plane (Lay, 2002).

To find the rest of the $m-1$ orthogonal basis vectors the Gram-Schmidt process is then applied:

$$\begin{aligned}
\mathbf{b}_1 &= n \\
\mathbf{b}_2 &= \mathbf{v}_2 \\
\mathbf{b}_3 &= \mathbf{v}_3 - \frac{\mathbf{v}_3 \cdot \mathbf{b}_1}{\mathbf{b}_1 \cdot \mathbf{b}_1}\mathbf{b}_1 - \frac{\mathbf{v}_3 \cdot \mathbf{b}_2}{\mathbf{b}_2 \cdot \mathbf{b}_2}\mathbf{b}_2 \\
&\vdots \\
\mathbf{b}_m &= \mathbf{v}_m - \frac{\mathbf{v}_m \cdot \mathbf{b}_1}{\mathbf{b}_1 \cdot \mathbf{b}_1}\mathbf{b}_1 - \frac{\mathbf{v}_m \cdot \mathbf{b}_2}{\mathbf{b}_2 \cdot \mathbf{b}_2}\mathbf{b}_2 - \dots - \frac{\mathbf{v}_m \cdot \mathbf{b}_{m-1}}{\mathbf{b}_{m-1} \cdot \mathbf{b}_{m-1}}\mathbf{b}_{m-1}
\end{aligned} \tag{9}$$

where $\mathbf{v}_2, \ldots, \mathbf{v}_m$ are given by

$$\begin{aligned}
\mathbf{v}_2 &= \bar{f}_1^*(\mathbf{x}) - \bar{f}_2^*(\mathbf{x}) \\
\mathbf{v}_3 &= \bar{f}_1^*(\mathbf{x}) - \bar{f}_3^*(\mathbf{x}) \\
&\vdots \\
\mathbf{v}_m &= \bar{f}_1^*(\mathbf{x}) - \bar{f}_m^*(\mathbf{x}).
\end{aligned} \tag{10}$$

Each vector is then normalized to get the orthonormal basis and $\mathbf{p}_0$ is chosen as the mean of the individual minima, i.e. $\mathbf{m} = \frac{1}{m}\sum_{i=1}^{m} \bar{f}_i^*(\mathbf{x})$. The right plot in Figure 10 shows the basis vectors together with the convex hull of the individual minima. Note that the basis vectors are scaled to better fit the plot.

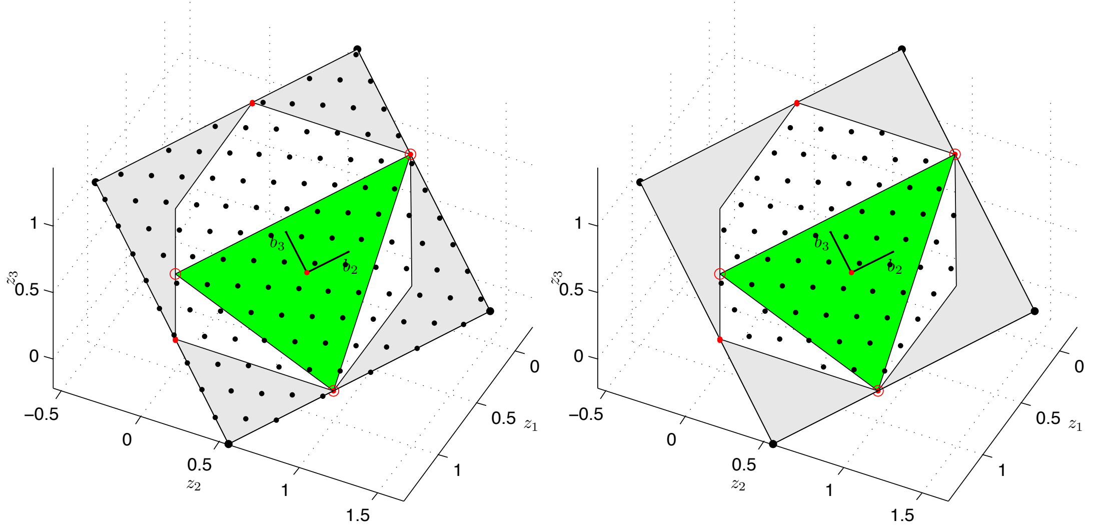


Figure 11: Left: The reference points created. Note that the distance between neighboring points are equal for all points. Right: Shows the remaining reference points after the usefulness filtration. Both plots are seen in the normal direction to the hyperplane spanned by the individual minima. The gray areas are not useful because their projection will not be inside the hyper-cube which contains all Pareto optimal solutions.

Before the reference points can be created the size of the reference point field must be calculated to guarantee to cover the entire Pareto frontier. A number of optimizations are performed to obtain the boundaries of the hyper-cube when projected to the hyperplane spanned by the basis $\mathbf{b}$. These optimization problems are a reformulation of the optimization problems described in Messac et al. (2003). The lower bounds, $\alpha_j^l, j = 2, \ldots, m$, on the weights in (8) are acquired by solving the problem

$$\begin{aligned}
\alpha_j^l = \underset{\mu,\mathbf{p}}{\text{minimize}} \quad & \alpha_j, \quad j = 2, \ldots, m \\
\text{s.t.} \quad & \mathbf{b}_i \cdot (\mu - \mathbf{p}) = 0, \quad i = 2, \ldots, m \\
& \mathbf{p} = \mathbf{m} + \sum_{k=2}^{m} \alpha_k \mathbf{b}_k \\
& 0 \leq \mu \leq 1, \quad \alpha_k \in \mathbb{R},\ k = 2, \ldots, m, \quad \mathbf{p} \in \mathbb{R}^m
\end{aligned} \tag{11}$$

where $\mathbf{m}$ is the mean of the individual minimas, $\alpha_k,\ k = 2, \ldots, m$, are the weights in (8), $\mu \in \mathbb{R}^m$ is a point in the hypercube and $\mathbf{p} \in \mathbb{R}^m$ is a point on the hyperplane. Note that the first equality constraint demands that the projection of the point $\mathbf{p}$ into the Pareto hypercube should be orthogonal to the reference hyperplane. In the same way the upper bound, $\alpha_j^u, j = 2, \dots, m$, of the weights can be found by performing (11) as a maximization instead of a minimization. Figure 10 shows an example in three dimensions where the rectangle ABCD is the result of the optimization.

A set of points is created by incrementing $\alpha_j, j = 2, \dots, m$ with a fixed length between the lower and upper bound. The left plot in Figure 11 shows the hyperplane for a set of reference points created in $\mathbb{R}^3$ shown in the normal direction. With an appropriate choice of number of steps and steplength, an equidistant sampling is acquired which can be seen in the both plots in Figure 11.

However, this method creates a number of points which lie outside the projection of the hyper-cube onto the hyperplane. This means that they cannot yield Pareto optimal solutions. The solution is to check the usefulness of every reference point. A reference point is useful if its projection intersects the Pareto hypercube and this is checked by solving

$$\begin{aligned} \underset{\mu}{\text{minimize}} \quad & 1 \\ \text{s.t.} \quad & \mathbf{b}_i(\mu - \mathbf{p}) = 0, \quad i = 2, \dots, m \\ & 0 \leq \mu \leq 1 \end{aligned} \tag{12}$$

where $\mu \in \mathbb{R}^m$ is a point in the hypercube and $\mathbf{p} \in \mathbb{R}^m$ is a point on the hyperplane. This gives a well distributed set of starting points which can give Pareto optimal solutions that are inside the hyper-cube with the direction of search perpendicular to the reference point hyperplane. The right plot in Figure 11 shows an example where the usefullness optimization has been performed on the reference points seen in the left plot.

3.3.2 Sequence of optimization

From a trade-off point of view the most interesting point of the Pareto set is the ideal point and points close to it. It is therefore beneficial to start with a reference point which is close to the projection of the ideal point on the reference points hyperplane and gradually move away from it.

When the number of variables and the complexity of the problem grows the initial starting guess is very important. One way to solve this is to use the solution from the previous optimization as initial guess for the next optimization as suggested by Das and Dennis (1998). With this in mind, it is important to use the reference points in a certain order to reach a solution correct and in some cases faster than without an initial guess.

The proposed solution used in this thesis is to use a span tree method. A start point is chosen and its nearest neighbors are found. Then the nearest neighbors of these are found, etc. This creates a number of branches which all spans from the original point. The left plot in Figure 12 shows the pattern for a small number of reference points. The point in the middle marked with a small circle is the point closest to the projection of the ideal point onto the reference hyperplane and is used as the first point to optimize.

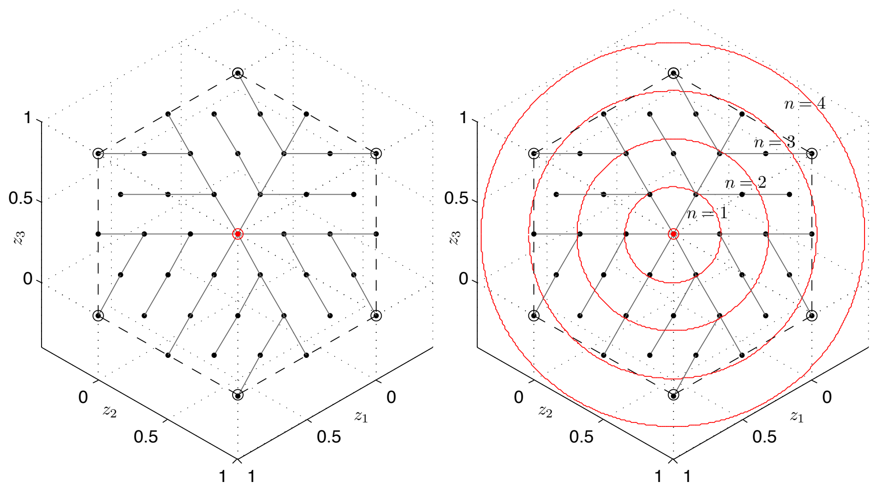


Figure 12: Left: Shows the connections created between the points in the set of reference points using the spanning tree method. The point in the middle was used as the starting point of the tree. Right: Shows the ordering sequence where level n has to be finished before level $n+1$. Note that the dashed line is the normal projection of the hypercube onto the reference hyperplane.

Since the points nearest to the projection of the ideal point are of most interest, the points are arranged in different levels. Each level is further away from the starting point. All points in level n are completed before the points in level $n+1$. The right plot in Figure 12 shows which points that belong to each level. Note that the circles only show which points that are included in each level and is not a part of the algorithm.

The span tree method is easy to implement, an arbitrary start point can be chosen and it is possible to generalize it to n dimensions. The big drawback is that all solutions in level $n-1$ have to be stored in memory until all points in level n are completed. This means that for large scale problems a large amount of memory might be needed.

3.3.3 Sampling the Pareto frontier

The set of reference points and direction created in the previous section is used to sample the Pareto frontier. As mentioned in Section 3.2 the direction method solves the problem

$$\begin{aligned} &\underset{\mathbf{x},\alpha,\mathbf{q}}{\text{maximize}}\ \alpha \\ &\text{subject to } \mathbf{x} \in X \\ &\qquad\qquad \bar{\mathbf{f}}(\mathbf{x}) = \alpha\mathbf{d} + \mathbf{z}^{R_i} - \mathbf{q} \end{aligned} \tag{13}$$

where $\bar{\mathbf{f}}(\mathbf{x})$ is the normalized objective vector according to Section 2.5, $\mathbf{z}^{R_i}$ is a reference point in the set of reference points created in the previous section, $\mathbf{d}$ is the direction and $\mathbf{q}$ is the slack vector.

Problem (13) is solved for $i = 1, \dots, p$, where p is the number of reference points in the set created in the previous section. The direction $\mathbf{d}$ is chosen equal to the negative normal of the reference hyperplane. To get equidistant sampling the slack vector is set to the zero vector since a solution on the search line is wanted.

This method is straight forward, easy to implement and understand. However, since it essentially is the NBI method it has one major drawback, it can produce non-Pareto optimal solutions (Marler and Arora, 2004). A solution to this problems will be be presented in the following section. Note that these problems also exists in the NC method (Messac et al., 2003).

3.3.4 Pareto filter

To deal with the fact that the NBI method might produce non-Pareto optimal solutions a *Pareto filter* is used. Messac et al. (2003) introduced a Pareto filter to complement the NC method. The filter is based on the definition of Pareto optimality and basically checks each point against all others to see if it is dominated.

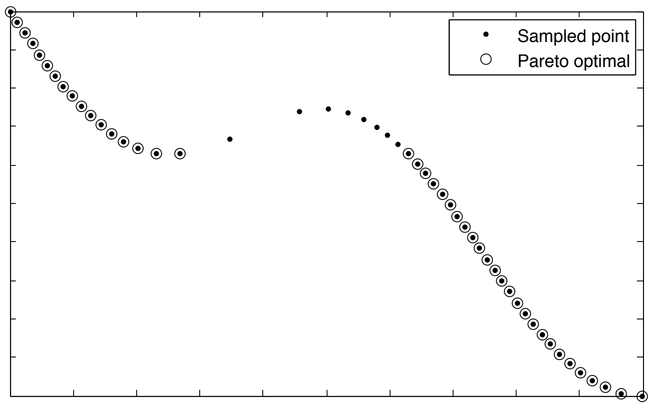


Figure 13: A two-dimensional Pareto set where the Pareto filter has been applied. Filtered points are marked with circles and are Pareto optimal.

The output from the filter is a set of globally Pareto optimal points. Figure 13 shows one example where the Pareto filter has been applied. Note that the original set consists of non-Pareto-, local Pareto- and global Pareto optimal points.

3.4 Interactive navigation in the Pareto set

The main objective of the interactive navigation part of the framework is to make it possible for the decision maker to smoothly move around on the Pareto frontier and search for a suitable solution. This is an iterative process where changes made by the decision maker triggers an optimization that results in a new solution that the decision maker can react on and possibly make new changes. This is an online step where the interaction between the application and the decision maker is high. Compared to the Pareto set generation it is however crucial that the iteration time is low so that the decision maker smoothly can maneuver on the Pareto frontier. This is possible when the optimization problem is small and preferably linear.

Several different interactive methods have been developed, e.g. Pareto Race by Kohronen (Miettinen, 1998) and Pareto Navigation by Monz (2006) but they mostly consider convex or even linear MOO problems. Methods used in the interactive part of this thesis have been inspired much by the latter.

In this framework the idea is that the decision maker can select the value and change the range of one or several objectives and/or decision variables. The solution is then visualized together with the feasible areas which the decision maker can navigate to from the current state. In this way the decision maker can investigate the Pareto frontier by changing values back and forth. The decision maker can control three parameters:

- Set a preferred value of an objective or a decision variable.
- Change the upper or lower bounds of the objectives or the decision variables.
- Fix objectives or decision variables to their respective current value.

Even if there are only a few parameters for the decision maker to control, there are several methods used internally. Next section will make an overall description of how the complete navigation step works and then the theory of every internal function is described thoroughly.

3.4.1 General idea of interactive navigation

The foundation of navigating on the Pareto frontier is to use a set of already approximated Pareto optimal points and then do a linear approximation of these to simplify the optimization problem and hence make it possible to optimize several times per second.

At first all the original Pareto points are in a feasible set. If the boundaries of the objectives or the decision variables are changed by the decision maker some of the points in this set are inactivated and hence excluded from the feasible set. The decision maker moves around on the Pareto frontier by selecting one or several preferred values in the objective space and/or decision space. This can be done continuously by using the linear combination of the set. However, this only works for convex Pareto sets and to solve this for non-convex Pareto set the feasible set is decomposed into several smaller convex sets. Whenever a desired level is changed by the decision maker all the convex sets are optimized one by one using the direction method. The best solution is then visualized to the decision maker.

3.4.2 Feasible points

For every objective there is a lower and an upper boundary z_i^l and z_i^u, $i = 1, \ldots, m$ and similarly for every decision variable x_k^l and x_k^u, $k = 1, \ldots, n$. These boundaries can be changed by the decision maker during navigation. The points that are outside any of the boundaries, are simply excluded from the set of feasible points Y and hence the set is constructed as

$$Y = \{(\mathbf{z} \in \mathbb{R}^m, \mathbf{x} \in \mathbb{R}^n) \mid \mathbf{z} \in Z_f, \mathbf{x} \in X_f\} \tag{14}$$

where

$$Z_f = \{\mathbf{z} \in \mathbb{R}^m \mid z_i^l \leq z_i \leq z_i^u,\ i = 1, \ldots, m\} \tag{15}$$

and

$$X_f = \{\mathbf{x} \in \mathbb{R}^n \mid x_k^l \leq x_k \leq x_k^u,\ k = 1, \ldots, n\}. \tag{16}$$

An example where the boundaries have been changed can be seen in Figure 14.

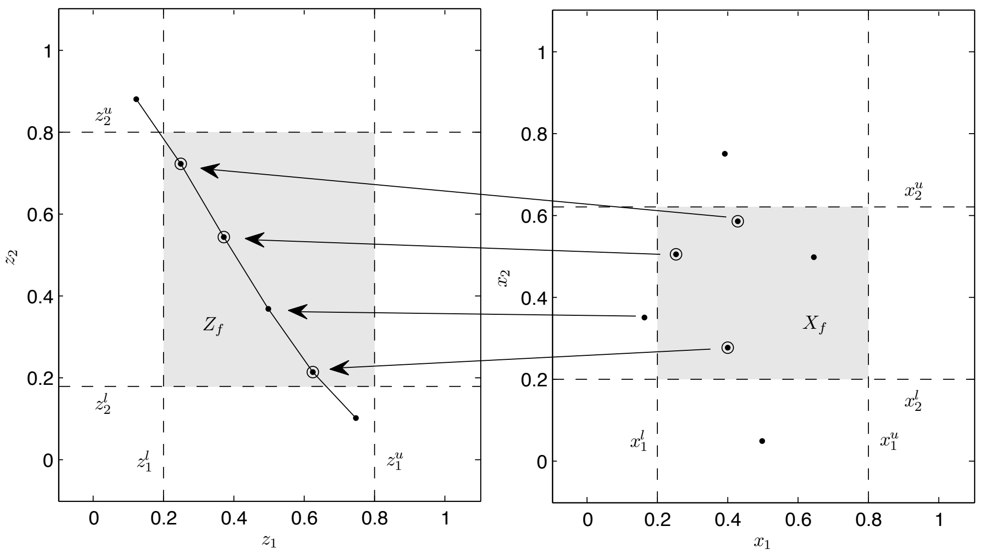


Figure 14: An example of feasible and infeasible points with set upper and lower boundaries of objective- $\mathbf{z}$ and decision space $\mathbf{x}$. Feasible points are marked with a circle. Note that the points have to be in both Z_f and X_f to be feasible.

3.4.3 Linear approximation of Pareto set

One simple method to approximate a set of points is to use the convex combination. This can be done in two levels with different accuracy of approximation as described by Monz et al. (2008). The first level is to do a linear approximation in the decision space and use the convex hull of the decision vectors $\mathbf{x}$ from the Pareto set and then approximate the objective vector as $\mathbf{f}(\mathbf{x}) \approx \mathbf{f}(\tilde{\mathbf{x}})$ where

$$\tilde{\mathbf{x}} = \sum_{i=1}^{p} \gamma_i \mathbf{x}^i, \tag{17}$$

where γ_i are the coefficients, $\sum_{i=1}^{p} \gamma_i = 1$ and p is the number of decision vectors.

The second level is to directly approximate in objective space and create the convex hull of the discrete set of objective vectors $\mathbf{y_k}$ from the Pareto set

$$\mathbf{f}(\mathbf{x}) \approx \sum_{i=1}^{p} \gamma_i \mathbf{y}^i. \tag{18}$$

It is important to note that the first level requires function evaluations where the functions might be nonlinear. For a convex Pareto frontier the approximations will always be above the true Pareto set and the second level of approximation will be above the first level (Monz, 2006).

3.4.4 Fix an objective or decision variable to a preferred value

This step is the actual navigation where the method tries to find a solution on the Pareto frontier that satisfies all objective- and decision variable-values that the decision maker prefers. A method to accomplish this is described by Monz (2006). The direction method described in Section 3.2 and (7) is used with the difference of fixing one or several objectives and decision variables to some specified values τ and μ respectively. The optimization problem then becomes

$$\begin{aligned} \underset{\mathbf{x},\alpha,\mathbf{q}}{\text{maximize}}\ & \alpha \\ \text{subject to}\ & z_i^R + \alpha d_i = f_i(\mathbf{x}) + q_i, \quad i \in \{1,\dots,m\} \setminus F_z \\ & f_j(\mathbf{x}) = \tau_j, \quad j \in F_z \\ & x_l = \mu_l, \quad l \in F_x \\ & q_i \geq 0, \quad i \in \{1,\dots,m\} \setminus F_z \end{aligned} \tag{19}$$

where $\mathbf{z}^R$ is the reference point and is set to the previous solution with $z_j = \tau_j, j \in F_z$, $\mathbf{q}$ is an additional slack and $\mathbf{d}$ is the search direction. The parameters τ_j and μ_l are the fixed values in the sets F_z and F_x which are fixed objectives and decision variables respectively. Note that F_z or F_x can be empty.

The search direction $\mathbf{d}$ controls the ratio of improvement/deterioration between the objectives except from $f_j(\mathbf{x}), j \in F_z$. In this framework all objectives should be improved/deteriorated equally much so the direction is set to $\mathbf{d} = [-1,\dots,-1]^T$.

An additional slack is added to always find a solution in case of no intersection with the Pareto front is found. This may occur if for instance the desired value is close to an edge or corner of the set.

An illustration of one search from the point $\mathbf{z}^{\text{old}}$ with one fixed objective τ can be seen in Figure 15. Note that as a result of setting the desired value, the objective space is sliced by a hyperplane, and hence the optimization problem is reduced with one dimension. In this case it turns into an optimization in only one dimension. The solution of the optimization problem is a weakly Pareto optimal point that fulfills the desired fixed values.

To be able to solve this optimization for large scale MOO problems in realtime, the set of points obtained from the offline part must be used. This is accomplished by using the linear approximation of the Pareto set. The best solution would be to use the first level of approximation, i.e. (17). However, for a large set of points the nonlinear

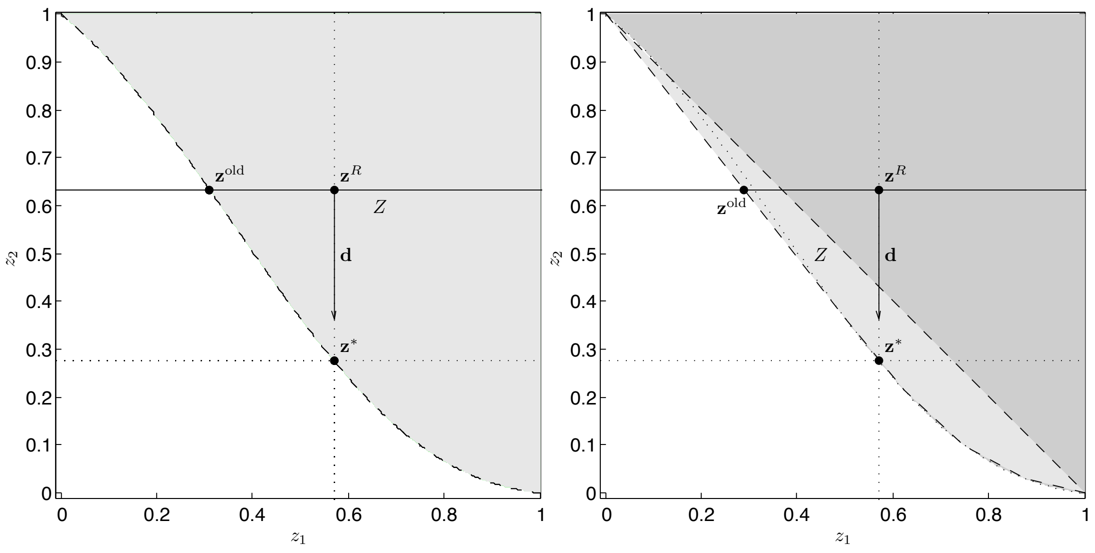


Figure 15: One optimization in the objective space with a desired objective value τ. The dashed line is the Pareto optimal set, $\mathbf{z}^{\text{old}}$ is the previous solution and $\mathbf{z}^*$ is the solution found. Note that $\mathbf{z}^R$ is the reference point used in the direction method obtained from $\mathbf{z}^{\text{old}}$ and τ. In a two-dimensional problem the optimization is in only one dimension and in this case only z_2 is optimized since the z_1 is fixed.

optimization problem would take too much time to solve and the software would not run smoothly. Instead the second level of linear approximation of the Pareto set is used. By using (18) the the optimization problem turns into a linear program which can be solved fast. The problem can be rewritten as

$$\begin{aligned}
&\underset{\mathbf{x},\alpha,\mathbf{q}}{\text{maximize}}\ \alpha \\
&\text{subject to } z_i^R + \alpha d_i = \sum_{k=1}^{p} \gamma_k y_i^k + q_i, \quad i \in \{1,\ldots,m\} \setminus F_z \\
&\qquad \sum_{k=1}^{p} \gamma_k y_j^k = \tau_j, \quad j \in F_z \\
&\qquad \sum_{k=1}^{p} \gamma_k x_l^k = \mu_l, \quad l \in F_x \\
&\qquad \sum_{k=1}^{p} \gamma_k = 1 \\
&\qquad q_i \geq 0,\ i \in \{1,\ldots,m\} \setminus F_z, \quad \gamma_k \geq 0,\ k = 1,\ldots,p
\end{aligned} \tag{20}$$

where γ_i are the linear approximation coefficients. The sets F_z and F_x are the fixed objectives and decision variables respectively. Note that this is a linear optimization problem.

3.4.5 Convex decomposition

To be able to continuosly navigate in a set of points, the convex combinations of the points can be used. One idea, which is used by Monz (2006), is to make a linear approximation of the complete set of feasible points and then find an optimal solution in this convex hull. This method however requires the set to be convex and cannot handle disconnected sets or holes that can emerge if for example a point somewhere in the middle of the set is infeasible which can be seen in the left plot in Figure 16. The convexity problem can be solved by decomposing the set of feasible points into a few smaller convex sets and then optimize on every set one by one and in the end choose the best solution from these. It is, however, difficult to decompose sets that are in more than two dimensions. An example of a decomposition in convex sets can be seen in the middle plot in Figure 16. The idea in this thesis is to use the extreme of the last method and decompose all points into polytopes where one polytope is connected to only m points where m is the number of dimensions of the MOO-problem which can be seen in the right plot in Figure 16.

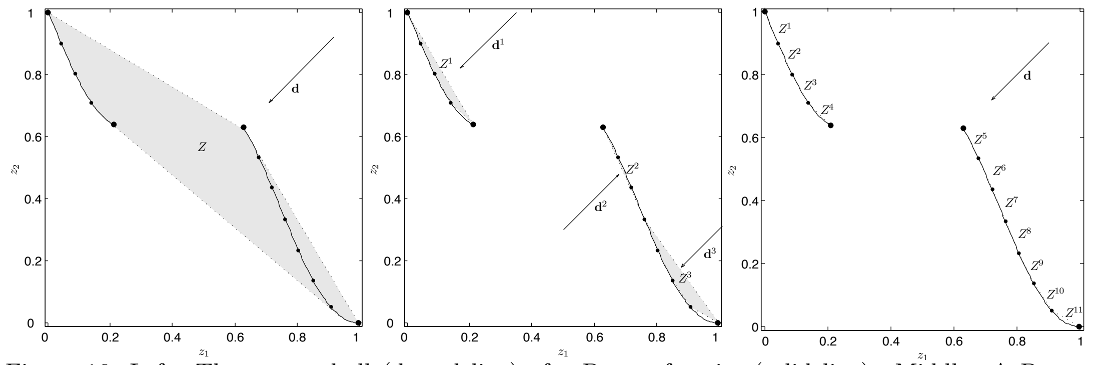


Figure 16: Left: The convex hull (dotted line) of a Pareto frontier (solid line). Middle: A Pareto frontier (solid line) decomposed into convex sets (dotted line). Right: A Pareto frontier (solid line) decomposed into a set of polytopes (dotted lines). Note that there is no polytope covering the hole in the frontier.

One polytope is called a d-simplex in the d-dimensional space. Note that for a d-simplex the number of vertices is $d+1$. This means that in one dimension a simplex is represented by a line, in two dimensions a triangle and in three dimensions a tetrahedron (Grunbaum and Shephard, 1969). An illustration of 1-simplex to 3-simplex can be seen in the left plot in Figure 17. Let $\mathbf{v} = \{\mathbf{z} \in \mathbb{R}^m, \mathbf{x} \in \mathbb{R}^n\}$ be one vertex in a simplex. Then the set of vertices in one m-simplex can be described as:

$$\mathbf{t} = \{\mathbf{v}_1, \ldots, \mathbf{v}_m\} \tag{21}$$

where m is the dimension in the objective space. An illustration of the vertices for one 2-simplex can be seen in the right plot in Figure 17.

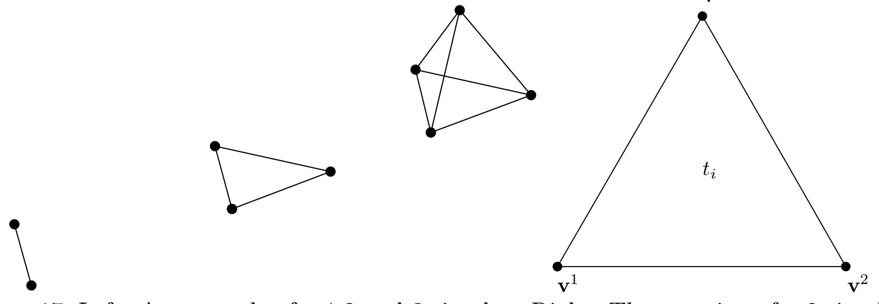


Figure 17: Left: An example of a 1,2 and 3-simplex. Right: The notation of a 2-simplex.

It is important to note that the Pareto frontier is a hypersurface in the m-dimensional space and therefore the decomposition should be represented in the $m-1$ dimensional space. For example, in a three dimensional problem the surface can be decomposed in the two dimensional space with 2-simplices, i.e. triangles with three vertices. To accomplish this the Pareto frontier must be projected to some hyperplane with one dimension less. In this framework the reference points used when obtaining the Pareto set are already a suitable projection of the frontier and hence it can be used directly for the decomposition. The left plot in Figure 18 shows an example of a decomposition of a set of points when looking in the normal direction of the reference point plane.

There are several methods to decompose a set of points to simplices but one method that works for p dimensions is the p-Delaunay triangulation. This method creates p-simplices such that no p-simplices intersects another and the minimum angle of all p-simplicies are maximized. How this method works is out of the scope for this thesis but see de Berg et al. (2000) for a description of how the decomposition is performed. All decompositions in Figure 18 was performed with this method.

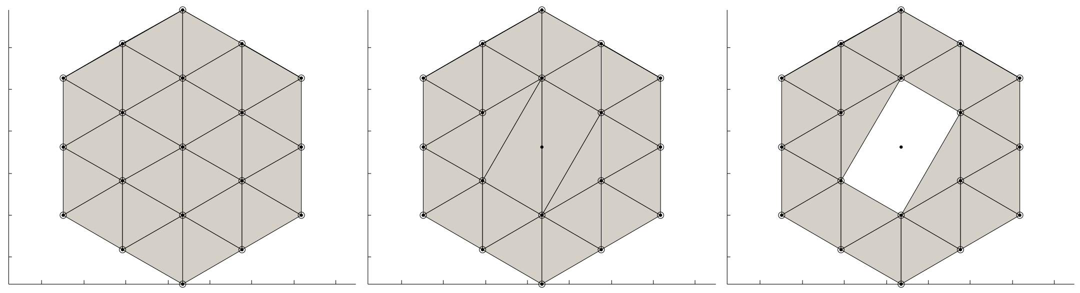


Figure 18: Three sets with decomposed convex sets on a two-dimensional hyperplane in three dimensions. Left: Shows a decomposition when all points in the set are feasible. Middel: Shows a decomposition when a point in the middle is infeasible. Right: A filtered decomposition of the middle plot. Note that the circles are feasible points in the set.

For a full set of feasible points, the Delaunay triangulation makes an appropriate decomposition of the set, seen in the left plot in Figure 18. However, if some points are not included, e.g. some points are infeasible, the Delanunay triangulation will still cover the complete set with the difference that some simplices have become larger. An example of this can be seen in the middle plot in Figure 18 where one point in the middle is infeasible. To solve this the decomposed set is filtered by checking the size of every simplex. There are several ways to filter the set but one method is to check the length of the edges and exclude the simplices where one or several of the edges are too long.

If the points are equidistant distributed, the general simplex have equidistant edges which can be seen in the left plot in Figure 19. The height of the simplex where the length of every edge is d is calculated as

$$h = \frac{\sqrt{3}}{2}d \tag{22}$$

by using Pythagoras theorem. If the decomposition made by Delaunay triangulation becomes as in the right plot in Figure 19 the two larger simplices 2 and 3 should not be included in the decomposition since they cover the point which is known to be infeasible. However, the two simplices 1 and 4 can still be a good approximation of the Pareto frontier and cover feasible areas and hence should be included. The longest edge of simplex 1 and 4 is $2h$ of the general simplex, calculated by using Pythagoras theorem. An appropriate way to filter the set of simplices is then to exclude simplices with edges longer than $2h$. Note that it would be desirable to also have simplices covering the upper en lower part of the ”hole” so that the hole is only a small hexagon. This is, however, not possible when using Delaunay since they would intersect each other.

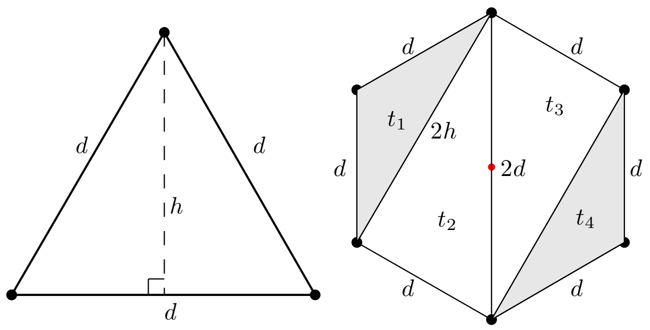


Figure 19: Left: The height h of an equilateral triangle where the length of every edge is d. Right: A decomposition of a set of points with one infeasible point in the middle. Simplex 2 and 3 should be excluded since they covering an infeasible point while simplex 1 and 4 can still be a good approximation. The longest edge of simplex 1 and 4 is $2h$ of the equilateral simplex in the left plot.

All filtered simplices created by the decomposition of the feasible points in Y are in the set $T = \{t_1, \dots, t_k\}$ where k is the number of simplices. An illustration of the set T can be seen in the right plot in Figure 20.

3.4.6 Approximated solution

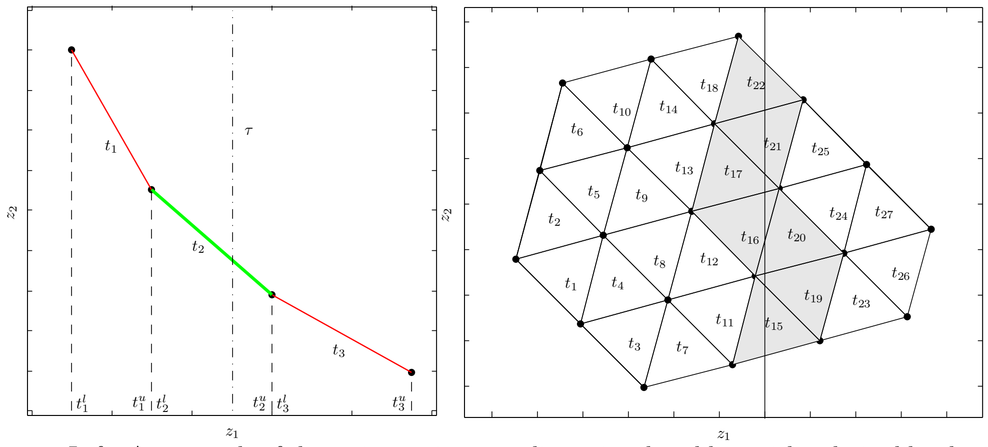
When the decision maker sets a desired level in the objective space or in decision space, each filtered simplex $t_i \in T$ is optimized one by one using (20) with $P \in \mathbf{v}_j$ and $\mathbf{v}_j \in t_i$ is the set of points in simplex t_i.

The simplex with the best solution, i.e. the largest objective value α, from optimizing the set T is chosen as the best solution of the optimization at the desired levels. An illustration of an optimization in a two dimensional problem with one objective set to τ can be seen in the left plot in Figure 20. In this example $\mathbf{z}^{1*}$ is attained by using slack but has a smaller value than $\mathbf{z}^{2*}$ and that $\mathbf{z}^{3*}$ has no solution since the slack must be positive. In this case simplex t_2 has the best solution $\mathbf{z}^{2*}$. Note that when a value in objective space has been changed and a solution is found on the Pareto frontier there is only one simplex that has no slack and is actually on the line of search. This is the optimal solution.

When one or several values are fixed in objective space and/or in decision space some simplices will probably be infeasible because they are simply not covering the desired levels. An example of this can be seen in Figure 20 where z_1 has been fixed to τ. In this case it can be seen that t_2 is the only simplex that can have a solution without slack at τ. This shows that it is not necessary to optimize the simplices that do not cover the desired levels since no solution will be found in these. The subset T_A, called the active set, is defined so that

$$\begin{aligned} \min(z_i^p) \le \tau_i \le \max(z_i^p), i \in F_z, p \in v_k \\ \min(x_j^p) \le \mu_j \le \max(x_j^p), j \in F_x, p \in v_k \end{aligned} \tag{23}$$

where $v_k, k = 1, \ldots, m$ is the vertices of simplex $t_l \in T$. The subset T_A is all the simplices which cover all the fixed levels in both decision space and objective space. In the left plot in Figure 20 the dashed lines shows the boundaries of every simplex and it can be seen that t_2 is the only simplex with τ inside the boundaries and hence it is the only simplex that needs to be optimized. An example of an active set in a three-dimensional problem can be seen in Figure 20. As can be seen in the figure this subset is smaller than the original set and in some cases heavily reduces the amount of optimizations needed.



t_1
t_2
t_3
τ
z_1
z_2
t_1^l
t_1^u
t_2^l
t_2^u
t_3^l
t_3^u
t_1 t_2 t_3 t_4 t_5 t_6 t_7 t_8 t_9 t_{10} t_{11} t_{12} t_{13} t_{14} t_{15} t_{16} t_{17} t_{18} t_{19} t_{20} t_{21} t_{22} t_{23} t_{24} t_{25} t_{26} t_{27}

Figure 20: Left: An example of the active set in a two dimensional problem with a desired level τ in z_2. The dashed lines shows the boundaries for every simplex. t_2 is the only simplex which is covering τ. An example of a three dimensional problem. The active set is seen as gray simplicies. The boundaries are not plotted for clarity.

By using the convex combination coefficients γ^* from the solution on the points in the best simplex an approximated value in objective space and decision space is calculated as

$$\mathbf{z} = \sum_{k=1}^{p} \gamma_k^* \mathbf{y}^k \tag{24}$$

where $\mathbf{y}^k$ are the objective vectors that are included in the best simplex t^* and p is the number of vertices in the simplex. The same works for decision space as

$$\mathbf{x} = \sum_{k=1}^{p} \gamma_k^* \mathbf{x}^k \tag{25}$$

where $\mathbf{x}^k$ are the decision vectors that are included in the best simplex t^*.

3.4.7 Find true Pareto optimal point

When the decision maker is satisfied with the solution or wants to check if the approximated solution is feasible it must be optimized in the original MOO problem. The approximated objective vector is used as a reference point in (7) and the approximated decision vector is used as an initial guess to the optimizer. If the optimization is successful the solution is presented to the decision maker.

3.4.8 Complement the obtained Pareto set

The idea of using an online step to navigate on an approximated Pareto frontier is to make a sparse sampling of the frontier and then use the linear approximation of the set. However, if the decision maker finds an area of the Pareto frontier which is of more interest than others, it would be beneficial to improve the Pareto set in that area. This can be done by specifying the area of interest and then iterating back to the offline step and do a new Pareto set. The new points can then be added to the already existing Pareto set and the decision maker can continue to navigate.

3.5 Implementation

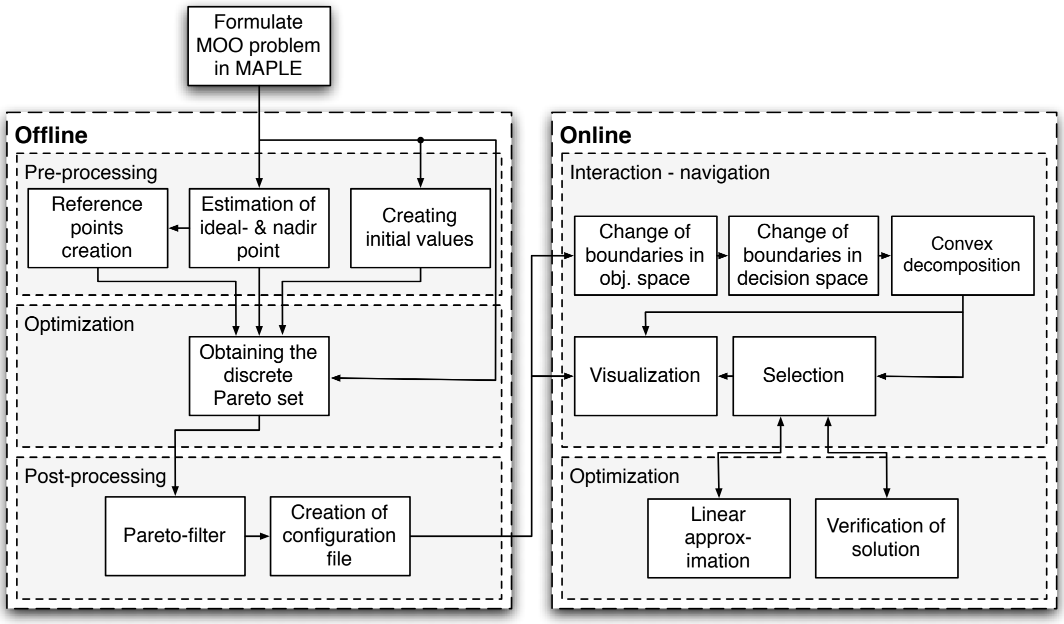

Formulate MOO problem in MAPLE
Offline
Pre-processing
Reference points creation
Estimation of ideal- & nadir point
Creating initial values
Optimization
Obtaining the discrete Pareto set
Post-processing
Pareto-filter
Creation of configuration file
Online
Interaction - navigation
Change of boundaries in obj. space
Change of boundaries in decision space
Convex decomposition
Visualization
Selection
Optimization
Linear approx-imation
Verification of solution

Figure 21: Overview of the framework. The offline block is explained in Section 3.3 and the Online block is explained in Section 3.4.

The implementation is divided in three part as mentioned in Section 3. The first part is the MAPLE/IPOPT-tool. All variables, constraints and objective functions of the optimization problem are entered in a MAPLE input file as symbolic equations. In the offline application it is possible to change settings of how the Pareto frontier should be obtained and then plot the resulting Pareto set. The set is exported to the online application which has a GUI that the decision maker can use to investigate the Pareto set and select a preferred final solution.

3.5.1 Offline

When the MOO problem has been setup in the MAPLE-tool it is parsed into C++ code with an interface to the IPOPT-package. IPOPT is an interior point optimizer for large scale nonlinear SOO problems. The interface between MAPLE and C++ was given by ABB Corporate Research and has only been changed to add features to optimize several points in one run. The C++ code is then compiled into an executable file that can be used to obtain the Pareto set.

The offline application has mostly been developed in MATLAB. MATLAB is used to call both the MAPLE-files and the executable files. Several MATLAB functions have been implemented to create the reference points, set up all files necessary for the executable file, filter the Pareto set, visualize the final Pareto set and export points to the online application.

The decision maker can control several settings how the optimization should be performed when obtaining the Pareto set such as the number of samples of the Pareto set and what kind of initialization features should be used. It is also possible to control what should be visualized and which variables in the decision space that are of interest to export to the online application. For example there might be variables that are internal in the original MOO problem. These are often not interesting to visualize and/or control and can be omitted in the navigation. A description of the exported variables can also be set such as a name and the physical unit of the variable.

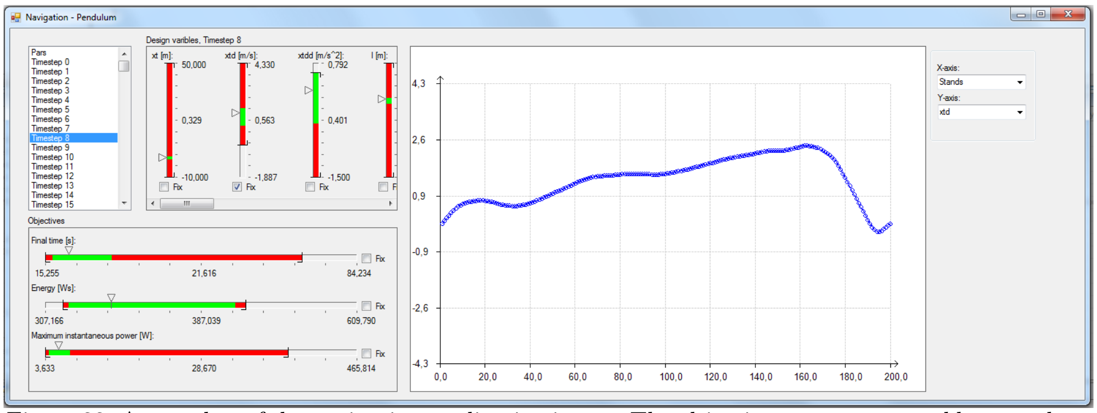
Before the Pareto set can be obtained the ideal points must be found. In the application it is possible to specify that a single objective should be optimized. The solutions are then saved so that the problem can be scaled and the reference points can be created. In the beginning of the procedure there are no previous solution at hand that can be used as an intialguess. It is therefore possible for the decision maker to create an initial guess by calculating appropriate values for some of the variables by using information about the model. For example an initial guess of the trajectory of some movement can be made which the decision maker expects to be close to the optimal trajectory.

The IPOPT-package has built-in functionality for initiating the solver to find a solution faster if much information about the problem is already known. There are two methods that have been implemented in the generic framework. The first is to use the built-in warmstart initialization which means that the values of the primal and dual variables from a previous optimization are used as a good initial guess for a new optimization. The other is to set the initial barrier parameter so that the search space is narrowed down from the beginning. As mentioned in Section 3.3.1 (Reference point creation) and 3.3.2 (Sequence of optimization), the reference points are created in a sequence so that there is high probability that the solutions are close to each other. In the implementation all primal- and dual variables are saved so that they can be used as initialization for the next point to solve. Note that even if the warmstart features are disabled, the primal variables can be used as an initial guess to solve the next point. There is also a "rerun" feature implemented in the framework that enables the possibility to run the optimization for a point again if the solver fails to find a solution when the warmstart features is enabled. In this case both warmstart features are automatically disabled in the next optimization to enlarge the search space.

3.5.2 Online

A GUI for the online step of the generic framework has been created. This has been implemented in C# with interaction to MATLAB and C++. The application imports a configuration file and all solutions in both objective space and decision space from the obtained Pareto set from the offline application. The configuration file is used to describe the size of the problem and the set of variables and decision variables that are of interest to control and/or visualize to the decision maker. It also includes the initial bounds and descriptive names of the objectives and decision variables.

In the application every objective and decision variable is represented by a range bar with a marker for the upper and lower bounds of the objective or variable and a marker for the current value. Figure 22 shows an example. All these markers can be changed by the decision maker and for every change an optimization is triggered. At startup the boundaries of the objectives are set to the ideal and nadir point in respective dimension. The boundaries of the decision variables are set to the same bounds as the variables in the original MOO-problem. This makes all attained points from the offline application feasible from the beginning.



X-axis: Stands

Y-axis: xtd

Figure 22: A snapshot of the navigation application in use. The objectives are represented by rangebars in the lower part of the window. A list of different sets of decision variables is shown the left. The selected set of decision variables are shown as rangebars in the middle. To the right there is a plotting area where solutions of objectives or decision variables can be visualized. Note the green and red fields of the rangebars which shows the feasible and infeasible regions to move to from the current solution.

One important feature when setting the level of one or several of the objectives or decision variables is to see the feasible regions of all the other objectives and decision variables. Especially when the problem is larger than three dimensions since the frontier cannot be visualized in an easy way. In the rangebars the feasible regions are visualized with green fields and infeasible regions with red fields which can be seen in Figure 22. This is done by using the set of simplices that are active, i.e. the set of simplices that intersects the hyperplanes of fixed values. The convex combination of the intersection is calculated for every simplex which are then used to create a set of ranges that are plotted in the rangebar. These fields show existing holes and disconnected sets in the Pareto frontier.

When the bounds of an objective or decision variable is changed some of the points in the Pareto set might be infeasible. In this case a new convex decomposition must be done since some polytopes will no longer be feasible. In Figure 18 it is possible to see that the set of polytopes has been recalculated because one point turned infeasible. If the set is not recalculated the hole will be larger. The convex decomposition is done in MATLAB for simplicity.

On every change of either a value or a boundary an optimization is executed. This is done with the linear solver CLP from Coin-OR which is written in C++. Every active simplex at the current fixed level is optimized one by one. The best solution is returned to the GUI and visualized on the rangebars. If no solution is found the rangebars are set to the previous value.

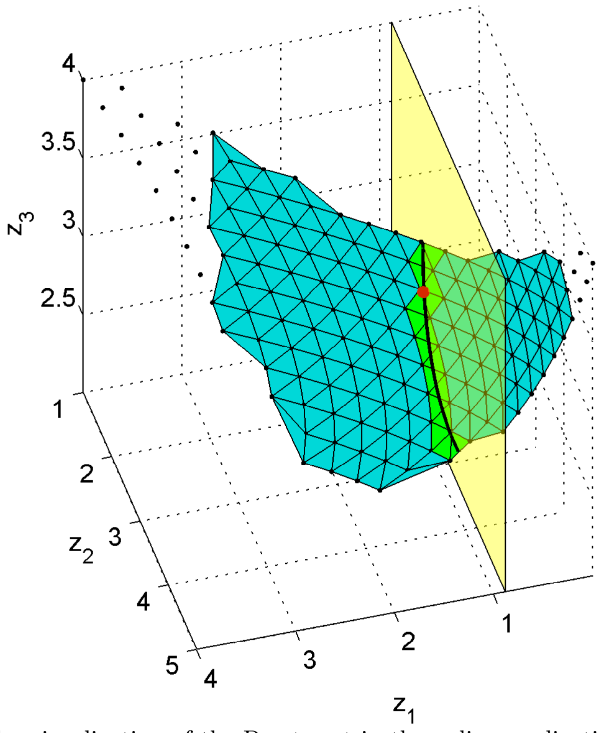


Figure 23: A plot of the visualization of the Pareto set in the online application. A desired level has been set on z_1 and both the upper and lower boundaries have been changed on z_1.

To the right in the GUI there is a two dimensional plot window. In this plot it is possible to show the values of decision variables at the current solution. The plot is updated whenever a new solution is found.

For problems with two or three dimensions it is possible to visualize the Pareto set when navigating in it. For simplicity the MATLAB plot function has been used instead of creating a plot function inside C#. A plot when navigating on a Pareto set can be seen in Figure 23. The lower and upper boundaries of z_1 have been changed so that some Pareto optimal points (black points) becomes infeasible (B) and the cyan colored area (A) are the feasible simplices. The yellow plane (C) shows the current set level on z_1 which intersects the Pareto set which can be seen as the thick black line. At this level the active simplices are shown in green (D) and the solution found by the optimization can be seen as a red point (E). When boundaries or desired levels have been changed and a solution has been found, the plot is updated.

4 Benchmarking problems

A set of smaller optimization problems with different properties have been used to test the functionality of the implemented framework and analyze the performance. The problems are mathematically simple and used in benchmarking of evolutionary MOO methods. However, these problems also works with traditional optimization tools. The shape of the Pareto front is known for these problems so the generation of the estimated Pareto front from the implemented framework can be evaluated.

4.1 Test problems with known Pareto frontiers

In this section, three MOO problems are described and evaluated. The two first of them have been created and analyzed by Yun et al. (2004) and Veldhuizen (1999) which have illustrations of the true Pareto frontiers. Test problem 3 is a modification of test problem 1 where an extra objective function has been added. In this case, the shape of the true Pareto front is unknown. The properties of the models can be seen in Table 1 and the mathematical description can be seen below:

Table 1: Properties of the test problems

	Dimensions	Variables	Shape	Other properties
Test problem 1	2	2	Non-convex	Disconnected
Test problem 2	3	2	Convex	
Test problem 3	3	2	Non-convex	Disconnected

Test problem 1 (Yun et al. (2004)):

$$\begin{aligned} z_1(\mathbf{x}) &= x_1 \\ z_2(\mathbf{x}) &= 1 + x_2^2 - x_1 - 0.2\text{sin}(3\pi x_1) \\ &\quad 0 \leq x_1 \leq 1, -2 \leq x_2 \leq 2 \end{aligned} \tag{26}$$

Test problem 2 (Veldhuizen (1999)):

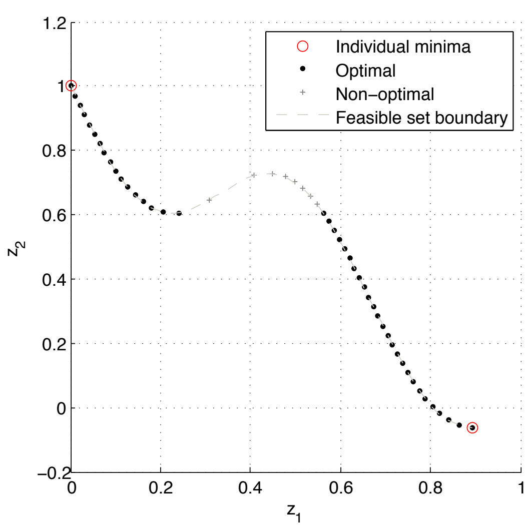
$$\begin{aligned} z_1(\mathbf{x}) &= x_1^2 + (x_2 - 1)^2 \\ z_2(\mathbf{x}) &= x_1^2 + (x_2 + 1)^2 + 1 \\ z_3(\mathbf{x}) &= (x_1 - 1)^2 + x_2^2 + 2 \\ &- 2 \leq x_1, x_2 \leq 2 \end{aligned} \tag{27}$$

Test problem 3 (Modified from Test problem 1):

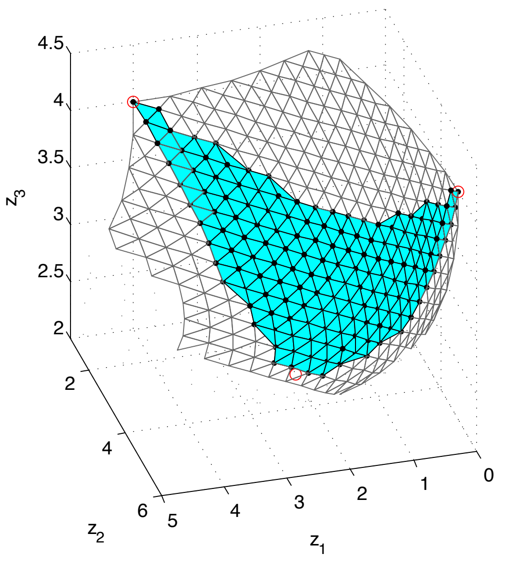
$$\begin{aligned} z_1(\mathbf{x}) &= x_1 \\ z_2(\mathbf{x}) &= 1 + x_2^2 - x_1 - 0.2\text{sin}(3\pi x_1) \\ z_3(\mathbf{x}) &= -x_2 \\ &\quad 0 \leq x_1 \leq 1, -2 \leq x_2 \leq 2 \end{aligned} \tag{28}$$

The three test problems were implemented in the framework and the Pareto frontiers were sampled with 51, 420 and 420 points respectively. The estimated Pareto frontiers of the problems can be seen in Figure 24.

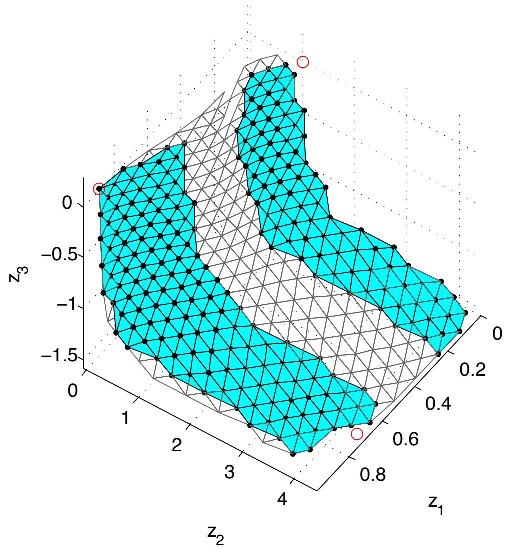
In Figure 24(a) the estimated Pareto optimal set (black markers) can be seen together with the boundary of the feasible set of the objective space. It can be seen that the



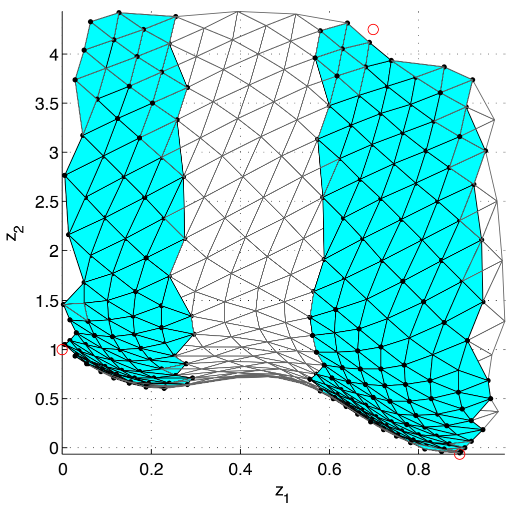
(a) Test problem 1



(b) Test problem 2



(c) Test problem 3



(d) Projection of test problem 3

Figure 24: Estimated Pareto frontiers of test problems

estimated Pareto set is a good approximation of the Pareto frontier and that the complete Pareto frontier is covered. Note that there are two disconnected sets since some points are not Pareto optimal (marked with gray crosses). This is due to the fact that between the two sets the first objective z_1 can be improved without deteriorating the second objective z_2.

In Figure 24(b) the estimated Pareto optimal set for test problem 2 can be seen. The method obtains the correct frontier and this test problem has been a great help while developing the framework.

In Figure 24(c) - 24(d) the Pareto optimal set for test problem 3 can be seen. Since this problem is an extension of test problem 1, the shape of the optimal set is not known. However, some verification can be performed. In Figure 24(d) the lower boundary of the frontier is equal to the boundary of the optimal set to test problem 1 seen in Figure 24(a) which is expected since test problem 3 is a extension of this problem.

4.2 A cart-pendulum problem

The well known optimal control problem of a cart-pendulum was analyzed. This problem has the benefits of being a real-world problem, well studied, nonlinear and scalable, i.e. it is possible to discretize the problem more densely to get a optimization problem with more optimization variables. The objective of the problem is to move the end-point of a pendulum from point A to B in a two dimensional space with limited variables, e.g. acceleration. Usually the only control signal for the cart-pendulum problem is to control a motor on the cart so that the pendulum can move back and forth. However, in this example the length of the pendulum can also be controlled. This modification was made to increase the degrees of freedoms in the system. An illustration of the setup can be seen in Figure 25.

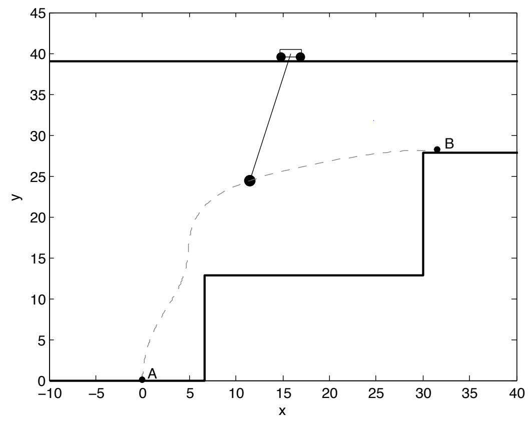


Figure 25: A plot of one solution of the cart-pendulum problem. The cart only runs sideways on the rail at the top of the plot. The end of the pendulum, marked as a large black marker, is supposed to move from point A to point B under certain conditions. The black lines can be seen as obstacles that the pendulum can not intersect. The dashed line shows the trajectory of the pendulum.

In this kind of problems there are several objective functions that can be of interest to optimize. However, in this thesis only three optimization objectives have been analyzed:

- f_1: Minimize the time of one cycle, i.e the time of moving the endpoint of the pendulum from point A to point B.

- f_2: Minimize the total energy used in one cycle.
- f_3: Minimize the maximum instantaneous power used by the system in one cycle.

The mathematical formulation of the objective functions is

$$\begin{aligned}
z_1 &= f_1(\mathbf{x}) = t_f \\
z_2 &= f_2(\mathbf{x}) = \sum_{i=1}^{N}(|P_c^i| + |P_p^i|) \\
z_3 &= f_3(\mathbf{x}) = \max(|P_c^1| + |P_p^1|, \dots, |P_c^N| + |P_p^N|) \\
&\qquad\quad \mathbf{x} \in \mathbf{X}.
\end{aligned} \tag{29}$$

Here, t_f is the total time of one cycle and P_c and P_p are the instantaneous power at every timestep when moving the cart and varying the length of the pendulum respectively. N is the number of time steps and $\mathbf{X}$ is the set of all decision variables.

The model of the pendulum was supplied by ABB Corporate Research as an example problem (Sjöberg et al., 2010). However, it should be noted that the model was discretized using the first order backward difference discretization where one cycle is divided into a set of timesteps. There are a set of variables for every timestep which describes the state of the model, for example acceleration, speed and position of the cart and the pendulum. The model used in this thesis had 43 optimization variables in each timestep where two of them are control signals (acceleration of the cart and length of the pendulum) and where some of the variables are internal to increase the speed of the optimization. Since there is a set of variables in each timestep the size of the MOO-problem can be scaled by varying the amount of timesteps for one cycle. In this thesis the amount of timesteps was set to 200 after suggestions from ABB which results in 8600 variables in the optimization problem.

4.2.1 Test case

The Pareto frontier of the cart-pendulum was sampled with 461 points by using the generic framework. All boundaries of the variables were set to reasonable values and were given by ABB Corporate Research. A three dimensional plot of the resulting Pareto set can be seen in Figure 26 and the projection of the Pareto set on every coordinate axis can be seen in Figure 27. The shape of the frontier can be described as a box with rounded edges and in the 3D plot one is looking from the inside of the box in the direction of one of the corners.

The black markers shows the Pareto optimal points that have been found with the Pareto filter and the cyan colored area is the convex decomposition of these points. There were 273 points that were Pareto optimal and the solver was not able to find a solution at all for 9 points. The gray lines shows non Pareto optimal surfaces. Note that some Pareto optimal points (black markers) are missing due to plotting issues. In every corner of all the colored faces there should a black marker.

The red circles are the individual minimas that were obtained by using SOO. It can be seen that two of them are close to each other which means that they are weakly conflicting. This can especially be noticed in the right plot in Figure 27 where z_2 and

z_3 are almost not conflicting at all. It is even so that when the time for one cycle is at the maximum value, in this case 100 s, the two objectives can almost reach their respective individual minimas at the same time (the point closest to origo).

In the left plot in Figure 27 the frontier is almost weakly Pareto optimal in the upper left area. This means that the energy consumption can be decreased and almost not deteriorate the time needed for one cycle. Actually, by increasing the total time with about 0.6 percent the energy consumption can be reduced with up to 35 percent. Since the energy consumption and the maximal instantaneous power are so closely connected, the relation between z_1 and z_3 which can be seen in the middle plot is quite similar. It is possible to reduce maximal instantaneous power with up to 58 percent and only increase the total time with 1.4 percent. The trajectories of the pendulum for the

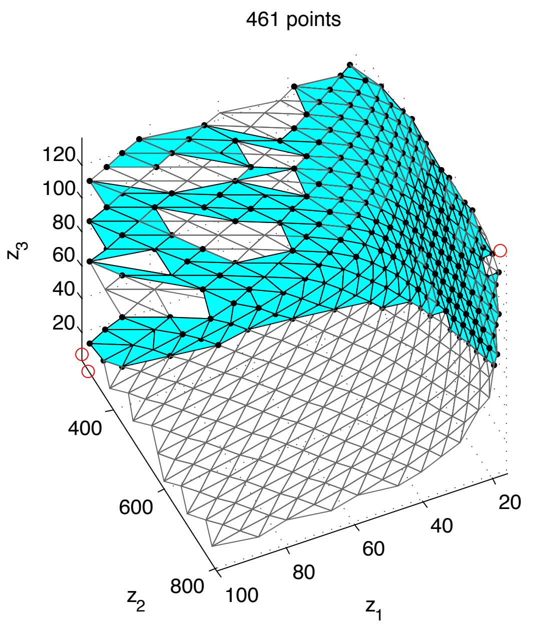


Figure 26: The obtained Pareto set with Pareto optimal (shaded faces) and non Pareto optimal (gray lines) areas. The individual minima of every objective can be seen as red circles. Note that the two circles to the left (individual minima of z_2 and z_3) are close to each other.

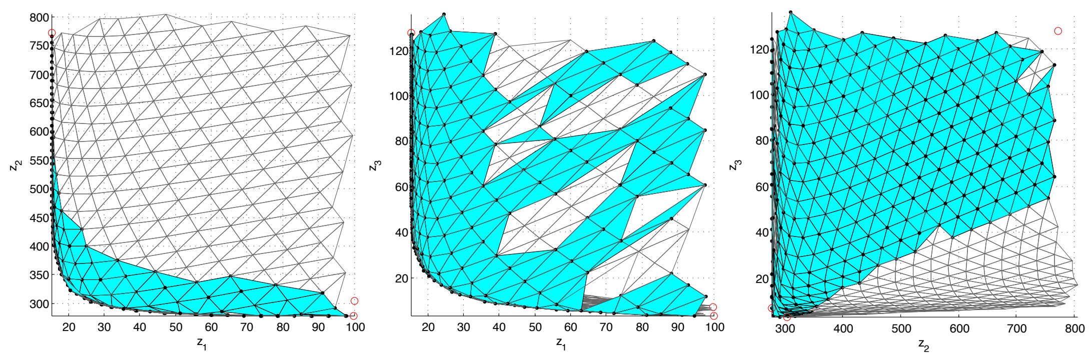


Figure 27: The projections of the obtained Pareto set on every coordinate axis. Left: The Pareto set of z_2 with respect to z_1. Middle: The Pareto set of z_3 with respect to z_1. Right: The Pareto set of z_3 with respect to z_2. Note the sharp shape in the lower left when looking at z_2 and z_3, which indicates that they are weakly conflicting.

obtained Pareto set can be seen in Figure 28. In the left plot all trajectories (gray lines) that were obtained in the optimization (including non Pareto optimal trajectories) are shown together with the individual minimas for each objective. In the right plot the Pareto filter was applied and shows only Pareto optimal trajectories. Note that almost all Pareto optimal trajectories have similar trajectories as the individual minimas. It should also be noted that the actual total time for one cycle is not illustrated in the figure and hence the difference in time between the trajectories cannot be seen. For example, when mimizing the time only (red line with cross markers), the total time for one cycle is 15.23 s and when minimizing energy or maximal power (square markers and point markers respectively) the time reaches the boundary of 100 s. This relation can instead be seen on the Pareto frontier in Figure 26 and 27.

When minimizing the time, the trajectory for $\mathbf{f}^{1*}$ is curvy. This is due to the fact that the pendulum reaches a higher point faster by moving the cart back and forth and swinging the pendulum. However, when minimizing only energy (green line with square markers) or power (blue line with point markers) the trajectories are smoother and the paths are shorter. As mentioned there are only two signals to control in this

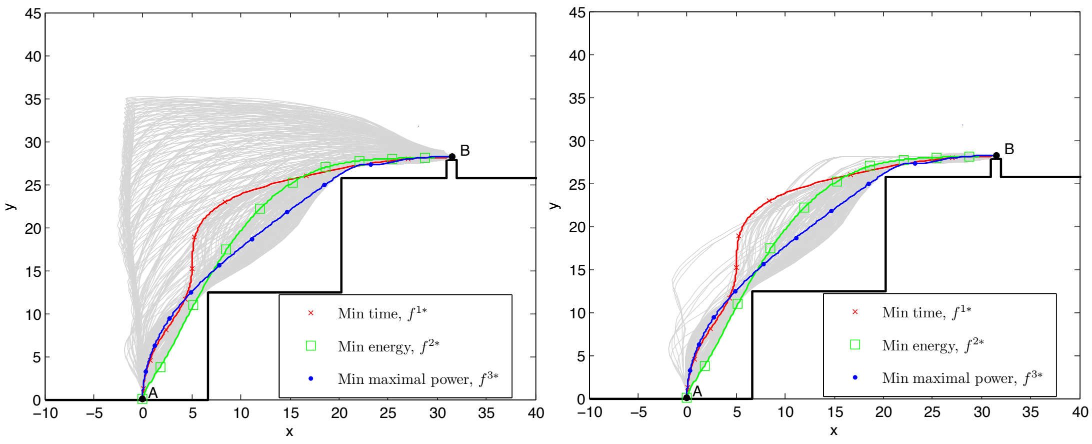


Figure 28: The trajectories (gray lines) of the pendulum for the obtained Pareto set. Left plot shows the trajectory of all sampled points and the right plot shows only Pareto optimal trajectories. The colored lines with markers shows the trajectory of the individual minima of respective objective.

system, the force of the motor to move the cart and the force to change the length of the pendulum. In this thesis the control signals are directly connected to the acceleration of the cart and to the acceleration of the length of the pendulum for simplification of the model. Control signals of respective actuator for Pareto optimal solutions can be seen in Figure 29. Note that the actuators in solution $\mathbf{f}^{1*}$ are faster and more active than the other two objectives. Most Pareto optimal solutions (gray lines) gives control signals that are similar to $\mathbf{f}^{1*}$. This can also be seen in the Pareto set in Figure 26 where most Pareto optimal points are in the area where the total time is low.

4.2.2 Benchmarking

Different settings of the warmstart feature in the offline application were compared in the cart-pendulum problem. Four Pareto sets were solved with the settings:

- Set 1: Warmstart disabled and μ set to adaptive.
- Set 2: Warmstart enabled and μ set to adaptive.
- Set 3: Warmstart disabled and $\mu = 10^{-5}$.
- Set 4: Warmstart enabled and $\mu = 10^{-5}$.

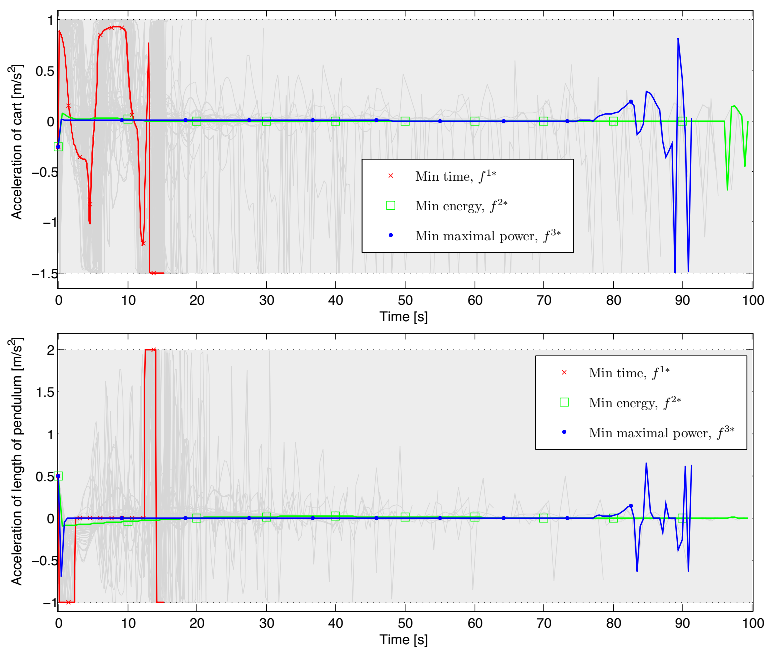


Figure 29: The control signal of the cart (upper plot) and the length of the pendulum (lower plot). The shaded area is the feasible area of respective control signal where the dotted lines are the upper and lower boundaries. The colored lines with markers shows the control signals for the individual minima of respective objective.

In every set 200 points were solved and the same initialguess was used for the first point in all sets. Both the time to find the solutions and the objective value of the direction method α were measured during the test. Figure 30 shows the time to solve all the points with the four different settings. It can be seen that using some kind of initiating feature decreases the time to solve the points and that initiating μ to some smaller value than default has most impact on the time to solve the Pareto set.

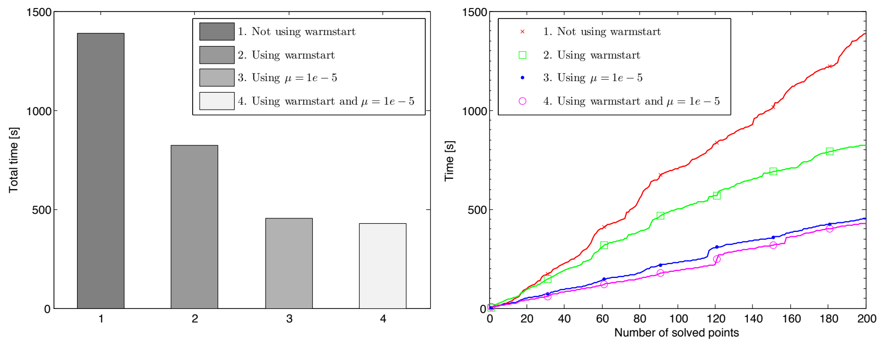


Figure 30: The time for the offline application to solve 200 points with four different settings of the warmstart feature. Left: The totalt time for the complete Pareto sets. Right: The accumulating time for every point.

The objective value α for every setting was compared by using α for every point in set 1 as a reference values and then calculate the ratio between the reference values and α in the other sets. Figure 31 shows the ratios of set 2-4 compared to set 1. A ratio of 1 means that both optimizations reached the same value. In the middle plot it is possible to see that when only μ is set the solver is able to find both better (about 11 percent) and worse (about 64 percent) objective values than in set 1.

The Pareto set seen in Figure 26 and 27 both used warmstart and an initial barrier parameter set to $\mu = 10^{-5}$.

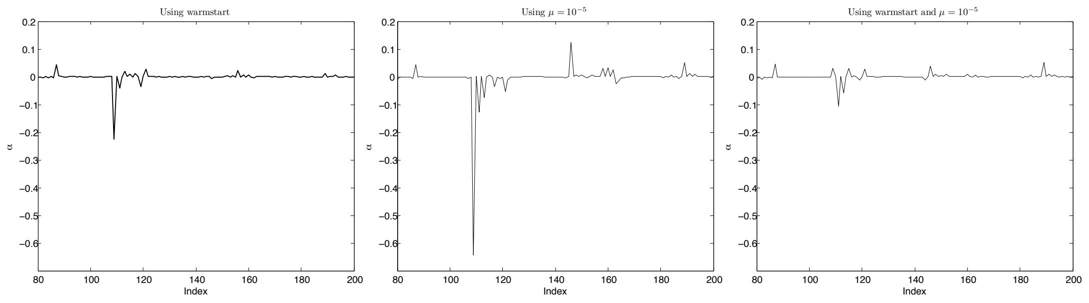


Figure 31: The ratio of α between set 1 and the other sets (2-4). A ratio of 1 means that both optimizations reached the same value. Note that when only $\mu = 10^{-5}$ is used (set 1), the optimizer sometimes reaches a value that is much worse than when no feature is used.

A test was made with different sampling size to find the relation between denser sampling and average solution time per point. Three Pareto sets were obtained, one with 461 points, one with 979 points and one with 3088 points. Figure 32 shows the Pareto sets obtained respectively. All three were obtained by using the warmstart feature. The most important result to note is that the average time for solving one point for each complete set was 2.14 s, 1.44 s and 1.23 s respectively. There are some differences between the Pareto sets in the sense that different areas are Pareto optimal. However, in all sets almost all points were solved except from some points to the lower left where z_3 is minimized (cannot be seen in the figure).

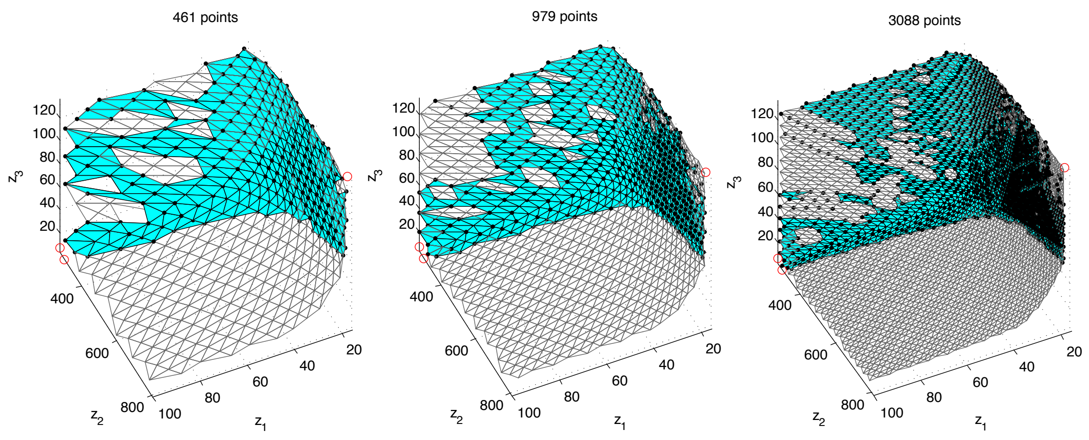


Figure 32: Three Pareto sets obtained with 461 points (left), 979 points (middle) and 3088 points (right). Note that to the upper left there are some differences between the sets.

4.2.3 Navigation

The obtained Pareto set with 461 sampled points (only 273 were Pareto optimal) was tested in the online application. Every point consisted of three objective values and 8600 decision variable values which were imported into the online application. It was possible to navigate on the frontier smoothly both in objective space and the decision space. When navigating close to a hole in the Pareto set the solution either "walked around" the hole or jumped to the other side depending of the size of the hole. When changing the boundaries of either objectives or decision variables the application had some noticeable lag. It was possible to plot the control signals and the trajectory of the pendulum in the plot window without feeling that the application was slower.

4.2.4 Summary

The offline application was applied successfully on the cart-pendulum problem. A Pareto set was obtained which shows that the energy consumption and the maximal instantaneous power are only weakly conflicting which can be expected since lower power also means lower energy consumption. It was also realized that the energy consumption (and hence also power) can be heavily reduced by increasing the total time of one cycle with a fraction of a second.

When warmstart features were used during optimization the Pareto set was solved faster than when only using the previous solution as an initalguess. However, it can be seen that in some cases the objective value only reached locally optimal solutions if the warmstart features were used, especially when only the barrier parameter was set. Since the next point to solve actually is some distance from the previous, the solver is not started close to the solution and if the barrier is too small from the beginning the solver can be lead into a locally optimal point. When warmstart is used, the solver is initiated with more information so that it is easier to find better solutions even if they also are locally optimal. It could also be seen that both warmstart and a set barrier parameter could find better solutions than without any features used. This means that it might be good to always use the warmstart features, but if the optimization time is not an issue for the decision maker, it might be good to also optimize without warmstart features.

In this specific MOO problem it can be seen that the time needed to solve a Pareto set does not increase linearly with the amount of points. The average time for one point was decreased when making a denser sampling of the Pareto frontier. One explanation to this can be that a denser sampling means that the solutions are closer to each other which improves the performance of the warmstart features.

The online application worked successfully on the cart-pendulum problem. It was possible to reach all Pareto optimal solutions and the navigation was smooth. However, when changing the boundaries the application was slower.

5 The hot rolling mill optimization problem

A hot rolling mill is used to process steel to a desired shape and quality. The Roll.LABB model developed by ABB Corporate Research describes a profile hot rolling mill in steady state and provides many objectives related to production, quality and energy. The modeling of the process is out of the scope for this thesis and the actual model is supplied by ABB Corporate Research. The beginning of this section should give the reader an overview and some insight on the complexity of the hot rolling mill model. The purpose is to help understand and to interpret the MOO results. Three objectives of particular interest in MOO are used in this thesis. These are:

- Minimization of power consumption.
- Maximization of production speed.
- Quality – minimization of grain size.

These objective functions are conflicting with each other and the MOO of the Roll.LABB model is supposed to clarify these dependencies.

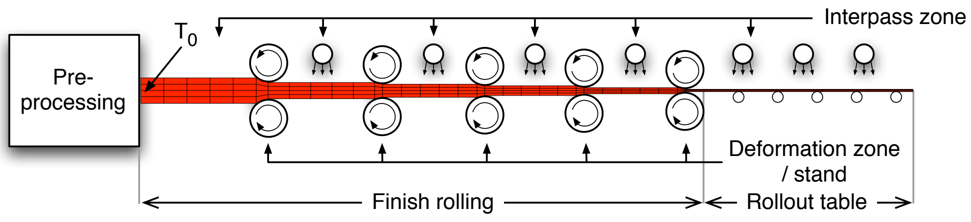


Figure 33: Principle sketch of the hot rolling mill process. Note that re-heating, descaling and rough rolling are grouped as pre-processing. The sketch is not according to scale, e.g. the rollout table is usually much longer than this sketch shows.

A hot rolling mill consists of a number of sub processes and one setup is

- Re-heating – the pre-cast metal is heated to a specified temperature.
- Descaling – The scale, which is an oxide layer formed in the re-heating stage, is removed.
- Rough rolling – In this stage huge reduction of the material is made with less concern for the finish.
- Finish rolling – A number of stands (pair of rolls) makes small reductions to better control the quality and finish of the product.
- Rollout table – The metal is cooled by water on the rollout table.

Figure 33 show a principle sketch where re-heating, descaling and rough rolling are grouped as pre-processing.

The process is modeled as interpass- and deformation zones, see Figure 33. The interpass zones model the zones before and after the stands and describe the cooling of the metal due to radiation, convection and active cooling from water cooling pipes. The deformation zones model the actual deformation exerted by the rolls in the stand and describe the change in material thickness, the change in width, the change in speed and the temperature change due to friction, plastic deformation and contacts with the rolls. There are only a small number of free (control) variables in the optimization model and the most important ones are roll gap, roll speed, temperature of material when it enters the process T_0 and the cooling in the interpass zones.

The temperature is assumed to be homogeneous with temperature T_0 through the entire material when entering the first interpass zone. The temperature is then described as a temperature field which is discretized both in the longitudinal and lateral direction. The mechanical properties are affected by the temperature of the bar, which affects the rolling process and the temperature is connected to the rolling process and hence, the problem is a coupled mechanical and thermodynamic problem. The temperature field is the single largest source of optimization variables in the model. It is scalable and the accuracy can be improved by denser sampling. However, this heavily increases the number of optimization variables (Ekh, 2007).

The metal bar is driven by the friction between the rolls and the metal bar itself. The mass flow through the process is constant and because of the height reduction in the deformation zone the speed of the bar after the rolls is increased.

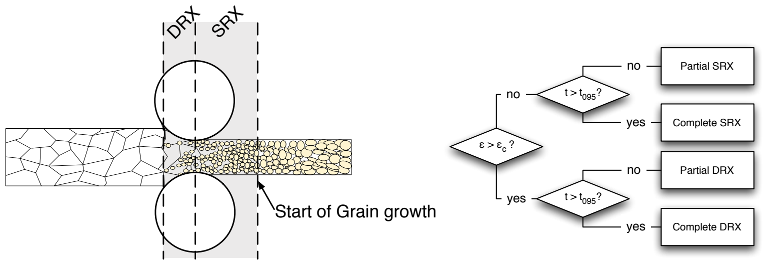


Figure 34: Left: Recrystallization phases and grain growth. Right: The combinatorial behavior of the recrystallization.

The microstructure evolution, i.e how the microstructure changes during the complete process, is a combination of recovery, recrystallization and grain growth. In Roll.LABB recrystallization is of special interest and is the process which creates new grains. These new grains grows on the expense of old deformed grains. Recrystallization which happens during derformation is called *dynamic recrystallization* (DRX) and recrystallization which happens after deformation is called *static recrystallization* (SRX). Recrystallization reduces strength and soften the metal, but it also gives an increased ductility (Pietrzyk et al., 1999). See left plot in Figure 34 for a schematic overview of the microstructure evolution.

The recrystallization in the metal bar is dependent on temperature, strain, strain rate and the current microstructure. The microstructure model is in some sense "combinatorial", e.g. the grain size growth is different depending on the strain and how long time the metal bar is in a certain stage. The right plot in Figure 34 show the combinatorial behavior of the process.

If the strain is less than a specific critical strain then the grain growth is governed by SRX. Depending on the time the metal bar is in the interpass zone, partial or complete SRX is achieved. In the same way the DRX is dependent on strain and time. The details are outside the scope of this thesis and the interested reader will find more information in Pietrzyk et al. (1999). This combinatorial behavior is present at every stand in the hot rolling model. This means that there is a large number of possible combinations, $4^{\text{number of stands}}$. Introduction of combinatorial parameters creates problems in the optimization and to simplify the model, the switching behavior in the model is softened. The switch (step function) is modeled as a function with a variable shape-factor. A larger shape-factor means a harder switch. Figure 35 show the switching behavior for different shape-factors.

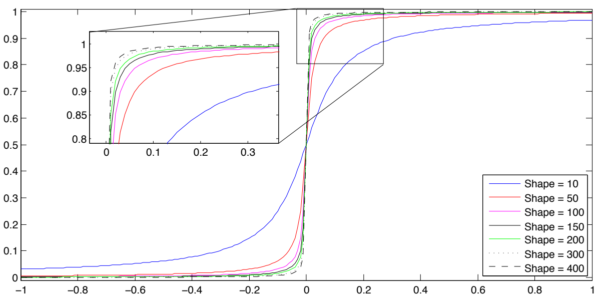


Figure 35: The switch behavior for different shape-factors.

5.1 Test case

The Roll.LABB model and an example of an operation condition with constants, boundaries and initial values were supplied by ABB Corporate Research. The values used in the model are physically reasonable but are not connected to any real hot rolling mill. The example operation condition was modified, the shape factor was set to 10 and the boundaries of roll gap and roll speed were relaxed ±20 percent. Also, the initial temperature was relaxed with ± 50 C° from a fixed value of 1050 °C and partial cooling was turned on, in the interpass zones. The problem was sampled with 420 points using the generic framework and Figure 36 show the result. This Pareto set will be used as the *basis Pareto frontier* for comparison.

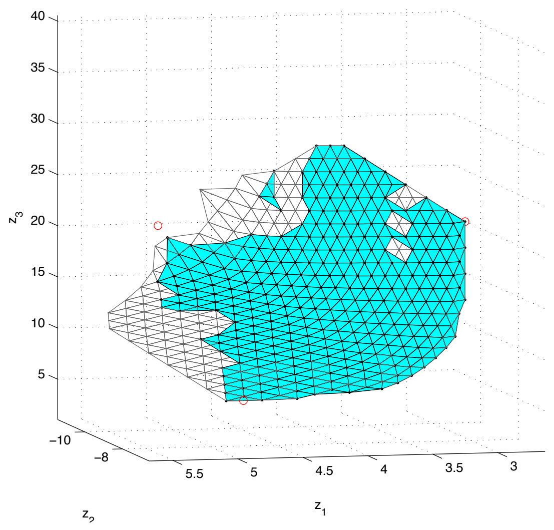


Figure 36: The Pareto frontier for the example operation condition with relaxed bounds on the control variables.

The black markers show the Pareto optimal points that have been found with the Pareto filter and the shaded area is the convex decomposition of these points. The gray lines show non Pareto optimal surfaces. Note that some Pareto optimal points (black markers) are missing due to plotting issues. In every corner of all the shaded faces there should be a black marker.

The red circles are the individual minimas that were obtained by optimizing each objective individually (SOO).

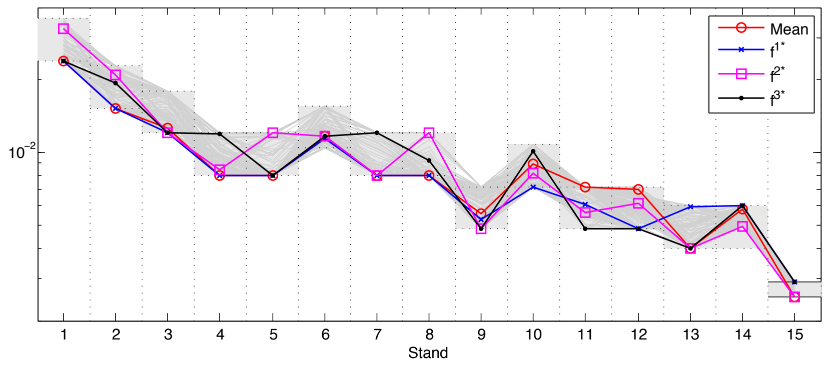


Figure 37: The values of the control variable gap plotted for all points on the Pareto frontier with the mean- and extreme points marked. Note that the solutions are spread over the entire allowed gap range (except for stand 14). This might indicate that this limits the solution.

As mentioned the primary free variables are roll gap, roll speed, temperature into the first interpass and the amount of cooling in each interpass. Figure 37 - 39 show gap, amount of cooling and roll speed respectively. The figures show the feasible area in each stand (gray area with dotted border), the “trajectories” for each feasible point in the Pareto frontier (gray lines) and four interesting points marked with colors. The plots clearly show that there is a spread of solutions ranging over the entire feasible set for each variable in all stands except for roll gap in stand 14. This indicates that the variables themselves are restricting the optimization problem.

The blue line (cross marker) in Figure 37 - 39 shows the settings for the extreme case to minimize power consumption. The temperature into the first interpass T_0 is maximized and the cooling is minimized in order to keep the temperature up. This means that it is easier to reduce the height of the material in each stand. The reduction is kept to a minimum and most of the reduction is done in the beginning of the mill where the material is as hot as possible. At the last stand the gap is the maximum in the allowed tolerance in order to keep the reduction to a minimum. Note that the roll speed is not minimized even though it has a very strong connection to the power consumption. This is a trade-off between power consumption due to roll speed and power consumption due to increased torque needed when the material cools down.

The magenta line (square marker) in Figure 37 - 39 shows the settings for maximizing production speed. The roll speed is maximized in each stand and the height out from the deformation zone in the last stand, is at its lowest tolerance. The temperature at the first interpass T_0 is maximized and the cooling is minimized in order to minimize power consumption needed to reduce the height of the material.

The black solid line (dot marker) in Figure 37 - 39 shows the control signals for minimizing grain size. The temperature T_0 is not minimized nor maximized but lower than for the other extreme cases and the cooling is maximized since a low temperature is

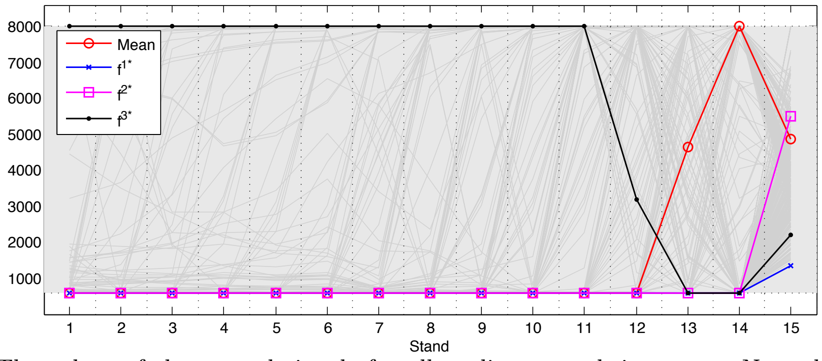


Figure 38: The values of the control signals for all cooling at each interpass. Note that this only includes the free variables in each stand.

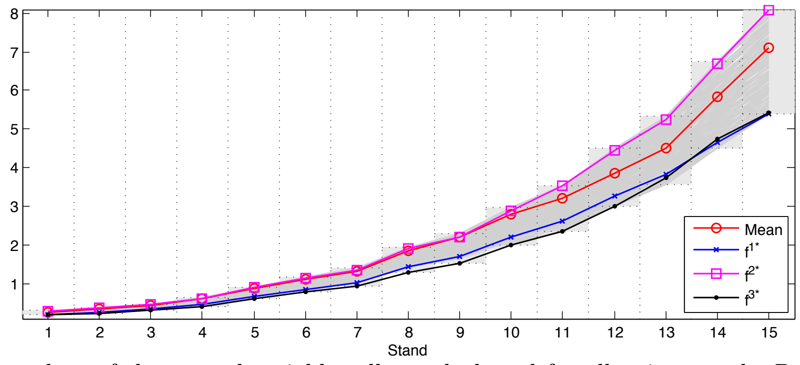


Figure 39: The values of the control variable roll speed plotted for all points on the Pareto frontier with the mean- and extreme points marked. Note that the solutions are spread over the entire roll speed range. This might indicate that this limits the solution.

very favorable to decrease grain size. The reduction is maximized at the first stand which stores enough energy in the material to initiate DRX. This gives a huge grain size decrease in the interpass after the first stand.

The red line (circle marker) in Figure 37 - 39 shows the solution closest to the projection of the ideal point onto the Pareto frontier, i.e. in the middle of the frontier. It is mainly there to have as a reference.

5.2 Spread of solutions

Besides looking at the spread of the free variables of all solutions in the Pareto frontier, the spread in a small set in objective space versus the spread of the corresponding variables in decision space is one way of measuring how well-behaved or “nice” the problem is. This measure can be an indication of how likely it is that a navigation solution, which is a linear combination of a set of neighboring points, is feasible since it is less likely that points that are close to each other in decision space will be separated with an infeasible region.

The spread was measured by picking a point in the Pareto set (obtained from the basis operation condition) and finding its 18 closest neighbors. The spread of solutions were then checked for the four free variables initial temperature, roll gap, cooling and roll speed. This was done for every point in the Pareto set and it could be seen that the spread was low for the complete Pareto set. Three example sets are described below and the chosen sets can be seen in Figure 40.

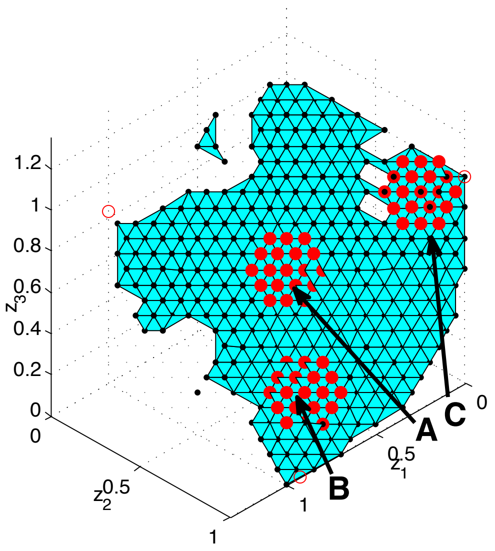


Figure 40: Pareto frontier where all free variables are relaxed ±20 percent from the original model and with shape factor = 10. Three points and their 18 closest neighbors have been chosen to see the spread.

The first set of points are the points closest to the projection of the ideal point onto the Pareto frontier and its 18 closest neighbors marked with A in Figure 40. Figure 41 shows the free variables for the mean point (red thick line) and the neighboring points (dark gray). The point in the centre of this set is in a central position on the Pareto frontier and is an equal trade-off between power consumption, production speed and grain size. The neighboring solutions show equal spread in all free variables around the solutions of the middle point. Since the solutions are central there are solutions which favor each of the objectives, e.g. more cooling for smaller grain size.

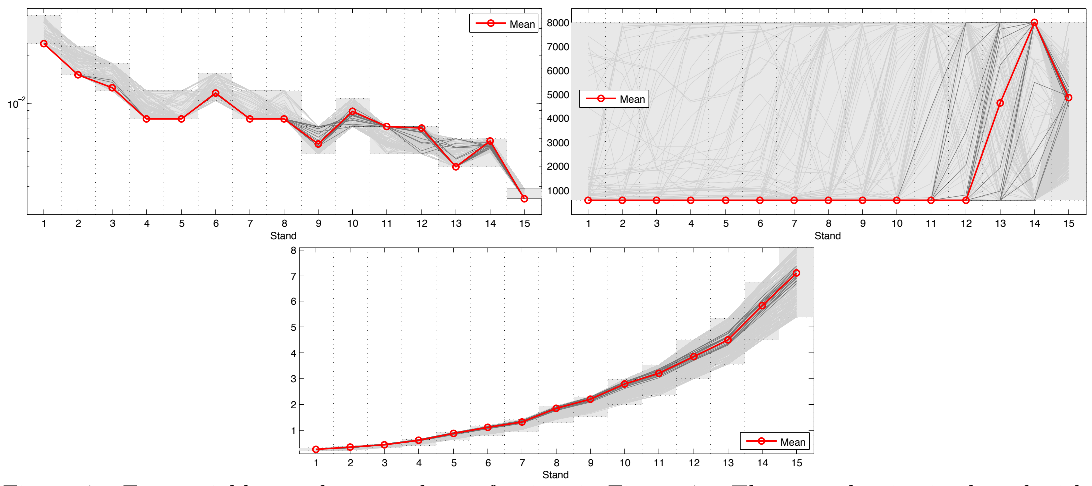


Figure 41: Free variables to the central set of points in Figure 40. The central point is plotted with a red thick line and all neighboring points are plotted as dark gray. The spread is even around the central point and the solutions do not favor any extreme cases. Note that all Pareto optimal solutions are plotted in light gray.

The second set is the set of points in Figure 40 which is closest to $\mathbf{f}^{3*}$ marked with B in Figure 40. Figure 42 shows the free variables for this set. The spread of the solutions in decision space of the neighboring points (dark gray line) is evenly spread around the central point (red thick line). These points have free variables which favors small grain size, i.e. low speed and a lot of cooling.

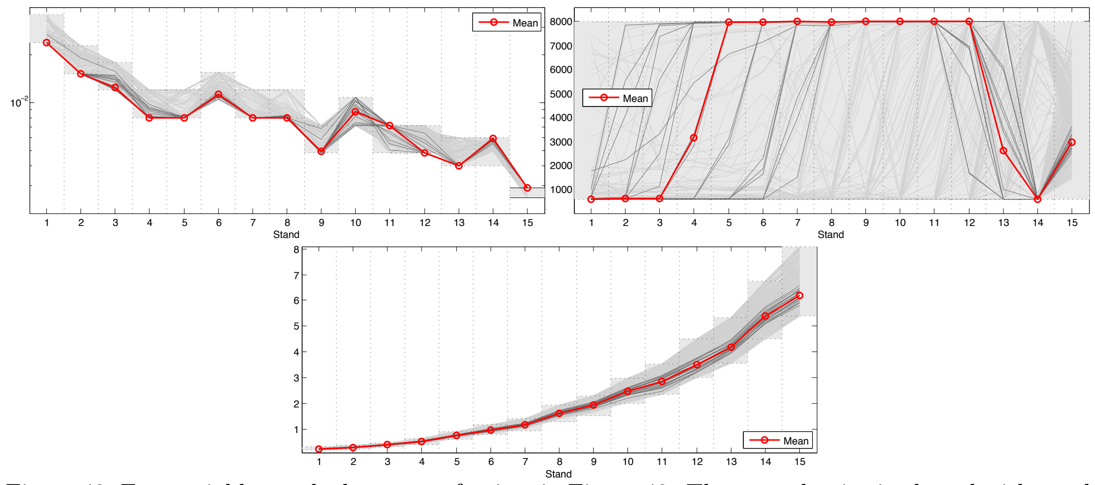


Figure 42: Free variables to the lower set of points in Figure 40. The central point is plotted with a red thick line and all neighboring points are plotted as dark gray. Note that all Pareto optimal solutions are plotted in light gray.

The last set of points in Figure 40 are the set closest to the ideal point of minimum power marked with C. Figure 43 shows the free variables of these points. Since this set is very close to an extreme value, i.e. the point of minimum power, the free variables has a very low spread and obviously favors low power consumption, i.e. no cooling and low speed.

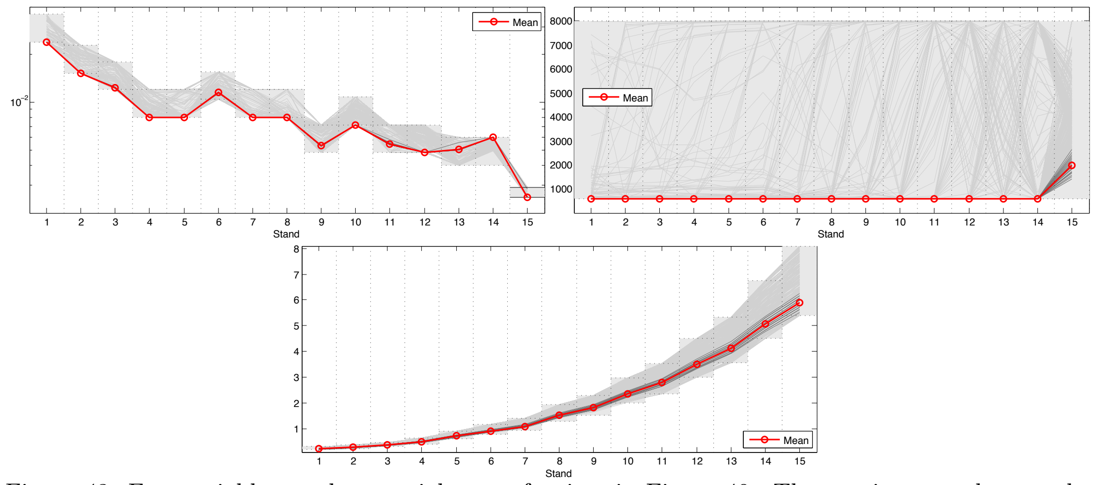


Figure 43: Free variables to the top right set of points in Figure 40. These points are close to the point low power consumption which means that the solutions are closer together. The central point is plotted with a red thick line and all neighboring points are plotted as dark gray. Note that all Pareto optimal solutions are plotted in light gray.

5.3 Effects of boundary changes

Since the example operation conditions were relaxed, it is interesting to see the effect that the relaxation has on the Pareto frontier. This was done by limiting one of the controlling variables and re-sampling both individual minimas and the Pareto frontier. Figure 44 shows from the left, the basis Pareto frontier, the Pareto frontier for the case where roll speed is limited and the Pareto frontier for the case where the initial temperature T_0 is fixed to 1050 °C. Figure 45 shows from the left, the case where the gap is limited and the case where the cooling between the deformation zones is turned off.

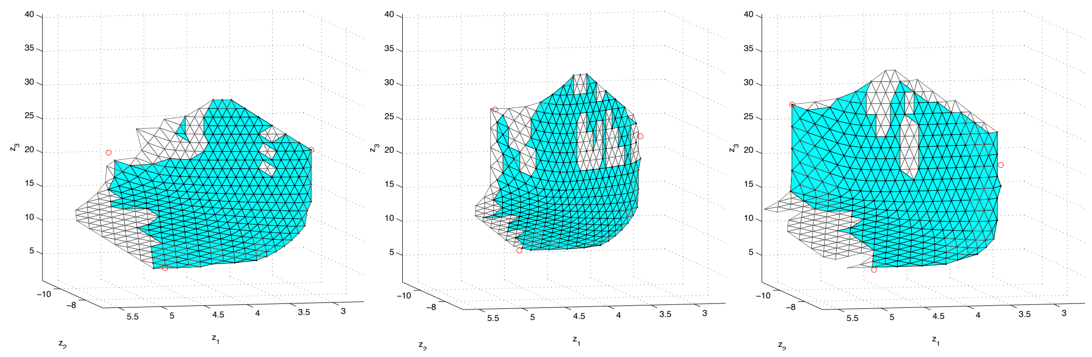


Figure 44: The Pareto frontier for the example operation condition with relaxed bounds on the control variables. Left: Basis operation condition. Middle: Pareto frontier with limited roll speed. Right: Pareto frontier with limited initial temperature T_0.

The contour of the Pareto frontiers projected on the z_1–z_2- , z_1–z_3- and z_2–z_3- planes are plotted in Figure 46-47 in order to better visualize the difference between the frontiers. The largest impact on the Pareto frontier is by cooling which can be seen in the right plot in Figure 46. When the cooling is turned off (dotted line) the frontier is much worse (higher grain size for the same power consumption). However, in the left plot it can be seen that it does not change the relation between production speed and power consumption. The minimum grain size grows from 6.72 in the basis model to 22.91 without active cooling between the deformation zones. The right plot in Figure 47 show an example where the amount of available cooling has been increased an additional 50 percent. However, this did not make any visible changes on the Pareto frontier. This indicates that some other variable is limiting the solution.

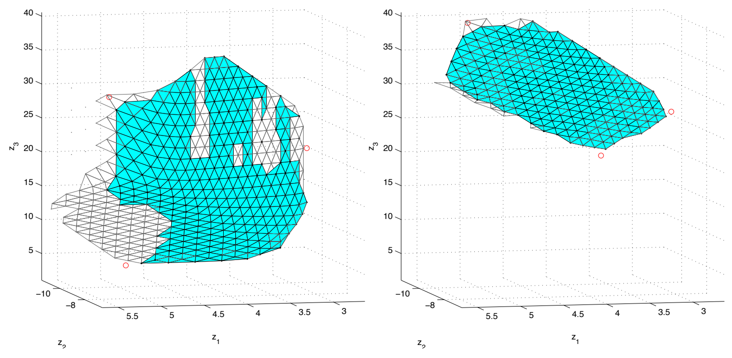


Figure 45: The Pareto frontier for the example operation condition with relaxed bounds on the control variables. Left: Limited roll gap. Right: Cooling turned off.

The roll speed is almost directly connected to the production speed (except for some slip) and hence, limiting the roll speed limits the production speed.

Setting the initial temperature T_0 to a fixed value does not impact the grain size considerably. However, as seen in the left plot in Figure 46 the entire Pareto frontier

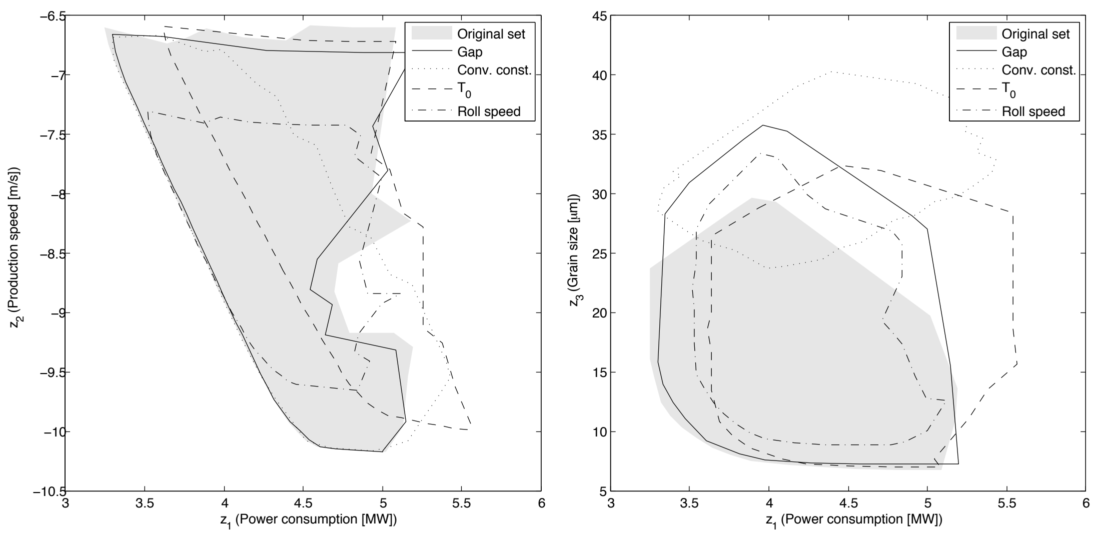


Figure 46: The contours of the projection of the Pareto frontiers, with limitations on the control variables, onto the z_1-z_2- and z_1-z_3-planes. Solid line: Limitation in roll gap. Dashed line: Setting T_0 to a fixed value. Dotted line: Cooling turned off. Dot-dashed line: Limitation in roll speed.

is shifted to the right which means that more power is needed for the same production speed. There is also a slight decrease in maximum production speed due to higher power requirements.

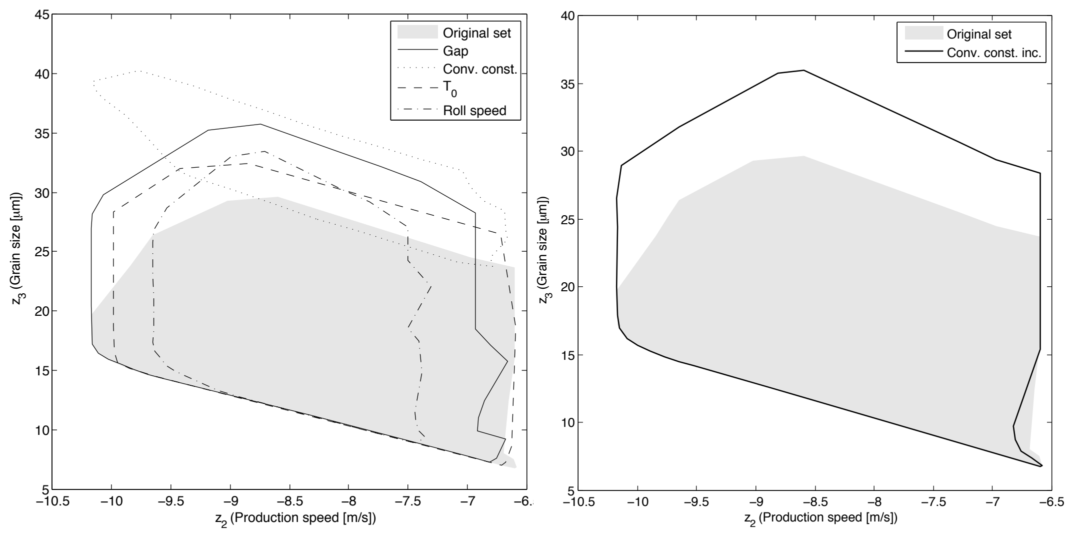


Figure 47: Left: The contours of the projection of the Pareto frontiers with limitations in the control variables onto the z_2-z_3-planes. Solid line: limitation in roll gap. Dashed line: Setting T_0 to a fixed value. Dotted line: Cooling turned off. Dot-dashed line: Limitation in roll speed. Right: The contours of the projection of the Pareto frontier for the basis operation condition (gray area) and the projection of the Pareto frontier with a 50 percent increase in cooling capacity (solid line).

Limiting the roll gap gives very small changes. The minimum power consumption increases, the grain size increases slightly and the lowest possible speed increases from the basis operation condition.

5.4 Effects of combinatorial behavior

The combinatorial behavior introduced in the microstructure model creates added difficulties when the shape factor is increased. The same operation condition was tested for a range of values on the shape factor. Three of the Pareto frontiers and their projections can be seen in Figure 48 - 50. These cases are from the left, for shape factor 10, 50 and 400. Figure 35 show a plot of the relaxed step function for different shape factors. The first holes in the Pareto frontier start to emerge already at shape factor 80 and continues to grow with increased shape factor. Already as low as with shape factor 50 the unevenness of the Pareto frontier increases and continues to increase with the shape factor. However, the unevenness are situated in the same areas and a trend can be seen in the solutions with an increasing shape factor.

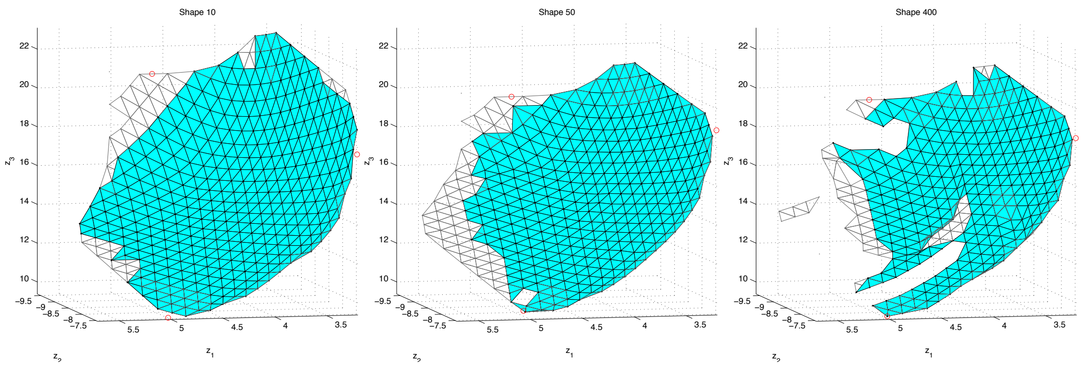


Figure 48: From the left, the same problem with shape factor 10, 50 and 400 respectively. Note that there are non-convex behavior already at shape factor 50. With shape factor 400 there has emerged large holes due to infeasible- and non-Pareto optimal pints. The Pareto frontier has obvious non-convex behavior and the set of Pareto optimal points has ben reduced.

The change of the Pareto optimal points which gives maximum production speed, minimum power or a trade-off between them are hardly not affected at all. For that reason the projections of the Pareto frontier on the z_1-z_2-plane are left out.

Figure 49 show the projection of the Pareto frontiers shown in Figure 48 onto the z_1-z_3-plane. Its lower boundary points are clearly affected by the increase in shape factor. It is mainly the steepness of the curve through the lower boundary points which are affected. The grain size is increased by 4.83 percent with the increase of shape factor from 10 to 50 at 3.5 MW and by 6.6 percent with the increase from shape factor 10 to 400.

In the same way, Figure 50 show the projection of the Pareto frontiers shown in Figure 48 onto the z_1-z_2-plane. At the production speed 9.2 m/s the increase in grain size when the shape factor is increased from 10 to 50 is 7.5 percent but with shape factor 400 the increase is only 4.1 percent. The unevenness of the curve through the lower boundary points starts with shape factor 50 and at shape factor 400 there are unevenness and a hole in the curve through lower boundary points.

5.5 Effects of denser sampling of the Pareto frontier

The basis operation condition were sampled with 420, 1191 and 1641 points. The increase of time for optimization per point can be seen in Table 2.

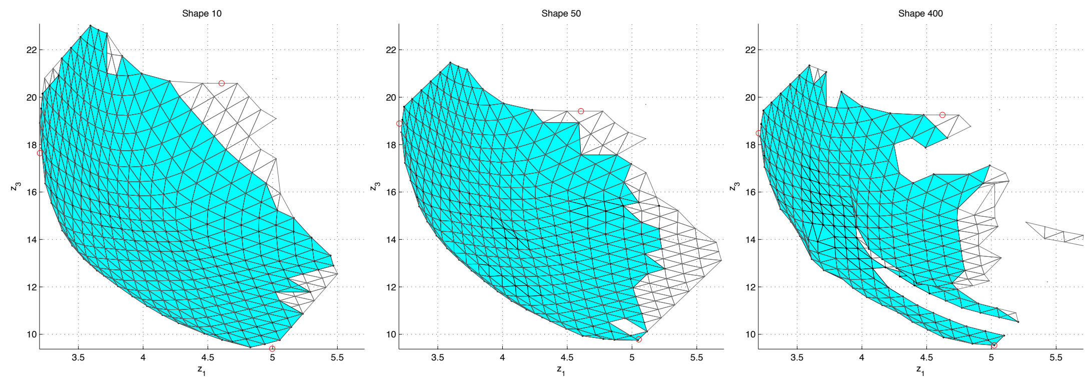


Figure 49: These plots show grain size versus power consumption for the same case as in Figure 48. The grain size increases with the shape factor and the boundary towards the axes exhibit unevenness. Note that this means that with an increased shape factor, the power requirement increases.

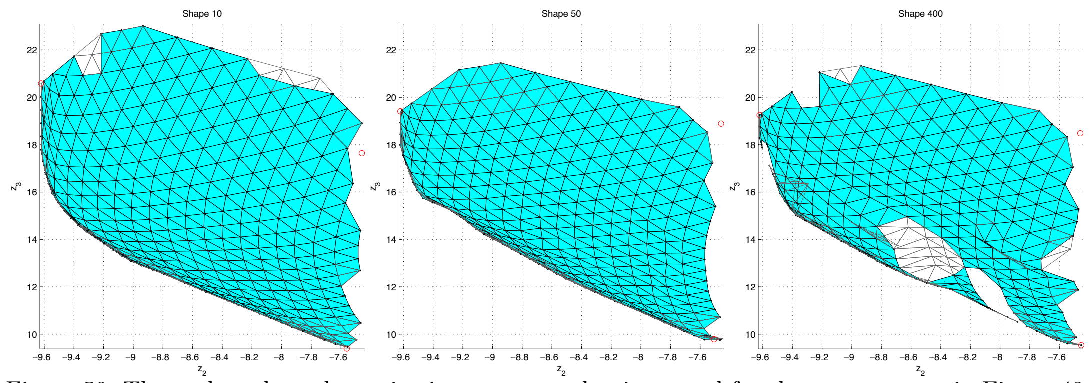


Figure 50: These plots show the grain size versus production speed for the same case as in Figure 48. Even for a small increase in shape factor the increase of grain size at for the same speed is substantial. Note that with shape factor 400 there is even a hole in the boundary towards the axes.

Table 2: The time for optimization per point with different sampling.

Number of points	420	1191	1641
Mean time per point	12.16 s	6.67 s	8.24 s

5.6 Summary

The Roll.LABB model has evolved over years of work and ABB are well aware of its properties. The solutions obtained when performing SOO only gives some information. In this application large amount of power is used and even small percentages of power savings means a lot of money. Since the Pareto frontier is implicitly defined by the optimization problem, only an idea of the exact relation has been known. The benefits of MOO in this application are clear, especially when using the navigation tool where the decision maker can look at the control signals for the current solution. The offline part of the framework was successfully applied to the Roll.LABB model. There is no obvious best single optimal solution on the Pareto frontier since the three objectives are strongly conflicting which means that it is up to the decision maker to choose his/her preferred solution.

The four primary control variables have been investigated and the most important is the cooling. It gives a decrease in the grain size without sacrificing production speed or power. The upper boundary of the amount of cooling allowed is not a limiting factor.

The spread in solutions in objective space versus the spread in decision space is low which indicate that the model behaves “nice” which means that an approximated solution is more likely to be close to a feasible solution.

Holes emerges in the Pareto frontier when the shape factor is increased. This might be due to combinatorial behavior and/or it might as well be local solutions which are easier for the solver to end up in, when the shape factor is ramped up.

6 Discussion

This section discusses the methods used and the implementation of the generic framework. Discussion about the pendulum- and hot rolling mill problems are considered in Section 4.2.4 and Section 5.6.

6.1 Reference point creation

As mentioned in Section 3.3.2 on page 18, the pattern of the created reference points is very important. This is because the method for sampling the Pareto frontier is based on the direction method without slack. However, there is a possibility that the created reference points do not sample the individual minima. This can be seen in Figure 11 on page 17, where all three individual minima are “missed”. This is not a problem in a trade-off point of view since the extreme points are of less interest. However, in some cases, e.g. to get a better understanding of the problem, it is interesting to be able to see the individual minima. This is a minor drawback since the individual minima are found during the pre-processing and the Pareto set can be complemented with these solutions.

Since the objectives are scaled according to Section 2.5 on page 8, the scale of the objectives are relative. If the original objective functions are badly scaled, i.e. the range is much greater in one of the objective functions, the sampling will not be equidistant in the unscaled case. In cases where the relative importance of the objective must be kept, i.e. the absolute value of one objective value compared to another, the scaling can be skipped and the reference points can be created in the unscaled problem. However, working with the implemented generic framework has shown that scaling of the constraints are important. Since the objectives are incorporated in the direction method constraints, scaling of the objective functions should be done too.

The use of the span tree method in the generic framework has both benefits and disadvantages. Large benefits have been seen by using a neighboring solution as an initial value to the next optimization and this clearly motivates the use of a neighbor search algorithm. However, it is possible that a “bad” local minima is found in one point and then subsequent initial values are based on this solution. This might lead to areas on the Pareto frontier with unwanted local solutions.

The reference point creation has been created as a part of the generic framework, hence the method is quite specific to accommodate the needs of the navigation, i.e. simplex creation. The equidistant sampling is useful to get a good approximation of the Pareto frontier since it gives an even coverage and there is no clustering of solutions. This also means that information gained from the sampling, e.g. infeasible solutions, can be used in the creation of the Pareto frontier approximation in a smart way. This makes it possible to exclude regions which do not contain solutions, i.e. holes.

6.2 Sampling

No time was spent on evaluating the nonlinear solver since the IPOPT/Maple-tool was supplied by ABB. The IPOPT-solver is an interior point solver for nonlinear large scale problems. The solutions found by IPOPT are not guaranteed to be globally optimal, hence unwanted local solutions can be obtained. If “bad” local optimal solutions are found only at individual points, this might result in making the point non Pareto optimal and a hole might emerge. Another possibility is that the solver finds “bad” local optimal points over the entire Pareto frontier, which shifts the entire Pareto frontier. In some cases where the aim is to improve on existing solutions local optimization is “good enough”, but the decision maker should be aware of the risks. Cases of local solutions have been seen in the work with IPOPT, especially in the pendulum problem where many trajectories produce the same objective value.

The choice not to use slack with the direction method during sampling have both benefits and disadvantages. The method used for creation of simplicies requires that no slack is used since the samples are presumed to be equidistant sampled. The non-Pareto optimal solutions obtained when no slack is used complements the Pareto optimal solutions and helps the understanding of the boundary of the feasible set , i.e. helps the understanding of the problem. However, without slack the possibility to find extreme solutions, e.g. solutions close to holes, decreases.

Practical experience from the thesis shows that re-runs, i.e. the possibility to solve the same point many times, is beneficial and gives better solution in some cases, especially cases for the pendulum problem where there are many different local solutions. How the re-runs should be combined with initial values and warm start has not been thoroughly investigated.

6.3 Navigation

When using the convex decomposition method to navigate on the Pareto frontier several optimizations must be performed. One for every simplex at the current preferred levels. How much the amount of simplices that have to be optimized affects the time for one iteration has not been measured. For a Pareto set with large amount of sampled points there will also be many simplices to optimize over which might take too much time to optimize to make the application feel real-time. The method implemented in this thesis to optimize only the active set of simplices for a desired level is only one out of several ideas. This method does however work for problems with any dimensions and is simple to implement.

When selecting levels of decision variables and no objectives are fixed there can occur jumps between different solutions. An alternative idea has been suggested by Winterfeld (2010) which would be to minimize the deviation in objective space instead, i.e. minimize$\|f_i(\mathbf{x}) - z_i^R\|, i \in \{1, \ldots, m\} \setminus F_z$. This gives a smooth transition between solutions when navigating in the decision space. This can, however, create an irregular pattern of search in the objective space since there are in many cases several solutions with the same deviation.

One problem that has been detected is that simplices that have vertices on both sides of a boundary in either objective- or decision space become infeasible even if some areas of the simplex might be feasible. This is due to the fact that infeasible points, even if they are close to a boundary, cannot be connected to a simplex. One idea is to make ”softer” boundaries that also include points that are outside but close to a boundary as feasible points. Another idea is to ”cut” a simplex in the intersection of the boundary and the simplex and then ”create” temporary points in the intersection that can be used in the convex decomposition.

In some cases when many points of the Pareto set are infeasible there might be regions where only one or two points are neighbors and that are feasible. In these cases for MOO problems with three or more objective functions the convex decomposition does not cover the entire set of sampled points. However, it should be possible to navigate to these points. One idea is to continue decomposing the set into smaller simplices. For example in a three dimensional MOO-problem, first decompose to simplices with three vertices. The remaining feasible points would then be decomposed with 1- (lines) and then 0-simplices (points) until all feasible points are in at least one simplex. It would however not be possible to continuously navigate to the 0-simplices since there is no area to approximate on.

One limitation of the methods used for convex decomposition in this thesis is that the Pareto frontier must be sampled with certain methods. If some other method would be used it might be difficult to make an appropriate projection of the Pareto frontier and hence difficult to obtain the correct decomposition.

In the current implementation the convex decomposition is filtered so that no simplex has an edge that is longer than $2h$, see the left plot in Figure 18 on page 26. This is only one way to approximate areas with holes without covering the infeasible points. It has not been investigated how good this approximation is and no other method has been tested. To cover more of the area around an infeasible point with simplices might increase the risk that a selected solution close to the infeasible point is also infeasible. If less area is covered, the decision maker might not be able to navigate to a solution that is actually feasible. However, a selected solution that the decision maker might see as the final final solution should always be verified with the original MOO problem. This means that it might be better to cover as much as possible of a hole with simplices and then realize that a selected solution close to a hole might be infeasible.

It might be a useful feature for the decision maker to be able to rank the different objectives according to importance, e.g when one objective is more important than others. In the current method, the direction used in the navigation step is -1 on all objectives which means that all objectives are improved/deteriorated equally much. An idea is to make it possible for the decision maker to change this direction in the application and hence control which of the objectives that is more (or less) important.

The most important advantage with the online application is that it is possible to move around on the Pareto frontier almost in real-time even for large scale problems. In the current implementation it does not matter how many dimensions the decision space has since it is only the set of points in objective space that are used in the optimization. This means that it is only the number of sampled points on the Pareto frontier and the number of dimensions in objective space that effects the speed of the navigation. However, when boundaries in either objective- or decision are changed all points must be checked if they are feasible or infeasible and the convex decomposition must be recalculated. In the current implementation all Pareto points must be checked in every decision variable and objective which makes the application lag for large scale problems. By only checking the boundaries of the decision variable or objective that is currently being changed the application would probably run much faster.

The direction method works well for navigation in objective space as described by Monz (2006), but navigation in decision space has not been investigated. In this thesis the direction method has been used in both spaces and works well for the problems it was applied to. However, there might exist MOO problems where the spread of solutions in decision space is larger and it is not known how the implemented application works for these.

In the current implementation the verification feature has not been implemented. The idea of sending the selected objective values as reference point and the solution as initial guess to the offline application should work but has not been tested.

Also the feature of complementing the obtained Pareto set with new points has not been very much investigated. It is however known that with the current solution of decomposing the Pareto points with simplices, some problems would occur if new Pareto points were added. This is due to the fact that the decomposition requires the mesh of reference points to be equidistant. Since the length of the edges of the simplices are used to filter which simplices that are feasible, there would be problems with simplices that are smaller than the original ones.

The method to decompose a set of Pareto points to small convex sets to be able to navigate in non-convex sets with interactive optimization has not been seen before by the authors. A comprehensive study on how to do the convex decomposition should be performed.

6.4 Dimension

The methods developed in the generic framework should in theory work well in 4 dimensions. However, the extensive testing which has been performed on 2- and 3 dimension, has not been performed on the 4 dimension case since there has been trouble to find problems to benchmark with.

7 Conclusions

In this thesis the methods for a generic interactive multiobjective optimization framework have been developed. The framework consists of two parts. One offline part to sample the Pareto frontier and one online part to continuously navigate on the discrete Pareto set in real-time.

The implementation of the methods for obtaining the Pareto set works well for 4 objectives in large scale problems.

The interpolation method developed to be able to navigate on the Pareto set is an extension of the one described in Monz (2006); Monz et al. (2008) but is able to deal with non-convex sets with holes. A method for decomposing the discrete Pareto set was developed which makes it possible to describe non-convex sets with holes. This is possible due to that our approach uses information from the sampling in the creation of the Pareto frontier approximation in a smart way. This method makes it possible to navigate and limit the problem in both objective- and decision space. The implementation of the navigation works very well even on large problems since the speed of navigation only depends on the objective dimension and how densely sampled the Pareto frontier is.

Overall the implementation of the generic framework works very well on all tested problems and especially on the Roll.LABB model.

7.1 Future work

The method of convex decomposition should be investigated further. e.g. how the set is decomposed, the speed of the optimization for extreme decomposition as used in this thesis versus larger sets and the impact on the reference point creation.

Since the navigation requires equidistant sampling the effects of slack on the Pareto frontier has not been fully investigated. However, if another method of convex decomposition is used it might be possible to use slack and in that case a thorough investigation of slack should be performed.

References

I. Das and J. E. Dennis. Normal-boundary intersection: A new method for generating the pareto surface in nonlinear multicriteria optimization problems. *SIAM Journal on Optimization*, 8:631–657, March 1998. ISSN 1052-6234. URL `http://dx.doi.org/10.1137/S1052623496307510`.

M. de Berg, M. van Kreveld, M. Overmars, and O. Schwarzkopf. *Computational Geometry: Algorithms and Applications*. Springer-Verlag, second edition, 2000. ISBN 978-3-540-77973-5.

K. Deb. *Multi-Objective Optimization Using Evolutionary Algorithms*. Wiley, June 2001. ISBN 047187339X.

G. Eichfelder. *Adaptive Scalarization Methods in Multiobjective Optimization (Vector Optimization)*. Springer, 2008. ISBN 9783540791577.

J. Ekh. Thermal modeling in long products rolling. Internal ABB report., 2007.

T. Erfani and S. Utyuzhnikov. Directed search domain: A method for even generation of pareto frontier in multiobjective optimization. *Engineering Optimization*, 43(5): 467–484, 2010. URL `http://dx.doi.org/10.1080/0305215X.2010.497185`.

P. Eskelinen, K. Miettinen, K. Klamroth, and J. Hakanen. Pareto navigator for interactive nonlinear multiobjective optimization. *OR Spectrum*, 32:211–227, 2010. ISSN 0171-6468. URL `http://dx.doi.org/10.1007/s00291-008-0151-6`.

B. Grunbaum and G. C. Shephard. Convex polytopes. *Bulletin of The London Mathematical Society*, 1:257–300, 1969. URL `http://dx.doi.org/10.1112/blms/1.3.257`.

M. Hartikainen, K. Miettinen, and M. Wiecek. Constructing a pareto front approximation for decision making. *Mathematical Methods of Operations Research*, 73:209–234, 2011. ISSN 1432-2994. URL `http://dx.doi.org/10.1007/s00186-010-0343-0`.

M. Insall and E. W. Weisstein. Connected set, June 2011. URL `http://mathworld.wolfram.com/ConnectedSet.html`.

K. Klamroth and J. Tind. Constrained optimization using multiple objective programming. *Journal of Global Optimization*, 37:325–355, 2006. ISSN 0925-5001. URL `http://dx.doi.org/10.1007/s10898-006-9052-x`.

P. Korhonen and G. Yu. Quadratic pareto race. Technical report, International Institute for Applied Systems Analysis, September 1997.

P. Korhonen and J. Wallenius. A pareto race. *Naval Research Logistics (NRL)*, 35(6):615–623, 1988. ISSN 1520-6750. URL `http://dx.doi.org/10.1002/1520-6750(198812)35:6<615::AID-NAV3220350608>3.0.CO;2-K`.

D. C. Lay. *Linear Algebra and Its Applications (3rd Edition)*. Addison Wesley, 2002. ISBN 0201709708.

R. Marler and J. Arora. Survey of multi-objective optimization methods for engineering. *Structural and Multidisciplinary Optimization*, 26:369–395, 2004. ISSN 1615-147X. URL `http://dx.doi.org/10.1007/s00158-003-0368-6`.

A. Messac, A. Ismail-Yahaya, and C. Mattson. The normalized normal constraint method for generating the pareto frontier. *Structural and Multidisciplinary Optimization*, 25:86–98, 2003. ISSN 1615-147X. URL `http://dx.doi.org/10.1007/s00158-002-0276-1`.

A. Messac and C. A. Mattson. Normal constraint method with guarantee of even representation of complete pareto frontier. *AIAA Journal*, 42(10):2101–2111, 2004. ISSN 0001-1452.

K. Miettinen. *Nonlinear Multiobjective Optimization.* Springer, 1998. ISBN 0792382781.

M. Monz, K. H. Küfer, T. Bortfeld, and C. Thieke. Pareto navigation – algorithmic foundation of interactive multi-criteria IMRT planning. *Physics in Medicine and Biology*, 53(4):985, 2008. URL `http://stacks.iop.org/0031-9155/53/i=4/a=011`.

M. Monz. *Pareto Navigation – interactive multiobjective optimisation and its application in radiotherapy planning.* PhD thesis, Universität Kaiserslautern, 2006.

A. Pascoletti and P. Serafini. Scalarizing vector optimization problems. *Journal of Optimization Theory and Applications*, 42:499–524, 1984. ISSN 0022-3239. URL `http://dx.doi.org/10.1007/BF00934564`.

M. Pietrzyk, L. Cser, and J. Lenard. *Mathematical and Physical Simulation of the Properties of Hot Rolled Products.* Elsevier Science, 1999. ISBN 0080427014.

J. Sjöberg, J. Lunding, and J. Öhr. Computional optimal trajectory of cart-pendulum system. Internal ABB report., 2010.

S. Utyuzhnikov, P. Fantini, and M. Guenov. A method for generating a well-distributed pareto set in nonlinear multiobjective optimization. *Journal of Computational and Applied Mathematics*, 223(2):820 – 841, 2009. ISSN 0377-0427. URL `http://dx.doi.org/10.1016/j.cam.2008.03.01`.

D. A. V. Veldhuizen. Multiobjective evolutionary algorithms: Classifications, analyses, and new innovations. Technical report, Evolutionary Computation, 1999.

A. R. Warburton. Quasiconcave vector maximization: Connectedness of the sets of pareto-optimal and weak pareto-optimal alternatives. *Journal of Optimization Theory and Applications*, 40:537–557, 1983. ISSN 0022-3239. URL `http://dx.doi.org/10.1007/BF00933970`.

A. Winterfeld. Interpolation techniques in hotpropt. Presentation held att ABB by Fraunhofer ITWM, 2010.

Y. Yun, H. Nakayama, and M. Arakawa. Generation of pareto frontiers using support vector machine. In *MCDM*, August 2004.