Design space exploration of RF-circuit blocks

Karer, E.

Award date:
2007

TECHNISCHE UNIVERSITEIT EINDHOVEN
Department of Mathematics and Computer Science

Design Space Exploration of RF-Circuit Blocks

by
E. Karer

Supervisors:

Prof. Dr. W.H.A. Schilders (TU/e)
Dr. ir. J.A. Croon (NXP Semiconductors)

Eindhoven, July 2007

Acknowledgments

This thesis was written in the framework of an internship at NXP Semiconductors. It describes the results of a six months master project. I was supervised by Prof. Dr. W.H.A. Schilders of NXP Semiconductors and the Technical University Eindhoven and furthermore, by Dr. ir. J. A. Croon of NXP Semiconductors.
Herewith, I want to express my deep gratitude to Prof. Schilders, who has guided me during the project and for proofreading of the thesis. Furthermore, I want to thank Dr. Croon sincerely for the helpful discussions, for the detailed corrections of the thesis and furthermore for the interesting introduction to semiconductor device modeling.
Additionally, I want to thank Univ.-Prof. Dipl.-Ing. Dr. H. Gfrerer of the Johannes Kepler University Linz for reviewing this work and for his useful suggestions during the project.

Contents

Abbreviations and Notation **ix**

1 Introduction **1**
1.1 Problem Statement and Aim of the Thesis 1
1.1.1 Reverse Modeling and Design Space Exploration 1
1.1.2 Problem Statement 3
1.1.3 Goals of the Project 4
1.2 Outline 5

2 Electrotechnical Background **7**
2.1 Introduction to the Semiconductor Design Process 7
2.2 RF-Circuit Block Models 9
2.3 Design and Performance Specifications of Low-Noise Amplifiers 11
2.4 Chapter Summary 14

3 Theory of Multiobjective Optimization **15**
3.1 Introduction 15
3.2 Basic Concepts in Multiobjective Optimization 17
3.2.1 Preference and Domination Relations 18
3.2.2 Pareto Optimality 20
3.3 Existence and Optimality Conditions 21
3.3.1 Existence of a Pareto Optimal Point 22
3.3.2 First-Order Optimality Conditions 23
3.4 Trade-Offs 27
3.5 Summary 28

4 Evolutionary Algorithms for MOPs **29**
4.1 General Framework of Evolutionary Algorithms 30
4.1.1 Probabilistic Genetic Operators 32
4.1.2 Fitness Function 34
4.1.3 Selection Operator 34
4.1.4 General Evolutionary Algorithm 35
4.2 Adaption of EAs to MOPs 36

4.3 Multiobjective Evolutionary Algorithms . . . 38
4.3.1 The Niched Pareto Genetic Algorithm - NPGA . . . 38
4.3.2 Pareto Envelope-Based Selection Algorithm - PESA . . . 39
4.3.3 Elitist Nondominated Sorting Genetic Algorithm - NSGA-II . . . 39
4.3.4 Summary and Conclusion . . . 40
4.4 The Strength Pareto Evolutionary Algorithm 2 . . . 40
4.4.1 Fitness Assignment . . . 42
4.4.2 Archive Truncation - Clustering . . . 43
4.4.3 Selection . . . 44
4.4.4 Recombination and Mutation . . . 44
4.4.5 Stopping Criterion . . . 46
4.5 Investigations on the Convergence Properties of SPEA2 . . . 46
4.5.1 Previous Results Available in Literature . . . 47
4.5.2 Preliminaries and Definitions . . . 48
4.5.3 Assumptions . . . 50
4.5.4 Mating Selection . . . 52
4.5.5 Recombination . . . 52
4.5.6 Mutation . . . 52
4.5.7 Environmental Selection - Clustering . . . 60
4.5.8 Summarizing the Convergence . . . 60
4.5.9 Experimental Verification of the Convergence Considerations . . . 61
4.6 Summary . . . 63

5 A Deterministic Approach for MOPs 65
5.1 Other Deterministic Solution Methods . . . 66
5.1.1 Weighting Method . . . 66
5.1.2 ε-Constraint Method . . . 67
5.2 Normal-Boundary Intersection Method . . . 68
5.2.1 Central Ideas of NBI . . . 69
5.2.2 Further Considerations on NBI . . . 71
5.3 Summary . . . 73

6 Applications 75
6.1 Implementation . . . 75
6.1.1 General Considerations on the Implementation . . . 75
6.1.2 Implementation of SPEA2 . . . 76
6.1.3 Implementation of the NBI . . . 80
6.2 Application to a Simple Example . . . 83
6.2.1 Test Environment . . . 84
6.2.2 SPEA2 with Exact Knowledge of Distance . . . 85
6.2.3 A Closer Look at the Mutation Parameters of SPEA2 . . . 87

6.2.4 Stopping Criterion of SPEA2 88
6.2.5 Normal Application of SPEA2 89
6.2.6 Comparison of SPEA2 to NBI 91
6.3 Application to a Performance Model of an LNA 93
6.3.1 Design Space Exploration of a Low-Noise Amplifier 93
6.3.2 Conclusions on the Design Space Exploration 98
6.4 Summary 99

7 Conclusion **101**
7.1 Conclusions 101
7.2 Suggestions and Future Work 102

Bibliography **105**

Abbreviations and Notation

RF	**R**adio-**F**requency	page 8
RSM	**R**esponse-**S**urface **M**odel	page 10
DSE	**D**esign **S**pace **E**xploration	Problem 1.1, page 2
MDSE	**M**ultiobjective **D**esign **S**pace **E**xploration	Problem 1.2, page 3
MOP	**M**ultiobjective **O**ptimization **P**roblem	Problem 3.1, page 16
SOP	**S**ingle Objective **O**ptimization **P**roblem	page 17
MOEA	**M**ulti**o**bjective **E**volutionary **A**lgorithm	Section 4.3, page 38
NPGA	**N**iched Pareto **G**enetic **A**lgorithm	Subsection 4.3.1, page 38
PESA	**P**areto **E**nvelope-Based **S**election **A**lgorithm	Subsection 4.3.2, page 39
NSGA-II	Elitist **N**ondominated **S**orting **G**enetic **A**lgorithm II	Subsection 4.3.3, page 39
SPEA2	**S**trength **P**areto **E**volutionary **A**lgorithm 2	Section 4.4, page 40
NBI	**N**ormal-**B**oundary **I**ntersection	Section 5.2, page 68
$\mathbb{R}^n$	n-dimensional Euclidean space	
$\mathbb{R}^+$	positive real numbers	
$\mathbf{p}$	performance figures (objective function)	page 2
t	number of performance figures	page 2
$\mathbf{d}$	design (input) parameters	page 2
n	number of design parameters	page 2
D	set of available (restricted) design parameters	page 2
Y	feasible objective values	Problem 1.1, page 2
k	number of performance figures to be optimized	page 3
X_f	feasible design parameters	Definition 3.2, page 16
P_a	attainable performance figures	Definition 3.2, page 16
P_f	Pareto optimal front	Definition 3.12, page 21
P_t	population at generation t	page 31
N	size of (normal) population P_t	page 31
$\bar{P}_t$	external set (archive)	page 41
$\bar{N}$	size of external set $\bar{P}_t$	page 41

p_i^*	individual minima	page 69
$\mathbf{p}^*$	shadow minimum/utopia point	page 69
$\mathbf{x}_i^*$	input parameter corresponding to p_i^*	page 69
Φ	pay-off matrix	page 69
$CHIM$	**C**onvex **H**ull of **I**ndividual **M**inima	page 69
$CHIM_\infty$	**C**onvex **H**ull of **I**ndividual **M**inima Infinity	page 70
$CHIM_+$	**C**onvex **H**ull of **I**ndividual **M**inima Plus	page 70
$\mathbf{n}$	normal vector in direction of $\mathbf{p}^*$	page 70
β	vector that determines the point on the CHIM	page 70
NBI_β	Single objective subproblem of NBI determined by β	page 71
n_β	step size for NBI-parameter β in each direction	page 71
$\tilde{\mathbf{n}}$	quasinormal vector in direction of $\mathbf{p}^*$	page 71

Chapter 1

Introduction

This thesis deals with the problem of exploring the design space of RF-circuit blocks. Out of the title several questions might emerge. We could ask, what design space exploration actually is, and furthermore what RF-circuit blocks are. The first question will be answered in this introductory chapter. The background of RF-circuit blocks will be treated later.

First, we will introduce the problem and the aims of this work in more detail. Afterwards, an overview of the remaining chapters will be given.

1.1 Problem Statement and Aim of the Thesis

In this section we will introduce the problem of exploring the design space of an RF building block (or any other system that behaves similarly) and additionally, the goals of this project will be stated. First, the general framework of this project will be described. After introducing the task of reverse modeling and pointing out the occurring difficulties when dealing with it, we will define the so-called *design space exploration*(DSE) problem. The reverse modeling problem can be seen as a starting point of this project, because it is the first question arising, when dealing with the inversion of functions. The later on defined DSE problem is a generalization of reverse modeling.
Then, we will first point out the general issues with solving the design space exploration problem. Based on the needs of end-users, i.e. RF-circuit designers, it will be justified to reduce the DSE problem to a related one, namely to a multiobjective optimization problem (MOP). This will be the general problem we are focusing on in this thesis. Finally in Subsection 1.1.3 the aims and goals of this project will be stated.

1.1.1 Reverse Modeling and Design Space Exploration

First of all, the general setting of this project will be explained and the initially stated problem will be discussed. Then it will be transformed into another quite interesting form,

i.e. the design space exploration problem, which will be the basis for further discussions and for the problem statement of the next subsection.

Generally speaking, the task of this project is to explore the design domain of a given model. We limit ourselves to so-called *Response-Surface Models* (RSM), which for our case are multidimensional functions built by polynomial, rational or spline interpolation. In other words, we assume to be given a model with a certain smoothness.

Generally speaking, the mathematical relations consist of t performance characteristics $\mathbf{p} = (p_i)_{i=1,\dots,t}$ as functions of n design parameters $\mathbf{d} = (d_j)_{j=1,\dots,n}$. The input parameters are restricted to a certain domain $D \subset \mathbb{R}^n$. In the framework of RSMs for RF-circuit building blocks those restrictions can be due to physical and practical limitations. Hence, the output domain P is given by

$$P = \mathbf{p}(D) := \{\mathbf{p} \,|\, \exists \mathbf{d} \in D \,:\, \mathbf{p} = \mathbf{p}(\mathbf{d})\} \subset \mathbb{R}^t \,. \tag{1.1}$$

The task of exploring the design space of a model is named reverse modeling. This is, because we are given the performance figures and we want to calculate the corresponding input parameters. The first task in the framework of reverse modeling that usually comes to our mind is the following: For a given performance vector $\mathbf{y} \in \mathbb{R}^t$, we want to compute the corresponding design or decision vector $\mathbf{d} \in D$, such that $\mathbf{p}(\mathbf{d}) = \mathbf{y}$. But unfortunately, this task is ill-posed in general, especially in higher dimensions. Typically one of the two following properties hold:

1. there does not exist any vector $\mathbf{d} \in D$, such that $\mathbf{p}(\mathbf{d}) = \mathbf{y}$ for any $\mathbf{y} \in \mathbb{R}^t$,
2. the solution $\mathbf{d}$ is not unique, i.e. there are at least two decision vectors that lead to the same performance vector $\mathbf{y}$.

The problems, mentioned before, are typical for higher dimensional functions, since the function $\mathbf{p}$ has to be surjective and injective to avoid these possibilities from happening. Of course, due to a restriction to the range of $\mathbf{p}$, i.e. P, surjectivity would be an immediate consequence and hence, the first point of the previous enumeration would never eventuate. Furthermore, the set P is not known for general models. Injectivity is even a bigger problem. It is equivalent to strictly increasing or decreasing monotonicity. Note, that especially polynomial and rational interpolation functions are not monotonic in general.

Consequently, due to the explanations given above, the inverse map should be regarded as a point-to-set mapping, i.e. $\mathbf{p}^{-1} : P \to \mathcal{P}(D)$, with $\mathbf{p}^{-1} : \mathbf{y} \mapsto \mathbf{p}^{-1}(\mathbf{y}) := \{\mathbf{d} \in D \,:\, \mathbf{p}(\mathbf{d}) = \mathbf{y}\}$. Since additionally P is not known explicitly, a subset of Y of $\mathbb{R}^t$ is chosen instead of a single performance $\mathbf{y}$. Hence, the problem of *design space exploration* can be stated as follows:

Problem 1.1 (Design Space Exploration). *For a given a set $Y \subset \mathbb{R}^t$, find the maximal set $X \subset D$, such that*

$$Y = \mathbf{p}(X) \,. \tag{1.2}$$

Additionally notice, that the design space exploration is the inverse problem of the so-called *performance space exploration*, which is to compute Y for given X. This problem is well-posed, if $\mathbf{p}$ is well-defined.

In the following subsection the difficulties of this problem will be stated. Furthermore, we will make some practical remarks, that are closely related to the task of circuit design. Based on this, Problem 1.1 will be transformed into a final form, namely into a multiobjective optimization problem. This problem provides the basis for all further investigations in this thesis.

1.1.2 Problem Statement

Problem 1.1 is in general very hard to handle. It is not clear at all how to treat it. One attempt to obtain at least a subset $\tilde{X}$ of X could be the following. Assuming simply connected domains[1] X and Y and continuity of $\mathbf{p}$, we could investigate only on the boundary of Y, i.e. on ∂Y. The corresponding values in the design space will cover than at least the performances of Y. One reason for this approach could be to reduce the dimensionality of the considered set by 1. If the function $\mathbf{p}$ is monotone in each direction, this method would obtain ∂X. But in general it is impossible to make any statement on those results.

Now, if we think of the work of a circuit designer. In his/her case, the function $\mathbf{p}$ describes the performance of an (in our case analog) electrical circuit. The designer wants the performance variables to lie in some certain range, i.e. the set Y. Additionally he/she is not interested in the whole feasible domain X, which would yield the complete feasible performance space Y, since the feasible domain Y contains maxima as well as minima of all performance variables. Usually, the designer is interested in an optimal behavior of the circuit. A possible example in the case of an amplifier could be, that the gain should be maximized, whereas the noise figure is to be minimized, and additionally the other performances should only behave nicely, i.e. they should be bounded. In such a case it is not necessary to compute the whole set of design variables X. Instead, it would be sufficient to compute only the boundary of the domain, which results in optimal values of those specific predefined performances. In other words, the aim could be to obtain the so-called *Pareto front*[2], i.e. we want to obtain a set of in a certain sense optimal values.

Due to the previous considerations, we will reduce our problem into one that is easier to handle and more useful for a circuit designer. A transformation of Problem 1.1, in the way explained above, leads to a so-called *multiobjective optimization problem* (MOP). Without loss of generality let us define the first k ($1 \leq k \leq t$) performance figures as the optimization variables. Then, we are able to state our new problem:

Problem 1.2 (Multiobjective Design Space Exploration). *Let $t, n, k \in \mathbb{N}$ with $1 \leq k \leq t$. Furthermore, given the connected compact domain $D \subset \mathbb{R}^n$, the smooth*

[1] A simply connected domain in $\mathbb{R}^n$ does not contain any holes.

[2] The basic theory to multiobjective optimization is treated in Chapter 3. The Pareto optimal front is given in Definition 3.12 on page 21.

function $\mathbf{p} : D \to \mathbb{R}^t$ *and the connected compact domain* $Y \subset \mathbb{R}^t$*. The multiobjective design space exploration (MDSE) problem is given by*

$$\min_{\mathbf{d} \in D} \mathbf{p_o} = \mathbf{p_o}(\mathbf{d}) := (p_1(\mathbf{d}), \ldots, p_k(\mathbf{d}))^T \tag{1.3}$$

subject to

$$\mathbf{p} = (p_1(\mathbf{d}), \ldots, p_t(\mathbf{d}))^T \in Y.$$

Note, that in the definition of Problem 1.2 we assumed without loss of generality, that all optimization performances are to be minimized. If otherwise, we have to multiply single or all functions by -1. Furthermore, the term smooth means, that $\mathbf{p}$ has to be at least continuous. If later on more smoothness is required, we will mention it at the respective position.

Furthermore, we limit the dimensions such that the solution approach is manageable, i.e. the dimensions of design space and corresponding range of the model should not exceed 10. This is a realistic assumption, since a few parameters determine already the main behavior of analog RF-circuit blocks.

The special difficulty of this problem is, that we are only given a function $\mathbf{p}$ in form of a black box. Hence, we are not able to compute the derivatives of $\mathbf{p}$ analytically. Furthermore, it is not possible to make general statements about the behavior of $\mathbf{p}$ without calculating its values. This means, that we do not now anything about monotonicity or other properties of the performance function, and hence we have to assume a certain regularity.

Additionally observe, that the relation of the performance space exploration and the multiobjective design space exploration. For this sake, we can trim the constraints of Problem 1.2 in the way, that we restrict the design variables to the set X and we do not prescribe any constraints on the performances $\mathbf{p}$. Then by optimizing the performances in certain directions we get an impression of the corresponding performance set Y, at least of its boundary ∂Y.

1.1.3 Goals of the Project

After transforming the reverse model via the definition of Problem 1.1 into our final Problem 1.2, we are now able to formulate the goals of this project:

- *Theoretical aspects.* Investigate on the theoretical properties of Problem 1.2 and on conditions that an optimal solution has to fulfill.
- *Design space exploration.* Find efficient ways to solve Problem 1.2.
- *Application.* Apply the approaches to RF-circuit block models to test the applicability.

1.2 Outline

In the previous section we explained the general problem of design space exploration and the transformation to a manageable one. Furthermore, the aims and goals of the project were explained. Hence, it is basically clear what the remaining tasks are. This leads to the following structure of this thesis.

As an introduction, the electrotechnical background of this project will be outlined in Chapter 2. First, the design flow for semiconductor devices will be explained. Furthermore RF circuit blocks will be introduced with focus on the low noise amplifier, and we will answer the question, why there is the need for compact models for this family of circuits. Furthermore an overview about the models and the reverse modeling in terms of compact RF circuit models will be given.

In Chapter 3 the theoretical aspects of multiobjective optimization problems (MOP) will be treated. After an introductory section, where the terms related to optimality of such problems are introduced, the existence of an, in some sense, optimal solution will be investigated. Additionally, we will present necessary and sufficient first order optimality conditions. Afterwards, the term trade-off will be mathematically defined. It provides insight into the drawback between single performances, which can be important information for the designer.

In order to solve the MDSE problem, we will choose two completely different methods, namely a probabilistic approach and a deterministic method.

Chapter 4 will treat the probabilistic approach. First, the basic concepts of so-called *evolutionary algorithms*(EAs) will be considered. The standard case, the optimization of a single function, i.e. single objective optimization problem (SOP), is used to examine the steps of the general algorithm. After this, the adaption of EAs from SOPs to MOPs will be discussed. Some examples of existing methods are given. The method of choice, the *Strength Pareto Evolutionary Algorithm 2* (SPEA2) is introduced in more detail. Finally the convergence properties of SPEA2 are investigated.

In Chapter 5 deterministic methods to deal with multiobjective optimization are presented. Again the method of choice, the *normal-boundary intersection*(NBI), is treated in detail and the relation to other methods will be stated.

Chapter 6 deals with the implementation details of both algorithms, SPEA2 and NBI. The methods are compared by means of a simple example, for which the exact solution is known explicitly. Furthermore, SPEA2 and NBI are applied to investigate the design space of a low noise amplifier.

Finally, Chapter 7 summarizes the work and presents our main conclusions. Moreover, suggestions for future work will be listed.

Chapter 2

Electrotechnical Background

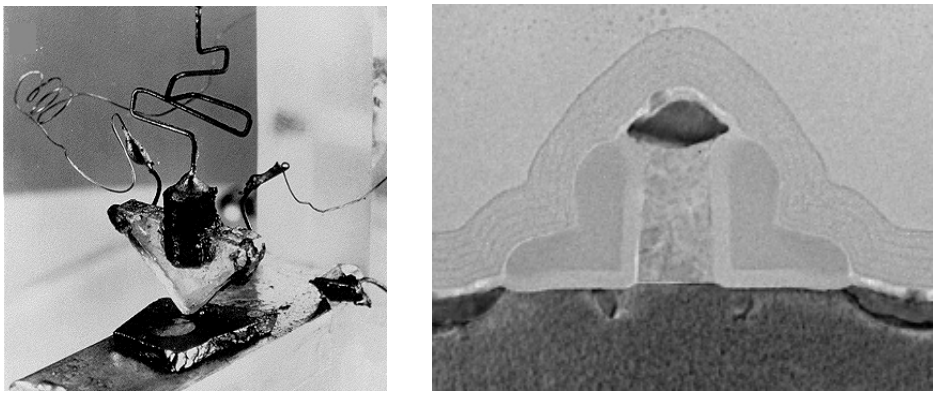
In the previous chapter we stated the problem we are dealing with in this thesis. We mentioned, that the models, i.e. the function **p**, describe the performance behavior of an electrical circuit in dependence of some design parameters. Therefore, this chapter aims to create some insight in the framework within this thesis is situated.

First a general introduction of the semiconductor design process is presented. Furthermore, the aim of the RF-building block modeling project will be stated. Later, the different components of an electrical RF-circuit will be listed. Furthermore, the two different directions of the circuit design flow are presented. The standard approach is to compute the performance of a circuit with given design parameters. In the other direction, the so-called *reverse modeling*, we compute the corresponding input parameters for given desired performances. As mentioned in Chapter 1 this is the main task of this work. In our specific case, the compact models will be so-called *response surface models*(RSM). Furthermore, the design and performance specifications of a *low noise amplifier* (LNA) will be explained. The LNA acts as the RF circuit block on which we will demonstrate the methods developed in this thesis.
Finally, this chapter will be summarized.

2.1 Introduction to the Semiconductor Design Process

In modern electronic applications integrated circuits consist of up to millions of components. There are so-called passive components, like resistors, capacitors and inductors. However, a circuit consists mainly of active devices, namely transistors. As an example for the improvements in technology, compare the first transistor which was developed in 1947, depicted in Figure 2.1(a), and an integrated 60nm NMOS transistor of the *Intel® Corporation* (shown in Figure 2.1(b)). The term 60nm is a design note and it denotes the (gate) width of the transistor. The state of the art is now at around 45nm.

The production of a chip consists of several disciplines. First, there is the process step development. Secondly, the process steps are used to produce the single devices and thirdly,



(a) (b)

Figure 2.1: The first transistor, invented in 1947 (left, found in [Lee]), and an integrated 60nm NMOS transistor of the Intel® Cooperation(right, found in [T+02]).

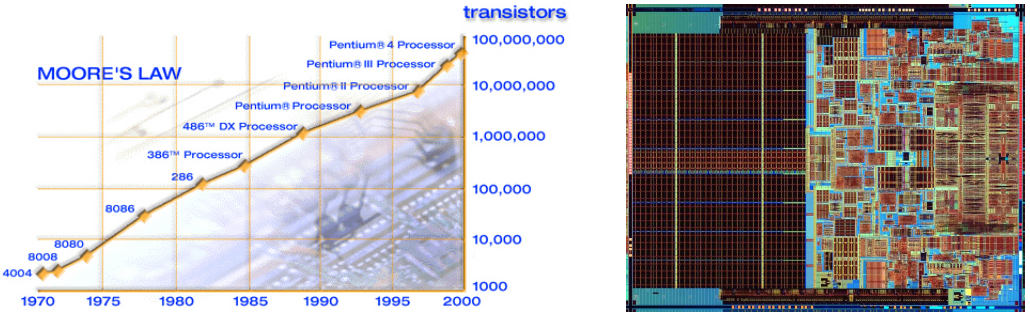
circuit blocks(cells) are made of the single devices.
Furthermore, the circuit design joins different types of basic circuit cells together. An electrical circuit mainly consists of digital cells(digital circuits), memory cells and an RF[1]/analog interface to the outside world, consisting of RF building blocks(RF front-end). So-called *standard cells* increase the design speed and the accuracy due to less design iterations. So far, such cells do only exist for digital circuits and memories, but not for RF building blocks. The reason for this lies in the different nature of digital and analog/RF design.

For digital circuits, the increasing amount of devices[2] makes it very hard to deal with the different influences of all components. For example, in Figure 2.2(a) Moore's law is applied to Intel®'s processors. It is seen that the number of transistors in Intel®'s processors increased from several thousands in the 1970's up to 100 millions and even more nowadays. Furthermore, to get an impression of the complexity of, especially digital, circuits, see Figure 2.2(b), where the photo of a Intel® Core™2 Extreme mobile processor die is depicted. For analog circuit blocks the design complexity does not arise from the number of transistors, which usually does not exceed 100 transistors. In this case the design complexity originates from the sensitivity to details in the design. While digital devices can be considered on or off, for analog design we need to take the full behavior of the devices into account, which makes it difficult to standardize.

The complexity of the whole design process means, that if the designs would be made purely based on experiments this would require many design iterations. This would be too expensive and too time consuming. Therefore special simulation tools are used in all phases of the design process. Usually, the impact of the production process on semiconductor devices is simulated via *process simulation* tools. Going one step further, single components are modeled and simulated, respectively, by use of *device simulation* packages.

[1] Radio-Frequency. It includes all signals that can be transmitted wireless. Therefore, frequencies from $10kHz$ to $300Ghz$ are regarded as RF.

[2] According to the famous Moore's law (cf. [Moo65]) the amount of transistors on an integrated circuit for minimum component cost doubles every 18 months.



(a) (b)

Figure 2.2: Moore's law applied to Intel's CPUs (left, see [M+02]) and a photo Intel® Core™2 Extreme mobile processor die (right, see [Int]).

The behavior of a whole circuit is treated with *circuit simulators* and on the highest level, *system simulation* tools are used to deal with whole systems.

The underlying models for single devices are based on semiconductor physics and are quite complicated. Using the same models in circuit simulation, it would result in too complex simulations and therefore, they would be too time consuming. Hence, compact models of devices are used. For digital systems again simplified models are used for cells. At this time, behavioral models for RF-circuit blocks are typically not very accurate, because they are not based on a standardized cell.

The aim of the RF-building block-modeling activity within NXP Semiconductors is to develop models that accurately describe the behavior of RF circuit blocks and offer insights in relevant design trade-offs. This activity is briefly introduced in the next section.

RF circuit blocks are (small) RF circuits that perform a specific function in a receiver for instance, e.g. low-noise amplifiers, mixers, oscillators, ... In the framework of this thesis we are dealing with such circuit blocks, focusing on LNAs. An LNA is an amplifier that has a very low noise figure NF (see Section 2.3), which means that almost no disturbances, so-called *noise*[3], are added to the amplified signal. For details see [Lee04] (especially chapter 12) or [LvdTV01].

2.2 RF-Circuit Block Models

In the following, we will outline the basic components of RF circuit block models. Furthermore, we will state two different ways to design a circuit. An electrical circuit model as defined in [CK07] consists of the following three main parts (see Figure 2.3(a)):

- *Circuit Part:* This part consists of the circuit topology modeled and its design parameters that are considered to be variable. Those will serve as input parameters for the model.

[3] According to [Lee04], the most general definition of noise is: "everything except the desired signal". Therefore, all disturbing signals are regarded as noise.

- *Electrical Representation:* It models the electrical behavior of the circuit and can be implemented in circuit simulators. It describes the relations between the occurring voltages and currents at the input and output, from which the performance specifications are derived. For the purpose of this thesis the electrical representation acts as a black box model to link the design parameters(input) to the performance specifications. This black box model consists of response-surface models (RSM), constructed by polynomial, rational or spline interpolation. The corresponding data points are obtained by circuit simulations for different parameter settings.

- *Performance Specifications:* Properties of interest that give the designer insight in the performance of the considered circuit. Performance figures are explained in Subsection 2.3. They are derived from the electrical behavior of the building block.

Additionally observe, that building-block models are only useful, if they are available for different topologies and several circuit block classes. Hence a lot of models have to be set up. By using physics-based models it would take several years to generate a suitable library. Consequently this technique is not applicable. Therefore, the RSM-approach is used in a first instance(cf. [CK07]).

In Figure 2.3 the structure of an overall model is depicted. Additionally to the already stated parts, there are two more:

- *Process Technology:* It defines the technology used to build the circuit. It can be based on device measurements or on simulations of certain technologies.

- *Circuit Environment:* This part mainly consists of the source and load conditions of the circuit block. Possible load and source conditions are the impedances of the source and the load (cf. Section 2.3).

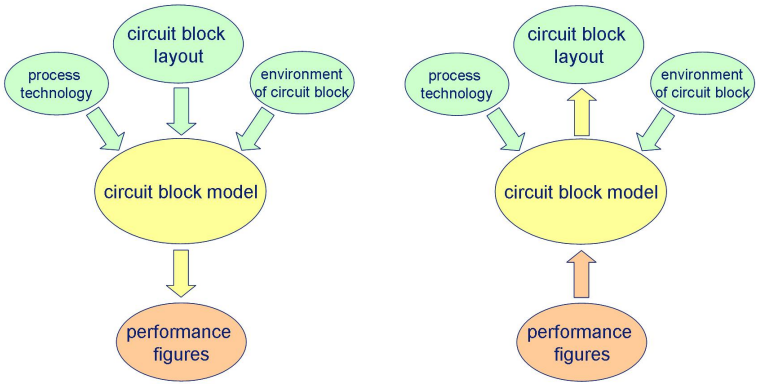


Figure 2.3: Two different operation modes for a model structure.

Furthermore, two different relationships between the parts of the model were established. In Figure 2.3(a) the usual flow of model development is depicted. Given the input parameters, the performance of the circuit can be computed.

But a more efficient or more natural way of designing a circuit is, that the designer prescribes certain performances that he/she wants to achieve. Then, the task is to compute the design parameters corresponding to this performance figures. This flow of modeling is displayed in Figure 2.3(b) and it is designated as *reverse modeling*. As explained already in Chapter 1 reverse modeling is the topic of this thesis.

2.3 Design and Performance Specifications of Low-Noise Amplifiers

In the following we will give a short summary of possible input(design) parameters and output parameters(performance figures) of LNAs. For more details on LNA design see chapter 12 of [Lee04]. Besides, typical values are presented for the parameters. Note, that in the framework of RSMs, a typical value for the design parameter is chosen. Then, the considered range covers a variation of one or two decades from this typical value. First, we will state some possible design (circuit) parameters:

- *Frequency f.* The frequency of the signal is determined by the application. It influences the magnitude of certain impedances (complex resistance), since the impedance of an ideal capacitor C and of an ideal inductor L are given by

$$Z_L = j\omega L \qquad \text{and} \qquad Z_C = \frac{1}{j\omega C}\,, \tag{2.1}$$

  where $\omega = 2\pi f$ denotes the radial frequency, j the imaginary unit, and L and C denote the inductance and the capacitance of the components respectively. In this work, a frequency of $5\,GHz$ is considered.

- *Width of the transistors W.* Typically W is in the range of $200\,\mu m$. Note, that those values are representative for LNAs, which are analog circuits. As noted above, the scales for digital transistor widths are up to a factor 1000 smaller.

- *Inductor L_s, used for source degeneration.* In amplifier design there are two different relations. The power transfer and the noise transfer. They determine the ratio between the power and noise, respectively, at the output and the input. In order to obtain optimal power transfer and near optimal noise transfer, the input impedance of the LNA needs to match the source impedance (usually $50\,\Omega$). L_s is used in *source degenerated LNAs*[4](cf. Figure 2.5(a) for the narrowband LNA, which is used for further considerations). It should drive the real part of the input impedance to the desired value. In standard source-degenerated LNAs we are considering L_s to be in the range $250\,pH$ for our purposes.

- *Inductor L_m, used for input matching.* The introduction of the inductor L_s drives the real part of the input impedance to a desired value. L_m is used to compensate

[4]The 's' in the notation L_s arises from the term "source"

the occurring imaginary parts for a certain frequency. The values of L_m are in the range of $2.5\,nH$. In Figure 2.5(b) a simple model, corresponding to the LNA shown on the left hand side of Figure 2.5, is depicted. Below we will compute the input impedance Z_{in} for this circuit, and by means of Z_{in} we will see the purpose of the inductors L_s and L_m.

- *Load impedance Z_l.* An imaginary resistance is called *impedance*. Its imaginary part corresponds to a phase shift of the signal. Ideal inductors and capacitors have purely imaginary impedances. Since every real load is not merely resistive, it has to be modeled in a general way. It is assumed that the magnitude of Z_l lies in between $20\,\Omega$ and $20\,k\Omega$.

In the following we will discuss some output variables, i.e. performance figures:

- *Power consumption P.* The power consumption determines the overall power consumption of the circuit. Usually P should be minimal. It is given in mW.

- *Available Power Gain G_A* is a measure of the power gain of the amplifier; usually $G_A > 10\,dB$.

- *Voltage gain A_v* is the ratio of output voltage divided by input voltage in a logarithmic scale. In typical applications A_v is greater than $18\,dB$.

- *Input reflection Γ_a.* Γ_a is the ratio between the reflected and transmitted signal at the input of the amplifier. Preferably it is as small as possible, hence the constraint $\Gamma_a < -10\,dB$ is typical.

- *Noise figure NF.* Every electronic device adds noise to the signal, even a simple wire. The ratio between total output noise power and the output noise due to the input source is termed the *noise factor F*. NF is F expressed in decibels. For a typical LNA we have $NF < 3\,dB$.

- *Intermodulation $IIP3$.* Ideally, an amplifier does not change the frequency behavior of a signal. Its only use is to amplify the signal to a certain extend, i.e. the transfer function should be linear in the signal v. In reality it is not. When expanding the transfer function with respect to v, especially the third order terms are very important. That is, because this term is responsible for a perturbation in frequencies close to the signals frequency. The *input-referred third-order intercept point $IIP3$* is a measure for the magnitude of this third order coefficient and it is usually greater than $0\,dBm$. In Figure 2.4 the determination of the $IIP3$ is sketched. Due to the power coefficient 3, the third-order terms grow faster than the first-order terms for increasing input, i.e. they have slope 3 compared to the slope 1 of the first order terms, if everything is viewed in logarithmic scales. Hence, the $IIP3$ is the intersection point of both terms for varying input power. For more details see Subsection 12.6 of [Lee04].

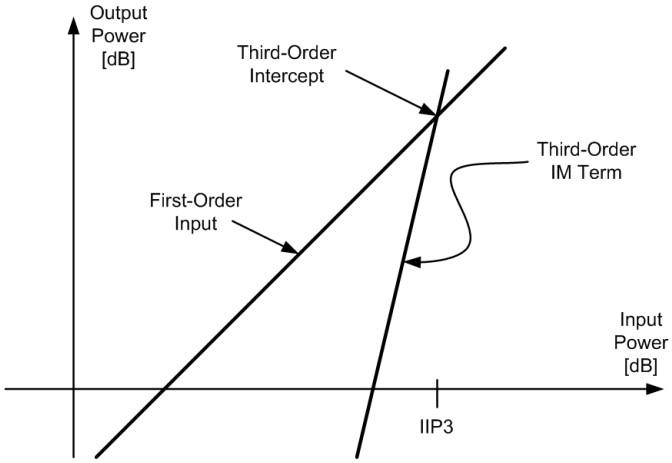


Figure 2.4: Illustration of the LNA performance parameter $IIP3$ concerning linearity.

- *Intermodulation* $IIP2$. Similar to $IIP3$, it determines the second-order nonlinearity. Since, the second order terms do not cause perturbations in close-by frequencies, they are not of main interest for the circuit designer. $IIP2 > 0\,dBm$ is usually required.

As indicated above, we will now compute the input impedance of the source degenerated LNA shown in Figure 2.5. Using the ohmic law[5] and (2.1), we obtain

$$v_{\text{in}} = i_{\text{in}} \left(j\omega L_m + \frac{1}{j\omega C_{gs}} + j\omega L_s \right) + g_m v_{gs}\, j\omega L_s \,, \tag{2.2}$$

where C_{gs} and v_{gs} denote the gate-source capacitance of the transistor and the voltage along this capacitor respectively. Furthermore, g_m terms the amplification factor of the voltage controlled current source. Using the relation

$$v_{gs} = \frac{i_{in}}{j\omega C_{gs}} \,,$$

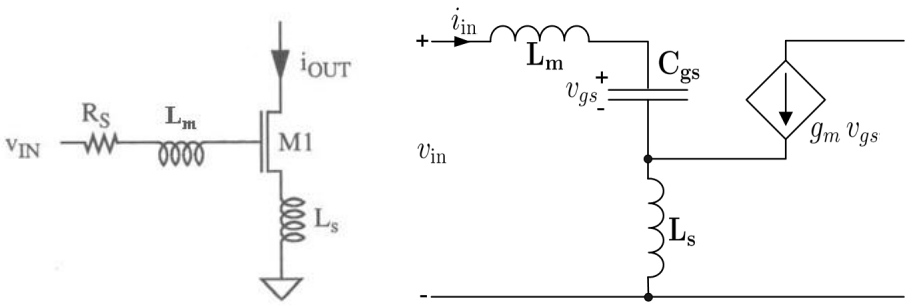
we get for the input impedance Z_{in} the following value:

$$Z_{in} = \frac{v_{in}}{i_{in}} = j\omega(L_m + L_s) + \frac{1}{j\omega C_{gs}} + g_m \frac{L_s}{C_{gs}} \,. \tag{2.3}$$

In (2.3) we see, that the last term of the sum is real. Hence by adjusting L_s we can trim the real value of the input impedance to the desired value. In addition, L_m can then be used to drive the imaginary part of Z_{in} for a certain frequency f_r to zero. Since for other frequencies, the input match and in addition the circuit's transfer function is not optimal any more, inductively source degenerated LNAs are mainly narrowband amplifiers.

Finally bear in mind, that Figure 2.5(b) is the most simple model for the depicted amplifier. In reality this model contains much more components.

[5]The ohmic law states, that the voltage v of a circuit is given by the current i times its resistance, i.e. impedance Z. Speaking in terms of formulas we have $v = i\,Z$.



(a) (b)

Figure 2.5: Narrowband low-noise amplifier with inductive source degeneration(left, cf. [Lee04] p. 378). Furthermore a simple model for this LNA is depicted on the right hand side.

2.4 Chapter Summary

In this chapter we briefly summarized and explained the electrotechnical background. We outlined the design flow of semiconductor devices. Furthermore, we pointed out, why there is a need for compact models for RF-circuit blocks. Additionally the basic parts of a model were explained. For our sake, we will use response-surface models, which map the design parameters to the electrical representation, from which the performance figures are derived analytically.

Furthermore, we also introduced the flow of reverse modeling in circuit design. This problem will be treated in the following chapters mathematically. Finally, some design and performance parameters for low noise amplifiers(LNAs) were discussed. Compact models for LNAs will be used in Chapter 6 to show numerical results of the introduced methods.

Chapter 3

Theory of Multiobjective Optimization

In Chapter 1 we introduced the problem of design space exploration(see Problem 1.1). Furthermore, we pointed out, that it is advisable to transform this problem into a multiobjective design space exploration problem, or in other words, into a multiobjective optimization problem, i.e. Problem 1.2. Furthermore, we outlined the electrotechnical reasoning behind this problem in the previous chapter. Now, we want to investigate on the theoretical background of multiobjective optimization. First, we will reformulate our problem into another form, i.e. we have to assume some constraint functions corresponding to the feasible design domain D of Problem 1.2. This will lead to the formulation given in Problem 3.1, which is the basis for all further investigations.

Note, that it is not clear at all, which feasible performances should be considered to be optimal and which of them are not optimal in some certain sense. This difficulty will be treated later. There, order relations will be defined, which will finally lead to the important definition of the so-called *Pareto optimality*. In this work, we will consider a solution to Problem 1.2 and Problem 3.1 respectively, as optimal, if it is Pareto optimal.
Furthermore, we will investigate on the existence of Pareto optimal solutions and for completeness, some first-order optimality conditions based on the Lagrangian formulation are listed. Finally, the important definitions of *trade-off* and *trade-off rate* will be introduced. They are very important in the framework of optimization along a trade-off front, since they determine the rate of drawback of one performance variable if we want to improve another one. The hard task for the circuit designer is then to balance this trade-off.

3.1 Introduction

To be able to state some analytical results, we have to assume that we are given the constraint functions that determine the sets D and Y, additionally to the requirements

made in Problem 1.2. That is, we are focused with a so-called *multiobjective optimization problem* (MOP). Formally it is defined by

Problem 3.1 (Multiobjective Optimization Problem). *Let k, l, m, $n \in \mathbb{N}$, $\mathbf{p} : \mathbb{R}^n \to \mathbb{R}^k$, $\mathbf{e} : \mathbb{R}^n \to \mathbb{R}^l$ and $\tilde{\mathbf{e}} : \mathbb{R}^k \to \mathbb{R}^m$. A general MOP includes n parameters (design variables) $\mathbf{d}$, k objective functions (performance figures) $\mathbf{p}$, l design space constraints $\mathbf{e}$ and m objective constraint functions $\tilde{\mathbf{e}}$. The optimization problem reads as*

$$\min_{\mathbf{d}\in\mathbb{R}^n} \mathbf{p} := \mathbf{p}(\mathbf{d}) = (p_1(\mathbf{d}), \ldots, p_k(\mathbf{d}))^T \tag{3.1}$$

subject to

$$\begin{aligned} \mathbf{d} &\in D := \{\mathbf{x} \in \mathbb{R}^n \,|\, e_i(\mathbf{x}) \leq 0,\, i = 1, \ldots, l\}\,, \\ \mathbf{p} &\in Y := \{\mathbf{p} \in \mathbb{R}^k \,|\, \tilde{e}_i(\mathbf{p}) \leq 0,\, i = 1, \ldots, m\}. \end{aligned}$$

Comparing Problem 1.2 with Problem 3.1 we see, that the performance figures not used for optimization, are transferred to the constraint functions $\mathbf{e}$. Additionally, the set Y has to be customized, i.e. the dimension of Y decreases from t to k. We assume that the feasible design and performance domains are defined by continuously differentiable functions, i.e. $\mathbf{e} \in C^1(\mathbb{R}^n, \mathbb{R}^l)$ and $\tilde{\mathbf{e}} \in C^1(\mathbb{R}^k, \mathbb{R}^m)$. We mentioned already in Section 1.1.2, that the performance function is smooth due to its origin. Hence, it is feasible to expect $\mathbf{p} \in C^r(\mathbb{R}^n, \mathbb{R}^k)$ for some $r \geq 1$. It is therefore safe to state, that for our situation Problem 1.2 reduces to Problem 3.1. In Figure 3.1 a schematic picture of the relation between the sets and functions occurring in Problem 3.1 is depicted. For simplicity we assumed linear constraint functions $\mathbf{e}_i$ and $\tilde{\mathbf{e}}_i$.

In the following, we define the set of feasible input parameters $\mathbf{d}$ and attainable performances $\mathbf{p}$:

Definition 3.2. *The feasible set X_f of the MOP, given by Problem 3.1, is defined by*

$$X_f := \{\mathbf{d} \in D \,|\, \mathbf{p}(\mathbf{d}) \in Y\}\ . \tag{3.2}$$

The set of attainable performance vectors P_a is given by

$$P_a := \{\mathbf{p}(\mathbf{d}) \,|\, \mathbf{d} \in X_f\}\ . \tag{3.3}$$

The sets X_f and P_a are defined to simplify the treatment in the next chapters. This is, because each vector $\mathbf{d} \in X_f$ is feasible and there exists a $\mathbf{y} \in P_a$ such that $\mathbf{p}(\mathbf{d}) = \mathbf{y}$. The other way around, we have similarly, that for every $\mathbf{y} \in P_a$ there exists at least one corresponding $\mathbf{d} \in X_f$.

In the following we will basically limit ourselves, especially for the theory, to convex functions and domains. For completeness, those definitions will be stated. Furthermore, the definition of a convex hull will be given, since it will be needed for the treatment of the normal-boundary intersection approach, which will be introduced in Section 5.2.

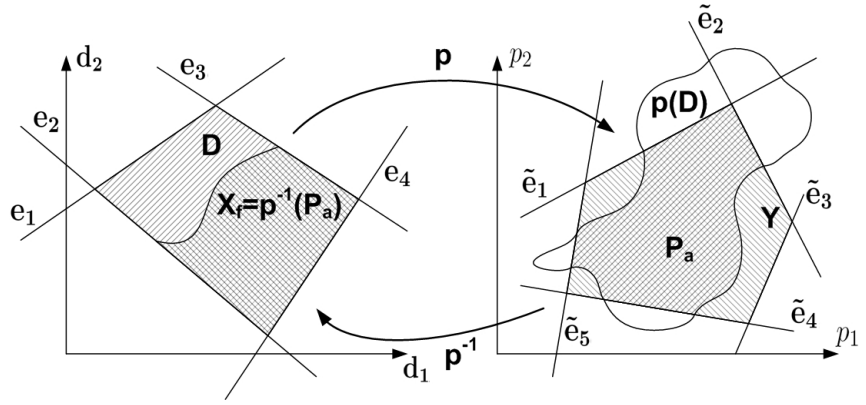


Figure 3.1: Illustration of the problem setting of a general MOP.

Definition 3.3. *Let $n \in \mathbb{N}$. A function $f : \mathbb{R}^n \to \mathbb{R}$ is called* convex *if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^n$ $f(\beta\mathbf{x} + (1-\beta)\mathbf{y}) \leq \beta f(\mathbf{x}) + (1-\beta)f(\mathbf{y})$ holds true for all $0 \leq \beta \leq 1$.*
A set $S \subset \mathbb{R}^n$ is convex *if $\mathbf{x}, \mathbf{y} \in S$ implies that $\beta\mathbf{x} + (1-\beta)\mathbf{y} \in S$ for all $0 \leq \beta \leq 1$.*
Furthermore, the convex hull $\mathrm{co}(S)$ of a finite set S is given by

$$\mathrm{co}(S) = \{\beta\mathbf{x} + (1-\beta)\mathbf{y} \,|\, \mathbf{x}, \mathbf{y} \in S,\ \beta \in [0,1]\} \ .$$

Convexity of functions and sets is important in optimization, since for a convex function defined on a convex set, the terms local and global minimum of a single objective optimization problem(SOP) coincide. Hence, if any solution method yields a stationary point, it is a global minimum, and hence the (numerical) solution process is simplified tremendously. For multiobjective problems the effects of convexity are similar, since then, the optimal front(see Definition 3.12) is connected.

3.2 Basic Concepts in Multiobjective Optimization

The problem we are concerned with, when dealing with performance vectors is, that it is not clear at all how to assess different vectors in comparison to each other. Generally speaking this means, that it is not obvious which order relation should be applied. If we would use lexicographic order, for instance, each element is comparable to each other. But this would imply that we rank the entries of the vectors, i.e. we would prefer single performance figures to others. But without preferring any performance function to another, we might get several solutions that cannot be compared to each other. Or in other words, the set P_a is not totally ordered by this relation. This leads to the terminology of *Pareto optimality*, which will be defined in Subsection 3.2.2.

But before, in the following subsection we have to introduce the concepts of reference and dominance relations, which are needed to set up the terminology of Pareto optimality. We will mainly follow [SNT85]. It deals with the theoretical background of multiobjective optimization problems.

3.2.1 Preference and Domination Relations

As mentioned above we want to define some order relation for the performance vectors or the corresponding design parameters. Before being able to order the elements of a set X in any way, we have to define the preference relation $\prec$.

Definition 3.4. *The binary relation $\prec$ is called* strict preference relation *on the set X. If $x \prec y$ for some $x, y \in X$, we say x is preferred to y. Consequently, we can define the relations $\precsim$ and $\sim$ as*

$$\begin{aligned} \mathbf{x} \sim \mathbf{y} \quad &:\Leftrightarrow \quad \neg(\mathbf{y} \prec \mathbf{x}) \ \wedge \ \neg(\mathbf{x} \prec \mathbf{y}) \\ \mathbf{x} \precsim \mathbf{y} \quad &:\Leftrightarrow \quad \mathbf{x} \prec \mathbf{y} \ \vee \ \mathbf{x} \sim \mathbf{y} \end{aligned}$$

The relation $\sim$ is called indifference *(x is indifferent to y) and $\precsim$ is called* preference-indifference relation *(y is not preferred to x).*

X can be any vector space. Later on, X will be the design space, i.e $X = \mathbb{R}^n$. Usually the preference (indifference) relation is an order relation. We use now these general definitions of relations to introduce order relations. But first we have to state some fundamental definitions for binary relations, which are used below to define the different order relations.

Definition 3.5. *Let R be a binary relation on the set X. Then R is said to be*

- irreflexive*, if* $\neg yRy$, $\forall y \in X$,
- transitive*, if* $\forall x, y, z \in X$*:* $xRy \wedge yRz \Rightarrow xRz$,
- negatively transitive*, if* $\forall x, y, z \in X$*:* $\neg xRy \wedge \neg yRz \Rightarrow \neg xRz$,
- asymmetric*, if* $\forall x, y \in X$*:* $xRy \Rightarrow \neg yRx$,
- weakly connected*, if* $xRy \vee yRx$, $\forall x, y \in X$, $x \neq y$.

In order to show the relations between the order relations, which will be defined below, we have to show, the following relations:

Lemma 3.6. *1. irreflexivity and transitivity imply asymmetry,*

2. asymmetry, transitivity and weak connectedness imply negative transitivity,

3. asymmetry implies irreflexivity,

4. negative transitivity and asymmetry imply transitivity.

In [SNT85] no proof for this lemma and no reference for a proof can be found. Hence, we show it in here.

Proof. Throughout the proof let $x, y, z \in X$ arbitrary but fixed. Furthermore, let R be the order relation on the set X.

1. Let xRy hold. Additionally assume yRx. Then, with transitivity we obtain xRx, what contradicts the irreflexivity. Hence $\neg yRx$ holds.
2. Let $\neg xRy$ and $\neg yRz$ hold. With the weak connectivity, we get xRy and yRz. Hence, with transitivity we obtain zRx. Asymmetry implies $\neg xRz$.
3. Assume yRy. Then with asymmetry we get $\neg yRy$, what contradicts the assumption.
4. We will show this implication by contradiction. We assume $\neg zRy$. Thus we have to show $\neg xRy \vee \neg yRz$. Therefore, we firstly assume yRz. Hence, with asymmetry we get $\neg zRy$. With negative transitivity $\neg xRy$ holds. Secondly, if xRy holds, we get similarly $\neg yRz$.

□

Now we are able to introduce the order relations. In general, there are three different types of orders(cf. [SNT85] p. 26 ff.), namely

Definition 3.7. *Let R be a binary relation on a set X. R is said to be a*

(i) strict partial order *if it is irreflexive and transitive,*

(ii) weak order *if R is asymmetric and negatively transitive, and a*

(iii) (strict) total order *if it is irreflexive, transitive, and weakly connected.*

The following connection between these order relations holds:

Corollary 3.8. *Let R be an order relation on a set X. The following holds:*

- *R is a weak order if it is a total order,*
- *if R is a weak order, it is a strict partial order too.*

Proof. The corollary follows from Lemma 3.6. □

Furthermore, if a set X is totally ordered by a relation R, then each element is comparable to each other. Additionally, we get that a bounded totally ordered set has a unique maximal (or respectively minimal) element. In the case of such a setting, there would exist a unique "optimal" value. As mentioned before, we are usually not confronted with this case in the framework of MOPs.

For MOP's especially the so-called *Pareto order*, or the *weak Pareto order* respectively, are of interest(see [SNT85] p. 30), since they are the most natural form of introducing a neutral order. Thereby the term neutral is related to variable preferences. The Pareto orders are defined by

Definition 3.9. *The* Pareto order *is defined by* $\prec := \leq$*, i.e. for* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$

$$\mathbf{x} \prec \mathbf{y} \;\Leftrightarrow\; \mathbf{x} \leq \mathbf{y} \;:\Leftrightarrow\; x_i \leq y_i \quad \forall i = 1, \ldots, k \quad \wedge \quad \mathbf{x} \neq \mathbf{y},$$

and the weak Pareto order *is defined by* $\prec := <$*, i.e. for* $\mathbf{x}, \mathbf{y} \in \mathbb{R}^k$

$$\mathbf{x} \prec \mathbf{y} \;\Leftrightarrow\; x_i < y_i \quad \forall i = 1, \ldots, k.$$

When solving an MOP, we want the optimal solutions $\mathbf{p}$ to be preferred to or indifferent to others in the performance space. But the constitutive values are the corresponding design vectors $\mathbf{d} \in X_f$, such that $\mathbf{p}(\mathbf{d}) = \mathbf{p}$. Therefore we have to adapt the definition of Pareto order, and define the *(weak) Pareto dominance.*

Definition 3.10. *Let* $\mathbf{x}, \mathbf{y} \in X_f$*. Then based on Problem 3.1, the Pareto dominance relation is defined by*

$$\mathbf{x} \prec \mathbf{y} \;(\mathbf{x} \textit{ dominates } \mathbf{y}) \quad :\Leftrightarrow \quad \mathbf{p}(\mathbf{x}) \leq \mathbf{p}(\mathbf{y})$$

Analogously to the Pareto dominance we can define the weak Pareto dominance

$$\mathbf{x} \prec \mathbf{y} \;(\mathbf{x} \textit{ weakly dominates } \mathbf{y}) \quad :\Leftrightarrow \quad \mathbf{p}(\mathbf{x}) < \mathbf{p}(\mathbf{y})$$

The indifference of two elements always depends on the used dominance relation, i.e.

$$\mathbf{x} \sim \mathbf{y} \;(\mathbf{x} \textit{ is indifferent to } \mathbf{y}) \quad :\Leftrightarrow \quad \mathbf{x} \not\prec \mathbf{y} \;\wedge\; \mathbf{y} \not\prec \mathbf{x}$$

Note, that in the previous definition, we are using the same symbol for Pareto and weak Pareto dominance. The used domination relation always follows from the context, namely it depends which Pareto order is used, either the weak or the normal Pareto order. Additionally bear in mind, that the set of weakly dominated elements of an element is smaller than the set of dominated elements.

3.2.2 Pareto Optimality

Above we defined the preference relation, which led to the different types of order relations. Then we introduced the (weak) Pareto order, which was extended to the (weak) Pareto dominance relation. Using this setup, we are now able to define an optimal solution of Problem 3.1.

Definition 3.11. *A point* $\hat{\mathbf{d}} \in X_f$ *is said to be a* (weak) Pareto optimal solution *to Problem 3.1 if there is no* $\mathbf{d} \in X_f$ *such that* $\mathbf{d} \prec \hat{\mathbf{d}}$*, i.e. if all points* $\mathbf{d} \in X_f$ *do not (weakly) dominate* $\hat{\mathbf{d}}$ *(cf. Definition 3.10).*

The previous definition was stated for Pareto optimal and weak Pareto optimal solutions. Again, it depends on the used relation $\prec$ (compare to Definition 3.10). Note, that for an

optimal front with some constant performance variables, only the minimum of the other variables will be accepted when Pareto optimality is used, while we consider all solutions along this front to be weakly optimal. An example to illustrate the difference between these definitions is shown in Figure 3.2(a).

The definitions 3.10 and 3.11 are also quite often used for vectors in the range of the performance functions. That is, a vector $\tilde{\mathbf{p}} \in Y$ (weakly) dominates $\bar{\mathbf{p}} \in Y$, if $\tilde{\mathbf{p}} \leq \bar{\mathbf{p}}$ ($\tilde{\mathbf{p}} < \bar{\mathbf{p}}$). Similar we define the optimality of two performance vectors $\tilde{\mathbf{p}}$, $\bar{\mathbf{p}} \in Y$.

In Definition 3.11 we defined *global Pareto optimal* solutions. Additionally, there exists the local (weak) Pareto optimality. Intuitively its definition is clear, since a decision vector $\mathbf{x}^*$, i.e. design vector, is called *locally (weak) Pareto optimal* if it is (weak) Pareto optimal in $D \cap B(\mathbf{x}^*, \delta)$ for some $\delta > 0$, where $B(\mathbf{x}^*, \delta)$ denotes the sphere with center $\mathbf{x}^*$ and radius δ. Note the relation between local and global optimal points for single objective optimization. As mentioned previously, the terms coincide for convex functions which are defined on convex domains. Similar, we obtain equivalence of Pareto and local Pareto optimal solution, if the attainable set P_a is convex. An additional similarity is, that we cannot expect the solutions obtained by (especially deterministic) numerical solution methods to be global Pareto optimal[1].

In general, the task is to compute all Pareto optimal solutions. The set of all optimal solutions is called the *Pareto front*.

Definition 3.12. *The* Pareto optimal front *(or in short Pareto front) is defined to be the set of all feasible (weak) Pareto optimal solutions, i.e.*

$$P_f := \{\tilde{\mathbf{p}} \in Y \mid \exists \mathbf{d} \in X_f \,:\, \mathbf{p}(\mathbf{d}) = \tilde{\mathbf{p}} \wedge \nexists \mathbf{d} \in X_f \,:\, \mathbf{p}(\mathbf{d}) \leq \tilde{\mathbf{p}}\,(\mathbf{p}(\mathbf{d}) < \tilde{\mathbf{p}})\} \tag{3.4}$$

The Pareto front can vary if either the weak Pareto or the Pareto optimality condition is used (cf. Figure 3.2(a)). We will basically focus on weak Pareto optimal solutions, since in the framework of design space exploration we are more interested in the trade-off between certain design parameters than on the strict Pareto optimal solutions. In Figure 3.2(b) a typical example of a convex Pareto front is depicted for a two-dimensional objective space.

3.3 Existence and Optimality Conditions

In this section, we will first investigate the existence of Pareto optimal solutions. We will see below that the existence of at least one Pareto optimal point is guaranteed. Afterwards, in Subsection 3.3.2 we will bridge the gap to single objective optimization. We will state some first-order necessary and sufficient optimality conditions for multiobjective optimization problems, which are the counterpart to the optimality conditions based on the Lagrangian formulation for SOPs. Note, that this section was only added for completeness.

[1]This holds true for the method that will be introduced in Section 5.2. An exception of this fact is the weighting method(see Subsection 5.1.1), where the obtained solution is always global Pareto optimal.

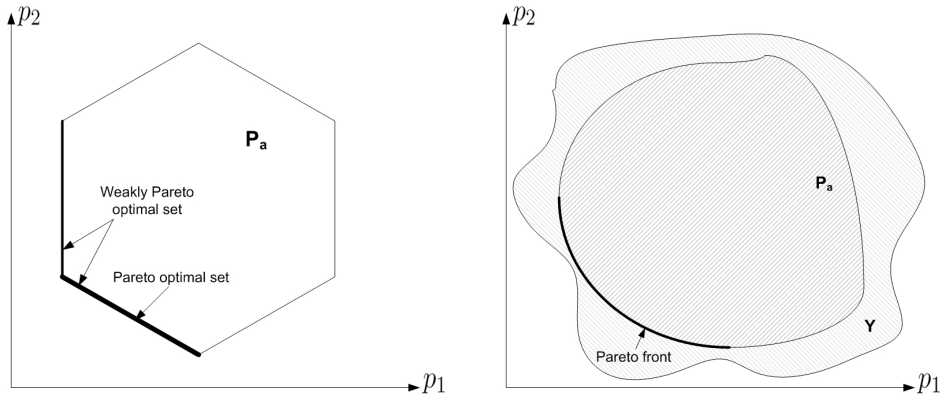


Figure 3.2: Illustration for Pareto optimal and weak Pareto optimal solutions(left). On the right, a typical example for a convex Pareto front is shown.

3.3.1 Existence of a Pareto Optimal Point

First of all, the question arises whether there exists a Pareto optimal front P_f, i.e. $P_f \neq \emptyset$. For Theorem 3.14, which states existence of a Pareto optimal solution of the multiobjective optimization problem, we need the definition of lower semicontinuity.

Definition 3.13. *A function $p_i : \mathbb{R}^n \to \mathbb{R}$ is lower semicontinuous (l.s.c.), if for any $\alpha \in \mathbb{R}$ $p_i^{-1}((\alpha, \infty))$ is an open set.*

Similarly we could define upper semicontinuity. Note, that continuity of a function implies semicontinuity. Now, we are able to state the theorem (cf. Corollary 3.2.1 of [SNT85] (p. 59)).

Theorem 3.14. *Let $\mathbf{p} = (p_1, \ldots, p_k)^T$ be a vector valued function from $\mathbb{R}^n$ into $\mathbb{R}^k$. Let X_f be a nonempty compact set in $\mathbb{R}^n$ and each p_i $(i = 1, \ldots, n)$ be lower semicontinuous on X_f. Then the Multiobjective Optimization Problem 3.1 has a Pareto optimal solution $\mathbf{d} \in X_f$.*

The proof of Theorem 3.14, as carried out in [SNT85], uses several definitions and lemmas and therefore, it is not so easy to follow. But the proof can also be carried out in a different way. Thereby, use is made of the properties of compact sets and lower semicontinuous functions. The proof is as follows:

Proof. Since we are dealing with finite dimensional domains, compactness is equivalent to boundedness and closeness. Since, X_f is compact and p_i is lower semicontinuous we have

$$\exists \mathbf{x}_{0,i} \in X_f : \quad p_i(\mathbf{x}_{0,i}) = \inf\{p_i(\mathbf{x}) : \mathbf{x} \in X_f\}. \tag{3.5}$$

This holds true since with $a_i := \inf\{p_i(\mathbf{x}) : \mathbf{x} \in X_f\}$, $A_{n,i} := \{\mathbf{x} \in X_f : p_i(\mathbf{x}) \leq a_i + \frac{1}{n}\}$ is closed. Now assume, that $\cap_{j=1}^{\infty} A_{j,i} = \emptyset$. This would imply that $A_{n,i}^C := X_f \backslash A_{n,i}$ is an

open cover of X_f. Hence, there would exist a finite subcover[2] $A^C_{n_k}$. Due to the finiteness of the n_k this cover cannot contain all elements of X_f which contradicts the assumption. Therefore we get for each $i \in \{1, \ldots, k\}$ the so-called *individual minima* $\mathbf{x}_{0,i}$. All of them are already weak Pareto optimal points. If two individual minima $\mathbf{x}_{0,i}$ and $\mathbf{x}_{0,j}$ have the same value in the performance space in either the i-th or the j-th component, we have to discard either $\mathbf{x}_i$ or $\mathbf{x}_j$. The elements left over are a subset of the Pareto optimal front with respect to the Pareto dominance. At least one element "survives" this elimination procedure. □

Note as general rule of thumb, that we need closeness and boundedness of X_f and continuity of the performance figures $\mathbf{p}$ to guarantee the existence of Pareto optimal points. The problems we are dealing with in this work are assumed to fulfill these requirements and hence we always suppose the existence of at least one Pareto optimal point.

Another question, that might arise is, if the Pareto optimal solution is unique. This is only for special cases true, i.e. in cases where the Pareto optimal solution lies in an acute corner. Therefore, we will not perform further investigations on this question.
In the following we will turn our concentration on first-order optimality conditions for (weak) Pareto optimal points.

3.3.2 First-Order Optimality Conditions

For the investigations of this subsection we will follow [Mie99], which treats nonlinear multiobjective optimization problems and states several deterministic solution methods for the solution of Problem 3.1. We will outline necessary and sufficient first-order conditions for (weak) Pareto optimality based on the Lagrangian description of the optimization problem. Thereby, the terminology first-order means as in the context of SOPs, that first derivatives are used to set up the conditions. This shows the relation to single objective optimization, where the optimality conditions are often used to set up solution approaches for those problems.

Since the general description of an MOP, needed for establishing optimality conditions, does not contain any restriction on the vector $\mathbf{p}$, i.e. the constraint $\mathbf{p} \in Y$, we have to customize our problem. In other words, we have to shift the constraints on $\mathbf{p}$, i.e. the functions $\tilde{\mathbf{e}}$, into the set D. Hence, for the sake of optimality conditions our main Problem 3.1 has to be rewritten in the equivalent form:

Problem 3.15. *Let k, l, m, $n, \mathbf{p}, \mathbf{e}$ and $\tilde{\mathbf{e}}$ be defined as in Problem 3.1. Problem 3.1 can be rewritten as*

$$\min_{\mathbf{d} \in \mathbb{R}^n} \mathbf{p} := \mathbf{p}(\mathbf{d}) = (p_1(\mathbf{d}), \ldots, p_k(\mathbf{d}))^T \tag{3.6}$$

[2]The general definition of compact sets is, that for each cover A_n, $n \in \mathbb{N}$ of a compact set A, there exists a finite subcover A_{n_k}, such that $\cup A_{n_k} \supset A$.

subject to

$$\mathbf{d} \in D := \{\mathbf{x} \in \mathbb{R}^n \,|\, (e_i(\mathbf{x}) \leq 0,\, i = 1,\, \ldots,\, l) \wedge (\tilde{e}_i(\mathbf{p}(\mathbf{x})) \leq 0,\, i = 1,\, \ldots,\, m)\}$$

The so-called *active constraints* are very important in constrained optimization and they always need a special treatment. The active constraints are the ones, that fulfill the constraint $(\mathbf{e}_j(\mathbf{x}^*) \leq 0)$ with equality. That is, for continuous non-redundant constraints $\mathbf{x}^*$ lies on the boundary of D. Similarly we can define the active performance constraints. We denote the sets of active constraints at a point $\mathbf{x}^*$ by $J(\mathbf{x}^*)$ and $\tilde{J}(\mathbf{x}^*)$. They are defined as follows

$$J(\mathbf{x}^*) := \{j \in \{1,\, \ldots,\, l\} \,|\, e_j(\mathbf{x}^*) = 0\} \tag{3.7}$$

$$\tilde{J}(\mathbf{x}^*) := \{j \in \{1,\, \ldots,\, m\} \,|\, \tilde{e}_j(\mathbf{p}(\mathbf{x}^*)) = 0\}\,. \tag{3.8}$$

In the following we will give necessary and sufficient conditions of first-order for (weak) Pareto optimality. For our problem Theorem 3.1.1 of [Mie99] has to be customized and additionally we combine it with Corollary 3.1.2, which claims that the conditions hold for weak Pareto optimality too.

Theorem 3.16 (Fritz John necessary condition for (weak) Pareto optimality). *Let the objective and constraint functions* $\mathbf{p}$, $\mathbf{e}$ *and* $\tilde{\mathbf{e}}$ *of Problem 3.15 be continuously differentiable and furthermore, let* $\mathbf{x}^* \in D$. *A necessary condition for* $\mathbf{x}^*$ *to be (weak) Pareto optimal is that there exist vectors* $\mathbf{0} \leq \lambda \in \mathbb{R}^k$, $\mathbf{0} \leq \mu \in \mathbb{R}^l$ *and* $\mathbf{0} \leq \tilde{\mu} \in \mathbb{R}^m$ *for which* $(\lambda,\, \mu,\, \tilde{\mu}) \neq (\mathbf{0},\, \mathbf{0},\, \mathbf{0})$ *such that*

$$\sum_{i=1}^{k} \lambda_i \nabla p_i(\mathbf{x}^*) + \sum_{j=1}^{l} \mu_j \nabla e_j(\mathbf{x}^*) + \nabla \mathbf{p}(\mathbf{x}^*)^T \left(\sum_{j=1}^{m} \tilde{\mu}_j \nabla_{\mathbf{p}} \tilde{e}_j(\mathbf{p}(\mathbf{x}^*)) \right) = \mathbf{0}\,,$$

$$\mu_j e_j(\mathbf{x}^*) = 0 \qquad \textit{for all} \;\; j = 1,\, \ldots,\, l\,,$$

$$\tilde{\mu}_j \tilde{e}_j(\mathbf{p}(\mathbf{x}^*)) = 0 \quad \textit{for all} \;\; j = 1,\, \ldots,\, m\,.$$

For the proof [Mie99] refers to other sources. Note, that the additional constraints $\tilde{\mathbf{e}}$ do not change the problem. We stated them explicitly to point out the additional constraints on the performances. In the following theorems this holds true again, since $\tilde{\mathbf{e}}$ can always be appended to the "normal" constraints $\mathbf{e}$. The gradients in Theorem 3.16 are assumed to be row vectors and $\nabla \mathbf{p}$ is defined by $(\nabla \mathbf{p}(\mathbf{x}))_{ij} := \frac{\partial p_i}{\partial x_j}(\mathbf{x})$.

Beside the Fritz John optimality conditions for single objective optimization, the *Karush-Kuhn-Tucker* optimality conditions are quite common as well. Their difference to the Fritz John conditions for single objective optimization problems(SOPs) is, that the multiplier λ is assumed to be positive. This should emphasize the important role of the objective function for optimization. To guarantee the positivity of λ some regularity constraints are needed.
Similarly the Karush-Kuhn-Tucker optimality conditions for MOPs guarantee λ to be not

equal to zero. For this sake we need again some so-called *constraint qualifications*. We will follow [Mie99] and define the *Kuhn-Tucker constraint qualification*. Due to the special form of our constraints, we have to adapt this definition as follows:

Definition 3.17. *Let the objective and constraint functions* $\mathbf{p}$, $\mathbf{e}$ *and* $\tilde{\mathbf{e}}$ *of Problem 3.15 be continuously differentiable at* $\mathbf{x}^* \in D$. *The problem satisfies the* Kuhn-Tucker constraint qualification *at* $\mathbf{x}^*$ *if for any* $\mathbf{d} \in \mathbb{R}^n$ *such that* $\nabla e_j(\mathbf{x}^*)^T\mathbf{d} \leq 0$ *for all* $j \in J(\mathbf{x}^*)$ *and* $\nabla_{\mathbf{p}}\tilde{e}_j(\mathbf{p}(\mathbf{x}^*))^T\nabla\mathbf{p}(\mathbf{x}^*)\mathbf{d} \leq 0$ *for all* $j \in \tilde{J}(\mathbf{x}^*)$, *there exists a function* $\mathbf{a} : [0, 1] \to \mathbb{R}^n$ *which is continuously differentiable at 0, and some scalar* $\alpha > 0$, *such that*

$$\begin{aligned} &\mathbf{a}(0) = \mathbf{x}^*, \\ &\mathbf{e}(\mathbf{a}(t)) \leq 0 \quad \textit{and} \quad \tilde{\mathbf{e}}(\mathbf{p}(\mathbf{a}(t))) \leq 0 \qquad \textit{for all } 0 \leq t \leq 1 \quad \textit{and} \\ &\mathbf{a}'(0) = \alpha\mathbf{d}\,. \end{aligned}$$

Now, let us have a closer look on the previous definition. Therefore, we assume for simplicity linear constraints. The gradient ∇e_i points outwards of the feasible set D, or X_f respectively. Furthermore, the hypersurface defined by e_i is perpendicular to ∇e_i. Hence, $\mathbf{d}$ points to the interior. This is at least valid for an arbitrary small distance and therefore we get non-positive constraint functions e_j in this area. In this case the function $\mathbf{a}$ is given by $\mathbf{a} = \mathbf{x}^* + t\alpha d$ with some suitable α.
They are also fulfilled, if we assume $\mathbf{x}^*$ to lie on a corner of D, then there is an infeasible region within this 180° domain. But this infeasible domain is discarded by the other active constraints. For an illustration of two different points fulfilling the Kuhn-Tucker constraint qualification see Figure 3.3. Additionally note, that the conditions of Definition 3.17 are satisfied, if the constraints are polynomials, or convergent power series. Hence, the Kuhn-Tucker constraint qualification are satisfied for problems with smooth constraint functions.

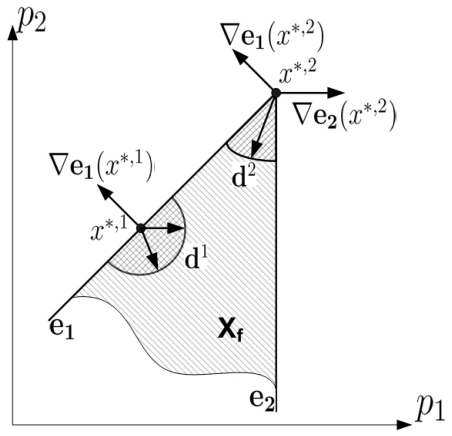


Figure 3.3: Example of two points $\mathbf{x}^{*,1}$ and $\mathbf{x}^{*,2}$ satisfying the Kuhn-Tucker constraint qualification.

By assuming some regularity of the constrained domain in $\mathbf{x}^*$, given by Definition 3.17, we are able to state stricter necessary conditions, i.e.

Theorem 3.18 (Karush-Kuhn-Tucker necessary condition for (weak) Pareto optimality). *Let the assumptions of Theorem 3.16 be satisfied by the Kuhn-Tucker constraint qualification defined in Definition 3.17. Theorem 3.16 is then valid with the addition that* $\lambda \neq \mathbf{0}$.

The proof of the Pareto optimality is given in [Mie99](p. 39). A reference for the proof of the weak Pareto optimality is quoted therein. So far, we considered necessary conditions for weak Pareto and Pareto optimality. Additionally it is desirable to have some sufficient optimality conditions. In the following theorem they will be alleged. It is obvious, that the restrictions have to be tightened again to guarantee an optimal solution.

Theorem 3.19 (Karush-Kuhn-Tucker sufficient conditions for Pareto optimality). *Let the objective functions* $\mathbf{p}$ *and the constraint functions* $\mathbf{e}$ *and* $\tilde{\mathbf{e}}$ *of Problem 3.15 be convex and continuously differentiable at a decision vector* $\mathbf{x}^* \in D$. *A sufficient condition for* $\mathbf{x}^*$ *to be Pareto optimal is that there exist multiplier* $\mathbf{0} < \lambda \in \mathbb{R}^k$, $\mathbf{0} \leq \mu \in \mathbb{R}^l$ *and* $\mathbf{0} \leq \tilde{\mu} \in \mathbb{R}^m$ *such that*

$$\sum_{i=1}^{k} \lambda_i \nabla p_i(\mathbf{x}^*) + \sum_{j=1}^{l} \mu_j \nabla e_j(\mathbf{x}^*) + \nabla \mathbf{p}(\mathbf{x}^*)^T \left(\sum_{j=1}^{m} \tilde{\mu}_j \nabla_{\mathbf{p}} \tilde{e}_j(\mathbf{p}(\mathbf{x}^*)) \right) = \mathbf{0}\,,$$

$$\mu_j e_j(\mathbf{x}^*) = 0 \qquad \textit{for all } j = 1, \ldots, l\,,$$

$$\tilde{\mu}_j \tilde{e}_j(\mathbf{p}(\mathbf{x}^*)) = 0 \quad \textit{for all } j = 1, \ldots, m\,.$$

The necessary Fritz John conditions and Karush-Kuhn-Tucker first-order optimality conditions presented in Theorem 3.16 and Theorem 3.18 do not distinguish between weak Pareto and Pareto optimality, i.e. the conditions hold for the weak and strong optimality term. Hence, especially for the Pareto optimality the restrictions are not as tight as they probably could be. To the contrary, the sufficient conditions differ for the two terms. While the stricter requirements for Pareto optimality are stated in Theorem 3.19, the sufficient qualification for $\mathbf{x}^* \in D$ to be weak optimal is given by

Theorem 3.20 (Karush-Kuhn-Tucker sufficient conditions for weak Pareto optimality). *The condition in Theorem 3.19 is sufficient for a decision vector* $\mathbf{x}^* \in D$ *to be weakly Pareto optimal for* $\mathbf{0} \leq \lambda$ *with* $\lambda \neq \mathbf{0}$.

The proofs of theorems 3.19 and 3.20 can be found in [Mie99]. Furthermore we remark, that due to the convexity condition we obtain global optimal solutions, while the theorems 3.16 and 3.18 are valid for locally as well as for globally optimal points.

Additionally to the first-order optimality conditions, there exist second-order requirements as well. Therefore, the objective and constraint functions have to be twice continuously differentiable. On the one hand they reduce the set of candidate solutions produced by the first-order solutions, but on the other hand they prescribe stronger regularity conditions to the problem. In [Mie99](p. 42 ff.) necessary and sufficient second-order conditions for Pareto optimality are mentioned. We do not treat them here, since they are not of main

concern for our problem, especially because this subsection should only provide additional information to fill some gaps.

3.4 Trade-Offs

In the way we are dealing with multiobjective optimization problems, namely the approximation of the Pareto optimal front, it is of big interest to know about the trade-off of one performance p_i corresponding to a change of another one, e.g. p_j. Therefore, different terms were introduced in the past. Following [Mie99] (p. 26 ff.) we introduce the *partial* and the *total trade-off*.

Definition 3.21. *Let* $\mathbf{d}^1$ *and* $\mathbf{d}^2 \in X_f$ *be two feasible design vectors corresponding to Problem 3.1. We denote the ratio of change between the functions* p_i *and* p_j *by*

$$\Lambda_{ij} = \Lambda_{ij}(\mathbf{d}^1, \mathbf{d}^2) = \frac{p_i(\mathbf{d}^1) - p_i(\mathbf{d}^2)}{p_j(\mathbf{d}^1) - p_j(\mathbf{d}^2)}\,, \tag{3.9}$$

where $p_j(\mathbf{d}^1) - p_j(\mathbf{d}^2) \neq 0$.
Now, Λ_{ij} *is called a* partial trade-off*, involving* p_i *and* p_j *between* $\mathbf{d}^1$ *and* $\mathbf{d}^2$ *if* $p_l(\mathbf{d}^1) - p_l(\mathbf{d}^2) = 0$ *for all* $l = 1, \ldots, k,\ l \neq i, j$. *If* $p_l(\mathbf{d}^1) \neq p_l(\mathbf{d}^2)$ *for at least one* $l = 1, \ldots, k$, *and* $l \neq i, j$, *then* Λ_{ij} *is called a* total trade-off*, involving* p_i *and* p_j *between* $\mathbf{d}^1$ *and* $\mathbf{d}^2$.

Note, that the desired case is the partial trade-off, because then the performance depends only on one other objective, while a total trade-off is quite hard to survey due to the dependence on other performance variables. Note that for a two dimensional performance space these terms coincide. Emanating from the definition of the total trade-off a similar terminology can be introduced. Let $\mathbf{d}^* \in X_f$ and let $\tilde{\mathbf{d}}$ be a direction emanating from $\mathbf{d}^*$, then the *total trade-off rate* at $\mathbf{d}^*$ is given by

$$\lambda_{ij} = \lambda_{ij}(\mathbf{d}^*, \tilde{\mathbf{d}}) = \lim_{\alpha \to 0^+} \Lambda_{ij}(\mathbf{d}^* + \alpha\tilde{\mathbf{d}}, \mathbf{d}^*)\,. \tag{3.10}$$

If for all $l = 1, \ldots, k,\ l \neq i, j$ p_l is constant in direction $\tilde{\mathbf{d}}$ in a vicinity of $\mathbf{d}^*$, then we call λ_{ij} *partial trade-off rate*. Furthermore, if the objective function is continuously differentiable, we have

$$\lambda_{ij} = \frac{\nabla p_i(\mathbf{d}^*)^T \tilde{\mathbf{d}}}{\nabla p_j(\mathbf{d}^*)^T \tilde{\mathbf{d}}}\,, \tag{3.11}$$

where again, the denominator differs from zero. Finally in the case of continuously differentiable functions, we can define the partial trade-off rate as follows.

Definition 3.22. *Let the performance functions* $\mathbf{p}$ *be continuously differentiable at a decision vector* $\mathbf{d}^* \in X_f$. *Then a* partial trade-off rate *at* $\mathbf{d}^*$*, involving* p_i *and* p_j*, is given by*

$$\lambda_{ij} = \lambda_{ij}(\mathbf{d}^*) := \frac{\partial p_i(\mathbf{d}^*)}{\partial p_j}\,. \tag{3.12}$$

For investigations on the trade-off in the case of DSE we will use Definition 3.22. Note, that for a numerical treatment we have to exploit Definition 3.21 because we are not given an analytical expression for the derivative of RSMs with respect to other performances and hence, the trade-off has to be computed numerically.

3.5 Summary

In this chapter we presented the basic theory for multiobjective optimization. The general MOP, we are dealing with, was stated in Problem 3.1. Thereby, we introduced the constraint functions $\mathbf{e}$ and $\tilde{\mathbf{e}}$, which limit the sets D and Y. This definition of the multiobjective optimization problem(MOP) serves as the basis for further mathematical investigations.
Furthermore, we defined the relations for multi objectives, the Pareto dominance and Pareto optimal solutions. Subsequentially, the existence of at least one Pareto optimal solution was treated and additionally necessary and sufficient first order optimality conditions were listed. These conditions align on the optimality conditions for single objective optimization problems, which are based on the Lagrangian description of the nonlinear constraint optimization problem.
Finally, we introduced the term trade-off rate. This is an important expression for dealing with MOPs, because it gives an impression of the drawback of one performance figure with respect to an improvement of another objective function.

Chapter 4

Evolutionary Algorithms for Multiobjective Optimization Problems

In principle, there exist two totally different approaches to solve any kind of optimization problems, namely deterministic and probabilistic approaches. There is a big variety of deterministic methods available in literature. Most of them were established for single objective optimization, like Trust-Region methods or Line Search with different choices of the search direction. In general, probabilistic solution methods are termed *evolutionary algorithms*.

As outlined in Chapter 1, our aim is to explore the design space of an RSM. Therefore, we stated and explained the transformation of Problem 1.1 to a general multiobjective optimization problem, i.e. Problem 3.1. In the following we want to choose a deterministic and a probabilistic method to solve Problem 3.1. Furthermore we want to compare these methods to each other by means of the application.

In this chapter we will state the setting of evolutionary algorithms. First, the standard approach, which is the application to single objective optimization problems, will be explained. We will outline the basic concepts of fitness, mutation, recombination and selection. Afterwards, we will point out the differences and give remedies for the difficulties, when applying evolutionary algorithms to multiobjective problems. Furthermore, the used method will be explained in detail and additionally convergence of this method under certain assumptions is shown. Especially, we could not find any analytical investigation on the convergence of the used probabilistic method for MOPs in the literature. Therefore, we will adapt existing results and we will apply them to our approach. Finally, a summary will conclude this chapter.

4.1 General Framework of Evolutionary Algorithms

In most of the modern applications certain quantities have to be optimized. Therefore, most often deterministic approaches are utilized. They usually make use of the gradient of the sought function and hence it is possible to obtain faster than linear convergence. Using some knowledge about the smoothness, it is even possible to succeed in finding an optimum without knowing the function of concern analytically. But sometimes these methods fail, because they get captured in stationary points or they cannot override stationary points between the actual iteration point and the global optimum. In such cases a wise choice of the initial solution can help to remedy these problems. But quite often it is very difficult or even impossible to find a suitable initial solution, e.g. if the constraints on the performances are very strict and if there are additionally very weak or almost no constraints on the design variables.

Especially the mentioned problems caused various scientists to investigate different ways to solve optimization problems (cf. [Gol89, B$\ddot{9}$6, ZT98, HNG94, SD94, ...]). Motivated by natural incidents, methods based on probabilistic considerations were developed. Nowadays, there exist a multitude of different types of them. Basically all these variations are termed *evolutionary algorithm* (EA). The basic concept of EAs is, that an initial population, also called generation, changes through recombination and mutation. Afterwards in a selection step, the best or fittest individuals of the new population survive. This helps, since a fitter individual indicates a smaller distance to an optimal value. From this short explanation we can already imagine the vast number of variations of evolutionary algorithms. We will focus on those that are suitable for the task of design space exploration. In general it is not possible to make any statements about the best method, since they all have their advantages and disadvantages. Usually it depends on the application which one to choose.

In the following we will give a brief introduction to evolutionary algorithms. Furthermore, basic definitions are presented. They are mainly based on [B$\ddot{9}$6].

Definition 4.1 (General Evolutionary Algorithm). *An* Evolutionary Algorithm (EA) *is defined as an 8-tuple*

$$EA = (I,\ f,\ \Omega,\ \Psi,\ s,\ \tau,\ N,\ \lambda)$$

where I is the space of individuals. $f : I \to \mathbb{R}$ denotes a fitness function. *Ω is a set of probabilistic genetic operators $\omega_{\Theta_i} : I^\lambda \to I^\lambda$, which are controlled by specific parameters summarized in the set Θ_i. s denotes the* selection operator*, which may change the number of individuals from λ or $\lambda + N$ to N with $\lambda, N \in \mathbb{N}$. N is the number of parent individuals, while λ denotes the number of offspring individuals. Furthermore is τ a termination criterion, and $\Psi : I^N \to I^N$ describes the complete process of transforming a population into a subsequent one, by applying the operators s and ω_{Θ_i}.*

As mentioned previously, EAs are population based approaches. An initial population P_0 transforms due to the overall operator Ψ to a subsequent one P_1 and so on. Thereby

denotes the subscript t the generation index. In the following we will denote by P_t the set of individuals at generation t, also called population at time t. The number of individuals in P_t equals N. In general, the behavior of EAs is determined by the operators f, s and the probabilistic genetic operators ω_{Θ_i}, but we have to bear in mind, that Definition 4.1 is based on single-objective optimization problems. In Figure 4.1 the flow of a general evolutionary algorithm is depicted. The operator ω_{Θ_i} generates λ offspring. Then they are assessed by the fitness assignment f. Afterwards, the operator s selects N individuals to form the next generation.
In the next subsections, the three steps within the main loop will be investigated. Later on, in Section 4.2, we will adapt the occurring ingredients of this definition to our purpose of MOPs.

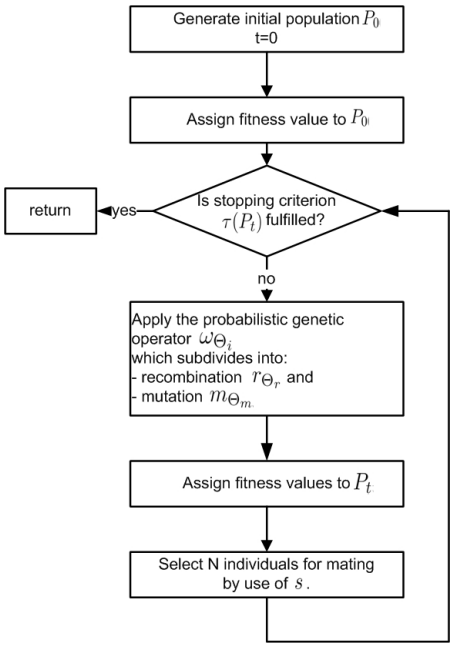


Figure 4.1: Flow chart of a general evolutionary algorithm.

There are a lot of different possibilities to choose the operators to the single steps. Which one is the best, always depends on the specific problem. Three different basic concepts to choose these operators and the space of individuals I were developed(cf. [B̈96]), namely *genetic algorithms*(GAs), *evolution strategies*(ES) and *evolutionary programming*(EP), which will be introduced in Subsection 4.1.4. The only thing we need to know now is, that the set of individuals I can be discrete or continuous, i.e. it can either be a binary string ($I = \{0, 1\}^l$, $l \in \mathbb{N}$) or I is given by $I = \mathbb{R}^n$.

Before going into more detail, we need to have a closer look on each step of the flow sketched above. We start with an arbitrary initial population P_0. This can be generated through any available method. For more details see Section 6.1. Then we assess this population by means of the fitness assignment. Afterwards we proof if the stopping criterion[1] is fulfilled. If the requirements are not met by now, we enter the loop. The single steps of the loop,

[1] see again Section 6.1

will be investigated in more details below.

4.1.1 Probabilistic Genetic Operators

The first operator that is applied in the loop, is the probabilistic genetic operator ω_{Θ_i}. Note, that for the choice of ω_{Θ_i} we distinguish between combinatorial and continuous optimization. Furthermore consist probabilistic genetic operators of two different steps, which are *recombination* and *mutation*:

Recombination

The first step in the procedure is recombination. The recombination operator r_{Θ_r} is usually a probabilistic operator.
For continuous problems it is basically a linear combination of two individuals. First two individuals $\mathbf{x}, \mathbf{y} \in P_t$ are chosen at random and then the new individual is generated by a random linear combination $\chi\mathbf{x} + (1 - \chi)\mathbf{y}$, with χ being uniformly distributed, i.e. $\chi \sim \mathcal{U}([a, b])$, where typically $a = 0$ and $b = 1$. Thereby it is also possible to choose different coefficients χ_i for the different dimensions i. Additionally, the probability p_c can be introduced. Then, recombination is only conducted with probability p_c. On the other hand, with probability $1 - p_c$ no recombination is executed. Then, an individual of P_t is selected randomly and passed on to the next step of the procedure.

For discrete problems the so-called *crossover* is usually used. With some probability p_c two individuals[2] $\mathbf{x}$ and $\mathbf{y}$ are chosen. Then, one or several distinct ordered points $p_i \in \{1, \ldots, l - 1\}$ are randomly selected[3]. Up to the first point p_1, inclusive it, the bits from the first candidate are copied to the new individual. From the first to the second point, or to the end if it is only a one-point crossover, the bits of the second candidate are copied to the new one. Then the bits of the first individual are used again, and so on.
As an example, consider a one-point crossover with $\mathbf{x} = 10100110$, $\mathbf{y} = 00011110$ and $p_1 = 3$. Then the new individual $\mathbf{z}$ is given by $\mathbf{z} = 101|11110$. Note, that with this method, using suitable points, almost every possible combination can be generated, i.e. from all zeros to all ones.

Mutation

Also in the second step of the genetic probabilistic operator, the mutation, discrete and continuous methods differ. In general, the mutation of an individual, in the continuous case, is to add an arbitrary vector $\mathbf{z}$ to the existing one $\mathbf{x}_{\text{old}}$, i.e. the new individual $\mathbf{x}_{\text{new}}$ is given by

$$\mathbf{x}_{\text{new}} = \mathbf{x}_{\text{old}} + \mathbf{z}.$$

[2] If only one individual is chosen, no crossover is applied. $p_c \in [0.6, 0.95]$ is suggested in [B̈96](p. 114).
[3] l denotes the length of the binary string.

We can choose almost every distribution for the random variable $\mathbf{z}$. But some of them are advantageous to others. In the following we will examine three different possibilities. They are chosen according to literature(cf. [B$\ddot{\text{o}}$96, MSV93] and the references given therein). The considered possibilities are

1. $\mathbf{z}$ can be a uniform distribution, i.e. $\mathbf{z} \sim \mathcal{U}([-A,\, A])$, whereby the range A of the distribution has to be defined in dependence of the domain of interest.

2. Another possibility is to choose a normal distribution $\mathbf{z} \sim \mathcal{N}(0,\, \Sigma)$. In that case the choice of the covariance matrix Σ is very crucial, and has to be done in a smart way to obtain optimal results.

3. The third method was introduced in [MSV93]. The algorithm is called *Breeder Genetic Algorithm* (BGA)[4]. It uses a special type of mutation, which looks for continuous problems as follows:

   First, we have to determine a typical value A_i for the range in each direction i. Furthermore we have to constitute an integer k, typically $k = 15$. Then the algorithm chooses one of the $2(k+1)$ points

   $$\pm(2^{-k}A_i,\, \ldots,\, 2^{0}A_i)$$

   for each direction i. The probability of a point to be chosen is uniform, i.e. $1/2(k+1)$. Note, that you can only achieve an accuracy of $2^{-k}A_i$ in each direction. But after an adaption of the A_i it is possible to achieve any accuracy we want.

   Another important fact is the influence of k. The higher we choose k the more smaller mutations are preferred and especially, we obtain

   $$\lim_{k\to\infty} P(\mathbf{z} \in B(\mathbf{0},\, \varepsilon)) = 1\,,$$

   for all $\varepsilon > 0$. The last considerations imply, that the higher k the lower is the expected value of the mutation in every direction i.

Later on, the adapted normal distributed and Breeder mutation procedures will be applied to MOPs. The quite natural uniformly distributed mutation will not be used, since it is more important to choose a suitable A in dependence of the distance to the optimum, than to choose Σ of the normal distribution or A in the case of Breeder-mutation. Furthermore, uniform distribution is uncommon in literature.

On the other hand, the mutation approach used for combinatorial problems is, to invert every bit with a probability of p_m, the so-called *mutation probability*. Often $p_m = 1/l$ is suggested[5] as a guideline, but also explicit values are mentioned, i.e. $p_m \in [0.001,\, 0.01]$ (cf. [B$\ddot{\text{o}}$96], p. 113).

[4] One might associate a discrete method with the name genetic algorithm. But in fact, in this article the easy generalization to continuous problems of the mutation procedure is explained.

[5] l denotes the length of the binary string.

4.1.2 Fitness Function

The fitness function f assigns a so-called fitness value to each individual of the population. The fitness value expresses the quality of the individual. Without loss of generality a lower fitness value is better than a higher one. Hence, for minimization problems (SOP) it is quite common to use the objective function as the fitness assignment or a slight adaption, e.g. some scaling of the objective function.

4.1.3 Selection Operator

Selection operators may be probabilistic or deterministic. Two typical examples for deterministic selection are the following

- The $(N + \lambda)$-*selection* selects the N best individuals out of the union of parents and offspring to form the next parent generation.
- The (N, λ)-*selection* selects the N best individuals out of the offspring. Therefore $\lambda > N$ is required. In the case of $N = \lambda$ no selection takes place and the EA acts like a random walk.

At first glance, the $(N + \lambda)$-selection seems advantageous against the (N, λ)-selection. But especially because of the better capability to overcome local optima in multimodal topologies(cf. [B̈96], p. 79) the (N, λ)-selection is recommended.
On the other hand, two possible probabilistic selection schemes are

- The most famous probabilistic selection scheme is the *q-tournament selection*, with $q \in \mathbb{N}$. Basically, the tournament selection chooses randomly q individuals. There are two possibilities for the set of possible candidates. Either we can choose only from the offspring, which are generated by recombination and mutation, or we can select from the union of parent generation and offspring. Out of this q candidates, only one individual, namely the best one in terms of the fitness function is copied to the next population. This procedure is repeated N times, until the next generation contains N individuals, which are the winners of the q-tournament selection. Note, that for increasing q the tournament selection converges to an $(N+\lambda)$- or an (N, λ)-selection, depending on the set we are selecting from.
- Another probabilistic method, called *proportional selection*, is to introduce a selection probability depending on the fitness value. For this purpose, a higher fitness value has to correspond to better individuals. Additionally, the fitness values have to be positive, i.e. $f : I \to \mathbb{R}_+$. The selection probability p_s of individual $\mathbf{x}_{i,t}$ is then defined by
$$p_s(\mathbf{x}_{i,t}) := \frac{f(\mathbf{x}_{i,t})}{\sum_{j=1}^{N} f(\mathbf{x}_{j,t})}.$$

The probabilistic selection schemes were listed, since they are the most common ones used in practice. The q-tournament selection is the most common approach used, especially for continuous methods. Investigations on it were made in [BT95].
On the other hand, the proportional selection is a quite natural one and hence, it is usually used for discrete problems (cf. [B̈96] p. 117 ff.).

4.1.4 General Evolutionary Algorithm

In the last subsections, the single steps of a general evolutionary algorithm corresponding to the flow chart depicted in Figure 4.1 were discussed. Thereby, the focus was on SOPs. Summarizing the previous investigations, we are able to state a general evolutionary algorithm:

Algorithm 4.2 (General Evolutionary Algorithm).

Input: N *population size*
Θ *parameters for termination, selection, recombination and mutation*
Output: P_t *final population that fulfills the stopping criterion* τ
$t = 0$;
initialize $P_0 = \{\mathbf{x}_{1,0}, \ldots, \mathbf{x}_{N,0}\}$;
evaluate fitness of P_0: $\{f(\mathbf{x}_{1,0}), \ldots, f(\mathbf{x}_{N,0})\}$;
while $\tau(P_t) \neq$ *true*
  recombine $P'_t = r_{\Theta_r}(P_t)$;
  mutate $P''_t = m_{\Theta_m}(P'_t)$;
  evaluate fitness of P''_t: $\{f(\mathbf{x}''_{1,t}), \ldots, f(\mathbf{x}''_{N,t})\}$;
  select: $P_{t+1} = s(P''_t \cup Q)$;
  $t = t + 1$;
end while

Note, that the set Q in the selection step can be the parent generation P_t (e.g. for $(\lambda+N)$-selection), or the the empty set (for (λ, N)-selection for instance).

Above, we introduced the three different concepts of evolutionary algorithms, namely Genetic Algorithms(GA), Evolution Strategies(ES) and Evolutionary Programming(EP). But up to now, we did not explain these terms. We only outlined the difference in the space of individuals, namely that for GAs $I = \{0, 1\}^l$, $l \in \mathbb{N}$, while for ES and EP $I = \mathbb{R}^n$. Hence, genetic algorithms are usually used for discrete optimization problems. But notice, that every continuous MOP can be transformed through discretization of I to a combinatorial problem. Hence, the set I is no limitation in general. Nevertheless bear in mind, that for continuous optimization problems a discretization is usually avoided, since it limits the accuracy of the approximation. That is why we insist on the distinction between the discrete GA and the continuous ES and EP.

Following [B̈96], the continuous methods ES and EP differ essentially in the selection and the probabilistic genetic operators from each other. Basically both use the same fitness

assignment and the same mutation operator, namely some kind of normal distribution. But they differ in the selection and recombination steps. While the evolution strategies use deterministic selection operators and a kind of linear combination for recombination, in EP no recombination is used at all, hence $\lambda = N$. Furthermore, usually tournament selection is used in evolutionary programming, where the candidates are chosen from the union of parents and offspring.

In genetic algorithms the proportional selection together with crossover and the special mutation method for discrete problems, which was introduced above, is used most often.

Of course, due to the big amount of people that were, and that still are, investigating on evolutionary algorithms, all kinds of combinations of these operators were tried out. In general, it is possible to combine all different operators to set up an evolutionary algorithm. But the ones mentioned previously are the most common in literature (cf. [B$\ddot{9}$6, BT95, MSV93, ...]).

4.2 Adaption of Evolutionary Algorithms to Multiobjective Optimization Problems

In this section we will show, how evolutionary algorithms have to be adapted to be applicable to multiobjective optimization problems, and which concepts are needed to obtain a suitable solution method.

First of all bear in mind, that evolutionary algorithms are population-based approaches. That is, we try to obtain N approximations to Pareto optimal points. The first task is to define the goals we have. Basically there are two important aims.

- Firstly, we want to approximate the Pareto front as accurate as possible. In other words this means, that it is desired to obtain a population, of which all points are as close as possible to the Pareto optimal front P_f.
- Secondly, the population that we obtain by the method should cover most of the desired front, i.e. the in some sense most outer individuals should be close to ∂P_f. Furthermore, the approximated points should exhibit a certain distribution. To be more precise, they should be uniformly distributed over the Pareto front. It should be avoided, that the resulting set accumulates around certain points.

To achieve this aims, we want to apply an evolutionary approach. Thereby, we encounter an additional goal. Namely, the solution method should be as fast as possible. Usually the speed of an algorithm is determined by the computational complexity. In the framework of evolutionary algorithms, the complexity is usually given by $\mathcal{O}(c(N)N^p)$, with some exponent $p \in \mathbb{R}_+$ and a value c that might depend on N again. This value is then usually

of lower order in N than the term N^p. The complexity as given above means, that the number of basic calculations[6] $B_C(N)$ is limited by

$$B_C(N) \leq \alpha\, c(N)\, N^p\,, \tag{4.1}$$

for $N \to \infty$ with some $\alpha \in \mathbb{R}_+$.

First of all, we exclude the goal of low complexity. Hence, for now the goals are to achieve accurate and diverse approximations. For this task, especially the fitness assignment of evolutionary algorithms has to be redesigned in order to be capable of MOPs. It has to be altered anyway, because the assignment as used for SOPs deals with only one objective function.
In the following we will introduce two concepts which are important for a successful MOEA.

To obtain a certain diversity of the individuals, it is necessary to apply the so-called *fitness sharing*, first suggested by Goldberg ([Gol89]). Thereby, the function values of several individuals are used to compute the fitness values. The typical approach is to degrade the fitness of individuals that have a large number of individuals in a certain vicinity. For a typical fitness assignment that applies fitness sharing see Subsection 4.3.1 and for more details [SD94, HNG94, FF95, Zit99, ...].
Additionally to the concept of fitness sharing, use is made of the Pareto dominance relation to set up the fitness values for each generation. Thereby, individuals that dominate others get a better fitness value than the dominated individuals. For the implementation of this strategy, a lot of different approaches were developed (cf. [SD94, HNG94, FF95, ZLT01, ...]).

Another strategy, called *elitism* is quite common in MOEAs (cf. [Zit99, ZLT01, DAPM00, CKO00]). Originally, elitism is the concept to use the b fittest individuals of generation t in the next generation $t+1$ again. Clearly, the $(\lambda+\mu)$-selection as introduced in Section 4.1.3 applies elitism.
This approach could be easily transferred to MOEAs, if we are given a suitable fitness assignment procedure. Note, that it is sometimes implemented (cf. [DAPM00]). Another quite popular strategy to apply elitism, is to maintain an external set, also called archive, in which the nondominated individuals of the union of archive and actual generation are stored.
Note, that the concept of elitism aims in faster convergence of the procedure, what corresponds to more accurate approximations of Pareto optimal points. Faster convergence is obtained, since nondominated individuals corresponding to the actual best approximations cannot be deleted, what might happens if elitism is not used.

Additionally to the fitness assignment, the selection step to fill the mating pool is sometimes adapted. Especially, if an archive is sustained.
In the following section we will introduce some available methods, that are aware of the two concepts of fitness sharing and elitism.

[6] Usually addition, multiplication and logical operators are denoted as basic calculations.

4.3 Multiobjective Evolutionary Algorithms

Now, we want to outline some existing multiobjective evolutionary algorithms (MOEA). Since most of them are basically defined by one or two steps, we will only mention the important ones. Usually, the difference of these methods lies in the fitness assignments.

In Subsection 4.3.1 the Niched Pareto GA will be outlined. Afterwards, in Subsection 4.3.2 the Pareto Envelope-Based Selection Algorithm PESA will be explained. The Elitist Non-dominated Sorting Genetic Algorithm NSGA-II will be discussed in Subsection 4.3.3. Finally, in the concluding subsection of this part, we will summarize the properties of these methods. Furthermore, they will be compared by means of available literature and their advantages and disadvantages will be outlined. Then, the conclusion will be to use another evolutionary algorithm called the Strength Pareto Evolutionary Algorithm 2 (SPEA2), which will be discussed in detail in the Sections 4.4 and 4.5.

4.3.1 The Niched Pareto Genetic Algorithm - NPGA

This approach was introduced in [HNG94] and it is a typical representative of MOEA. It uses the *Pareto domination tournament*, a method similar to tournament selection. First, 2 individuals are picked at random. Then, t_{dom} individuals are chosen at random to form the *comparison set*. The winner of this tournament is the individual that is dominated by less elements of the comparison set compared to the other candidate. In the case of equality in the Pareto domination tournament, fitness sharing, as generally explained in Section 4.2, is applied.
Thereby, the objective fitness f_i (it can be any fitness value as long as higher fitness indicate better individuals) is divided by the *niche count* m_i of the individual i. m_i counts the individuals in the neighborhood (determined by the *niche radius* σ_{share}) of the element, i.e.

$$m_i := \sum_{\mathbf{j} \in P_t} s(d(\mathbf{i}, \mathbf{j}))\,,$$

where $d(\mathbf{i}, \mathbf{j})$ is the distance[7] between the individuals $\mathbf{i}$ and $\mathbf{j}$. Furthermore, the *sharing function* s is defined by

$$s(d) := \begin{cases} 1 - (\frac{d}{\sigma_{\text{share}}})^{\alpha}\,, & \text{if } d < \sigma_{\text{share}}\,, \\ 0\,, & \text{otherwise}\,. \end{cases}$$

This means that points close to other points have a lower fitness than individuals in sparsely populated areas. Note, that the fitness sharing as explained here is the standard approach used in this context.
Finally note, that elitism is not used in this algorithm.

[7] Usually the distance of individuals is calculated in the objective space, since we are mainly interested in diversity at the Pareto front. But it is also possible to choose the design space. Then, diversity is obtained there.

4.3.2 Pareto Envelope-Based Selection Algorithm - PESA

This algorithm was suggested in [CKO00]. The algorithm is purely based on elitism. Thereby the nondominated individuals are stored in an external set EP. Note, that the basic concept of this algorithm is the same as the one used in the Strength Pareto Evolutionary Algorithm 2 (SPEA2), which will be introduced in Section 4.4.
The terminology purely based on elitism means, that the algorithm does not contain any concrete fitness assignment and, or selection. The basic algorithm of PESA is as follows:

1. Generate initial 'internal' population (IP) of P_I individuals and set the 'external' population (EP) to the empty set.
2. Incorporate nondominated elements from IP into EP, i.e. afterwards, EP contains the nondominated elements of $IP \cup EP$.
3. If termination criterion is fulfilled, then stop and return EP.
4. Delete the current content of IP and repeat the following steps until P_I new candidates are generated:
   - Apply recombination with probability p_c on EP and then mutate the obtained elements((see Subsection 4.1.1).
5. Return to Step 2.

4.3.3 Elitist Nondominated Sorting Genetic Algorithm - NSGA-II

The Nondominated Sorting Genetic Algorithm NSGA was introduced in [SD94]. The only difference to other GAs is in the fitness assignment and selection. The individuals are chosen by stochastic proportional selection to fill the mating pool. But before a fitness assignment has to be applied. This is realized in the following way: The nondominated individuals are determined and a high 'dummy' fitness value is assigned to them. Note, that in this case the better the individual is, the higher the fitness value. Afterwards, with fitness sharing diversity is obtained, i.e. the fitness is divided by a quantity proportional to the number of individuals around it (compare with Subsection 4.3.1). The nondominated values form the first front. Then this first front is temporarily ignored and the second front is identified and the fitness is again assigned. The new dummy fitness has to be smaller than the smallest fitness value of the previous front. The procedure continues, until all individuals were treated. Then the selection is carried out.
Although this algorithm is maintaining diversity and shows acceptable convergence results (due to preferring dominating individuals), it has a too high complexity and furthermore no elitism is implemented.

The Elitist NSGA-II, introduced in [DAPM00], improves in the way of a faster sorting procedure to obtain the dominating front in the fitness assignment step. But contrary the

memory consumption increased. Additionally, the fitness sharing was exchanged by a faster method. Moreover, the selection operator changed too. To apply elitism, $(\lambda + N)$-selection is used.

4.3.4 Summary and Conclusion

In this section we outlined three different MOEAs, namely NPGA, PESA and NSGA-II. The NPGA contains the strategy of fitness sharing but no elitism. Contrary, the Pareto-Based Envelope Selection Algorithm PESA is a purely elitism based approach, where no fitness assignment was implemented. Since it was shown in literature ([TLK02, ZLT01, BEAG04]), that algorithms which use these concepts converge faster than the ones that do not, we will not use NPGA and PESA for our purpose, i.e. the multiobjective design space exploration.

Hence, NSGA-II seems to be the best choice. It has a complexity of $\mathcal{O}(kN^2)$, where N is the number of individuals in each generation. Additionally it requires $\mathcal{O}(N^2)$ of memory. Especially, the computational complexity is the same as for NPGA and PESA.

But nevertheless, we chose another approach for our aims, namely the Strength Pareto Evolutionary Algorithm 2 (SPEA2, [ZLT01]), which incorporates both important strategies. Due to comparative studies like [TLK02, ZLT01, BEAG04] SPEA2 is at least as good as NSGA-II to meet our requirements. Although the computational complexity of SPEA2 is $\mathcal{O}(k \log(N) N^2)$, we expect almost the same runtime, since the constant α in (4.1) is expected to be higher for NSGA-II.

Summarizing the previous considerations, both methods, SPEA2 and NSGA-II, are equally suitable. But nevertheless, we chose SPEA2 for our sake. This method will be introduced in detail in the next section.

4.4 The Strength Pareto Evolutionary Algorithm 2

In [ZT98] and in [Zit99] the first version of the *Strength Pareto Evolutionary Algorithm* (SPEA) was presented. After further investigations, an improved version of this algorithm was presented in [ZLT01], called SPEA2. The SPEA2 method was compared to other available methods in several papers. Since it showed a satisfying behavior, we choose this algorithm to compute an approximation to the Pareto optimal front. In the following we will explain this approach in detail by discussing the specific choices of the operators introduced in Section 4.1. As mentioned previously, the major differences to evolutionary algorithms for single optimization problems are the fitness assignment and the selection step.

First, we will point out the two basic concepts used, which apply elitism and fitness sharing.

- *Environmental Selection/ Elitism.* In Section 4.2, we mentioned already the concept of elitism. SPEA2 uses this idea. There, an archive or external set is maintained, which contains a representation of the nondominated front among all individuals treated so far during the whole simulation run.

  A member of this archive is only removed if

  - it is dominated by a new individual. In this case the new individual is copied to the archive.
  - if the archive exceeded its maximum size. In that case, the external set is too crowded and some individuals have to be properly chosen, to be removed from the archive. The selection of the individuals is very crucial to guarantee diversity maintenance of the external set.

  This procedure is called *environmental selection* or *clustering*.

- *Mating Selection/ Fitness Sharing.* In Section 4.2 the very important concept of fitness sharing was introduced. Additionally to a Pareto dominance based so-called *raw fitness*, a different version of fitness sharing compared to the one shown in Subsection 4.3.1 is used. For more details see Subsection 4.4.1.

Following the definition of the normal (internal) population P_t and its size N, we will denote the external set or archive at generation t by $\bar{P}_t$. The number of individuals in $\bar{P}_t$ is defined by $\bar{N}$. Before going into further details, we introduce the framework of SPEA2. Therefore, remember our goal, which is to compute approximations to the Pareto front P_f. Thus, the return value of the algorithm is a set A that contains only nondominated individuals. The algorithm is given by

Algorithm 4.3 (SPEA2).

Input: N *population size*
$\bar{N}$ *archive size*
Θ *parameters for termination, selection, recombination and mutation*
Output: A *nondominated set*
$t = 0$;
initialize $P_0 = \{\mathbf{x}_{1,0}, \ldots, \mathbf{x}_{N,0}\}$;
initialize $\bar{P}_0 = \emptyset$;
while $\tau(P_t, \bar{P}_t, t) \neq true$
  Fitness assignment of $P_t \cup \bar{P}_t$: $\mathbf{f} = \{f(\mathbf{x}_{1,t}), \ldots, f(\mathbf{x}_{M,t})\}$;
  Environmental selection (clustering): $\bar{P}_{t+1} = c(P_t \cup \bar{P}_t)$;
  Mating selection: $P'_{t+1} = s_q(P_t \cup \bar{P}_t, \mathbf{f})$;
  Recombination: $P''_{t+1} = r_{\Theta_r}(P'_{t+1})$;
  Mutation: $P_{t+1} = m_{\Theta_m}(P''_{t+1})$;
  $t = t + 1$;
end while
$A = \{\mathbf{x} \in \bar{P}_t \mid \nexists \mathbf{y} \in \bar{P}_t : \mathbf{y} \prec \mathbf{x}\}$;

On the next pages, we will investigate the single steps of Algorithm 4.3 in more detail. First, the fitness assignment used in SPEA2, will be presented. Then, the environmental selection step will be outlined. After that, selection, recombination and mutation are explained. Finally, the stopping criterion of this method will be discussed.

4.4.1 Fitness Assignment

The fitness assignment of SPEA2 comprehends two important basic concepts. On the one hand, use is made of the Pareto dominance order and on the other hand fitness sharing, i.e. diversity maintenance is applied. Hence, it is executed in two steps. First, the set of individuals is ranked using the the Pareto dominance order, according to Definition 3.10. Afterwards, in a second step, diversity information is incorporated into the fitness values. The final fitness is given by

$$f(\mathbf{i}) := r(\mathbf{i}) + d(\mathbf{i})\,, \tag{4.2}$$

where $r(\mathbf{i})$ is the so-called *raw fitness* incorporating Pareto dominance relation. The *density* $d(\mathbf{i})$ contains diversity information, i.e. it incorporates fitness sharing.

For calculating the raw fitness, the so-called *strength* S of all individuals $\mathbf{i} \in P_t \cup \bar{P}_t$ is determined before. That is the number of individuals each element dominates, i.e.

$$S(\mathbf{i}) := |\left\{\mathbf{j} \in P_t \cup \bar{P}_t \,|\, \mathbf{i} \prec \mathbf{j}\right\}|\,, \tag{4.3}$$

where the the (weak) Pareto dominance relation $\prec$ is given by Definition 3.10. As already mentioned before, we will use the weak Pareto dominance, since the trade-off front is desired. In the following the *raw fitness* of the elements is defined by

$$r(\mathbf{i}) = \sum_{\mathbf{j} \in P_t \cup \bar{P}_t \wedge \mathbf{j} \prec \mathbf{i}} S(\mathbf{j})\,. \tag{4.4}$$

It is the sum of all strengths of the individuals $\mathbf{j}$ dominating $\mathbf{i}$. Note, that non-dominated individuals have the raw fitness value 0. This task has a complexity of $\mathcal{O}(M^2)$, where the total number of individuals M is defined by

$$M := |P_t \cup \bar{P}_t| = N + \bar{N}\,. \tag{4.5}$$

In Figure 4.2 an example to illustrate the raw fitness assignment is shown. There, the values in brackets are the strength values S. The nondominated individuals are marked by a ring. In this figure, we can see that individuals who are dominated by a larger number of points have a higher fitness value. This incorporates already the concept of diversity maintenance.

In the second step, a new kind of fitness sharing is applied. In [ZLT01] this is done by introducing the density d via

$$d(\mathbf{i}) := \frac{1}{2 + \sigma_{\mathbf{i}}^k} \tag{4.6}$$

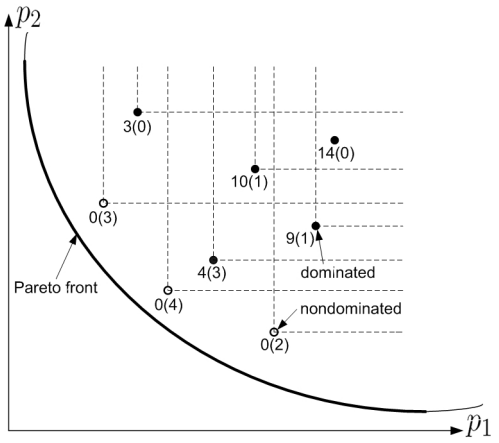


Figure 4.2: Example of raw fitness values. The numbers of the points denote the raw fitness r and the numbers in the brackets the strength S. The points marked by a ring are the nondominated individuals and the filled ones are the dominated ones. The dashed lines denote the domination ranges.

In (4.6) $\sigma_{\mathbf{i}}^k$ denotes the distance of $\mathbf{i}$ to the k-th closest individual in the set $P_t \cup \bar{P}_t$. Note, that the distance of individuals could be measured either in the design space(parameter space) or in the performance space(objective space). Since we want to obtain diversity in the objective space, the distance is measured there. According to [ZLT01], k should be chosen around $\sqrt{M}$ to obtain best distributed results. Higher k leads to a smoother characteristics, while a lower k leads to sharper behavior, what could lead to disadvantages(cf. [ZLT01] and the references given there). The complexity of the density computation is $\mathcal{O}(M^2 \log M)$, i.e. this part of the algorithm mainly determines the runtime of the algorithm.

Note, that $f(\mathbf{i}) < 1$ for nondominated individuals $\mathbf{i}$, where individuals with smaller f are preferred.
At a first glance, the density value does not seem to be very important, since the raw fitness can assume values up to[8] $M(M-1)/2$. But if many individuals of an actual population are indifferent to each other, only very little information can be obtained from the raw fitness assignment. In that cases, the density function plays the decisive role in the fitness assignment.

4.4.2 Archive Truncation - Clustering

The concept of elitism was already mentioned above. In SPEA2 it is implemented by storying the nondominated elements in an external set $\bar{P}_t$, also called archive. We prescribe a certain size of this set $\bar{N}$. In difference to early approaches (cf. [Zit99]), where the size of $\bar{P}_t$ was at most $\bar{N}$, in SPEA2 the size of the archive is held constant to $\bar{N}$. The so-called *clustering algorithm* limits the number of individuals in $\bar{P}_t$ to $\bar{N}$ or it fills $\bar{P}_t$ up. The method works as follows.

[8] That is the case if the individuals are totally ordered by the Pareto dominance relation $\prec$.

First, the new external set is defined by

$$\bar{P}_{t+1} = \left\{ \mathbf{i} \in P_t \cup \bar{P}_t \mid f(\mathbf{i}) < 1 \right\} , \tag{4.7}$$

where the fitness f is given by (4.2). If the nondominated set has size $\bar{N}$, nothing else has to be done. Otherwise, if $|\bar{P}_{t+1}| < \bar{N}$ the external set is filled up. Therefore, the individuals are sorted by their fitness and then the first $\bar{N} - |\bar{P}_{t+1}|$ individuals with fitness $F(\mathbf{i}) \geq 1$ are copied to the archive. On the other hand, if $|\bar{P}_{t+1}| > \bar{N}$ $\bar{P}_t - \bar{N}$ individuals are removed by using the order $\mathbf{i} \leq_d \mathbf{j}$. Thereby, an element $\mathbf{i}$ is chosen for removal, if

$$\mathbf{i} \leq_d \mathbf{j} \qquad \forall \mathbf{j} \in \bar{P}_{t+1} , \tag{4.8}$$

where the total order $\leq_d$ on $\bar{P}_{t+1}$ is defined by

$$\begin{aligned} \mathbf{i} \leq_d \mathbf{j} \quad :\Leftrightarrow \quad & \forall 0 < k < |\bar{P}_{t+1}| : \sigma_{\mathbf{i}}^k = \sigma_{\mathbf{j}}^k \quad \vee \\ & \exists 0 < k < |\bar{P}_{t+1}| : \left[\left(\forall 0 < l < k : \sigma_{\mathbf{i}}^l = \sigma_{\mathbf{j}}^l \right) \wedge \sigma_{\mathbf{i}}^k < \sigma_{\mathbf{j}}^k \right] . \end{aligned} \tag{4.9}$$

In the definition of the order $\leq_d$, i.e. (4.9), $\sigma_{\mathbf{i}}^k$ denotes, like in the definition of the fitness assignment (4.6), the distance of the individual $\mathbf{i}$ to the k-th closest element in $\bar{P}_{t+1}$. Again, the distance is measured in the performance space, since we want to prevent the boundary elements in the objective space from being deleted. We have to bear in mind, that the minimal distance occurs always at least two times. Hence, the element with the closest distance to other individuals is deleted. Moreover, this drop-out differs from the density approach of fitness assignment, since it discards the individual that is closest to the others.

Finally it has to be mentioned that in the case of missing elements, the auxiliary addition of elements helps to maintain a focus on the dominating front. Because in the case of only one or very few dominating elements, it is very likely, that individuals far away from the dominating element(s) are chosen for recombination and mutation and hence, the method would show a very slow convergence.

4.4.3 Selection

For the selection operator s a q-tournament selection (see Subsection 4.1.3) is chosen in [ZLT01], where usually $q = 2$, i.e. binary tournament. In the selection step, the operator s chooses N individuals from the candidate set $P_t \cup \bar{P}_t$. The selected N individuals form the so-called *mating pool*. As the name indicates, these individuals will be used in the following steps for mating in terms of recombination and mutation.

4.4.4 Recombination and Mutation

Now we introduce the operators for recombination r_{Θ_r} and mutation m_{Θ_m} used in our approach. In [ZLT01] it is only mentioned to use standard recombination and mutation,

while in this work it was chosen to use a discrete version with 1-point crossover and point mutations. Since we are dealing with continuous problems and due to the lack of information available in literature on the best choices for these operators, we will use standard approaches, which were already investigated in Subsection 4.1.1.

For recombination we use a linear combination with $\chi \sim \mathcal{U}([0, 1])$. Usually the probability for recombination $p_r = 1$, since it shows better behavior as will be outlined in Section 6.2.

It is more difficult to determine a satisfactory mutation operator. As outlined in Subsection 4.1.1, according to [B96], usually normally distributed mutation operators are used, i.e. $m_{\Theta_m} \sim \mathcal{N}(0, \Sigma)$ with Σ being the covariance matrix. Different possibilities to choose Σ exist:

1. Choose the same standard deviation σ for each direction. Hence, we obtain

$$\Sigma = \sigma^2 I \tag{4.10}$$

   with I being the identity matrix of suitable dimension n. The corresponding domain can be expressed as a hypersphere.

2. A step further would be to introduce different standard deviations in each direction, but without any correlation between them. Consequently the covariance matrix Σ is given by

$$\Sigma = \text{diag}(\sigma_1^2, \ldots, \sigma_n^2) \tag{4.11}$$

   and can be viewed as an ellipsoid with non-rotated main directions.

3. One can choose an arbitrary covariance matrix Σ. The determined region can be depicted as an arbitrary ellipsoid.

Firstly, we will focus on the first two cases, especially on the second, since the scale of each variable can differ. Note, that after a scaling of the design variables, the first and second possibility usually coincide. Moreover, we do not use the third possibility, since it is too complex. Especially a suitable choice is very difficult. For more information see [B96] p. 68 ff.

The problem is now, how to choose σ or σ_i, $i = 1, \ldots, n$ to achieve optimal progress toward the dominating Pareto front. In [MSV93] this task was investigated by means of single optimization problems. The measure, that was used to compare the results, is the *expected progress*. It is, according to its name, the integral over the domain where the individual is not discarded[9] is achieved of the progress times the probability density. We will further discuss this issue in Section 4.5, Subsection 6.1.2 and Subsection 6.2.3.

For comparison, we will additionally use the Breeder mutation, which was introduced in Subsection 4.1.1. For more details on implementation and numerical results see Chapter 6.

[9] Basically this domain is given by the domain of individuals $\mathbf{x} \in X_f$, that are not dominated by an individual $\mathbf{y} \in \bar{P}_t$.

4.4.5 Stopping Criterion

In general optimization problems, the ideal stopping criterion would be to measure the error of the actual approximated solution and to stop if a desired accuracy is reached. Since, it is typically not possible to measure the error, i.e. the distance of the actual approximation to the solution, a relative measure is usually used. This means that the iteration is stopped, if the change in the objective function value is smaller than some ε. This approach is quite common in deterministic numerical solution methods for SOPs.

The problem in the framework of evolutionary algorithms is, that the Pareto front is not known beforehand. Hence an absolute error measure cannot be applied. Additionally it is not clear at all how to set up a relative error term corresponding to the explanations given above. Therefore, the most common stopping criterion in evolutionary algorithms is to prescribe a maximum number of generations T, what is of course not optimal, since there is no guarantee at all, that the procedure is already close enough at the solution. Hence, also other concepts are suggested in literature. For instance, another possible termination criterion could be the existence of an individual with sufficient fitness, or a sufficient behavior. But since we want to compute an approximation for the whole Pareto dominant front, this is not a useful alternative. Furthermore, we could stop the process if a stagnation in the nondominating front has established. This approach is the pendant to the criterion that watches the relative function change, which was mentioned above. In [PG02] this method was treated in more detail. First, the coverage function f_C is defined by

$$f_C(P',\, P'') := \frac{|\{\mathbf{p}'' \in P'' \,:\, \exists \mathbf{p}' \in P' \,:\, \mathbf{p}' \prec \mathbf{p}''\}|}{|P''|}\,. \tag{4.12}$$

It represents the ratio of points in P'' that are dominated by at least one point in P'. Using (4.12) we can define the *convergence index* q:

$$q(i,\, G) := f_C(\bar{P}_{i+G},\, \bar{P}_i) - f_C(\bar{P}_i,\, \bar{P}_{i+G})\,. \tag{4.13}$$

Note, that as long as the evolutionary process improves the solution, we continue the process. Improving the solution corresponds to the relation $f_C(\bar{P}_{i+G},\, \bar{P}_i) \geq f_C(\bar{P}_i,\, \bar{P}_{i+G})$, i.e. $q(i,\, G) \geq 0$. For the application of this approach, we have to compute every G iterations the convergence index. If $q(i,\, G) \leq T_C$ we stop the process. Thereby, T_C (corresponding to ε) is some predefined threshold value. Additionally the step size G has to be chosen properly. We will use this criterion or the maximum iteration criterion in our implementation (see Subsection 6.1.2 for details on the choice of T_C and G).

4.5 Investigations on the Convergence Properties of SPEA2

In the following the convergence behavior of the Strength Pareto Evolutionary Algorithm 2 will be investigated. First we will give some references to other convergence proofs, since at least one of them is used as a guideline for our proof. Afterwards, in Subsection 4.5.2 the

route of the proof will be outlined. We are dealing with a probabilistic approach therefore the term convergence has to be defined. Furthermore, the theorem stating the convergence will be quoted there. Afterwards, in Subsection 4.5.3 the assumptions needed to show convergence will be listed and justified.
For the proof of this theorem, the single steps of SPEA2 have to be studied. This will be done in the subsections 4.5.4 to 4.5.8. Finally, the calculations of Subsection 4.5.6 will be verified by numerical tests.

4.5.1 Previous Results Available in Literature

Now, we want to outline some convergence results available in literature. There are several approaches, which show convergence for algorithms or a class of algorithms under certain assumptions. But unfortunately, no analytical results on the convergence of SPEA2 is available. Therefore, the literature was searched to get an idea how to prove convergence of SPEA2. Available results are:

In [Rud98] convergence of a $(1 + 1)$-selection MOEA is examined. Especially convergence is exemplarily shown, but with the decisive restriction, that the range of the mutation operator is related to the distance to the optimal solution.
In [RA00] convergence of EA is shown for a large class of discrete evolutionary algorithms[10]. Since we are dealing with continuous problems, i.e. the domains are uncountable, those considerations are not useful for us.

In another article ([Han99]) convergence is shown for so-called *efficiency preserving algorithms*. Those are algorithms, for which the individuals that dominate the actual nondominated individuals of P_{t+1} are a subset of the ones dominating the nondominated individuals in P_t. That is, algorithms for which $\mathrm{Dom}(P_{t+1}) \subseteq \mathrm{Dom}(P_t)$, where $\mathrm{Dom}(P_t)$ is defined by $\mathrm{Dom}(P_t) := \cup_{\mathbf{a} \in E(P_t, \mathbf{p})} \mathrm{Dom}(\mathbf{a})$, with $\mathrm{Dom}(\mathbf{a}) := \{\mathbf{x} \in X_f : \mathbf{x} \prec \mathbf{a}\}$, i.e. the set of feasible individuals $\mathbf{x}$ that dominate the actual point $\mathbf{a}$. Thereby contains $E(P_t, \mathbf{p})$ the nondominated individuals of P_t with respect to the objective function $\mathbf{p}$.
Note the fact, that SPEA2 does not have this property. For instance, if a new point $\mathbf{a}$ is indifferent to all existing points $\mathbf{x}$, i.e. $\mathrm{Dom}(\mathbf{a}) \not\subseteq \mathrm{Dom}(\mathbf{x})$ for all $\mathbf{x}$. For illustration imagine a boundary point. If the new point extends the front to the external, then the efficiency preserving property is not fulfilled. Especially we have to mention, that algorithms with this intrinsic property cannot extend the range of their actually approximated front any more, which is not desirable in general.

In [B$\ddot{9}$6] a convergence proof for evolution strategies applied to SOPs is depicted for $(\lambda+N)$-selection. But in this proof the crucial step of recombination was neglected. Furthermore, we have to mention, that SPEA2 is not a evolution strategy in the sense of [B$\ddot{9}$6] due to the additional clustering step, which does not fit in the general concept outlined therein. This fact and the immanent difficulty of multiobjective optimization problems due to the absence of a total order of X_f make it impossible to adapt this proof to MOPs.

[10] There, the behavior of the algorithms is expressed by use finite Markov chains.

The previously listed results were not useful, except [Rud98]. We will exploit the ideas of this proof to show convergence of SPEA2 in the worst case under certain assumptions.

4.5.2 Preliminaries and Definitions

In this part, we will introduce the necessary basics for the proof of convergence of SPEA2. Below, the definition of the term convergence in the framework of random variables will be introduced and furthermore, the key theorem of the proof will be stated. Additionally a structure of the proof is outlined below.

We are dealing with probabilistic algorithms. Hence, the populations P_t are random variables. Since there are several definitions for convergence of random variables, we will repeat the two we are using:

Definition 4.4. *Let $\{X_i\}$, $i \in \mathbb{N}$ be a sequence of random variables. The sequence $\{X_i\}$ is said to* converge in probability *to a random variable X (denoted by $X_i \xrightarrow{pr} X$) if for every $\varepsilon > 0$*

$$\lim_{i \to \infty} P\left(d(X_i, X) < \varepsilon\right) = 1\,, \tag{4.14}$$

with D being a suitable nonnegative distance measure. Furthermore, the sequence $\{X_i\}$ converges almost surely *(a.s.) to a random variable X (denoted by $X_i \xrightarrow{a.s.} X$) if*

$$\lim_{i \to \infty} P\left(d(X_i, X) = 0\right) = 1\,. \tag{4.15}$$

Almost sure convergence implies the convergence in probability. In other words, almost sure convergence means that with probability one the sequence $\{X_i\}$ converges to the limit X. On the other hand, convergence in probability does have the meaning that with probability one we can come as close as we want to the limit X, but nothing can be said about the limit $\varepsilon \to 0$. Therefore, additional requirements are needed to follow a.s. convergence from convergence in probability.
In an almost general framework, we will show almost sure convergence of the approximated front to the Pareto front for a worst case example. Moreover, it is worth noticing, that we will only show convergence in terms of distance to P_f. No information on the distribution of the approximations can be concluded.

First of all, we have to define the distance measure. There are various possibilities to define D needed for Definition 4.4. If we assume the reference set $X = P_f$ to be known, we could use one of the following distance function

$$\left\|\bar{P}_t\right\|_1 := \frac{1}{\bar{N}} \sum_{\mathbf{p}_i \in \bar{P}_t} d(\mathbf{p}_i, P_f) \tag{4.16a}$$

$$\left\|\bar{P}_t\right\|_2 := \frac{1}{\bar{N}} \sqrt{\sum_{\mathbf{p}_i \in \bar{P}_t} d(\mathbf{p}_i, P_f)^2} \tag{4.16b}$$

$$\left\|\bar{P}_t\right\|_\infty := \max_{p_i \in \bar{P}_t} d(p_i, P_f)\ , \tag{4.16c}$$

with the distance function d, being defined by

$$d(\mathbf{p}_i, P_f) := \inf_{\mathbf{p} \in P_f} \|\mathbf{p}_i - \mathbf{p}\| \,. \tag{4.16d}$$

Note, that almost sure convergence is the desired property. For this sake we will use a customized version of Theorem 6.3 of [BJ68] (p. 84). In [Rud98] (Theorem 1, p. 514) this theorem was adapted and applied. If we customize it further to the use of a whole front, which is no restriction, the theorem is given by

Theorem 4.5. *Let $(D_t : t \geq 0)$ be a sequence of nonnegative random variables, i.e. distance measures related to random sets P_t, and let $\gamma : \mathbb{R}_+ \to \mathbb{R}_+$ be a continuous function vanishing only at the origin. Furthermore, let $\mathcal{F}_t$ be an event induced by the previous states*[11] *$\tilde{P}_0, \ldots, \tilde{P}_{t-1}$. If $E(D_t) < \infty$ and*

$$E(D_{t+1}|\mathcal{F}_t) \leq D_t - \gamma(D_t) \tag{4.17}$$

for all $t \geq 0$, then the sequence $(D_t : t \geq 0)$ converges to zero with probability one as $t \to \infty$.

The population $\tilde{P}_t$ at time t only depends on $\tilde{P}_{t-1}$ and not on the previous ones, and therefore $\mathcal{F}_t$ is usually stated as $\tilde{P}_t = P$. E denotes the (conditional) expected value. In the used relation, D_t denotes the distance measure $\left\|\bar{P}_t\right\|_k$, $k = 1, 2, \infty$ defined by (4.16a), (4.16b) or (4.16c) of the actual approximated nondominated front to the Pareto optimal front P_f.

If we are able to show that (4.17) is fulfilled, then convergence follows immediately from the theorem. Hence, we have to find a γ satisfying the conditions of Theorem 4.5. Therefore we can rewrite condition (4.17) in the following form

$$E(D_{t+1}|\tilde{P}_t = P) \leq D_t - \gamma(D_t) \quad \Leftrightarrow \quad D_t - e(D_t) \leq D_t - \gamma(D_t)\,,$$

and hence, we have to find $\gamma : \mathbb{R}_+ \to \mathbb{R}_+$, such that

$$e(d) \geq \gamma(d) \tag{4.18}$$

holds, for all $d \in \mathbb{R}_+$. $e(d)$ denotes the expected progress for the dominating and the indifferent area for an arbitrary population $\tilde{P}_t$ with distance d. The dominated area is not regarded, because a new individual is discarded if it is dominated by the old one, what is equivalent to zero progress. For our sake, the expected progress $e(\mathbf{x}, d)$ of an individual $\mathbf{x}$ with distance d is schematically given by

$$e(\mathbf{x}, d) := \int_{D(\mathbf{x}) \cup I(\mathbf{x})} \text{progress}(\mathbf{y}) \, \text{probability}_d(\mathbf{y}) \, \mathrm{d}\mathbf{y}\,, \tag{4.19}$$

where $D(\mathbf{x})$ denotes the the individuals that dominate $\mathbf{x}$ and $I(\mathbf{x})$ designates the indifferent

[11] The states $\tilde{P}_t$ combine the originally considered regular generation P_t and the archive P_t.

individuals. Both sets lie in the parameter space.

It turns out, that we have to investigate the single steps of Algorithm 4.3, which are in the order we will treat them:

1. Mating selection (Subsection 4.5.4)
2. Recombination (Subsection 4.5.5)
3. Mutation (Subsections 4.5.6)
4. Environmental selection - Clustering (Subsection 4.5.7)

Finally the results will be summarized in Subsection 4.5.8. But first of all, we have to make some assumptions to be able to show convergence according to Theorem 4.5. They will be listed in the next paragraph.

4.5.3 Assumptions

First of all, we assume the feasible performances Y to be bounded and hence $E(D_t) < \infty$ can be guaranteed. In the following we have to investigate the condition (4.17). For this sake, we want to mention the assumptions we have to make in order to be able to prove convergence.

- Due to the big amount of possible combinations of n and k we choose $n = k$ for simplicity. Note, that always the space of smaller dimension determines the behavior of the function in general. If $n < k$, $\mathbf{p}(D)$ is a surface of dimension less or equal to n in the objective space. Otherwise if $n > k$, for most of the objective vectors there exist a $(n-k)$-dimensional surface corresponding to this vector. Therefore, the space of lower dimension determines the behavior and hence, our assumption is not too restrictive.
- Additionally we have to make some assumptions on the performance function $\mathbf{p}$. We choose $\mathbf{p} := I$, i.e. $\mathbf{p}$ is the identity mapping from $\mathbb{R}^n$ to $\mathbb{R}^n$. The feature in this context is, that design domains with equal volume are mapped to performance areas of the same size. Note, that if a parameter set corresponding to a dominating domain in the performances of nonzero measure would have measure zero, with probability one the algorithm would not reach this areas and hence we would expect no convergence at all to those areas. Therefore, we need that $\mathbf{p}$ exhibits a "nice" behavior as mentioned before.
- Furthermore, we choose $N = 1$ and $\bar{N} \in \mathbb{N}$. This is done, since it is too hard to handle the effect of interference between several new points. $N > 1$ implies then faster convergence, but not proportional to N, since some of the new points might be excluded through other new ones.

- Additionally, we suppose the archive $\bar{P}_t$ to be completely filled with nondominated points. This is not very severe, because with recombination of two dominating individuals we obtain already a point, which is indifferent to both parents.

- Another restriction we have to suppose is, that the optimal front is convex. Thus, it reaches from a straight line to a corner. In our special case, we show convergence only for a special case, depicted in Figure 4.3. We are regarding only the front on the left and not on the bottom. This is done, since the additional front at the bottom would yield a higher expected progress. But in the case that **p** is already very close to the front, the distance to the bottom front would be, considered relatively to the distance, very large. Hence, this worst case has to be assumed and the assumption is justified.
  Note, that this sketch can be applied to any straight Pareto front by use of a coordinate transformation.

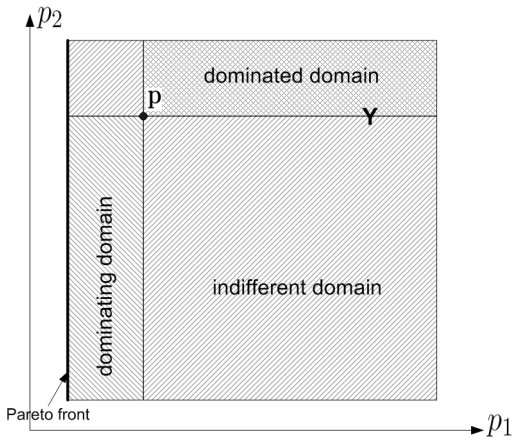


Figure 4.3: Pareto front assumed for the convergence proof.

  This assumption is not as severe, as one might think. It is only stated, because then, the front depicted in Figure4.3 is the worst case.

- Finally, we have to assume for the mutation step, that we know the distance r_1, given by 4.16d, of each individual to the Pareto optimal front P_f. This is necessary, since a wrong choice of the mutation parameters Θ_m can lead to an expected regress in distance. For the approaches, that will be used in reality see Subsection 6.1.2.

The case depicted in Figure 4.3 is the worst case in terms of the expected progress, when a normally distributed mutation operator is used, because the lower bound would limit the maximal regress after mutation. But especially, as will be explained later, this worst case assumption is satisfied since we consider all distances in terms of multiples of the distance to the front. And when the chosen individual's distance converges to zero all other distances relative to the determining one converge to infinity.

Now, we are able to investigate the single steps of SPEA2 to prove the existence of a function γ which fulfills the requirements of Theorem 4.5.

4.5.4 Mating Selection

The first step is the mating selection. We want to consider the cases of improvement, i.e. the cases when the newly generated individual is not discarded. Thus we try to select a nondominated element, because it has the best chances to result in an improvement. That is, $\mathbf{x} \in P_t$ will not be considered. The distance measure D_t of $\bar{P}_t$ is given by (4.16a), (4.16b) or (4.16c). Thereby any possible distribution of $\bar{P}_t$ with distance D_t is possible.
The extremal case in terms of distance and furthermore, in terms of expected progress, is, whenever $\bar{N}-1$ individuals have distance zero and one has the distance $\bar{N} \cdot D_t$ (if (4.16a) or (4.16b) are used) to the Pareto front. Furthermore, it might happen that this individual has the worst fitness above all[12], including $\mathbf{x} \in P_t$. Therefore, we have to select this individual q times in order to guarantee, that this individual is chosen by the tournament. In the following we will compute a γ which is linear in d, i.e. $e(d) \geq \zeta d$ with $\zeta > 0$. Hence, the assumed case is indeed the worst case. The selection probability is then given by

$$p_{s,d} = \frac{1}{(\bar{N}+1)^q} \,. \tag{4.20}$$

The distance to the optimal front r_1 is given by $r_1 = \bar{N} \cdot D_t$ (for (4.16a) and (4.16b)). When using (4.16c) $r_1 = D_t$ and hence we obtain a different ζ. From now on, we will carry out the computations for (4.16b) without loss of generality. For the other two possibilities, only the positive constant ζ would change.

4.5.5 Recombination

Now, let us assume we selected the desired element of $\bar{P}_t$. Since we chose $N = 1$, no recombination is carried out. In almost all convergence proofs for EAs provided in literature, recombination was neglected (cf. [BÖ96] p. 89 or [Rud98]). It is difficult to be taken into account, since under certain circumstances the distance of the generated new individual to the Pareto front can become arbitrarily large compared to the distances of the parents. An example for such a case is depicted in Figure 4.4. There, the individuals $\mathbf{x}$ and $\mathbf{y}$ are Pareto optimal points, i.e. they have distance 0. Thus, the generated linear combination has an infinitely high distance to P_f compared to the parents distances. But nevertheless, it is advisable to perform recombination, because a high diversity of the contour is designated as well, what then might lead to a decrease of the convergence speed.

4.5.6 Mutation

The next step is mutation. This step is a very crucial step in terms of convergence properties as will be outlined below. Therefore, we need some preliminary investigations on the mutation step. Afterwards, rates for the expected progress will be derived by means of the assumptions made above.

[12] In the assumed case the fitness is merely determined by the density D(see (4.6)) as outlined in Subsection 4.4.1.

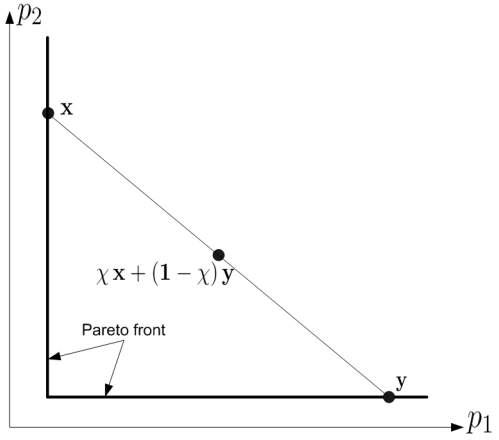


Figure 4.4: Example for a "bad" recombination in terms of convergence.

Preliminary Considerations for the Mutation Step

We will only consider the normal distribution, but for the other mutation operators introduced in Subsection 4.1.1 similar conclusion can be drawn[13]. The n-dimensional normal distribution $\mathcal{N}(\mu, \Sigma)$ is given by the following probability density function, but for the other mutation operators introduced in Subsection 4.1.1 similar conclusion can be drawn. The n-dimensional normal distribution $\mathcal{N}(\mu, \Sigma)$ is given by the following probability density function

$$p(\mathbf{x}) = \frac{\exp\left(-\frac{1}{2}(\mathbf{x} - \mu)\Sigma^{-1}(\mathbf{x} - \mu)\right)}{(2\pi)^{n/2}(\det \Sigma)^{1/2}} \tag{4.21}$$

with the expectation μ, which is $\mathbf{0}$ in the case of mutation, and the covariance matrix Σ. For the investigations on the expected progress of the mutation step, we assume the simplest case for Σ, i.e. $\Sigma = \sigma^2 I$. This restriction is not so severe, as outlined in Subsection 4.4.4.

In Figure 4.3 the shape of the considered front was introduced. As already mentioned previously, we have to consider the expected progress. It is outlined in equation (4.19), and it is given by the integral over the feasible set, i.e. the feasible domain except the dominated area, of the progress times the probability. Note that it is possible for the progress to become negative in indifferent areas. In Figure 4.5 the worst case in terms of the expected progress is sketched for $n = 2$. If the height of the indifferent area is bigger than r_1, then the optimal expected progress (related to an optimal σ) will be higher, since for any point in this domain we obtain a progress in distance. Of course, the height of this area can also be smaller up to zero. This case will be treated later on.
Note, that the expected progress will decrease for $\delta_2 \to \infty$, because the expected progress is negative in this indifferent area. Furthermore, we assume $\delta_1 \to \infty$ too, since otherwise for some regions we would have a smaller distance to the front at the bottom than to the considered one-sided front. This would lead to an increased expected progress. Hence, the assumption $\delta_1 \to \infty$ completes the worst case scenario.

[13] See [MSV93] for the treatment of the uniform distribution, the normal distribution and the mutation used in the Breeder Genetic Algorithm, explained in Subsection 4.1.1, for single objective mutation.

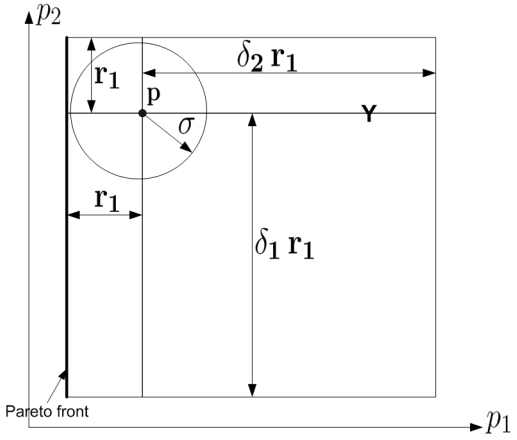


Figure 4.5: Sketch of the assumed worst case. δ_1 and δ_2 are the determining values for the distance of $\mathbf{p}$ to the boundaries of the domain. The point $\mathbf{p}$ has distance r_1 to the Pareto front and furthermore, the circle with radius σ denotes the "$1-\sigma$" domain of the mutation operator.

For the sake of computing the worst case of the expected progress, we have to bear already the next step of the algorithm in mind, namely the environmental selection. In general there are the following possibilities for the new individual to be located

1. The new individual dominates the parent individual. In this case, the parent will be discarded of the archive $\bar{P}_t$.
2. The generated element is indifferent to the parent individual. Since there are $\bar{N}-1$ other nondominated elements in the approximated front, two different possibilities for this new element arise:
   (a) The new individual is dominating at least one other individual of the set $\bar{P}_t$.
   (b) It is indifferent to all other elements of the set $\bar{P}_t$.

Additionally to this enumeration we have to state several remarks.
The case depicted in the figures 4.3 and 4.5 is the worst case in terms of the expected progress, except this small distance to the top of the domain which will be treated later. The distance to the top can also be limited by another nondominated individual. Furthermore, if there would be some more elements of $\bar{P}_t$ in the lower indifferent area, they would limit the considered domain, since only nondominated regions are of interest. If the new individual would be dominated by another one, it would be discarded and hence the progress in those areas is zero. Additionally note, that we assume the parent to be discarded in the case of indifference of the new individual too, because the lower indifferent area causes a decrease of the expected progress. That is, since we are dealing with the worst case.

Summarizing the previous considerations, we are treating the whole dominating and indifferent domain. The upper indifferent domain is considered for convergence reasons and with the other indifferent areas we are dealing due to the regress in those areas.

Compare our investigations with [Han99], where efficiency preserving algorithms are investigated as indicated in Subsection 4.5.1. To be able to apply those results, we would have to discard indifferent individuals to fulfill the requirements stated there. Therefore, these results are completely out of scope to show convergence of SPEA2.

Mutation - The Standard Case

First we will compute the probability to obtain a feasible point for given distance r_1 and a given standard deviation σ. We will limit ourselves to the two–dimensional case. Additionally the n-dimensional ($n > 2$) case will be sketched afterwards. This is possible due to our simple test problem, which reduces to a two-dimensional problem.

The probability to obtain a feasible point, i.e. a nondominated one according to Figure 4.5, in 2D is given by

$$\begin{aligned}
p_2(r_1,\,\sigma,\,\delta_1,\,\delta_2) &= \int_{-r_1}^{\delta_2 r_1}\int_{-\delta_1 r_1}^{r_1} p(x,\,y)\,\mathrm{d}y\,\mathrm{d}x - \int_{0}^{\delta_2 r_1}\int_{0}^{r_1} p(x,\,y)\,\mathrm{d}y\,\mathrm{d}x \\
&= \frac{1}{2\pi\sigma^2}\left\{\int_{-r_1}^{\delta_2 r_1}\int_{-\delta_1 r_1}^{r_1} e^{-\frac{x^2+y^2}{2\sigma^2}}\,\mathrm{d}y\,\mathrm{d}x - \int_{0}^{\delta_2 r_1}\int_{0}^{r_1} e^{-\frac{x^2+y^2}{2\sigma^2}}\,\mathrm{d}y\,\mathrm{d}x\right\} \\
&= \frac{1}{2\pi\sigma^2}\left\{\int_{-\delta_1 r_1}^{r_1} e^{-\frac{y^2}{2\sigma^2}}\,\mathrm{d}y \int_{-r_1}^{\delta_2 r_1} e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x\right. \\
&\quad \left. - \int_{0}^{r_1} e^{-\frac{y^2}{2\sigma^2}}\,\mathrm{d}y \int_{0}^{\delta_2 r_1} e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x\right\} \\
&= \frac{1}{2\sqrt{2\pi}\,\sigma}\left\{\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right) + \operatorname{erf}\left(\frac{\delta_1 r_1}{\sqrt{2}\,\sigma}\right)\right]\int_{-r_1}^{\delta_2 r_1} e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x\right. \\
&\quad \left. - \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\int_{0}^{\delta_2 r_1} e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x\right\} \\
&= \frac{1}{4}\left\{\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right) + \operatorname{erf}\left(\frac{\delta_1 r_1}{\sqrt{2}\,\sigma}\right)\right]\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right) + \operatorname{erf}\left(\frac{\delta_2 r_1}{\sqrt{2}\,\sigma}\right)\right]\right. \\
&\quad \left. - \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\operatorname{erf}\left(\frac{\delta_2 r_1}{\sqrt{2}\,\sigma}\right)\right\}\,, \qquad (4.22)
\end{aligned}$$

where $\operatorname{erf}(x)$ denotes the so-called *error function*, which is defined by

$$\operatorname{erf}(x) := \frac{2}{\sqrt{\pi}}\int_0^x e^{-s^2}\,\mathrm{d}s\,. \qquad (4.23)$$

The limit of (4.22) for $\delta_1 \to \infty$ and $\delta_2 \to \infty$ is given by

$$p_2(r_1,\,\sigma) = \lim_{\delta_1,\,\delta_2\to\infty} p_2(r_1,\,\sigma,\,\delta_1,\,\delta_2) = \frac{1}{4}\left[1 + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right) + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)^2\right] \qquad (4.24)$$

Now, we want to compute the expected progress e_n and especially the optimal σ depending on the distance r_1 resulting in a maximal progress, which can help us to increase the convergence speed of the algorithm. Since our relative starting point is $\mathbf{0}$ and the front is

situated at $x = -r_1$, the progress is given by $-x$. Thus, the expected value is given by

$$\begin{aligned}
e_2(r_1,\,\sigma,\,\delta_1,\,\delta_2) &= \int_{-r_1}^{\delta_2 r_1}\int_{-\delta_1 r_1}^{r_1} -x\,p(x,\,y)\,\mathrm{d}y\,\mathrm{d}x - \int_{0}^{\delta_2 r_1}\int_{0}^{r_1} -x\,p(x,\,y)\,\mathrm{d}y\,\mathrm{d}x \\
&= \frac{1}{2\pi\sigma^2}\left\{\int_{-r_1}^{\delta_2 r_1}\int_{-\delta_1 r_1}^{r_1} -x\,e^{-\frac{x^2+y^2}{2\sigma^2}}\,\mathrm{d}y\,\mathrm{d}x \right. \\
&\quad \left. -\int_{0}^{\delta_2 r_1}\int_{0}^{r_1} -x\,e^{-\frac{x^2+y^2}{2\sigma^2}}\,\mathrm{d}y\,\mathrm{d}x\right\} \\
&= \frac{1}{2\sqrt{2\pi}\,\sigma}\left\{\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)+\operatorname{erf}\left(\frac{\delta_1 r_1}{\sqrt{2}\,\sigma}\right)\right]\int_{-r_1}^{\delta_2 r_1} -x\,e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x \right. \\
&\quad \left. -\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\int_{0}^{\delta_2 r_1} -x\,e^{-\frac{x^2}{2\sigma^2}}\,\mathrm{d}x\right\} \\
&= \frac{1}{2\sqrt{2\pi}\,\sigma}\left\{\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)+\operatorname{erf}\left(\frac{\delta_1 r_1}{\sqrt{2}\,\sigma}\right)\right]\int_{-r_1}^{\delta_2 r_1} \frac{\mathrm{d}}{\mathrm{d}x}\left(\sigma^2\,e^{-\frac{x^2}{2\sigma^2}}\right)\,\mathrm{d}x \right. \\
&\quad \left. -\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\int_{0}^{\delta_2 r_1} \frac{\mathrm{d}}{\mathrm{d}x}\left(\sigma^2\,e^{-\frac{x^2}{2\sigma^2}}\right)\,\mathrm{d}x\right\} \\
&= \frac{\sigma}{2\sqrt{2\pi}}\left\{\left[e^{-\frac{\delta_2^2 r_1^2}{2\sigma^2}}-e^{-\frac{r_1^2}{2\sigma^2}}\right]\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)+\operatorname{erf}\left(\frac{\delta_1 r_1}{\sqrt{2}\,\sigma}\right)\right] \right. \\
&\quad \left. +\left[1-e^{-\frac{\delta_2^2 r_1^2}{2\sigma^2}}\right]\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\right\}, \qquad (4.25)
\end{aligned}$$

whereas its limit for $\delta_1 \to \infty$ and $\delta_2 \to \infty$ is given by

$$\begin{aligned}
e_2(r_1,\,\sigma) &= \lim_{\delta_1,\,\delta_2\to\infty} e_2(r_1,\,\sigma,\,\delta_1,\,\delta_2) \\
&= \frac{\sigma}{2\sqrt{2\pi}}\left\{\left[1-e^{-\frac{r_1^2}{2\sigma^2}}\right]\left[\operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)+1\right]-1\right\}. \qquad (4.26)
\end{aligned}$$

Note, that for all $r_1 > 0$, there exists a $\sigma > 0$, such that $e_2(r_1,\,\sigma)$ is bounded away from zero, which is very important to be able to apply Theorem 4.5. Furthermore, p_2 and e_2 depend only on the ratio $\alpha := \sigma/r_1$, except that e_2 is additionally multiplied by a σ what shows the linear convergence character. In Figure 4.7(a) on p. 58 $p_2(r_1,\,\sigma)$ and $e_2(r_1,\,\sigma)$ are shown for varying the ratio α. Due to the multiplicative factor σ in e_2 we set $r_1 = 1$. In this picture we can clearly see the positive part of e_2. We want to obtain the maximal expected progress $\tilde{e}_2(r_1)$. Therefore the root of the derivative of (4.26) with respect to σ for fixed r_1 is needed. It turns out, that the the derivative depends just on α and hence, the optimal parameter $\tilde{\sigma}_2(r_1)$ is linear in r_1. The optimal ratio $\tilde{\alpha}_2$ cannot be calculated analytically due to the complexity of the derivative. A numerical computation of the root yields

$$\tilde{\alpha}_2 = 0.431735\,, \qquad (4.27)$$

with a maximum expected progress given by

$$\tilde{e}_2 := e_2(r_1,\,\tilde{\alpha}_2\,r_1) = 0.0726904\,r_1\,. \qquad (4.28)$$

The scheme of the Pareto front as depicted in Figures 4.3 and 4.5 can be easily extended to higher dimensions. Similar to the two dimensional case, we assume the point $\mathbf{p}$ to have a distance to the other boundaries in x_i-direction, $i = 2, \ldots, n$ of at least r_1. We omit the detour of introducing δ_i, $i = 1, \ldots, n$ and calculate directly the limit for $\delta_i \to \infty$. The probability of the new individual for not being discarded, either because it is infeasible or because it dominates the parent, is given by

$$\begin{aligned}
p_n(r_1, \sigma) &= \int_{-r_1}^{\infty} \underbrace{\int_{-\infty}^{r_1} \cdots \int_{-\infty}^{r_1}}_{n-1 \text{ times}} p(\mathbf{x})\, \mathrm{d}\mathbf{x} - \int_0^{\infty} \underbrace{\int_0^{r_1} \cdots \int_0^{r_1}}_{n-1 \text{ times}} p(\mathbf{x})\, \mathrm{d}\mathbf{x} \\
&= \frac{1}{(2\pi)^{n/2}\,\sigma^n} \left\{ \underbrace{\int_{-\infty}^{r_1} \cdots \int_{-\infty}^{r_1}}_{n \text{ times}} e^{-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}}\, \mathrm{d}\mathbf{x} - \int_0^{\infty} \underbrace{\int_0^{r_1} \cdots \int_0^{r_1}}_{n-1 \text{ times}} e^{-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}}\, \mathrm{d}\mathbf{x} \right\} \\
&= \frac{1}{(2\pi)^{n/2}\,\sigma^n} \left\{ \left[\frac{\sqrt{\pi}\sigma}{\sqrt{2}}\right]^n \left[1 + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\right]^n \right. \\
&\quad \left. - \left[\frac{\sqrt{\pi}\sigma}{\sqrt{2}}\right]^n \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)^{n-1} \operatorname{erf}\left(\frac{s}{\sqrt{2}\,\sigma}\right)\Big|_{s=0}^{\infty} \right\} \\
&= \frac{1}{2^n} \left\{ \left[1 + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\right]^n - \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)^{n-1} \right\}. \qquad (4.29)
\end{aligned}$$

The progress in the considered case is given by $-x_1$ of the new point, and hence, the expected progress e_n in the limiting case can be computed as follows:

$$\begin{aligned}
e_n(r_1, \sigma) &= \frac{1}{(2\pi)^{n/2}\,\sigma^n} \left\{ - \underbrace{\int_{-\infty}^{r_1} \cdots \int_{-\infty}^{r_1}}_{n \text{ times}} -x_1\, e^{-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}}\, \mathrm{d}\mathbf{x} \right. \\
&\quad \left. - \int_0^{\infty} \underbrace{\int_0^{r_1} \cdots \int_0^{r_1}}_{n-1 \text{ times}} -x_1\, e^{-\frac{\mathbf{x}^T\mathbf{x}}{2\sigma^2}}\, \mathrm{d}\mathbf{x} \right\} = \ldots
\end{aligned}$$

$$\begin{aligned}
\ldots &= \frac{1}{(2\pi)^{n/2}\,\sigma^n} \left\{ - \left[\frac{\sqrt{\pi}\sigma}{\sqrt{2}}\right]^{n-1} \left[1 + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\right]^{n-1} \int_{-\infty}^{r_1} -x_1\, e^{-\frac{x_1^2}{2\sigma^2}}\, \mathrm{d}x_1 \right. \\
&\quad \left. - \left[\frac{\sqrt{\pi}\sigma}{\sqrt{2}}\right]^{n-1} \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)^{n-1} \int_0^{\infty} -x_1\, e^{-\frac{x_1^2}{2\sigma^2}}\, \mathrm{d}x_1 \right\} \\
&= \frac{\sigma}{2^{n-1}\sqrt{2\pi}} \left\{ -e^{-\frac{r_1^2}{2\sigma^2}} \left[1 + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)\right]^{n-1} + \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right)^{n-1} \right\}. \qquad (4.30)
\end{aligned}$$

For a later use, we define $\beta(r_1, \sigma)$ via the relation $e_n(r_1, \sigma) = \beta(r_1, \sigma)\sigma$, in which β depends exclusively on the ratio α.

Exploiting (4.30) it is possible to compute its derivative with respect to σ. In Figure 4.6 the optimal ratios $\tilde{\alpha}$ are shown for $n = 1, \ldots, 15$. We find out, that the optimal expected progress $\tilde{e}_n(1)$(depicted on the right-hand side) is positive, but it decreases exponentially by a factor of two for increasing n.

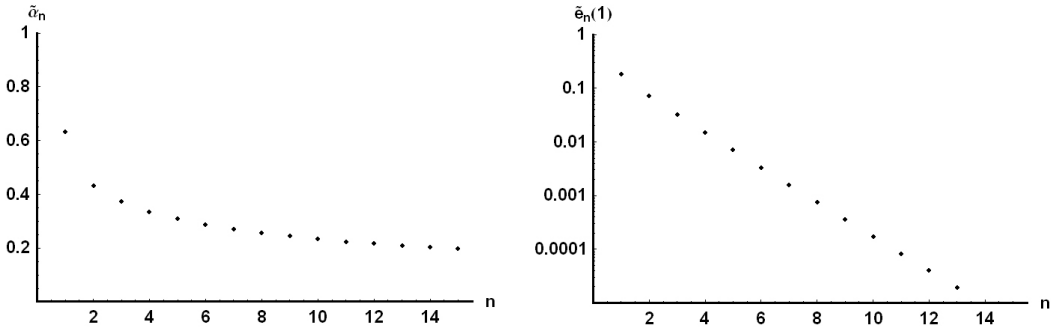


Figure 4.6: Optimal ratio $\tilde{\alpha}_n$ (left) for different n in the worst case sketched in the figures 4.3 and 4.5. The dots on the right hand side denote the corresponding optimal expected progress $\tilde{e}_n(1)$.

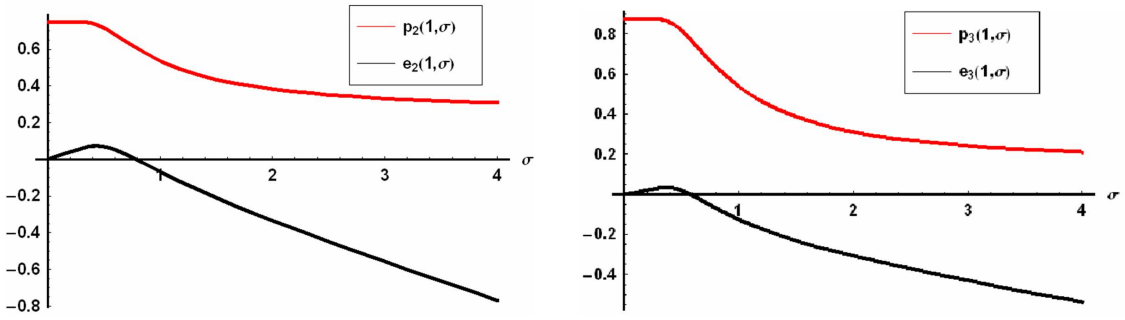
Since the 3-dimensional case is additionally to the 2-dimensional case of special interest, we quote the optimal ratio $\tilde{\alpha}_3$:

$$\tilde{\alpha}_3 = 0.371733\,, \tag{4.31}$$

which leads to a maximal expected progress rate of

$$\tilde{e}_3(r_1) := e_3(r_1,\, \tilde{\alpha}_3\, r_1) = 0.0325971\, r_1\,. \tag{4.32}$$

In Figure 4.7(b) $p_3(r_1,\, \alpha\, r_1)$ and $e_3(r_1,\, \alpha\, r_1)$ are shown for varying ratio α. Due to the linear dependency of e_3 on σ we choose again $r_1 = 1$ and vary α, i.e. respectively σ.



(a) (b)

Figure 4.7: Success probability $p_n(1,\, \sigma)$ and the expected progress $e_n(1,\, \sigma)$ for the assumed front for $n = 2$ (left) and $n = 3$ (right).

Reconsidering (4.30) and Figure 4.6 respectively, we find an exponential decrease of $\tilde{e}_n(r_1)$. That is, because the probability to obtain a dominating individual converges to $1/2^n$ if the ratio α converges to infinity, which is expected, because the volume of the dominated domain is in general, for the n-dimensional case, $\mathcal{O}(\frac{1}{2^n})$ with $\alpha \to \infty$. Hence, the probability to obtain a feasible point, either dominating or indifferent, increases logarithmically with

n. Especially since the volume of the so-called indifferent domain is of the order $1 - \frac{1}{2^{n-1}}$, we expect an logarithmic decrease of the expected progress.

Mutation - A Special Case

Before finalizing the calculations on the expected progress, we have to deal with the special case, in which the selected point $\bar{p}$ has a distance less than r_1 in at least one of the directions $x_2, \ldots, x_n$. The worst case is again if the distance is equal to zero. Then we obtain for $\sigma > 0$ $e_n < 0$. In this case we have to adapt the algorithm, to be able to still guarantee a positive expected progress in distance. We change the algorithm in the way, that we discard all new generated points if they do not dominate the parent. The only thing we have to know is, if we are close to the boundary of the feasible performance domain[14] or not. On the other hand, this situation can also occur if another individual, which is indifferent to the parent, has a smaller distance to the actual selected individual than r_1 in at least one of these specific directions. If so, the domain of improvement is limited by this individual. Hence, we are able to treat this case in the same way as the one described before.

By using the same setting as above, i.e. according to Figure 4.5 but with neglecting x_i, $i = 1, \ldots, n$, we get for the selection probabilities in the limit

$$\begin{aligned} p_{n,s}(r_1, \sigma) &= \int_{-r_1}^{0} \underbrace{\int_{-\infty}^{0} \cdots \int_{-\infty}^{0}}_{n-1 \text{ times}} p(\mathbf{x})\, \mathrm{d}\mathbf{x} = \frac{1}{(2\pi)^{n/2}\,\sigma^n} \frac{\sigma^{n-1}\pi^{(n-1)/2}}{2^{(n-1)/2}} \int_{-r_1}^{0} e^{-\frac{x_1^2}{2\sigma^2}}\, \mathrm{d}x_1 \\ &= \frac{1}{2^n} \operatorname{erf}\left(\frac{r_1}{\sqrt{2}\,\sigma}\right). \end{aligned} \tag{4.33}$$

Furthermore the expected progresses for the n-dimensional case are given by

$$e_{n,s}(r_1, \sigma) = \int_{-r_1}^{0} \underbrace{\int_{-\infty}^{0} \cdots \int_{-\infty}^{0}}_{n-1 \text{ times}} -x_1\, p(\mathbf{x})\, \mathrm{d}\mathbf{x} = \frac{\sigma}{2^{n-1}\sqrt{2\pi}} \left(1 - e^{-\frac{r_1^2}{2\sigma^2}}\right). \tag{4.34}$$

In that case, we obtain an optimal ratio $\tilde{\alpha}_n = \tilde{\sigma}_n(r_1)/r_1$ of

$$\tilde{\alpha}_n = 0.630835\,, \tag{4.35}$$

which result in the optimal expected progress given by

$$\tilde{e}_{2,s}(r_1) = e_{2,s}(r_1, \tilde{\alpha}_2\, r_1) = 0.0900126 r_1 \tag{4.36}$$

$$\tilde{e}_{3,s}(r_1) = e_{3,s}(r_1, \tilde{\alpha}_3\, r_1) = 0.0450063 r_1\,. \tag{4.37}$$

[14] Note, that we have to make the important assumption to have some measure for the distance of a point to the boundary of the feasible domain. It is needed for both domains, for the design domain as well as for the performance space, to be able to obtain convergence of infeasible points to the feasible domain. This was explained in Subsection 4.4.1.

Note that due to (4.34) the expected progress decreases also exponentially with the factor 2 for increasing n. This is due to the exponential decrease of the probability to obtain a new individual in the dominating domain. Note, that $\tilde{e}_{n,s}$ is still positive, but the improvement factor becomes very small too.

After this small adaption due to convergence problems note, that for further discussions we can neglect this case, because the optimal expected progress in the n-dimensional case, given by (4.36) and (4.37) for instance, is higher than in the "standard" case (cf. (4.28), (4.32) and Figure 4.6).

4.5.7 Environmental Selection - Clustering

Finally the selection step is performed. If the new point $\mathbf{x}$ dominates the parent individual, then the parent is discarded. Furthermore we assume as explained above, that the new individual is only not added to the archive if it is dominated by the parent. As mentioned previously too, we do not consider other limiting individuals, since they would only limit the valid domain, which would result in an increased expected progress.

4.5.8 Summarizing the Convergence

Summarizing the previous results, using $p_{s,d}$ given by (4.20), $\tilde{e}_n$ and $\tilde{\alpha}_n$, respectively, given by (4.28), (4.32), (4.27) and (4.31), or depicted in Figure 4.6, we find that under the assumptions made in Subsection 4.5.3 the following holds:

$$\begin{aligned} e(d) &= \|P_t\|_2 - E(D_{t+1}|\tilde{P}_t = P) = \|P_t\|_2 - \|E(P_{t+1}|\tilde{P}_t = P)\|_2 \geq d - \frac{1}{\bar{N}}\sqrt{d(\mathbf{p}_{\text{wc}}, P_f)^2} \\ &= \frac{1}{\bar{N}} p_{s,d}\; e_n(\bar{N}d,\, \tilde{\sigma}(\bar{N}d)) = \frac{1}{\bar{N}}\, p_{s,d}\; \tilde{e}_n(\bar{N}d) = \frac{\beta(\tilde{\alpha}_n)}{(\bar{N}+1)^q}\, d\,, \end{aligned} \tag{4.38}$$

with $\|\bar{P}\| = d$. $\mathbf{p}_{\text{wc}}$ denotes the new computed individual in the worst case as assumed in the previous chapters and therewith, by defining γ as

$$\gamma(d) := \zeta_n\, d \qquad \text{with} \qquad \zeta_n := \frac{\beta(\tilde{\alpha}_n)}{(\bar{N}+1)^q}\,, \tag{4.39}$$

we fulfill condition (4.18) and therefore we obtain almost sure convergence for this special worst case due to Theorem 4.5. In (4.38) and (4.39) β denotes the factor corresponding to $\tilde{e}_n$, i.e. $\tilde{e}_n(r_1) = \beta(\tilde{\alpha}_n)\, r_1$.
Note, that we assumed an optimal standard deviation σ depending on the distance d, but the worst case in terms of the setting with respect to the made assumptions. Additionally bear in mind, that linear convergence is the best we can obtain if no gradient information is used, since then no further information about the behavior of the function except its values is used.

Furthermore note, that this worst case lower bound decreases quadratically with $\bar{N}$, due to the mating selection step. We will compare the results derived in this section with

numerical tests in Subsection 6.2.2. To compare the convergence rate, we compute several values of ζ_n for $n = 2$, $q = 2$ and different $\bar{N}$. They are shown in Table 4.1.

$\bar{N}$	ζ_2
1	0.0181726
5	0.0020192
10	0.0006007
20	0.0001648
30	0.0000756
40	0.0000432

Table 4.1: The worst case expected progress factor ζ_2, given by (4.39), for $q = 2$ and different archive sizes $\bar{N}$.

4.5.9 Experimental Verification of the Convergence Considerations

In this short part we want to verify the convergence rates computed in the previous section. Thereby, the expected rates of the mutation operator are of special interest. For this sake, we arrange an experimental setting. For a certain dimension n and σ, we generate n_{exp} vectors by means of the $\mathcal{N}(0, \sigma^2 I)$ distribution with some $\sigma > 0$. First of all we are dealing with the case depicted in the Figures 4.3 and 4.5 with $r_1 = 1$. Therefore, we say that a generated point $\mathbf{d}$ is feasible, if it lies in the domain of interest, i.e. if

$$\mathbf{d} \in P_I := \left\{[-1,\, \infty] \times [-\infty,\, 1]^{n-1}\right\} \setminus \left\{[0, \infty] \times [0,\, 1]^{n-1}\right\} . \tag{4.40}$$

From the generated vectors $\{\mathbf{d}^n_{i,\sigma}\}_{i=1,\,\ldots,\,n_{\text{exp}}}$ we compute the following values

$$p_{n,\text{exp}}(1,\, \sigma) \;:=\; \frac{|\{\mathbf{d}^n_{i,\sigma} \in P_I \,:\, i = 1,\, \ldots,\, n_{\text{exp}}\}|}{n_{\text{exp}}} \qquad \text{and} \tag{4.41}$$

$$e_{n,\text{exp}}(1,\, \sigma) \;:=\; \frac{\sum_{\mathbf{d} \in \{\mathbf{d}^n_{i,\sigma} \in P_I \,:\, i=1,\,\ldots,\,n_{\text{exp}}\}} -d_1}{n_{\text{exp}}} . \tag{4.42}$$

In other words, $p_{n,\text{exp}}$ denotes the relative frequency of the event to obtain a feasible point and $e_{n,\text{exp}}$ indicates the corresponding progress of the feasible points.
For all experiments, we used $n_{\text{exp}} = 5000$ and for σ a step size of 0.02 was chosen. Furthermore, the experiments were performed for $\sigma \in [0,\, 2]$. In Figures 4.8(a) and 4.8(c) $p_{n,\text{exp}}(1,\, \sigma)$ is compared to $p_n(1, \sigma)$, derived in (4.24) and (4.29), for $n = 2$ and $n = 3$. Figures 4.8(b) and 4.8(d) show $e_{n,\text{exp}}(1,\, \sigma)$ in relation to $e_n(1, \sigma)$, computed in (4.26) and (4.30), again for $n = 2$ and $n = 3$.
Furthermore, in Figures 4.8(e) and 4.8(f) the success probability $p_{2,s,\text{exp}}(1,\, \sigma)$ and the expected progress $e_{2,s,\text{exp}}(1,\, \sigma)$ for the special case are compared to the derived values $p_{2,s}(1,\, \sigma)$ and $e_{2,s}(r_1,\, \sigma)$ (cf. (4.33) and (4.34)) for $n = 2$. For this case, the definitions (4.40), (4.41) and (4.42) have to be adapted to the different domain of interest.

For $n_{\text{exp}} \to \infty$ the graphs obtained by the experiments would converge with probability

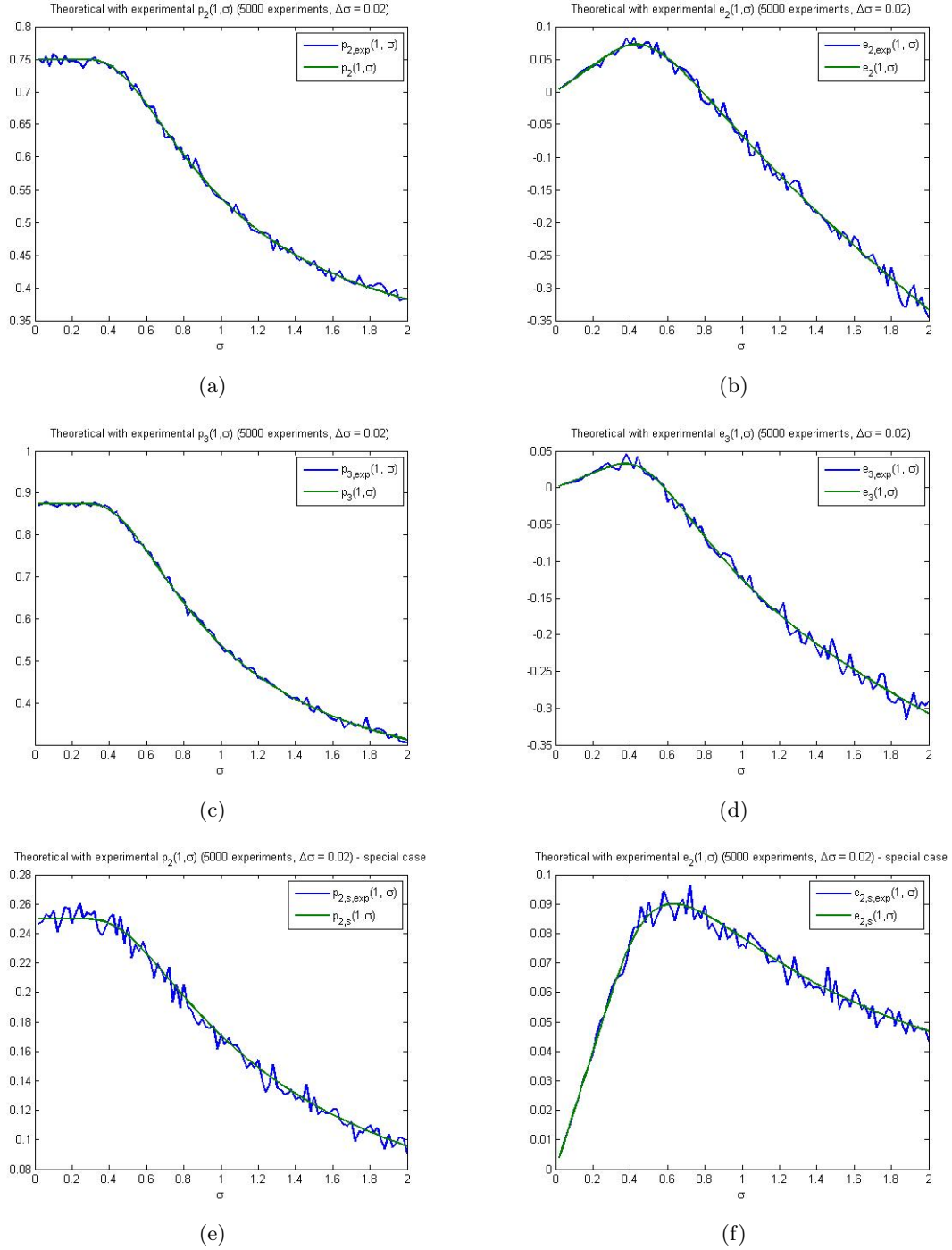


Figure 4.8: Experimental Investigations on the the mutation operator. On the left side the exact and experimental obtained success probabilities are shown. The corresponding expected progresses are depicted on the right hand side. The dimensionality of the experiments is $n = k = 2, 3$. Furthermore, in last row, the special case, where only dominating individuals are allowed, is depicted for $n = 2$.

one to the exact curves. Note, that due to a rule of thumb[15] we can expect an accuracy $10^{-1.85}$, because $5000 \approx 10^{3.70}$. The proof of this rule needs some knowledge about the normal distribution and probabilistic theory, but we will skip it in here.

[15]It states, that if we want to achieve an accuracy of the relative frequency compared to the exact probability of 10^{-k}, we have to carry out 10^{2k} random experiments.

4.6 Summary

In this chapter we first introduced the basic concepts of evolutionary algorithms. It was explained, that EAs are population based algorithms where the population changes due to deterministic or probabilistic selection, recombination and due to probabilistic mutation. These concepts orientate on incidents occurring in nature. The basic properties and ideas concerning evolutionary algorithms were explained. Furthermore, the components of these algorithms were listed and explained. In the beginning, the focus was still on single objective optimization problems. Three different kinds of EAs were introduced.

Afterwards, the adjustments of evolutionary algorithms to multiobjective optimization problems were treated. First, the aims of multiobjective evolutionary algorithms were presented, which are to obtain accurate and uniformly distributed approximations. Moreover, the basic concepts fitness sharing and elitism were introduced. They are important for an efficient MOEA. Then, other algorithms available were shortly summarized. Subsequently, SPEA2 was introduced in detail. It was preferred to the other methods, since it showed promising results in literature and because no drawbacks compared to other efficient methods could be determined.
Basically, SPEA2 is an Evolution Strategy, since recombination and mutation with normal distribution are used, although the additional step of environmental selection contradicts this definition.

Finally, almost sure convergence of this approach was shown under some assumptions. This was carried out by means of the worst case with respect to the prescribed assumptions. The convergence was shown in terms of an expected improvement of the population's distance to the Pareto front for every generation. The rate is linear, what is the best we can expect if no gradient information is used. Finally, the derived convergence rates for the mutation operator were verified by the help of probabilistic experiments.

Chapter 5

A Deterministic Approach for Multiobjective Optimization Problems

In this chapter we will investigate on deterministic numerical solution methods for MOPs with focus on so-called *Normal-Boundary Intersection*(NBI) method, which was introduced in [DD98] and further investigated in [Das98]. Moreover, in [Ste05] NBI was used to execute the design space exploration, which was termed performance space exploration there.

The task of multiobjective optimization is to determine the Pareto optimal front P_f. Since this is usually impossible to achieve, numerical solution methods aim to determine a finite set of Pareto optimal solutions instead. Basically there are two different goals when dealing with multiobjective optimization. Either one single or *final* optimal solution is desired, or we want to stress the trade-off between the different objectives. For the former goal a so-called *decision maker*, a person who has better insight in the background of the problem, has to pick one solution from the computed solution set, which fulfills his/her requirements best[1]. Since this case is not of importance for the sake of design space exploration we turn to the second goal. Thereby, the important task is to achieve a certain distribution of the optimal points. Usually it is desired to get a uniform distribution of optimal solutions along P_f.

The idea of most MOP solution methods is to transform the problem into parameterized nonlinear single optimization problems (SOP). Firstly, in Section 5.1 two traditional approaches are presented and their disadvantages will be outlined. Afterwards, in Section 5.2 we will deal with the NBI approach.

[1]For more details see [Mie99] for instance.

5.1 Other Deterministic Solution Methods

Before we will introduce the normal-boundary intersection, two other quite common methods to solve MOPs will be presented, namely the *weighting method* and the *ε-constraint method*. Furthermore, their relation to each other will be outlined and their disadvantages will be discussed.

5.1.1 Weighting Method

In the *weighting method* the multiobjective optimization problem is transformed into a single objective problem parameterized by some vector $\mathbf{w}$. The scalar cost function is obtained by assigning non-negative weights w_i to each performance variable and by summing up the weighted performances. The problem reads then as follows

$$\min_{\mathbf{d}\in X_f} \sum_{i=1}^{k} w_i p_i(\mathbf{d}) = \mathbf{w}^T \mathbf{p}(\mathbf{d})\,, \tag{5.1}$$

with $w_i \geq 0,\ i = 1, \ldots, k$ and $\sum_{i=1}^{k} w_i = 1$. In [Mie99] the following results on the weighting approach can be found as well as their proofs (cf. p. 78 ff.). Summarizing the statements in one theorem we obtain

Theorem 5.1. *The solution of the weighting problem* (5.1) *is weakly Pareto optimal. Furthermore the solution is Pareto optimal*

- *if the weighting coefficients are positive, i.e. $w_i > 0$ for all $i = 1, \ldots, k$.*
- *or if the solution of the weighting problem* (5.1) *is unique.*

Furthermore, after some investigations, the following quite obvious result can be stated:

Theorem 5.2. *Let the multiobjective optimization problem be convex. If $\mathbf{d}^* \in X_f$ is Pareto optimal, then there exists a weighting vector $\mathbf{w}$ ($w_i \geq 0,\ i = 1, \ldots, k,\ \sum_{i=1}^{k} w_i = 1$) such that $\mathbf{d}^*$ is a solution of the weighting problem* (5.1).

The proof is given in [Mie99] (p. 79). The standard procedure of the weighting method, for approximating the trade-off front, is to choose different weights, usually uniformly distributed, and to solve for each of them the single objective optimization problem (5.1). In that way we obtain several optimal points that can be used to set up an approximation for the Pareto front. Furthermore, for single $w_i = 1$ we obtain the individual minima, which limit the Pareto front.
It is worth noticing, that Theorem 5.2 states the existence of a weight $\mathbf{w}$ for each optimal point if the problem is convex. But we cannot make any conclusion about the distribution of those weights.

In [DD97] the disadvantages of the weighting method are investigated. Following the previous reasoning, the main weaknesses of this approach are

- If the Pareto curve is not convex, there does not exist any $\mathbf{w}$ for which the solution to problem (5.1) lies in the nonconvex part.
- Even if the Pareto curve is convex, an even spread of weights $\mathbf{w}$ does not necessarily produce an even spread of points on the Pareto curve.

In this article the two arguments mentioned above are investigated and additionally proven by means of examples.

5.1.2 ε-Constraint Method

Another interesting and often used MOP solution method is the *ε-constraint method*. The interesting intrinsic peculiarity of this approach will be pointed out later .
For some $s \in \{1, \ldots, k\}$ and $\varepsilon \in \mathbb{R}^{k-1}$ the ε-constraint method corresponding to Problem 3.1 consists of the following optimization problem

$$\begin{aligned} &\min_{\mathbf{d} \in X_f} p_s(\mathbf{d}) \\ \text{subject to} \quad &p_j(\mathbf{d}) \leq \varepsilon_j \quad \text{for all } j = 1, \ldots, k,\ j \neq s\,. \end{aligned} \tag{5.2}$$

In [Mie99] it is shown, that the solution of the ε-constraint problem (5.2) is weakly Pareto optimal (Theorem 3.2.1, p. 85). Furthermore the following theorem is proven therein:

Theorem 5.3. *A decision vector $\mathbf{d}^* \in X_f$ is Pareto optimal if and only if it is a solution of the ε-constraint problem* (5.2) *for every $s = 1, \ldots, k$, where $\varepsilon_j = p_j(\mathbf{d}^*)$ for $j = 1, \ldots, k,\ j \neq s$.*

Finally it is shown, that if the solution of problem (5.2) is unique for any s and any ε, the solution is Pareto optimal. In Figure 5.1 an example for several solutions of a two-dimensional problem for different ε is depicted. We can clearly see, that for ε^1 no solution exists, while for the other ε^j, $j = 2, 3, 4$ we obtain the distinct solutions $\mathbf{p}^j$. Hence by adjusting the bounds ε_j it is possible to obtain solutions along the whole Pareto front.

When reviewing this method, we find that also this method has the disadvantage, that a uniform distribution of the points cannot be achieved in general, since no relation between different ε and the distribution of the corresponding Pareto optimal solutions can be drawn. Therefore, this approach is not the method of choice too.

Under certain assumptions the equivalence between the different solution methods for multicriteria optimization can be established. In the following we will state the results on the relation between the weighting method presented in the previous subsection and the ε-constraint method, that are again derived in [Mie99] (p. 88).

Theorem 5.4. *Let $\mathbf{d}^* \in X_f$ be a solution of weighting problem* (5.1) *and let $\mathbf{0} \leq \mathbf{w} \in \mathbb{R}^k$ be the corresponding weighting vector. Then*

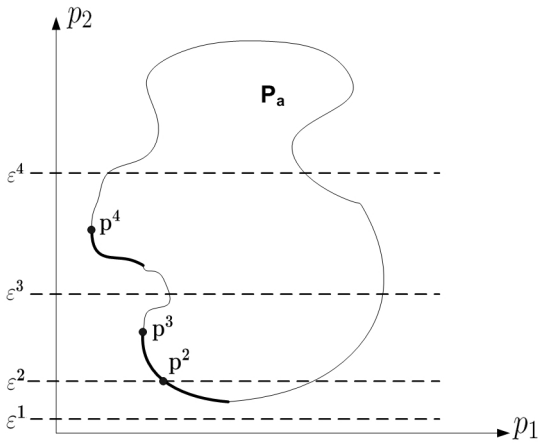


Figure 5.1: Different constraints for the ε-constraint problem (5.2). The bold line denotes the Pareto front P_f corresponding to the attainable set P_a.

1. *if* $w_s > 0$, $\mathbf{d}^*$ *is a solution of the* ε*-constraint problem for the objective function* p_s *and* $\varepsilon_j = p_j(\mathbf{d}^*)$ *for* $j = 1, \ldots, k,\ j \neq s$. *Or,*

2. *if* $\mathbf{d}^*$ *is a unique solution of the weighting problem* (5.1)*, then* $\mathbf{d}^*$ *is a solution of the* ε*-constraint problem when* $\varepsilon_j = p_j(\mathbf{d}^*)$ *for* $j = 1, \ldots, k,\ j \neq s$ *and for every* $p_s,\ s = 1, \ldots, k$, *as the objective function.*

Furthermore note the fact, that for convex problems, there exists a weighting vector $\mathbf{0} \leq \mathbf{w} \in \mathbb{R}^k$, $\sum_{i=1}^k w_i = 1$, such that $\mathbf{d}^*$ is a solution to problem (5.1) if $\mathbf{d}^*$ is a solution to problem (5.2) for some p_s to be minimized and $\varepsilon_j = p_j(\mathbf{d}^*)$ for $j = 1, \ldots, k,\ j \neq s$.
Summarizing we find the equivalence of ε-constraint problem and the weighting method for convex problems.

We mentioned above, that this method has an interesting property. To point this out, we assume that the objective and constraint functions are twice continuously differentiable. Suppose that a feasible point $\mathbf{d}^*$, which satisfies the ε-constraints, fulfills the second order sufficient solutions with the Karush-Kuhn-Tucker multipliers λ_{sj} for the ε-constraints[2]. Thereby, the index s denotes the choice of the minimization function. Then, if the multipliers are strictly positive, we get for the trade-off rates, introduced in Definition 3.22,

$$\lambda_{sj} = -\frac{\partial p_s(\mathbf{d}^*)}{\partial p_j} \quad \text{for all } \ j \neq s\,, \tag{5.3}$$

i.e. the Lagrange-multipliers λ_{sj} determine the trade-off rate involving p_s and p_j.

5.2 Normal-Boundary Intersection Method

In the previous section, we pointed out the disadvantages of the weighting method and the ε-constraint method. Therefore, we will introduce the *normal-boundary intersection*

[2]For more information see [Mie99], p. 92.

(NBI) method, which aims to remedy the problems of the previously explained methods.

As mentioned above, the normal-boundary intersection method was introduced in [DD98]. In the following we will explain the method and its background. In the next subsection the required general definitions, and the basic idea of NBI will be worked out. In Subsection 5.2.2 further investigations on the normal-boundary intersection are shown and the basic properties of the approach are mentioned. Furthermore in Chapter 6 some implementation issues are discussed (see Subsection 6.1.3) and the NBI is applied to a problem concerning the design space exploration of compact models.

5.2.1 Central Ideas of NBI

First of all, we have to define the so-called *individual minima* p_i^* of the performances corresponding to the MOP 3.1. They are obtained by the following single objective optimization problem:

$$p_i^* := \min_{\mathbf{d} \in X_f} p_i(\mathbf{d}) \qquad i = 1, \ldots, k\,, \tag{5.4}$$

where the feasible set X_f is given by Definition 3.2. The p_i^* form the *shadow minimum* or *utopia point* $\mathbf{p}^*$, i.e.

$$\mathbf{p}^* := \begin{pmatrix} p_1^* \\ p_2^* \\ \vdots \\ p_n^* \end{pmatrix}. \tag{5.5}$$

Henceforth, we assume the existence of an individual minimizer p_i^* for each $i = 1 \ldots, k$. This is guaranteed because we suppose X_f to be compact, i.e. closed and bounded, and $\mathbf{p}$ to be at least continuous. Additionally, let $\mathbf{x}_i^*$ be the minimizer corresponding to p_i^*. Clearly, the $\mathbf{x}_i^*$ do not have to be unique, but for the first considerations this is not decisive, since especially $\mathbf{p}(\mathbf{x}_i^*)$ is of interest. Furthermore observe, that the shadow minimum cannot be attained in general, but one of the goals could be to come as close as possible to $\mathbf{p}^*$.

Let us define the $k \times k$ *pay-off matrix* Φ by

$$(\Phi_{ij})_{i,j=1,\ldots,k} := p_i(\mathbf{x}_j^*) - p_i^*\,, \tag{5.6}$$

i.e. the matrix whose j-th column consists of $\mathbf{p}(x_j^*) - \mathbf{p}^*$. Note the special structure of Φ, namely

$$\begin{aligned} \Phi_{ii} &= 0 \qquad \text{for all } i = 1, \ldots, k \quad \text{and} \\ \Phi_{ij} &\geq 0 \qquad \text{for all } i, j = 1, \ldots, k,\ i \neq j\,. \end{aligned} \tag{5.7}$$

This structure can be exploited in the way that a negative entry Φ_{ij} signifies that $\mathbf{x}_i^*$ is not the global minimizer of p_i because then $\mathbf{x}_j^*$ results in a smaller value.

Now we are able to define the *convex hull of individual minima*($CHIM$[3]). As the name

[3]Note, that we use the terminology introduced in [DD98].

indicates, it is defined by

$$CHIM := \left\{ \Phi\beta \,:\, \beta \in \mathbb{R}^k,\ \sum_{i=1}^{k} \beta_i = 1,\ \beta_i \geq 0 \,\text{for all}\, i = 1, \ldots, k \right\}. \tag{5.8}$$

An example of a Pareto front including its corresponding $CHIM$ is depicted in Figure 5.2. Similar to the $CHIM$, we can define $CHIM_\infty$, the affine subspace of lowest dimension containing the $CHIM$. It is given by $CHIM_\infty := \left\{ \Phi\beta \,:\, \beta \in \mathbb{R}^k,\ \sum_{i=1}^{k} \beta_i = 1 \right\}$. Finally, using the definitions 3.2 and 3.3, we are able to set up $CHIM_+$, which is given by $CHIM_+ := \text{co}(\partial P_a \cap CHIM_\infty)$, where ∂P_a denotes the boundary of the attainable performances P_a. We have, that $CHIM = CHIM_+$ for $k = 2$, but for general $k > 2$ $CHIM$

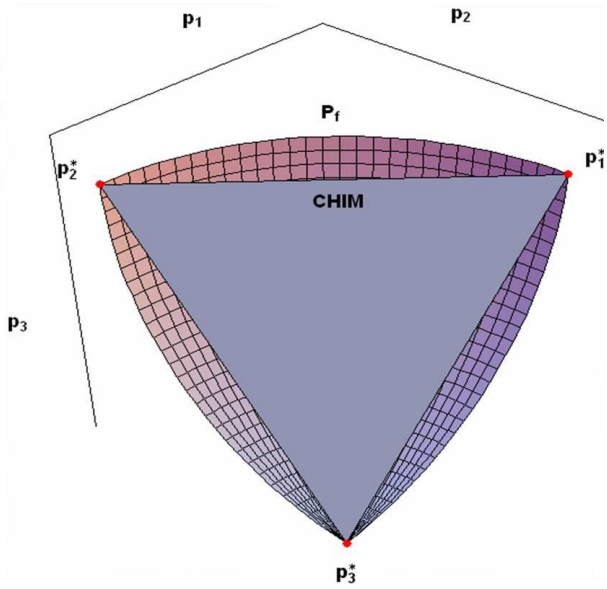


Figure 5.2: The Pareto optimal front P_f and the $CHIM$ for an example with $k = 3$. The red points denote the individual minima p_i^*.

is usually not equal to $CHIM_+$. As an example, Figure 5.2 shows a typical CHIM. We can see, that ∂P_a does not touch the $CHIM$ at all points of its boundary, and hence $CHIM \neq CHIM_+$ in this case. Note, that all points of ∂P_a lying below $CHIM_+$ are Pareto optimal if the surface is not "too concave" there. Furthermore we have to remind our original aim, which is to compute a trade-off front that does not have to be equal to the Pareto front. P_f is usually a subset of this trade-off front. For the sake explained before, it is desirable to have the $CHIM_+$. This is almost impossible and hence we are content with the approximation given by $CHIM$.

In the next step of the method several Pareto optimal points are computed in the following way:
We determine the normal vector $\mathbf{n}$ of the $CHIM$ pointing towards $\mathbf{p}^*$. Then, we choose an arbitrary point on this convex set corresponding to a vector $\beta \in [0,1]^k$ with $\sum_{i=1}^{k} \beta_i = 1$. In consequence we want to compute the feasible point $\mathbf{p}(\mathbf{d})$, which has the largest distance t of the point $\Phi\beta$ in direction $\mathbf{n}$. In other words, we want to solve the following single

objective optimization problem

$$\min_{(t,\,\mathbf{d})\in\mathbb{R}\times X_f} -t \tag{5.9}$$

subject to

$$\mathbf{p}(\mathbf{d}) - \mathbf{p}^* = \Phi\beta + t\mathbf{n}\,. \tag{5.10}$$

Note, that $\mathbf{d}$ has to be feasible, i.e. $\mathbf{d} \in X_f$. Furthermore, for fixed β we refer to this subproblem with NBI_β. Hence, the next task to do, is to choose the vectors β to obtain an as accurate as possible approximation of the front. The most simple choice is a uniform distribution. That is, given a certain positive integer n_β, we choose $\beta_1 \in \{\frac{s}{n_\beta} \mid s = 0, \ldots, n_\beta\}$ and the remaining β_i are chosen from the set $\{\frac{s}{n_\beta} \mid s = 0, \ldots, n_\beta \wedge s \leq n_\beta(1 - \sum_{j=1}^{i-1}\beta_j)\}$. The number of subproblems NBI_β with uniform distribution is given by $\begin{pmatrix} n + n_\beta - 1 \\ n_\beta \end{pmatrix}$.

In Figure 5.3 a sketch of the normal-boundary intersection method is depicted. Therein, we see the almost uniform distribution of the obtained points along the Pareto front P_f. Furthermore observe that the shadow minimum $\mathbf{p}^*$ cannot be assumed in this example.

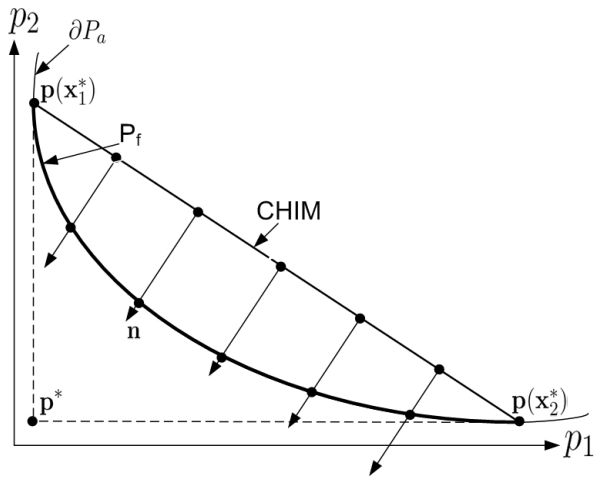


Figure 5.3: Sketch of the NBI method. The uniform distribution of the obtained points can be observed.

Concluding this treatment on NBI note, that NBI remedies the problems of the weighting and ε-constraint method, i.e. the obtained solutions are expected to be uniformly distributed along the trade-off front and moreover, it is possible to obtain non-Pareto optimal points, what is especially in the design space exploration desired.

5.2.2 Further Considerations on NBI

Since for general k it is quite tricky to set up the normal vector $\mathbf{n}$, the quasinormal $\tilde{\mathbf{n}} = -\Phi\mathbf{e}$ is often used instead, with $\mathbf{e}$ being the vector containing only ones. In [DD98] (p. 645 ff.) it is shown, that using the quasinormal $\tilde{\mathbf{n}}$ the obtained point is independent of any scaling of the performances for a particular β.

In Figure 5.4 a sketch of the normal-boundary intersection method is depicted, where $\mathbf{n}$ was replaced by $\tilde{\mathbf{n}}$. Therein, we see again the almost uniform distribution of the obtained points along the Pareto front P_f (compare to Figure 5.3), although the quasinormal was used. Hence, we see that for smooth objective functions $\mathbf{p}$ the normal $\mathbf{n}$ can be replaced by the quasinormal $\tilde{\mathbf{n}}$. Clearly, the steeper the slope of the CHIM becomes, the worse the results are.

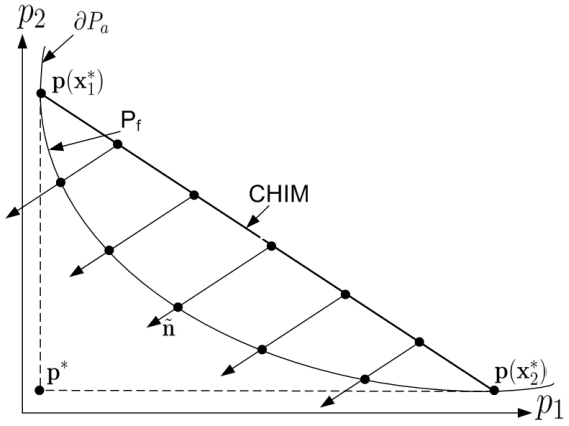


Figure 5.4: Sketch of the NBI method with the quasinormal $\tilde{\mathbf{n}}$.

We mentioned above, that originally the desired set would be $CHIM_+$. This set would cover the whole trade-off front. The big advantages of the $CHIM$ are its simple parameterization by means of the barycentric coordinates β, and the sufficiency in terms of recovering the trade-off front. It is worth noticing that in general design space exploration applications these boundary terms of the front are not of interest. But to extend the solutions we could apply the general principle of NBI to the $(k-1)$-dimensional subproblem at the boundary surfaces of the $CHIM$. There, we have one degree of freedom to choose the normal vector $\mathbf{n}$. Therewith it would be possible to obtain all the points left over in the boundary regions. Since these points are not of main interest in the framework of design space exploration, we do not pursue this topic further.

Now, we consider the necessary first order Karush-Kuhn-Tucker optimality conditions for the subproblem NBI_β, that is Problem (5.9). For a feasible pair $(t^*, \mathbf{d}^*)$ there have to exist the multipliers $\lambda_{\mathbf{1}}^* \in \mathbb{R}^k$, $\mathbf{0} \leq \lambda_2^* \in \mathbb{R}^l$ and $\mathbf{0} \leq \tilde{\lambda}_2^* \in \mathbb{R}^m$ such that

$$-\nabla_d \mathbf{p}(\mathbf{d}^*)^T \lambda_1^* + \nabla_d \mathbf{e}(\mathbf{d}^*)^T \lambda_2^* + \nabla_d \tilde{\mathbf{e}}(\mathbf{p}(\mathbf{d}^*))^T \tilde{\lambda}_2^* = \mathbf{0} \tag{5.11}$$

$$-1 + \mathbf{n}^T \lambda_1^* = 0\,. \tag{5.12}$$

Additionally $\lambda_{2,i}\, e_i(\mathbf{d}^*) = 0$ and $\tilde{\lambda}_{2,j}\, \tilde{e}_j(\mathbf{p}(\mathbf{d}^*)) = 0$ have to hold for every $i = 1, \ldots, l$ and every $j = 1, \ldots, m$. In [DD98] it is shown, that if $\sum_{i=1}^k \lambda_{1,i}^* \neq 0$ and $\mathbf{0} \leq \lambda_1^*$, then $\mathbf{d}^*$ is the solution of the weighting method, introduced in Subsection 5.1.1, with the weights

$$w_j = \frac{\lambda_{1,j}^*}{\sum_{i=1}^k \lambda_{1,i}^*}\,. \tag{5.13}$$

We can conclude that at least one of the weights λ_1^* corresponding to a solution of NBI$_\beta$ in a concave region has to be negative, because it is not possible to obtain points at ∂P_a in concave regions with the weighting method. On the other hand, from the previous statements it cannot be concluded that the multipliers have to be positive in convex areas. This is an interesting question. Nevertheless this statement can be assumed, since the Lagrange-multipliers always indicate the sensitivity of the corresponding value to a change in the input variables.

In order to save computation time, in [Ste05](p. 42-43) it is recommended to change the equality constraints (5.10) into inequality constraints, i.e. the following setting is suggested:

$$\mathbf{p}(\mathbf{d}) - \mathbf{p}^* \leq \Phi\beta + t\mathbf{n}\,. \tag{5.14}$$

It is said, that this change improves the number of function evaluations between 5% and 20% for typical applications. We have to bear the effects of this change in mind. Reconsidering the equivalence between the weighting method and the normal-boundary intersection we find out, that the conditions mentioned above do not change except the fact that $\lambda_1^* \geq 0$ has to hold if (5.10) is replaced by (5.14). This implies, that all solutions to the adapted NBI$_\beta$ are also solutions to the weighting method with $\mathbf{w}$ given by (5.13). Hence, we do not obtain any points on ∂P_a in concave regions. Since our task is to compute the trade-off front, this property is not desired.

The equivalence of NBI and the ε-constraint method is additionally shown in [DD98]. The conditions for equivalence are again that the $\lambda_{1,j}^* >= 0$ and that there has to be at least one positive one.

5.3 Summary

In this chapter we introduced deterministic solution approaches to deal with multiobjective optimization problems. First, we introduced the weighting and the ε-constraint method. Furthermore, we mentioned the relation between these methods under certain assumptions. But since these methods have some disadvantages we do not apply them to MOPs. The main disadvantages of the weighting method are the impossibility to obtain solutions in concave regions and that the solutions are usually not uniformly distributed along the Pareto front. On the other hand the main disadvantage of the ε-constraint method is the difficult choice of the constraints to obtain uniformly distributed solutions.

To remedy these problems the *normal-boundary intersection* was investigated instead. Thereby, we first compute the individual minima, which limit the Pareto front. Afterwards, the normal vector to the convex hull of the individual minima is set up. Moreover, for a choice of a point on this convex hull we compute a Pareto optimal solution by computing a feasible point that has a maximal distance to the chosen point along the normal vector. Then by a uniform distribution of the weighting vector, we get an close-to uniform distribution of the solutions. This method overcomes all the weaknesses of the other

two presented methods and therefore, we will apply NBI to the design space exploration method in the next chapter.

Chapter 6

Applications of the Evolutionary and the Deterministic Method

In the previous two chapters, the probabilistic approach SPEA2 and the deterministic method NBI were introduced and explained in detail. In this chapter we will emphasize on the practical side of both approaches.

Therefore, the issues and details concerning the implementation of SPEA2 and NBI will be explained. Afterwards, both methods will be assessed by means of a simple example. Furthermore, we will apply SPEA2 and NBI to a compact performance model of a low-noise amplifier. Thereby, a possible flow of designing a circuit to obtain the design parameters corresponding to desired performance figures will be outlined. Finally, the considerations and observations of this chapter will be summarized.

6.1 Implementation

In this section we will make, additionally to the already stated considerations in the previous Sections 4.4 and 5.2, some remarks concerning the implementation of SPEA2 and NBI. First we will point out general issues of implementing a MOP according to Problem 3.1. Afterwards, SPEA2 and NBI will be treated in detail.

6.1.1 General Considerations on the Implementation

We implemented both methods in *MATLAB®*[1], but basically they can be implemented in any programming language. Due to the various possibilities of visualization, the big amount of available vector manipulation functionalities and the function *fmincon*, which is an optimization algorithm for SOPs, we chose this program.

[1] for more information see [Mat].

First of all remember that in general we have t performance variables and k objectives to optimize. Thus, we introduce the vector `optimize` of length t, whose elements are in $\{-1, 0, 1\}$. These values determine, whether we want to maximize the performance, to use it as a constraint or to minimize the objective, respectively. k elements of `optimize` are not equal to zero. Furthermore, we assume that the domains D and Y of Problem 3.1 are given by hypercubes, i.e. we are given arrays `limitDS` and `limitPF` of the size $2 \times n$ and $2 \times t$, which contain the upper and lower bounds for the single design and performance parameters. We assumed this for simplicity[2] and since it is convenient for the general procedure of designing a RF-circuit by means of RF-circuit block models, which will be explained in Section 6.3. We summarized all the settings of an MOP in the framework of design space exploration in the class `Blockmodel_RSM`.

In Subsection 6.1.2 we will introduce the difficulties when implementing SPEA2 and in Subsection 6.1.3 the NBI approach will be investigated.

6.1.2 Implementation of SPEA2

Now, we want to elucidate the details on the implementation of SPEA2 as explained in Subsection 4.4. First of all note that not all steps need special treatment additionally to the already discussed items. Especially the mating selection, which chooses N individuals by means of the q-tournament selection, and the recombination step are straightforward to implement.

Initial Population

An important task for a fast success of SPEA2 is to make a good choice for the initial population P_0. There are three different possibilities to generate P_0:

1. *Random generation.* Randomly generate N points in the range of the design space, given by `limitDS`. Of course, not all individuals have to be in X_f. Actually, for problems with severely constrained performances it can be almost impossible, to generate any feasible initial individual. The treatment of infeasible points will be explained below.

2. *Precomputed data.* We could reuse the results of a previous optimization run, that maybe had similar constraints on the data. Note, that in the case of circuit design, as will be explained later, the designer restricts the problem more and more, until he/she obtains a unique solution. Thereby, the data of the previous computation could be used as an initial population for the next step.

3. *Reference table.* Another possibility is to pre-compute a table of performances $\mathbf{p}$ with a large number of different design parameters. To get N initial values, we

[2] In the following, we need to know the distance of infeasible points to the closest feasible one. This measure is needed in both spaces.

discard the infeasible points. Then we choose the $\lfloor \frac{N}{k} \rfloor$ minimal/maximal points of each performance p_i, $i = 1, \ldots, k-1$ that is to be optimized. For the last one, we choose $N - k\lfloor \frac{N}{k} \rfloor$ optimal individuals. This should result in an excellent initial population. Especially due to recombination, we will reach a uniformly distributed coverage of the whole front very fast, if the chosen points are situated close enough at the individual minima.
Of, course, selecting a uniformly distributed approximation of the front would be even better. But this is much more complicated too. Hence it is not used.

Stopping Criterion

The stopping criterion is a very crucial point in SPEA2. We implemented two different convergence criteria. The first one is the quite obvious maximum number of generations T and the second criterion was introduced in Subsection 4.4. It uses the convergence index $q(t, G)$ defined by (4.13). Thereby, the difficult task is to choose a suitable bound T_C and step size G. For the choice of T_C observe the following considerations:

In order to determine optimal settings we will investigate the probability $p_{\mathrm{s}}(j)$, to obtain j new dominating individuals, by neglecting recombination for simplicity. It can be computed as follows:
First of all we assume, that $\bar{N} \geq N$, since otherwise the so-called *selection pressure*[3] would be very high. This means, that it is very likely that a certain element is selected. This likelihood should not be too high, i.e. close to 1, because then the probability that an individual survives several generations is high. Such a property is not desired in order to obtain a fast convergence. Hence, this setting is justified. We are given the probability[4] $p \approx \frac{1}{2^k}$, that the new individual dominates the old one. Note, that there are several possibilities to choose the moments $i_0, \ldots, i_{j-1}$ when an individual $\mathbf{x}$ of the set $\bar{P}_t$ is chosen. Furthermore, if an individual $\mathbf{x} \in \bar{P}_t$ was already chosen, then it is considered to yield no improvement any more. Hence, the probability to obtain a progress, after i individuals of $\bar{P}_t$ have already been considered, is given by

$$p_{\mathrm{s}}(i) = \frac{\bar{N} - i}{\bar{N} + N}\, p\,.$$

The probability to obtain no progress, after considering i individuals with progress, is determined by

$$p_{\mathrm{ns}}(i) = 1 - p_{\mathrm{s}}(i) = \frac{N + i + (1-p)(\bar{N} - i)}{\bar{N} + N}\,.$$

This means, that we do not obtain a dominated individual, firstly, if the mutation of a nondominated individual results in no progress, what happens with probability $1 - p$, or

[3] Compare [Zit99] p. 24. There the selection pressure is related to the takeover time. For this definition we have to assume that P_0 has one optimal individual. The takeover time is the time t it takes until P_t, or $\bar{P}_t$ respectively, is completely filled with this single optimum, if no probabilistic genetic operators are used. That is, P_t contains only this optimum individual N times. A high selection pressure corresponds to a low takeover time, since then it is very likely that the optimum is selected.

[4] Assuming again, as in Section 4.5, a uniformly distributing performance function $\mathbf{p}$.

secondly, if we select an element of P_t or of $\bar{P}_t$ that was already chosen. From this, we get the probability to generate j new dominating individuals:

$$p(\bar{N},\, N,\, p,\, j) \;=\; \sum_{i_0=1}^{N-j+1} \cdots \sum_{i_{j-1}=i_{j-2}+1}^{N} p_{\mathrm{ns}}(0)^{i_0-1}\, p_{\mathrm{ns}}(1)^{i_1-i_0-1} \cdots p_{\mathrm{ns}}(j-1)^{i_{j-1}-i_{j-2}-1} \\ p_{\mathrm{ns}}(j)^{N-i_{j-1}}\, p_{\mathrm{s}}(0) \cdots p_{\mathrm{s}}(j-1) \tag{6.1}$$

Using (6.1), we are now able to compute the expected number of new dominating individuals E. It is given by

$$E(\bar{N},\, N,\, p) = \sum_{j=0}^{N} j\, p(\bar{N},\, N,\, p,\, j)\,. \tag{6.2}$$

The coverage function f_C ((4.12) on p. 46) is defined by the number of dominated individuals divided by the size of the set. Therefore we computed the value $\frac{E(\bar{N},\, N,\, p)}{\bar{N}}$, with $\bar{N} = \alpha N$, numerically for $k = 2$, 3, several α and N. The results are depicted in Table 6.1. We will exploit this table to get an impression of the stopping criterion. This will be carried out in Subsection 6.2.4. Note, that although we only computed these values for N up to 15, those ranges are valid for higher N, since the change between different N decreases asymptotically. We want to use $T_C = \frac{E(\bar{N},\, N,\, p)}{\beta \bar{N}}$, for some $\beta \in \mathbb{R}$. Additionally note the supposed assumption that no individual of $\bar{P}_{t+G}$ is dominated by an element of $\bar{P}_t$, what is quite realistic.

dimension	α	lower bound	upper bound
$k = 2$	1	0.117	0.1250
	2	0.080	0.0833
	3	0.060	0.0625
	4	0.048	0.0500
$k = 3$	1	0.0600	0.06250
	2	0.0408	0.04167
	3	0.0307	0.03125
	4	0.0247	0.02500

Table 6.1: The range of $\frac{E(\bar{N},\, N,\, p)}{\bar{N}}$, $\bar{N} = \alpha N$ for several $\alpha = 1, 2, 3, 4$ and $N = 1, \ldots, 15$. Furthermore, the values are computed for $k = 2$ and $k = 3$.

Nevertheless, when using this stopping criterion, there still exists a certain probability, that the algorithm stops too early. This observation is especially for smaller N valid, or if the feasible performance space Y is very small, compared to $\mathbf{p}(D)$. To prevent this, we could start to test this criterion after some preiterations, e.g. 40 or higher. Furthermore note, that the numerical results in Subsection 6.2.4 will show that this stopping criterion when recombination is used, is not efficient. Nevertheless, in the case of used recombination, $T_C = 0$ and $G = 8$ seem to be the best choice. The same setting can be used, if no recombination is applied. Bu then, this criterion makes much more sense as will be explained in Subsection 6.2.4.

Summarizing, if accurate results are desired, we should stop after T generations, whereby $T = 200$ or even higher is suitable. Especially, we will see later, that the stopping criterion

based on the coverage function should not be used if recombination is applied.

Fitness

The fitness step is quite straightforward to implement as explained in Subsection 4.4. But we have to make one additional remark. Infeasible individuals have to be treated separately, since they might dominate all other individuals. Therefore the standard fitness assignment is carried out on the set of feasible individuals. Then, we compute the distances d_d and d_p of the infeasible individuals to the feasible domains D and Y, i.e.

$$d_d(\mathbf{x}) := \min_{\mathbf{d}\in D} \|\mathbf{d}-\mathbf{x}\| \qquad \text{and} \tag{6.3}$$

$$d_p(\mathbf{x}) := \min_{\mathbf{p}\in Y} \|\mathbf{p}-\mathbf{p}(\mathbf{x})\| \,. \tag{6.4}$$

The fitness of an infeasible individual $\mathbf{x}$ is now given by

$$f(\mathbf{x}) := \delta(d_d(\mathbf{x}) + d_p(\mathbf{x}))\,, \tag{6.5}$$

where δ denotes the penalty term. We chose $\delta = 1000$, which worked very well for the tested constrained as well as unconstrained problems. This kind of punishment has the meaning, that closer individuals are preferred to ones that are farther away.

Environmental Selection - Clustering

Basically, this step was implemented according to Subsection 4.4. We have only to be aware of the normal case, namely if $\bar{N} \geq N$. Then, in the first step where $\bar{P}_t = \emptyset$, we have to copy several individuals more often than once into the archive. This is done in the following way:
Copy $\lfloor \frac{\bar{N}}{N} \rfloor$-times the set P_t and paste it into the set $\bar{P}_t$. Finally, the $\bar{N} - \lfloor \frac{\bar{N}}{N} \rfloor$ individuals of P_t with the lowest fitness value are copied to $\bar{P}_t$.

Note, that such a setting, i.e. $\bar{N} > N$, can be useful, since then the selection pressure is decreased, which was already outlined above.

Mutation

The mutation is the most crucial step of this evolutionary algorithm, as we already noticed in Subsection 4.5, where we investigated the convergence behavior of SPEA2. As stated in this section, when using the optimal standard deviation σ, we will expect linear convergence. This will be verified in Section 6.2. However, in general we do not know the distance of the single individuals to the Pareto front P_f. As we saw in the previous investigations, a too high σ may lead to an increase of the distance D_t. But on the other hand, a too small σ can lead to arbitrary small convergence speed.

The implemented mutation procedure, which uses the normal distribution, works as follows: First, we determine the ranges $r_i = \max_{\mathbf{x}\in P_0} |x_i|$, $i = 1, \ldots, n$ of the initial population P_0. All r_i are then multiplied by some mutation factor p_m, which according to Subsection 6.2.3 equals $0.5 - 0.7$. In the mutation step itself, we are using the same ranges for all individuals of the current set P''_{t+1} according to Algorithm 4.3 on p. 41. The ranges r_i are adapted every G_m steps, i.e. $r_i = \alpha_m r_i$ with $\alpha_m < 1$, depending on the convergence index $q(t, G_m) < T_m$(cf. (4.13)) with $T_m = \frac{E(\bar{N},\, N,\, p)}{\beta_m \bar{N}}$ with $\beta_m = 3$. σ is to be decreased, since we assume the algorithm to converge. Furthermore, this procedure is based on the observation, that a too large σ would imply, that too many infeasible individuals are generated and hence, $q(t, G_m)$ would be smaller in that case. Typically we choose $G_m = 2$ and $\alpha_m \in [0.9,\, 1)$ depending on N and $\bar{N}$. Especially, α_m is chosen via

$$\alpha_m(N) := 0.95^{\left(\frac{N}{30}\right)^{1/2}}. \tag{6.6}$$

This formula will be justified through numerical experiments, that will be performed in Subsection 6.2.3. There we will determine 0.95 to be optimal for $N = 30$. Since we assume linear convergence and additionally adapt the values at most in every second step, we conclude (6.6). Furthermore, the assumption that for higher N SPEA2 should converge faster proportional to the increase of N is applied. Summarizing, note that this formula is based on numerical experiments. The square root in the exponent was added to slow down the decrease of α_m for higher N, since then the progress is not expected to be proportional to N larger, than for $N = 30$.

Due to the vast number of uncertainties we are faced with, originating from the large number of parameters, when dealing with normal distributed mutation, we implemented a second mutation type, namely the Breeder mutation (cf. [MSV93]), which was introduced in Section 4.1.1. Therefore, we only have to choose the A_i which are chosen as the r_i in the normal distributed mutation, but with a higher mutation factor p_m (usually $p_m \in [1,\, 5]$). Furthermore, we choose $k = 15$ as suggested in [MSV93]. Additionally, we adapt the A_i as in the case of normal distributed mutation via the factor

$$\alpha_m(N) := 0.96^{\left(\frac{N}{30}\right)^{1/2}}. \tag{6.7}$$

The factor 0.96 will be determined in Subsection 6.2.3 too. Anticipatory bear in mind, that the Breeder mutation will turn out to work very well in the framework of SPEA2.

Finally notice, that we do not further decrease the r_i and A_i respectively, if they are below a certain threshold, i.e. 10^{-13}. We would not have any noticeable impact for smaller r_i and A_i, because the double-machine-precision is around 10^{-15}.

6.1.3 Implementation of the NBI

After the previous investigations on the implementation of SPEA2, we will now look into the details of the NBI implementation. As already mentioned above, the *MATLAB*-approach is based on the single objective optimization algorithm *fmincon*, which works

properly for our task. In the following, several practical issues concerning this method are examined in more detail.

Initial Solutions

We have the same possibilities as for SPEA2 to choose the initial guess for the individual minima. Either we can generate it randomly, reuse some data from previous investigations, or we can choose a suitable point from a precomputed table. In all cases, we choose for each direction i the point which is optimal in this certain direction. This should result in a best possible behavior. Note, that even in the case of a random generation it can be useful to produce more points and to choose the most suitable one afterwards.

For the choice of the initial solution for the single optimization problem NBI_β, the following promising method is suggested by [DD98]:
Assume, that $(t^*, \mathbf{d}^*)$ is a solution to NBI_β. Then for a feasible $\tilde{\beta}$ with $\|\beta - \tilde{\beta}\| \leq \varepsilon$ in a certain norm for some small $\varepsilon > 0$, we expect the solution of $\text{NBI}_{\tilde{\beta}}$ to be "as close at $(t^*, \mathbf{d}^*)$ as the distance ε of β and $\tilde{\beta}$". Hence, by using $(t^*, \mathbf{d}^*)$ as an initial guess for $\text{NBI}_{\tilde{\beta}}$ we expect a faster convergence of the procedure, compared to a random generation of an initial solution. Therefore, we start close to an individual minima and continue further by solving always near-by problems. This approach has some similarities to *continuation methods*, which are used to solve nonlinear equations.

On the other hand, we could use the following rule to set the initial solutions for NBI_β. According to the linear combination of the individual minima in the performance space to set up NBI_β, $\mathbf{x}_0$ can be chosen as

$$\mathbf{x}_0 = \sum_{i=1}^{k} w_i \mathbf{x}_i^* \,. \tag{6.8}$$

This setting aligns on the assumption of a linear objective function in at least the dimensions, in which it is optimized. We introduced this setting, since in Section 6.3 this approach will be used, since at outperforms the previously introduced method slightly

Individual Minima

The correct computation of the individual minima $\mathbf{x}_i^*$, $i = 1, \ldots, k$ is very important for a successful application of NBI. The constraints, that is the restriction to feasible points $\mathbf{d} \in X_f$, are implemented in the form of inequality constraints c. They are given by

$$c(\mathbf{d}) := \begin{pmatrix} d_d(\mathbf{d}) \\ d_p(\mathbf{d}) \end{pmatrix} \leq 0 \,. \tag{6.9}$$

Hence, the optimization procedure gets some feedback about the position of the points, i.e. c_1, c_2 decrease for points that approach to the feasible domains D and Y respectively.

If the function *fmincon* fails to produce a correct solution, we have to abort the method, because they are absolutely necessary for a proper functionality of NBI. In the case where only the maximum number of iterations or function evaluations was reached, we compute randomly a new initial guess and double these maximum bounds. Then the optimization procedure is started again. We continue this procedure, until a feasible individual minima is found. Otherwise, if the maximum number of function evaluations exceeds a certain bound, i.e. 4000, then we stop the procedure with an error message.
This is necessary, since incorrect individual minima $\mathbf{x}_i^*$ may lead to a completely misleading trade-off front.

Normal Vector n

Only for $k = 2$, the normal vector is computed exactly. Thereby, $\mathbf{n}$ is given by

$$\mathbf{n} = \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} (\mathbf{x}_1^* - \mathbf{x}_2^*) \,. \tag{6.10}$$

For higher dimensions we use the quasi-normal vector instead, i.e. $\tilde{\mathbf{n}} = -\Phi\mathbf{e}$, as was introduced in [DD98] (as outlined in Subsection 5.2.2) due to the increasing complexity of this issue for higher k. The influence on the obtained approximated points is negligible, as stated before.

Computation of β

We mentioned above, that we want to use the computed solutions $(t_\beta, \mathbf{d}_\beta)$ as a starting value for a close-by problem. For this we are using the following procedure to choose the vector β:

Algorithm 6.1 (Determination of next β).

Input: n_β *number of intervals in each direction, i.e.* $n_\beta + 1$ *is the number of points*
β *with* $\beta_i \in \{0, \ldots, n_\beta\}$
Output: β^{new} *new* β
b_{close} *indicates, if the next problem is a close-by problem or not*

determine the highest index i *for which* $\beta_i \neq 0$ *holds;*
$\beta^{new} = \beta$*;*
$b_{close} =$ *true;*
if $i \neq 1 \wedge \neg(\beta_{i-1}^{new} = n_\beta - 1 \wedge i = 2)$
  $\beta_k^{new} = \beta_i^{new} - 1$*;*
  $\beta_{i-1}^{new} = \beta_{i-1}^{new} + 1$*;*
  if $\beta_{i-1}^{new} = n_\beta$
    $\beta^{new} = \mathbf{0}$*;*
    $\beta_{i-2}^{new} = 1$*;*
    $\beta_k^{new} = n_\beta - 1$*;*

$b_{close} = false;$
else
for $j = i$ *to* $k - 1$
$\beta_j^{new} = 0;$
end for
end if
else
Error: We cannot compute further β*!;*
end if

Every time, when the inner *if*-condition is fulfilled, we generate a new β that differs too much from its predecessor. Hence, the previous result should not be used as an initial guess. To remedy this, we proceed as follows.
We store the solution of the first subproblem, i.e. with $\beta = \frac{1}{n_\beta}(0, \ldots, 0, 1, n_\beta - 1)$. Then, when b_{close} is *false* for the first time, with $\beta^{\text{new}} = \frac{1}{n_\beta}(0, \ldots, 0, 1, 0, n_\beta - 1)$, we use the stored value as an initial guess. Afterwards, we store the solution of NBI_β and use it the next time, when $b_{\text{close}} = false$. Otherwise, we always use the result of the previous computation as the initial value.

The Subproblems NBI_β

Although it seems, that the implementation of the subproblems NBI_β is easy, there are two points to mention.

The first important fact is, that the *fmincon* sometimes fails to produce a solution to NBI_β. This can happen if the maximum number of function evaluations or iterations is reached, or if it simply fails to produce an optimum. In this case, we will discard the produced solution. We do not extend the range of the maximum number to rerun the procedure as would be done when computing the individual minima $\mathbf{x}^{*,i}$, because it is quite likely to fail again. We proceed in this way, since our method should not be too time consuming.

The next noticeable remark is the one mentioned already in Subsection 5.2.2, that we could exchange the equality constraints (5.10) by the inequality constraints (5.14), as suggested in [Ste05]. However, both constraints were implemented. Anticipatory, they did not exhibit an obviously different behavior for the simple test problems. In the framework of RSMs for RF-circuit blocks (Section 6.3) both possibilities do not work satisfactory, what might come from the behavior of the RSM. But nevertheless, the equality constraints behaved slightly better than the inequalities.

6.2 Application to a Simple Example

In the previous section, we implemented the two methods. Now, in this section, we want to asses these methods by means of a simple example. First, we will investigate the con-

vergence of SPEA2. Therefore, we will exemplarily verify the results obtained in Subsection 4.5. Afterwards, we will examine some specific parameters for SPEA2. Furthermore, the two mutation methods of SPEA2 will be compared and finally, we will compare the two implemented methods SPEA2 and NBI.

6.2.1 Test Environment

Before we are able to asses the methods we have to state our test problem. It was chosen, because it is a standard example used for testing EAs in literature (e.g. [HNG94, SD94]). Throughout this section, we assume the following multiobjective optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^2} \mathbf{y}(\mathbf{x}) := \left(\|\mathbf{x}\|^2\,,\, \|\mathbf{x}-\mathbf{z}\|^2\right)^T \tag{6.11}$$

subject to

$$\begin{aligned} x &\in D := [-5,\, 5]^2\,, \\ y &\in Y := [0,\, 10]^2\,, \end{aligned}$$

with $\mathbf{z} = (2,\, 0)^T$. In Figure 6.1 the function $\mathbf{y}$, defined in (6.11), is depicted. It can be easily seen, that for our choice of $\mathbf{z}$, the Pareto front is given by

$$P_f = \{\mathbf{p} \in \mathbb{R}^n \,|\, \exists x \in [0,\, 2] \,:\, \mathbf{p} = (x^2,\, (x-2)^2)^T\}\,, \tag{6.12}$$

with the corresponding design parameter set X_P

$$X_P := \mathbf{y}^{-1}(P_f) = \{(t,\, 0)^T \,|\, t \in [0,\, 2]\}\,. \tag{6.13}$$

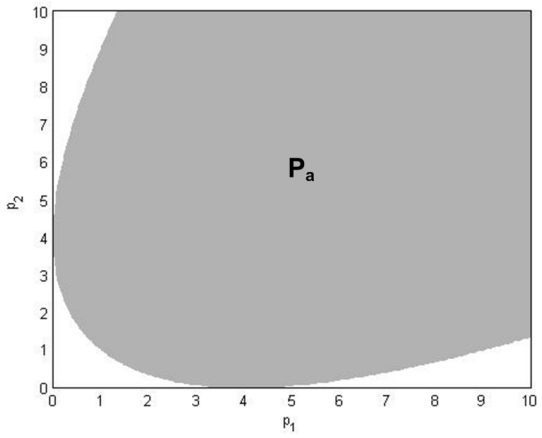


Figure 6.1: Function $\mathbf{y}$ used to assess the methods. The gray area designates the attainable domain P_a.

For the evaluation of the quality of a solution set, the l_2-distance measure defined by

equation (4.16b) (on p. 48) is used. Note, that for the simple test function (6.11) the optimal point corresponding to the distance function (4.16d) can be calculated analytically. First, we substitute the test function into the distance function and obtain

$$\begin{aligned} d(\mathbf{p}_i, P) &= \inf_{\mathbf{p}\in P} \|\mathbf{p}_i - \mathbf{p}\| = \min_{x\in[0,\,2]} \|\mathbf{p}_i - \left(x^2,\, (x-2)^2\right)^T\| \\ &= \min_{\mathbf{x}\in[0,\,2]} f(x,\, \mathbf{p}_i) := \sqrt{(x^2 - \mathbf{p}_{i,1})^2 + ((x-2)^2 - \mathbf{p}_{i,2})^2} \end{aligned} \tag{6.14}$$

Hence, we have to solve

$$0 = \frac{\partial f}{\partial x} = \frac{4x(x^2 - \mathbf{p}_{i,1}) + 4(x-2)((x-2)^2 - z_2)}{2\sqrt{(x^2 - \mathbf{p}_{i,1})^2 + ((x-2)^2 - \mathbf{p}_{i,2})^2}}\,. \tag{6.15}$$

After solving this equation, we find, that there is only one real solution given by

$$\begin{aligned} x = 1 &+ \frac{\mathbf{p}_{i,1} + \mathbf{p}_{i,2} - 6}{6^{1/3}\left(9(\mathbf{p}_{i,1} - \mathbf{p}_{i,2}) + \sqrt{3}\sqrt{27(\mathbf{p}_{i,1} - \mathbf{p}_{i,2})^2 - 2(\mathbf{p}_{i,1} + \mathbf{p}_{i,2} - 6)^3}\right)^{1/3}} + \\ &\frac{\left(9(\mathbf{p}_{i,1} - \mathbf{p}_{i,2}) + \sqrt{3}\sqrt{27(\mathbf{p}_{i,1} - \mathbf{p}_{i,2})^2 - 2(\mathbf{p}_{i,1} + \mathbf{p}_{i,2} - 6)^3}\right)^{1/3}}{6^{2/3}}\,. \end{aligned} \tag{6.16}$$

We will exploit (6.16) to compute the distance measure $\|P\|_2$ of a set of approximate vectors.

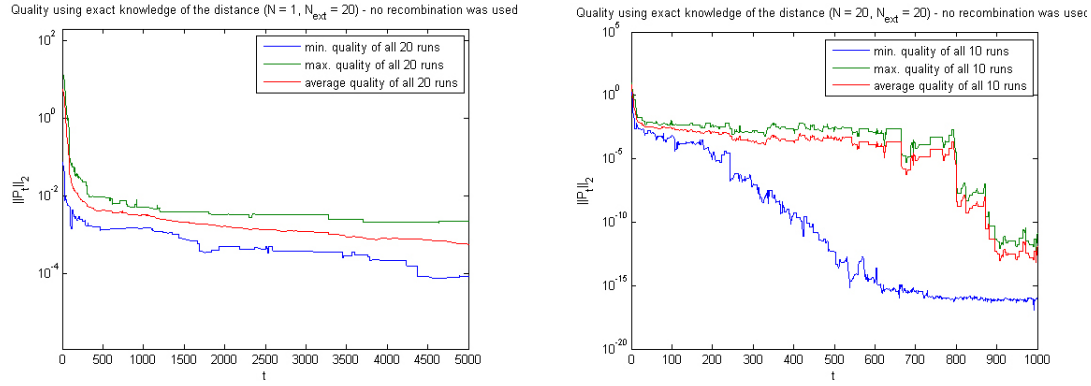
Moreover bear for the following subsections in mind, that we chose, except for the first part, $N = \bar{N}$, since this setting showed the best results and furthermore, it is suggested in [ZLT01]. N was always chosen as high as possible to obtain the results still in reasonable time. Additionally note the important remark, that SPEA2 is a probabilistic method. Therefore, a single run has no evidence and hence, several simulations have to executed. For each test case, we chose the highest possible (in terms of computation time) value according to N.

6.2.2 SPEA2 with Exact Knowledge of Distance

We showed in Subsection 4.5, that under certain assumptions the Strength Pareto Evolutionary Algorithm 2 with normal distributed mutation is expected to converge linearly. For this special case, we computed expected convergence factors, which are given by (4.26) for $k = n = 2$. Remember, that this rates were computed for $N = 1$ and $\bar{N} \in \mathbb{N}$ under the assumption $\mathbf{p} = I$. Furthermore, we listed the lower bounds of the expected progresses in Table 4.1, for $k = n = 2$ and several archive sizes $\bar{N}$.

For the reasons explained above, we chose 20 simulation runs. The minimal(blue), maximal(green) and the average(red) distance measure of the simulation runs are depicted in Figure 6.2. We chose $N = 1$, $\bar{N} = 20$ and the maximum number of iterations $T = 5000$ for Figure 6.2(a). For the right picture $N = \bar{N} = 20$ and $T = 1000$ iterations were chosen. Furthermore, we used the exact distance of the single individuals. That is, we computed

the distance of each point to the closest one in X_P, corresponding to (6.16). The distance was computed in the design space, because there the mutation is executed. Furthermore, we multiplied the distances by 0.6 aligning on the results in (4.27) and (4.35).



(a) (b)

Figure 6.2: For 20 simulations runs the minimal(blue), maximal(green) and the average(red) distance measure of the test example (6.11) with exact knowledge of the distance are depicted for $N = 1$, $\bar{N} = 20$ and $T = 5000$ iterations (left). For the picture on the right hand side 10 runs with $N = \bar{N} = 20$ and $T = 1000$ were executed. No recombination was applied.

Summarizing the results of Figure 6.2, we can see the convergence, but especially for the left picture with $N = 1$, the improvement factor is very low. The average distance measures are $\|\bar{P}_0\|_2 = 5.456$ for the initial population and $\|\bar{P}_{5000}\|_2 = 5.618 \cdot 10^{-4}$ for the generation $t = 5000$. Therefore, we obtain an average factor for the expected progress $\bar{\zeta}_2 = 1.834 \cdot 10^{-3}$. This value aligns with the value derived in Subsection 4.5.8, where in Table 4.1 we derived $\zeta_2 = 1.648 \cdot 10^{-3}$ for $\bar{N} = 20$. When comparing these results bear in mind that the results in Section 4.5 were computed with $\mathbf{p} = I$. But still, the considered problem (6.11) does not differ that much from the identity in terms of distribution, as explained in Subsection 4.5.3.

For the second graph, $N = \bar{N} = 20$ was chosen and no recombination was applied. We introduce this picture to show, that for higher N and fixed $\bar{N}$ we obtain faster convergence in terms of iterations, but the number of iterations does not decrease proportional to N, due to possible interferences of the newly produced individuals. In Figure 6.2 this can be seen, since $\bar{\zeta}_{2,1} = 1.834 \cdot 10^{-3}$ for $N = 1$ (left picture) and $\bar{\zeta}_{2,20} = 2.879 \cdot 10^{-2}$ for $N = 20$ (right picture), while $\bar{\zeta}_{2,20} = 1 - (1 - \bar{\zeta}_{2,1})^{20} = 3.606 \cdot 10^{-2}$ is expected when neglecting the influence of interference (see Section 4.5.

Summarizing the results, we verified the theoretical results on convergence. Furthermore we conclude that doubling of N for fixed $\bar{N}$ leads to a convergence that is only almost twice as fast. Additionally note, that we expect less interferences the closer we are at the Pareto front, because then the mutation is limited to smaller domains and only recombination can cause interfering individuals.

6.2.3 A Closer Look at the Mutation Parameters of SPEA2

In the previous subsection, we verified the results derived in Section 4.5. Since the distance of the individuals to the Pareto front is not available in general, the previous results cannot be applied. In Subsection 6.1.2 we mentioned, that we implement two different mutation operators, namely the normally distributed mutation and the Breeder mutation. In the following we compare the two and we will investigate on the parameters needed for those operators.

For this sake, the distance (4.16b), i.e. $\|\bar{P}_T\|_2$, of the archive $\bar{P}_T$ for $T = 300$ to P_f was computed. The simulations were executed for both mutation approaches for different settings of α_m and p_m according to Subsection 6.1.2. We chose $N = \bar{N} = 30$ since this is an acceptable value in terms of accuracy and time consumption, especially when looking forward to the design space exploration.
For each parameter setting 10 computation runs were performed, due to the probabilistic character of the method. The average values of these computations for the different settings are depicted in Table 6.2. For all computations the same initial population P_0 was used. It was randomly generated before.

Breeder mutation						
	p_m					
α_m	1	2	3	4	5	7
0.90	0.033936	0.000035	0.000026	0.033379	0.034304	0.063672
0.92	0.000028	0.000038	0.036501	0.000057	0.000033	0.000065
0.94	0.034760	0.000056	0.000073	0.020638	0.038576	0.000092
0.96	0.000135	0.038686	0.000286	0.000307	0.000694	0.000736
0.98	0.035397	0.056777	0.035591	0.001078	0.088688	0.001147
Normally distributed mutation						
	p_m					
α_m	0.1	0.3	0.5	0.7	0.9	1.1
0.65	0.000059	0.000070	0.000045	0.000039	0.000042	0.000050
0.75	0.000469	0.000056	0.000037	0.000046	0.000016	0.000017
0.85	0.000979	0.000032	0.020621	0.000033	0.000018	0.000051
0.90	0.000227	0.000043	0.000017	0.000021	0.000040	0.031107
0.95	0.000322	0.000019	0.000035	0.000017	0.000068	0.000021

Table 6.2: Comparison of $\|\bar{P}_T\|_2$ after $T = 300$ iterations for $N = \bar{N} = 30$ for different values of α_m and p_m for Breeder mutation and normally distributed mutation. 10 runs were performed for each setting.

When having a closer look at Table 6.2 we find that Breeder mutation seems to have a larger variation, since for slightly different settings we obtain completely different distances. For verification compare the values of ($\alpha_m = 0.94$, $p_m = 4$) and ($\alpha_m = 0.94$, $p_m = 5$). There $\|\bar{P}_{300}\|$ differs by more than two decades. Of course, this can be explained by means of the low number of simulation runs, i.e. 10. But still, by simply counting these big variations, we get the impression that the Breeder mutation varies more than the normally distributed mutation, at least in the considered ranges of the parameters.

When choosing the parameters α_m and p_m we have to bear in mind, that too low values may

lead to no convergence, while contrary too high values can lead to very slow convergence or even an stagnation. Nevertheless, it is usually more dangerous to choose too small values than too high ones. Taking these considerations into account, we suggest for $N = \bar{N} = 30$ the following typical values:

mutation method	α_m	p_m
Breeder	0.96	3
Normal distribution	0.95	0.7

Table 6.3: Typical values for Breeder and normally distributed mutation for $N = \bar{N} = 30$.

Since, we expect a faster convergence in terms of iterations for higher N, $\bar{N}$, we have to decrease the factor α_m. Contrary, for less individuals, the proportionality factor has to be increased. This and the expected linear convergence leads to the rules presented in Subsection 6.1.2.

6.2.4 Stopping Criterion of SPEA2

In this paragraph, we will have a closer look on the stopping criterion as introduced in the subsections 4.4.5 and 6.1.2. There, we saw that the two parameters T_C and G have to be chosen properly in order to reach some desired accuracy. Basically, for higher G we would expect a higher convergence index q, because we expect a certain progress in each iteration step and hence, the progress should increase with each step. But contrary bear in mind, that for a small distances of the individuals to P_f compared to the distances to the other elements of $\bar{P}_t$, we expect a significant decrease of this convergence indicator due to recombination. Therefore, we might expect the existence of a lower bound for the achievable accuracy when using this stopping criterion. It is obvious, that this bound depends on $\bar{N}$.

Based on these considerations, we will determine suitable values for the parameters T_C and G. This is done by comparing the accuracy $\|\bar{P}_t\|_2$ for different settings of these parameters. Table 6.4 compares the distance $\|\bar{P}_t\|_2$ after the iteration stopped for $G \in \{4, 8, 12, 16\}$ and $T_C \in \{\frac{\beta}{3}, \frac{\beta}{5}, \frac{\beta}{7}, \frac{\beta}{9}, 0, \frac{-\beta}{9}, \frac{-\beta}{5}\}$. The results were computed for Breeder mutation (upper row) and normally distributed mutation (lower row). Additionally, the number of iterations is shown on right-hand side of the corresponding distance value. 20 simulation runs were performed for each setting. The listed values are the averages of the obtained results.
We chose $N = \bar{N} = 40$, because it is a typical setting in 2D. Moreover, it is still possible to obtain results in acceptable time. The mutation parameters α_m and p_m were chosen according to Table 6.3 together with Subsection 6.1.2.

From Table 6.4 we can see, that for the used settings, the Breeder mutation stops much earlier than SPEA2 with normally distributed mutation. Nevertheless, the accuracy with Breeder mutation is on average higher than with normal distribution. But this can be due to experimental variation, because we only performed 20 runs. But still, the iteration

T_C	G							
	4		8		12		16	
$\beta/3$	0.00124	43	0.02793	49	0.02806	65	0.01428	65
	0.03065	44	0.00573	53	0.02988	78	0.04997	107
$\beta/5$	0.01712	49	0.01473	61	0.01451	92	0.00137	87
	0.05049	53	0.01612	73	0.03638	117	0.07148	140
$\beta/7$	0.04058	53	0.01757	70	0.00143	71	0.04265	86
	0.01294	52	0.03260	79	0.02859	116	0.03681	138
0	0.01551	53	0.05460	64	0.01538	80	0.00876	74
	0.01545	50	0.00378	78	0.01624	135	0.05905	178
$-\beta/9$	0.04522	111	0.02928	103	0.02447	166	0.05023	114
	0.11604	373	0.11684	313	0.06996	345	0.07319	340
$-\beta/5$	0.04239	146	0.04284	125	0.02759	129	0.07532	153
	0.10503	329	0.07464	364	0.06614	321	0.09306	319

Table 6.4: Comparison of $\|\bar{P}_t\|_2$ after the iteration stopped by using the convergence index. The tests were performed with $N = \bar{N} = 40$ for different values of α_m and p_m for Breeder mutation(upper row) and normally distributed mutation(lower row). 20 runs were performed for each setting. The left values indicate the average accuracy when the algorithm stopped after on average t iterations(right value).

count seems to be lower. This might be due to the discrete character of Breeder mutation, because a typical performance of SPEA2 with Breeder mutation is, that it stagnates for several iterations and then it performs a big step towards the Pareto front. Therefore, in the regions of stagnation the criterion is quite easily fulfilled.

Additionally note, that the doubts we mentioned above were verified. Due to the used recombination, the progress in terms of the coverage function f_C (4.12) and convergence index q (4.13) is not a practicable measure, since we do not obtain accurate results at all. No matter which parameter setting is used. It turns out, that if no recombination would be used, the needed iterations would increase decisively. But since SPEA2 with recombination outperforms the approach without recombination, as will be shown in Subsection 6.2.5, we will not further investigate on it.

Summarizing the results we obtain the fact, that for SPEA2 with recombination, the stopping criterion based on the convergence index q is certainly not the optimal stopping criterion. If we apply this criterion, distances of smaller 10^{-2} cannot be expected. Therefore, it seems that we achieve best results for $T_C = 0$ and $G = 8$ in the case of $N = \bar{N} = 30$.

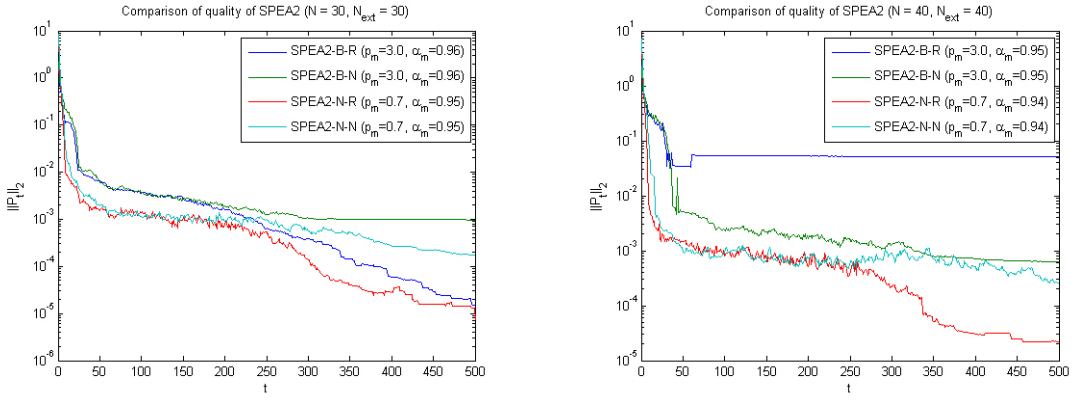
Finally bear in mind, that this stopping criterion is not suitable and therefore, we will use the maximum iteration count as a stopping criterion.

6.2.5 Normal Application of SPEA2

After the previous investigations, where the theoretical calculations were verified, the parameters for mutation were examined and the alternative stopping criterion was treated, we will now investigate on the convergence behavior of SPEA2 in the typical case.

For this purpose, we will consider the distance function (4.16b) for the generations $t = \{0, \ldots, T\}$ with $T = 500$. In Figure 6.3(a) the results are shown. The computations were

executed for four different settings, which are Breeder mutation(denoted by SPEA2-B) and normally distributed mutation(SPEA2-N) together with(-R) or without(-N) recombination. For the simulations the typical setting $N = \bar{N} = 30$ together with the mutation settings as depicted in Table 6.3 were used.
For each configuration 20 simulations were performed and the average of the obtained distances was plotted.



(a) (b)

Figure 6.3: For 20 simulations runs the distance measure of the test example (6.11) with exact knowledge of the distance is depicted for $N = \bar{N} = 30$ and $T = 500$ iterations (left). The picture on the right-hand side was computed with $N = \bar{N} = 40$ through 10 simulation runs. The results show SPEA2 with Breeder mutation with/without recombination (SPEA2-B-R/SPEA2-B-R) and additionally SPEA2 with normal distribution and with/without recombination (SPEA2-N-R/SPEA2-N-R) is depicted.

For comparison, Figure 6.3(b) shows the distance for $N = \bar{N} = 40$. Therefore, 10 simulation runs were performed. Furthermore, the parameters α_m were chosen according to (6.6) and (6.7). In this figure, we can clearly see, the nice behavior of the approach using the normally distributed approach. Contrary, the Breeder mutated SPEA2 shows bad performance. This shows the probabilistic character of this approach. The iteration in this case got caught in a bad setting and then it takes a long time to get out of this situation.

The example depicted in Figure 6.3(a) was set up to investigate on two points. First, we wanted to know more about the impact of recombination and second, the performance difference between Breeder and normally distributed mutation should be pointed out in more detail.

The algorithm works significantly better with recombination, than without. One reason might be, that especially for higher N and $\bar{N}$ respectively, the archive $\bar{P}_t$ is then filled with only nondominated individuals much faster. Moreover note, that an external set $\bar{P}_t$ that does not contain dominated points is the base for a fast converging SPEA2 approach.
Another explanation for this reason could lie in the nature of our problem (6.11). We chose a parameter domain D, which is symmetric around the optimal x_2 parameter. Therefore, half of the points are expected to lie on each side. In the following, with probability 1/2 points with different x_2-signs are chosen. In those cases the linear combination leads to an

improvement in at least x_2-direction.
We made this observation for all performed numerical tests on this example and furthermore for RSMs.

Additionally, we can see that in this case the normally distributed mutation performs approximately as well as the Breeder mutation. In general, throughout all the tests none of the mutation approaches was able to outperform the other one. This is, because Breeder mutation converges quite often very fast. But on the other hand it seems to be more unstable, i.e. sometimes it stagnates for several 100 iterations. Hence, it seems that both implemented methods work equally well, although Figure 6.3(b) states a counter-example.

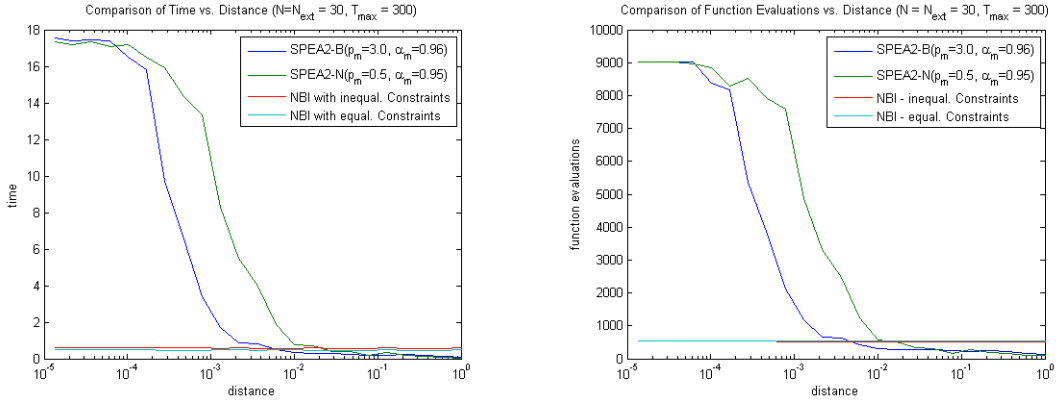
6.2.6 Comparison of SPEA2 to NBI

In the previous subsections, we exclusively dealt with SPEA2. This was due to the large number of parameters that can be adjusted in this approach. Moreover, it is a typical iterative approach, where easy performance measures are possible.

Now, we will compare SPEA2 and NBI by means of (6.11). Therefore, we will investigate on the time or objective function evaluations it takes, to reach a certain accuracy. The results on these considerations are depicted in Figure 6.4. The comparison of time to distance is depicted in Figure 6.4(a), while the number of function evaluations versus distance is shown on the right-hand side of this figure. SPEA2 was applied with the Breeder-mutation(denoted by SPEA2-B) and the normally distributed mutation(SPEA2-N). For NBI, the two different constraints for the NBI$_\beta$-subproblems are considered.

For this setting, we ran 6 experiments for each setting and calculated the obtained average. Furthermore, the same initial set P_0 with $|P_0| = N$ was used for both methods and all iterations, whereby $N = \bar{N} = 30$ was chosen. Additionally, for SPEA2 $p_m = 3$ in the case of Breeder-mutation and $p_m = 0.5$ for the normally distributed mutation was used. The factors α_m were set according to Subsection 6.1.2.
To guide the accuracy of the NBI approach, we tried to prescribe the thresholds for the error constraints. But, as can be seen, the accuracy of solutions of such simple examples by using the *fmincon*-function is always at least 10^{-7}, no matter how high we set this threshold value. That is, the algorithm converges very fast for this easy example.

As explained before, the NBI method has no problems at all with simple examples and hence, we do not recognize any difference between the two different cases. To be able to make a valuable assessment about the differences between equality and inequality constraints more complex problems have to be considered.
The other interesting result of this investigation is, that SPEA2 works, at least for the used setting, for both Breeder-mutation and for normally distributed mutation equally well.
Additionally note, that the computation time and the number of function evaluations are directly related to each other. For SPEA2, the number of function evaluations per iteration is N. Therefore, we have $N = 30$ evaluations per iteration and hence, due to $T = 300$



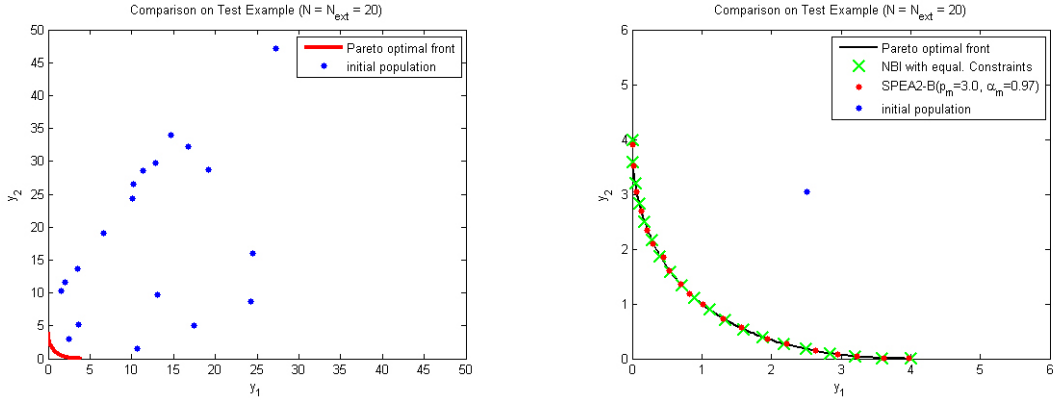
(a) (b)

Figure 6.4: For each 6 runs were performed and the average of the corresponding values was computed.

the maximal number of evaluations is limited to 9000 in that case. Note, that for an suitable choice of $r_i\,(\sigma_i)$ and A_i we expect (linear) convergence. But, to reach distances $\|\bar{P}_t\|_2 < 10^{-6}$ we will usually need more than 300 iterations. This bound was introduced to limit the computation time to feasible values.

Additionally, Figure 6.5 shows a typical initial population on the left-hand side, containing $N = 20$ individuals. Moreover, Figure 6.5(b) depicts the reconstructions of the front with NBI(green crosses) and SPEA2(red dots). For both methods $N = \bar{N} = 20$ points were computed to avoid overcrowded pictures. For SPEA2, the Breeder mutation with standard parameters was used. The depicted front is given by $\bar{P}_{150}$, i.e. the archive after 150 iterations.

To compare the results, the black line in the right picture denotes the Pareto front. After a detailed look, we recognize a small difference in the approximation of SPEA2. But in general, the results are accurate.
Additionally recognize the uniform distribution of both approximation sets, which was one main reason for the choice of both methods.



(a) (b)

Figure 6.5: The initial population for the computation including the Pareto front(left) and the front determined by SPEA2 with Breeder mutation(red dots) for $T = 150$ iterations and $N = \bar{N} = 20$ and NBI(green crosses) (right) for $n_\beta = 19$ are shown.

6.3 Application to a Performance Model of an LNA

In the previous chapters and sections we discussed mainly the two methods SPEA2 and NBI. We treated their analytical aspects, their implementation and in the previous section we tested them by means of a simple example.
In this section we will close the circle and come back to the original problem, namely the multiobjective design space exploration (see Problem 1.2 on p. 3) of compact RF-building block models, which are in our case response-surface models[5] as outlined in Chapter 2.

6.3.1 Design Space Exploration of a Low-Noise Amplifier

For the following investigations we will limit ourselves to a model for a source degenerated LNA as depicted in Figure 2.5(a) on p. 14.
The model parameters are the transistor width W, the inductances L_s and L_m as well as the load resistance R_l. They were explained in Section 2.3. Therein, typical values of these parameters for our example are stated. Other possible input parameters were set to fixed values, e.g. $f = 5\,GHz$. Additionally note, that the considered ranges of the parameters were transformed to $[-1, 1]$.
The performance variables were introduced in Section 2.3 too. In (6.17) a typical setting

[5]Note, that different approaches for design space exploration are available in literature(cf. [SG03, Ste05, PG02]). [SG03] computes the Pareto front by using circuit simulations for $\mathbf{p}$ together with EAs. [Ste05] uses NBI among other deterministic approaches. In [PG02], SPEA2 is used to derive local Pareto optimal points of digital system-on-a-chip(SoC) circuits.

for the constraints on the performances is depicted.

$$\begin{aligned} P &\quad \text{as small as possible} \\ \Gamma_a &< -10\,dB \\ G_A &> 10\,dB \\ A_v &> 18\,dB \\ IIP2 &> 0\,dBm \\ IIP3 &> -10\,dBm \\ NF &< 2dB \end{aligned} \tag{6.17}$$

In the following we will perform a typical procedure for design space exploration. Therefore, we will first examine some trade-off fronts with typical, but not too severe, constraints on the performances. We will accomplish this by prescribing the restrictions given in (6.17). Then the constraints will be tightened more and more, in order to obtain a desired performance.

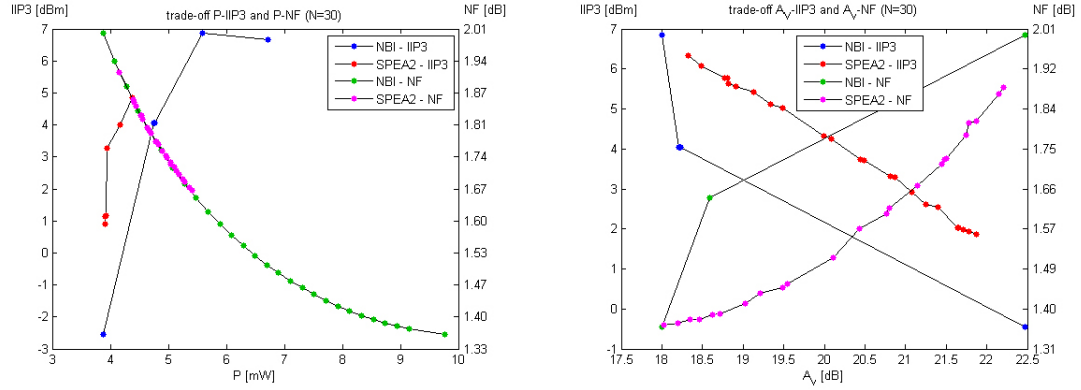
We compare the results of NBI with SPEA2. Since according to the investigations of the previous section, normally distributed mutation seems to be more stable, it will be used below. Furthermore, the equality constraints (5.10) are used, since they showed slightly better performance than the inequality constraints (5.14). Moreover, we used the linear combination of the individual minima, given by (6.8), as the initial guess for the NBI$_\beta$ subproblems, since it outperformed the continuation method-like procedure (see Subsection 6.1.3) in the numerical tests. For both methods $n_\beta + 1 = \bar{N} = N = 30$ points were attempted to be computed. For SPEA2, we chose $T = 800$ generations as a stopping criterion.

Design Space Exploration - Step 1

In Figure (6.6) the computed trade-off fronts are depicted. Thereby, Figure 6.6(a) shows the trade-offs $\min P$ versus $\max IIP3$ and $\min P$ versus $\min NF$. Moreover, Figure 6.6(b) depicts the computed front between $\max A_v$ and $\max IIP3$ and $\max A_v$ is compared to $\min NF$. In the following we will omit the preceding terms min and max, since it is clear from the context, which optimization direction is used. In the following we will write $P - IIP3$ for instance.

Considering the graphs we recognize several facts:

- With NBI we obtain a larger range of the front compared to the results of SPEA2, as can be especially seen in Figure6.6(a). Therefore, we could introduce a certain iteration number T_r, from which on, we neglect recombination. This can be advantageous, since recombination does not produce boundary elements, what is desired in the trade-off $P - NF$ for instance, to extend the range of the approximations. We chose $T_r = 500$.



(a) (b)

Figure 6.6: The trade-off between P and $IIP3$ as well as P and NF is shown in the left picture, and on the right-hand side, the trade-off between A_v and $IIP3$ as well as A_v and NF is depicted. For all computations NBI (with equality constraints and initial solution for NBI_β according to (6.8)) is compared to SPEA2 (normally distributed, $T = 800$). $N = 30$ points were computed. The constraints of the problem are (6.17).

To extend the front, we could attempt to change the restrictions on the performances. Considering the $P - NF$ front in Figure 6.6(b) for instance, we could tighten the restrictions, i.e. $P > 5.5$ and $NF < 1.65$ to obtain the trade-off in the additional areas.

- NBI fails quite often to produce feasible points for the NBI_β subproblems, or sometimes the obtained solutions are not global optimums, as can be seen in $P - IIP3$. In the latter case, the SOP-solution algorithm gets trapped in a stationary point, which is not optimal.

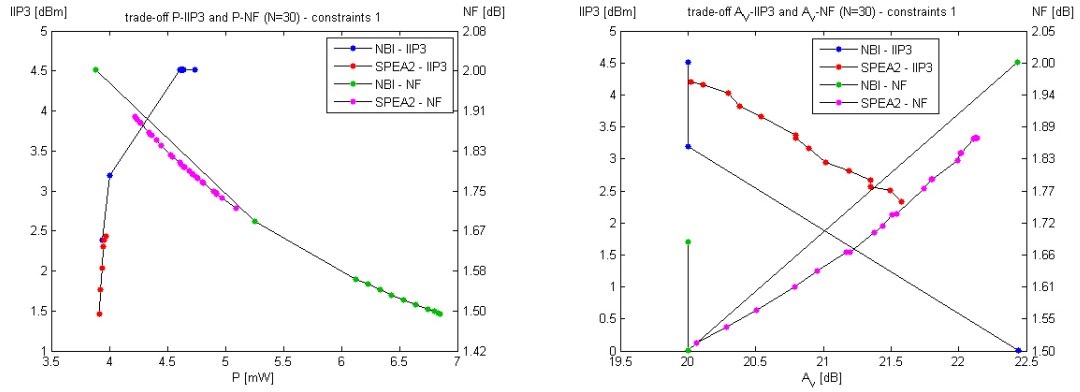
- The most surprising observation is the time consumption of the two methods. While SPEA2 needs $60-84$ seconds on a normal laptop to return a result, NBI took $150-500$ seconds. This is due to the complexity of the optimization algorithm. Although, SPEA2 performed 24000 function evaluations and NBI computed $\mathbf{p}$ $4800 - 14000$ times the objective function value, SPEA2 was much faster.

Design Space Exploration - Step 2

Coming back to the procedure of exploring the design space of the considered circuit we find that the ranges of the performance figures are still quite large. Therefore, we have still degrees of freedom, which can be used to apply additional constraints to the problem, in order to obtain a desired performance. Additionally to the constraints (6.17), in the next step, the following has to hold:

$$\begin{aligned} A_v &> 20dB \\ IIP3 &> 0dBm\,. \end{aligned} \tag{6.18}$$

In Figure 6.7 the same trade-off fronts as in Figure 6.6 are depicted. Contrary, the constraints (6.17) and (6.18) are investigated. Furthermore, for SPEA2 we increased the number of iterations to $T = 1200$, since the more severe the constraints become, the more careful and accurate we have to be in order to get correct and especially feasible results. Moreover, we reused the results obtained in the previous computation step, as outlined in Section 6.1. To guarantee a fair comparison, the previous results of each method served as an input for the same approach again. In Figure 6.7, we recognize again, that SPEA2 does



(a) (b)

Figure 6.7: The trade-off between P and $IIP3$ as well as P and NF is shown in the left picture, and on the right-hand side, the trade-off between A_v and $IIP3$ as well as A_v and NF is depicted. For all computations NBI (with equality constraints and initial solution for NBI_β according to (6.8)) is compared to SPEA2 (normally distributed, $T = 1200$). $N = 30$ points were computed. The constraints of the problem are (6.17) and (6.18).

not cover the whole front, at least after 1200 iterations. But contrary the quite interesting observation of $P - NF$ is, that NBI fails to produce solutions in areas where SPEA2 finds approximations and the other way around. The explanation for this incident is, that the design vectors used for both methods differ, even in the points that lie close together. This shows exemplarily the difficulty we are dealing with, namely that the inverse map of $\mathbf{p}$ is not injective.

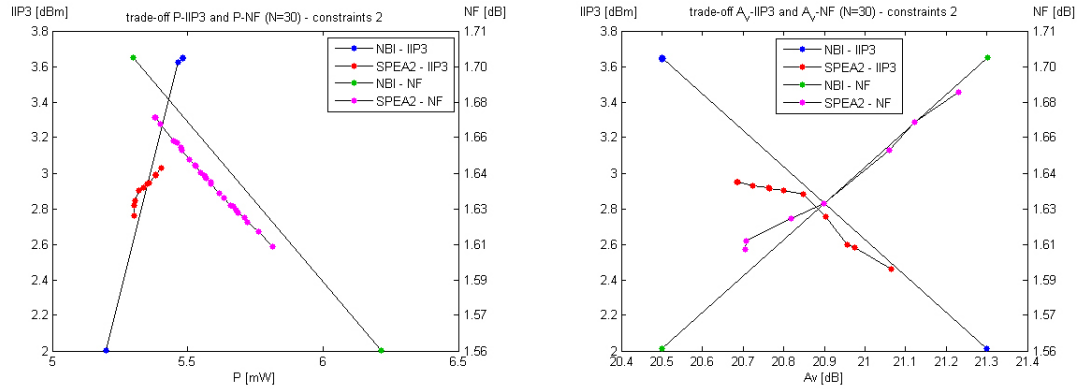
Design Space Exploration - Step 3

From the trade-off fronts for the constraints (6.17) and (6.18), we see clearly, that there is still space for further restrictions. Hence, we tighten the constraints further, namely we prescribe additionally

$$\begin{aligned} A_v &> 20.5dB \\ IIP3 &> 2dBm \\ NF &< 1.7dB\,. \end{aligned} \tag{6.19}$$

In Figure 6.8 the conditions (6.19) were additionally to (6.17) and (6.17) applied. Therefore, the same trade-offs as in the previous pictures were investigated. Due to the decreasing

scales we performed $T = 1600$ iterations for SPEA2.



(a) (b)

Figure 6.8: The trade-off between P and $IIP3$ as well as P and NF is shown in the left picture, and on the right-hand side, the trade-off between A_v and $IIP3$ as well as A_v and NF is depicted. For all computations NBI (with equality constraints and initial solution for NBI_β according to (6.8)) are compared to SPEA2 (normally distributed, $T = 1600$). $N = 30$ points were computed. The constraints of the problem are (6.17), (6.18) and (6.19).

Having a closer look at Figure 6.8 we see, that there is still enough freedom to restrict the performances further. But we will not execute this. Instead we will choose one point. For instance, let us consider

$$\mathbf{d} = \begin{pmatrix} W \\ L_m \\ L_s \\ R_l \end{pmatrix} = \begin{pmatrix} 153.47\,\mu m \\ 3.70\,nH \\ 152.70\,pH \\ 8.07\,k\Omega \end{pmatrix} \quad \text{with} \quad \mathbf{p}(\mathbf{d}) = \begin{pmatrix} P \\ \Gamma_a \\ GA \\ A_v \\ IIP2 \\ IIP3 \\ NF \end{pmatrix} = \begin{pmatrix} 5.40mW \\ -10.42\,dB \\ 13.52\,dB \\ 20.90\,dB \\ 16.29\,dBm \\ 2.76\,dBm \\ 1.69\,dB \end{pmatrix}. \tag{6.20}$$

It is interesting now, to get an impression of the trade-off between different performances around this point. Therefore, w finally come back to the definitions of trade-off made in Section 3.4. We compute exemplarily the partial trade-offs $\Lambda_{A_v,\mathrm{j}}$, as introduced in Definition 3.21, along the $A_v - IIP3$ front. They are given by

$$\begin{pmatrix} \Lambda_{A_v,\mathrm{P}} \\ \Lambda_{A_v,\Gamma_a} \\ \Lambda_{A_v,\mathrm{GA}} \\ \Lambda_{A_v,A_v} \\ \Lambda_{A_v,\mathrm{IIP2}} \\ \Lambda_{A_v,\mathrm{IIP3}} \\ \Lambda_{A_v,\mathrm{NF}} \end{pmatrix}(\mathbf{d}) = \begin{pmatrix} -0.76 \\ -5.92 \\ -4.49 \\ 1.0 \\ 0.12 \\ -2.35 \\ 0.08 \end{pmatrix}. \tag{6.21}$$

Hence we see, that Γ_a, GA and $IIP3$ are most sensitive to improvements in A_v. Nevertheless, we have to bear in mind, that this rates are absolute values and therefore, the magnitudes of these values cannot be compared directly to each other. Furthermore bear in mind, that since most performance figures are in logarithmic scale, the trade-off values depend on the treated performance value. To avoid these problems, the performances should be considered in the original scales and moreover, they should be scaled there to guarantee a fair comparison of the trade-off values.

After the previous treatments a final remark has to be stated corresponding to the used response-surface-model.

Remark: This example has only four input parameter. An additional load inductor L_l was neglected[6], since when taking L_l into account, the performance figures exhibited a very rapid behavior at the boundary of the considered design space ranges. This influenced the methods significantly. Therefore bear in mind, that it is desired to avoid the trade-off fronts of being situated at the boundary of the design space D. Because then, especially NBI might fail to compute the complete trade-off front, because it gets easily caught in those boundary points.

6.3.2 Conclusions on the Design Space Exploration

Now, we will shortly summarize the conclusions we can draw from the previous investigations on design space exploration of a source-degenerated LNA with SPEA2 and NBI.

- For NBI, the equality constraints (5.10) for the NBI_β subproblems showed an even better performance than the inequality constraints (5.14). This is contrary to the statements made in [Ste05].
- Moreover NBI performed worse if the initial solutions of the subproblems NBI_β were set to the solutions of the previously computed near-by problems (see Subsection 6.1.3). The choice of a linear combination of the individual minima $\mathbf{x}_i^*$, according to (6.8), showed a slightly better performance.
- With SPEA2 we obtained the results much faster as with NBI. This is due to the high complexity of the SOP solution routine *fmincon*.
- The results obtained by SPEA2 were accurate enough to provide a good impression of the occurring ranges especially for the first two steps. On the other side, NBI failed quite often to provide a suitable solution to the single objective optimization subproblems.
- But conversely to the previous point, the range of the SPEA2 approximation does usually not cover the whole Pareto front, as can be seen in Figure 6.6(a) for instance. Contrariwise, the individual minima $\mathbf{p}_i^*$ could always be computed in NBI. Hence, they give a better impression for the range of the Pareto front.

[6] This implies a purely resistive load impedance Z_l.

- When dealing with the trade-off $P-NF$ in step 2, i.e. in Figure 6.7(a), we saw, that NBI obtained approximations in regions where SPEA2 did not detect solutions, and also the other way around.
  The reason for this was, that different design domains were considered by both methods. Nevertheless, completely different design parameters led to similar performance figures, which verifies exemplarily the ill-posed character of the problem, i.e. the non-uniqueness of design parameters $\mathbf{d} \in X_f$ for attainable performances $\mathbf{p} \in P_a$.
  Furthermore, we cannot expect to get optimal solutions in general, and especially with NBI. For SPEA2, this might hold true too, since we decrease the mutation parameters r_i and A_i continuously (see Subsection 6.1.2). Hence, after some iterations, SPEA2 is not capable to overcome such locally optimal regions too. This was additionally verified by this trade-off example.

Taking all these considerations into account, we find, that both methods have their advantages and disadvantages. On the one hand, with NBI we obtain a larger range of the trade-off front, while SPEA2 is able to produce a lot of uniformly distributed approximations. But SPEA2 fails in the considered framework to cover the whole range. This all leads to the following hybrid approach:

1. First, compute the individual minima $\mathbf{p}_i^*$ according to the NBI procedure.
2. Use the individual minima to set up an initial population P_0.
3. Apply SPEA2 on the problem with the initial set P_0.

This method has the advantage, that it covers almost the whole range and furthermore, the ability of SPEA2 to achieve accurate and fast results is used.

6.4 Summary

In this chapter, we first treated the implementation details of the two methods SPEA2 and NBI. Especially the mutation operator and the stopping criterion in the SPEA2 approach needed some extra investigations. Finally, due to numerical tests we found, that the special stopping criterion, that makes use of the coverage function, is not suitable for our case, since the recombination operator does not align to this criterion. Hence, a maximal number of iterations should be used as the stopping criterion.
Furthermore, when dealing with the mutation operator, we found that normally distributed mutation seems to be more stable than Breeder mutation. On the other hand, for Breeder mutation it is quite likely to obtain a fast convergence behavior. Nevertheless, stability is an important issue and hence, normally distributed mutation is preferred.

In the framework of implementation issues for NBI, especially the choice of the initial guess for the single objective optimization subproblems is important. The numerical tests

showed that a linear combination of the individual minima performs slightly better than to use the solutions of the near-by problems. Furthermore, equality constraints for the subproblems outperform the inequality constraints.

Moreover, we verified the convergence results of SPEA2 by means of numerical results on a simple example. Additionally, SPEA2 and NBI were compared. For the simple example we found a very fast and accurate performance of NBI. Furthermore, the results showed, that recombination should be applied in SPEA2.

In the last part, we treated a typical example of design space exploration. Therefore, we considered different trade-off fronts to gain insight in the model. Furthermore, the restrictions were tightened more and more, in order to obtain a design parameter corresponding to some optimal performance figures. When dealing with this problem, we encountered several issues.
The most important observations are, that NBI takes much more time to proceed than SPEA2 due to the behavior of the model, that SPEA2 obtains sometimes even more accurate results, that NBI fails quite often to produce the single optima to the subproblems and that due to the individual minima the result of NBI gives at least a good impression of the range of the trade-off front.

Combining all these considerations, the conclusion is a suggestion for the following approach: Compute the individual minima according to NBI. Afterwards, incorporate them into an initial population, which is then passed to SPEA2. The obtained approximations are expected to cover almost the whole trade-off front and additionally, we expect them to be uniformly distributed along the front.

Chapter 7

Conclusion

In the following we will summarize this work and since there is space for improvements, we will point out further aims, that could be investigated in the future.

7.1 Conclusions

The aim of this thesis is to explore the design space of a compact model. Therefore, the problem was originally stated in the following form: For a given set of performance figures of a compact model, i.e. a black box, determine the corresponding set in the design space, which is equivalent to detect the inverse map. Since this task is too hard because we do not know any properties of the function of concern, we reduced the problem to a multiobjective optimization problem. That is, we choose some objectives to be optimized and transfer the remaining performances to the constraints of the problem.

In this work, the focus was on RF-circuit block models. The background and the importance of these models was explained. Furthermore, we briefly introduced a low-noise amplifier, which was used later to perform numerical simulations.

Coming back to the main problem of the work, which is to solve a multiobjective optimization problem, we have to bear in mind, that it is not obvious at all, how to optimize all objectives, since an improvement in one performance figure may lead to a setback of another one. Therefore, the aim is to compute the Pareto optimal front.

For the solution of optimization problems, there exist in general two totally different approaches, namely probabilistic (called evolutionary algorithms) and deterministic ones. We chose the Strength Pareto Evolutionary Algorithm 2 (SPEA2) and the deterministic Normal-Boundary Intersection (NBI) method.

In the following, the general framework of evolutionary algorithms was introduced. Evolutionary algorithms are population-based methods, in which the population changes due to fitness assignment, mating selection probabilistic genetic operators. For multiobjective

evolutionary algorithms mainly the fitness assignment and the selection have to be altered. Thereby, the concepts of fitness sharing and elitism are important to obtain an efficient method. These concepts are applied in SPEA2.
Furthermore, almost sure convergence was shown for SPEA2 under certain assumptions. This was carried out, by computing the expected progress in distance of the population in the next generation. The obtained rates, could be verified later by numerical experiments.

On the other hand, NBI was implemented. The special procedure guarantees that the obtained solutions are almost uniformly distributed along the trade-off front. This property and also that non-optimal solutions can be obtained were the reasons to implement this deterministic method.

Finally, we outlined the issues we are faced with when implementing both methods. For SPEA2, the mutation operator needs special treatment. Thereby, its parameters are adapted during the iteration process according to the expected progress of the method. Suitable parameter choice rules were established by means of numerical tests. On the other hand, especially the initial solutions for the single objective optimization subproblems are very crucial. Numerical tests showed a slightly better performance if the initial solutions were chosen according to the corresponding linear combination of the design parameters of the individual minima, than if the solutions of close-by problems are used.
Afterwards, the methods were tested. For simple problems NBI outperformed SPEA2 tremendously. But in the application of the methods to a realistic low-noise amplifier model, both approaches exhibited advantages and disadvantages. First of all, SPEA2 executed much faster than NBI. Additionally, NBI showed a lot of problems to obtain inner points along the trade-off curve. On the other hand, while SPEA2 did not have problems to obtain quite suitable results, the by SPEA2 approximated front did usually not cover at all the whole Pareto front.
To circumvent the problems of both methods, a hybrid method was suggested, which seems to combine the advantages of both approaches. But unfortunately it could not be implemented due to a lack of time.

7.2 Suggestions and Future Work

In this work, we treated and implemented two completely different methods to deal with multiobjective optimization problems. Furthermore, we investigated several issues concerning these methods. Some of them could be clarified, but others not. Hence, there are some tasks left to deal with in future.

- We saw, that the availability of the individual minima is essential for a successful application of NBI. Furthermore, they are of importance, because they limit the Pareto optimal front. Therefore, we could investigate on some guarantees for a correct computation of these minima.

- Investigate suitable stopping criteria for SPEA2. Since the stopping criterion, that we treated, performed not satisfactory when recombination was used, other methods except the maximum iteration count should be found.
- We encountered that the algorithms might get trapped around certain performance values. Such an behavior is well-known for deterministic methods, but since we adapt the mutation parameter of SPEA2, also this method might get trapped around certain performances. Therefore, approaches have to be made to remedy this problem.
- The behavior, as described in the previous point is quite likely around points, where the corresponding design parameters are not unique. Investigations could be made, how to overcome the problems of non-uniqueness. Remember, that the goal of multiobjective optimization is to determine some parameters that yield Pareto optimal points. Thereby, uniqueness-considerations are not an issue.
- It would be interesting to implement the hybrid method, which is expected to result uniformly distributed approximations that dominate the whole Pareto front.
- In this work, we treated only two-dimensional examples. But also higher dimensional problems are possible. Therefore, methods have to be developed to visualize the obtained results in higher dimension. In 3D this could be done by means of a suitable triangulation of the obtained performances.
- For a circuit designer it would be advantageous to have a graphical user interface in order to explore the design space of a given compact circuit block model.

Bibliography

[B̈96] Thomas Bäck. *Evolutionary Algorithms in Theory and Practice: Evolution Strategies, Evolutionary Programming, Genetic Algorithms*. Oxford University Press, 1996.

[BEAG04] Lam T. Bui, Daryl Essam, Hussein A. Abbass, and David Green. Performance analysis of evolutionary multi-objective optimization methods in noisy environments. In *The 8th Asia Pacific Symposium on Intelligent and Evolutionary Systems, pp. 29-39*, 2004.

[BJ68] Richard S. Bucy and Peter D. Joseph. *Filtering for Stochastic Processes with Applications to Guidance*. Interscience Publishers, New York, 1968.

[BT95] Tobias Blickle and Lothar Thiele. A mathematical analysis of tournament selection. In *Proceedings of the Sixth International Conference on Genetic Algorithms, pp. 9–16*, 1995.

[CK07] Jeroen A. Croon and D. B. M. Klaassen. Modelling RF building blocks: Narrow-band weakly-nonlinear LNAs. *Internal Report of NXP Semiconductors*, 2007.

[CKO00] David W. Corne, Joshua D. Knowles, and Martin J. Oates. The Pareto Envelope-based Selection Algorithm for multiobjective optimization. In *Proceedings of the Parallel Problem Solving from Nature VI Conference, pp. 839-848*. Springer. Lecture Notes in Computer Science Nr. 1917, 2000.

[DAPM00] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast Elitist Non-dominated Sorting Genetic Algorithm for multi-objective optimization: NSGA-II. Technical Report KanGAL report 200001, Indian Institute of Technology, Kanpur, India, 2000.

[Das98] Indraneel Das. On characterizing the "knee" of the Pareto curve based on Normal-Boundary Intersection. *SIAM Journal on Optimization, Vol. 8, Nr. 3, pp. 631-657*, 1998.

[DD97] I. Das and J. Dennis. A closer look at drawbacks of minimizing weighted sums of objectives for Pareto set generation in multicriteria optimization problems. *Structural Optimization, Vol. 14, pp. 63-69*, 1997.

[DD98] Indraneel Das and J. E. Dennis. Normal-Boundary Intersection: A new method for generating Pareto optimal points in multicriteria optimization problems. *SIAM Journal on Optimization, Vol. 8, Nr. 3, pp. 631-657*, 1998.

[FF95] Carlos M. Fonseca and Peter J. Fleming. Multiobjective genetic algorithms made easy: selection, sharing and mating restrictions. In *Proceedings of the 1st International Conference on Genetic Algorithms in Engineering Systems: Innovations and Application, pp. 45–52*, 1995.

[Gol89] David E. Goldberg. *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley Publishing Company, Inc., 1989.

[Han99] Thomas Hanne. On the convergence of multiobjective evolutionary algorithms. *European Journal of Operational Research, Vol. 117, pp. 553–564*, 1999.

[HNG94] Jeffrey Horn, Nicholas Nafpliotis, and David E. Goldberg. A Niched Pareto Genetic Algorithm for multiobjective optimization. In *Proceedings of the First IEEE Conference on Evolutionary Computation, Vol. 1, pp. 82-87*, 1994.

[Int] Intel® Cooperation, PRESS KIT - Intel® Core™2 extreme mobile processor. http://www.intel.com/pressroom/kits/mobile/core2extreme/index.htm.

[Lee] J.A.N. Lee. The history of computing. http://ei.cs.vt.edu/~history/.

[Lee04] Thomas H. Lee. *The Design of CMOS Radio-Frequency Integrated Circuits, Second Edition.* Cambridge University Press, 2004.

[LvdTV01] Domine Leenaerts, Johan van der Tang, and Cicero Vaucher. *Circuit Design for RF Transceivers.* Kluwer Academic Publishers, 2001.

[M+02] Ravi Mahajan et al. Emerging directions for packaging technologies. *Intel Technology Journal, Vol. 6, Issue 2, pp. 62–75*, 2002.

[Mat] The MathWorks, Inc. http://www.mathworks.com/.

[Mie99] Kaisa M. Miettinen. *Nonlinear Multiobjective Optimization.* Kluwer Academic Publishers, Boston, 1999.

[Moo65] Gordon E. Moore. Cramming more components onto integrated circuits. *Electronics Magazine, Vol. 38, Nr. 8*, 1965.

[MSV93] H. Mühlenbein and D. Schlierkamp-Voosen. Predictive models for the breeder genetic algorithm i: Continuous parameter optimization. *Evolutionary Computation Vol. 1, Nr. 1, pp. 25-49*, 1993.

[PG02] Maurizio Palesi and Tony Givargis. Multi-objective design space exploration using genetic algorithms. In *Proceedings of the 10th International Conference on Hardware Software Codesign, pp. 67-72*, 2002.

[RA00] Guenter Rudolph and Alexandru Agapie. Convergence properties of some multi-objective evolutionary algorithms. In *Proceedings of the 2000 Congress on Evolutionary Computation CEC00, pp. 1010-1016*. IEEE Press, 2000.

[Rud98] Günter Rudolph. On a multi-objective evolutionary algorithm and its convergence to the Pareto set. In *Proceedings of the 2nd IEEE World Congress on Computational Intelligence(WCCI 98), International Conference on Evolutionary Computation(ICEC 98), pp. 511–516*, 1998.

[SD94] N. Srinivas and Kalyanmoy Deb. Multiobjective optimization using nondominated sorting in genetic algorithms. *Evolutionary Computation, Vol. 2, Nr. 3, pp. 221-248*, 1994.

[SG03] Bart De Smedt and Georges G. E. Gielen. WATSON: Design space boundary exploration and model generation for analog and rf ic design. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, Vol. 22, Nr. 2, pp. 213-224*, 2003.

[SNT85] Yoshikazu Sawaragi, Hirotaka Nakayama, and Tetsuzo Tanino. *Theory of Multiobjective Optimization*. Mathematics in Science and Engineering, Vol. 176. Academic Press, Inc., 1985.

[Ste05] Guido Stehr. *On the Performance Space Exploration of Analog Integrated Circuit*. PhD thesis, Technical University Munich, September 2005.

[T+02] Scott Thompson et al. 130nm logic technology featuring 60nm transistors, low-K dielectrics, and Cu interconnects. *Intel Technology Journal, Vol. 6, Issue 2, pp. 5–13*, 2002.

[TLK02] K.C. Tan, T.H. Lee, and E.F. Khor. Evolutionary algorithms for multi-objective optimization: Performance assessments and comparisons. *Artificial Intelligence Review, Vol. 17, Nr. 4, pp. 253-290*, 2002.

[Zit99] Eckart Zitzler. *Evolutionary Algorithms for Multiobjective Optimization: Methods and Applications*. PhD thesis, ETH Zurich, December 1999.

[ZLT01] Eckart Zitzler, Marco Laumanns, and L. Thiele. SPEA2: Improving the Strength Pareto Evolutionary Algorithm. Technical Report TIK Report 103, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, 2001.

[ZT98] Eckart Zitzler and Lothar Thiele. An evolutionary algorithm for multiobjective optimization: The Strength Pareto Approach. Technical Report TIK Report 43, Computer Engineering and Networks Laboratory (TIK), Swiss Federal Institute of Technology (ETH) Zurich, 1998.