TECHNISCHE UNIVERSITÄT MÜNCHEN
Lehrstuhl für Entwurfsautomatisierung

Deterministic Performance Space Exploration of Analog Integrated Circuits considering Process Variations and Operating Conditions

Daniel Müller-Gritschneder

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzender:	Univ.-Prof. Dr. techn. Josef A. Nossek
Prüfer der Dissertation:	1. Priv.-Doz. Dr.-Ing. Helmut Gräb
	2. Univ.-Prof. Dr.-Ing. Lars Hedrich, Johann Wolfgang Goethe-Universität Frankfurt am Main

Die Dissertation wurde am 19.01.2009 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 26.06.2009 angenommen.

A paperback version of this thesis was published by Verlag Dr. Hut, Munich, in 2009.
ISBN 978-3-86853-167-1.

Acknowledgments

This work resulted from my five year long research activity at the Institute of Electronic Design Automation of the TU München. First of all, I want to thank Prof. Ulf Schlichtmann for giving me the chance to conduct my research at his institute and assist in his lectures. I also want to thank Dr. Helmut Gräb for his guidance during my research work. His advice, feedback and our discussions aided me greatly in developing the ideas and methods presented in this work. I would also like to thank the other committee members, Prof. Nossek and Prof. Hedrich, for their interest in my work.

I thank my colleagues Guido Stehr, Jun Zou and Husni Habal, with whom I worked closely together during my research. I also want to thank all my other colleagues that made the time at the institute interesting and fun.

Finally, I want to express my gratitude towards all my family, my wife Vreni, my kids Lukas, Teresa and Justus, my parents Wolfgang and Gabriele, my brothers Michael, Patrick, David and Gabriel and my sister Magdalena as well as my family-in-law, Bine, Benno, Nathalie, Matthias and Elisabeth, for their continuous support. I also want to thank my friends of the *'Teestube'* for all the good times during these years.

Contents

Chapter 1

Introduction

Recent years have shown a clear trend in integrated circuit (IC) design to implement functionality by means of digital hardware. Still, there is a growing need for analog circuitry. Real world signals are analog. Digital circuits need analog interfaces and analog-to-digital as well as digital-to-analog (AD/DA) converters to communicate with the real world. Additionally, analog circuits are used in power management and clock generation of digital circuits as well as for RF signal processing tasks. Modern mixed-signal system-on-chip (SoC) solutions implement the digital circuit together with analog blocks on one single chip [KCJ+00]. This requires the design of analog circuit blocks with the same digital complementary metal-oxide-semiconductor (CMOS) technology. Digital CMOS technologies have shown a constant rate of device shrinking and decreasing supply voltage in recent years. Due to device shrinking, more transistors can be realized on the same area, as described in the famous forecast by Gordon Moore known as 'Moore's Law' [ITR06].

The number of transistors in analog circuits is usually only a fraction of the number of transistors in digital circuits. Still, analog circuit design poses a major challenge and can be the bottleneck in SoC design. Due to the many physical effects and performance trade-offs in analog design, the introduction of higher abstraction levels is difficult. Many analog circuits are still designed manually by looking at the circuit at transistor level.

Currently, research in analog electronic design automation (EDA) has focused on automating analog design steps as well as to add hierarchy to the analog design flow. New design tools for circuit optimization are commercially available [Mar01, MAW07]. Hardware description languages (HDL), such as VHDL-AMS [APT02, DV03] and Verilog-A [VLR], can be used to describe analog circuits on higher abstraction levels. Still, the analog design flow is lacking behind the digital flow in terms of automation, offering a challenging field for the development of new design methodologies for analog and mixed-signal circuits.

1.1 Analog Design

1.1.1 Analog Design Flow

The analog design flow for CMOS circuits can be broken into three steps [CGRS96, Sch02, Gra07]:

1. *Circuit structure (topology) generation or selection*: First, a circuit structure, also known as topology, must be generated. Often, one can start from existing topologies (structure reuse).
2. *Circuit sizing*: One or more device parameters have to be chosen for each device of the analog circuit. This step is known as circuit sizing. The selectable device parameters are known as design parameters or sizing parameters. For example, CMOS transistors are described by their size, which is defined by the length and width of the channel.
3. *Layout generation*: Finally, a layout must be generated for the sized circuit. The layout holds the geometric information required to produce the integrated circuit in silicon.

Figure 1.1 illustrates the analog design flow. This thesis focuses on the circuit sizing step.

Figure 1.1: The analog design flow

1.1.2 Process Variations and Operating Conditions

Some device parameters are subject to statistical variances due to inaccuracies in the production process of integrated circuits. These process variations often have great impact on the electrical behavior of the circuit. Shrinking device sizes lead to a relative increase of the importance of process variations, since the absolute variations in the device parameters do not scale down by the same factor as the device sizes. Additionally, environmental conditions such as temperature affect the electrical properties of the circuit during operation.

The process variations and operating conditions - in the following referred to as tolerances - must be considered during the design phase of analog circuits. The circuit sizing step is separated into nominal sizing and tolerance sizing.

During nominal sizing, the influence of the tolerances is ignored. The device parameters that are subject to process variations are set to fixed values. Operating conditions are set to standard environmental conditions, such as, e.g., room temperature.

During the tolerance sizing, the influence of the tolerances on the electrical properties of the circuit is analyzed. This requires methods that calculate the influence of process variations and operating conditions on the electrical properties of the circuit.

1.1.3 Circuit Performances

The electrical properties of the circuit are usually described by a set of circuit performances. For example, the performances of operational amplifiers typically include the gain, the bandwidth and the phase margin. The circuit performances depend on the circuit structure and the design parameters. Additionally, process variations and operating conditions affect the circuit performances.

Usually, a set of requirements on the properties of the circuit is defined, the so-called specification. The specification is usually given as bounds on the performances.

In order to measure the circuit performances, the circuit is connected to a test bench. The test bench defines the input signal (stimulus), the output signals as well as test circuitry, which can include, e.g., load capacitances or voltage/current feedback. The circuit performances are usually classified by the type of stimulus, which is applied on the circuit. The three major types of stimulus are direct current (DC), alternating current (AC) and transient (TR).

Various methods are available to evaluate the circuit performances before the circuit is manufactured. These methods can roughly be classified in:

- Numerical analog circuit simulators such as SPICE [Nag75], Spectre [Kun95] or Titan [FWZ+92]. The simulator requires a netlist of the circuit and device simulation models such as BSIM [SSP87] or a macromodel of the circuit [WD06][GESSL95] as input.
- Circuit performance models based on symbolic analysis [GWS89, VDL+01, NBHB04].
- Circuit performance models based on numerical models [HS96, Ziz01, DV05].

Numerical simulation is very accurate due to the great effort taken in describing the behavior of the transistor by its device simulation model. This accuracy comes at the cost of relative high computational costs to evaluate the circuit performances.

Circuit performance models are used to reduce the time required to evaluate the performances. They are generated for each analog circuit topology either manually or, if available, by automatic methods. The generation of such models raises issues such as setup time, complexity, accuracy and number of required circuit simulations.

1.1.4 Automatic Circuit Sizing and Analog Synthesis

Most automatic sizing methods, also known as circuit optimization methods, require as input the circuit structure and the test benches. These methods, usually, apply a *simulator-in-a-loop* concept to automatically generate an optimized sizing.

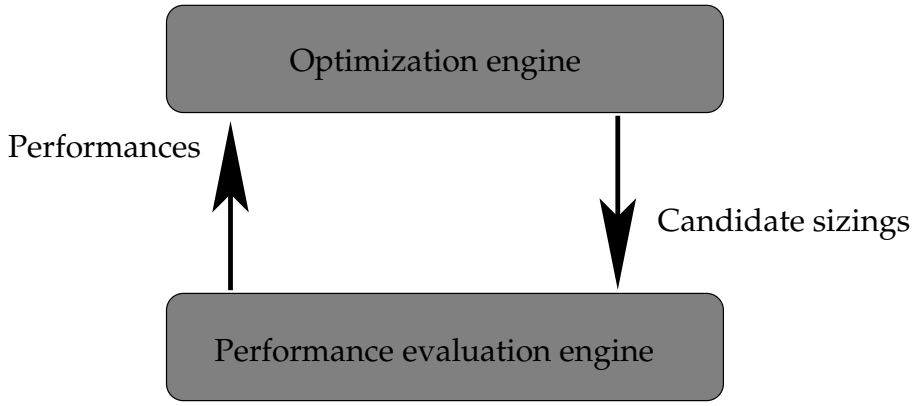


Figure 1.2: Simulator-in-a-loop concept

This concept is illustrated in figure 1.2. The optimization engine generates new candidate sizings. The performances of the candidate sizing are evaluated by the performances evaluation engine. The loop is repeated until a satisfactory sizing is found or a maximal number of iterations is reached. Usually, several performance evaluations are required per iteration. As performance evaluations are computationally expensive, the computational efficiency of such methods can be evaluated by the number of required performance evaluations.

State-of-the art automatic sizing methods can be classified either by the type of performance evaluation method:

- numerical simulation [LD81, NRSVT88, AEG+00, MFDCRV94, ORC96, KPRC99, PKR+00],
- numerical models [Ziz01, DGS03, ABD03, LGXP04, LGXP07] or
- symbolic equations [De87, ETP89, HRC89, KSG90, MC91, HEL92, dMHBL98, dMHBL01, VDL+01, GWS90, ORC96, DLG+98, VDL+01].

or by the type of optimization method:

- deterministic [LD81, NRSVT88, AEG+00, Ziz01, DGS03, LGXP04, LGXP07] or
- stochastic [MFDCRV94, ORC96, KPRC99, PKR+00, ABD03].

Analog optimization methods are already available in commercial tools [Mar01, MAW07, Cad]. On the other side, circuit structures are still mainly generated by hand. Some approaches for an automatic generation of circuit structures, referred to as analog synthesis, have been presented [KL95, HRC89, WH06].

1.2 Motivation

A major challenge in analog design is to find a good trade-off between the various circuit performances [BHT+03]. A range of different circuit performances can be realized by each circuit structure based on the choice of the design parameters (sizing). To find an optimized sizing, each performance becomes an objective of the optimization. Automatic sizing is, therefore, a multi-objective optimization problem. Usually, there is no single sizing, for which all performances take their optimal values at the same time. Due to trade-offs in the design, one performance can only be improved at the cost of the others. The designer must decide on a compromise between the competing performances.

The designer usually sets the importance of each performance, e.g., in form of weights before running an automatic sizing method. Based on these relative importance, automatic sizing leads to a certain compromise between the performances based on an internal objective function (also known as cost function). It is usually not known, which performance compromise is obtained for certain weights. Additionally, a single sizing only shows one performance compromise and, thus, does not give much insight in the performance capabilities of the circuit structure. To gain more insight in the performance trade-offs, new methods have to be developed to explore the complete range of the performance capabilities of a circuit structure, so-called performance space exploration methods.

1.2.1 Performance Space Exploration

Performance space exploration methods aim at calculating the complete range of feasible performances for an analog circuit structure, the so-called *feasible performance space*. Usually it is not required to generate the complete feasible performance space, since not all performance combinations in the feasible performance space are optimal. An optimal compromise between the performances is reached, if one performance can only be further improved at cost of another performance. Such an optimal compromise between the performances is called *Pareto-optimal*.

This is illustrated graphically for two performances noise and power for an example circuit in figure 1.3 left. Point B is sub-optimal since both noise and power can be improved (decreased). Point A is Pareto-optimal as lower noise requires higher power and vice versa in order to stay in the feasible performance space. All Pareto-optimal compromises between the performances make up the so-called *Pareto front*. The Pareto front is located at the boundary of the feasible performance space as shown in figure 1.3.

Most performance space exploration methods only generate the Pareto front, since it shows the most ambitious feasible performance compromises. Performance space exploration methods have two major application fields:

- Performance trade-off analysis and circuit structure (topology) selection.
- Automatic hierarchical sizing.

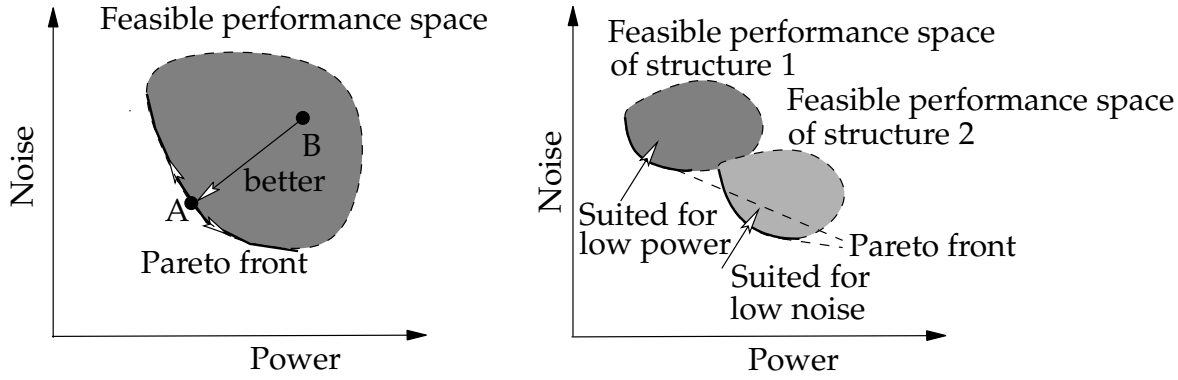


Figure 1.3: Feasible performance space and structure selection. *Left*, feasible performance space and Pareto front. Point B is sub-optimal, point A is Pareto-optimal. *Right*, structure selection based on performance space exploration.

1.2.2 Performance Trade-off Analysis and Circuit Structure Selection

After performance space exploration, all feasible compromises between the performances are available to the designer. It is shown what amount of one performance must be sacrificed to improve another. This knowledge can be used to decide on the most suited performance compromise. If there is more than one circuit structure to implement an analog circuit block for a specified functionality, the performance capabilities of each alternative structure can be generated. Based on the feasible performance spaces, the designer can easily choose the best fitting circuit structure. This is illustrated graphically in figure 1.3, right. Structure 1 can be sized to obtain low power, while structure 2 can be sized for low noise.

1.2.3 Automatic Hierarchical Sizing

Hierarchical sizing is required for large-scale analog and mixed-signal circuits such as Phase Locked Loops or Sigma-Delta AD Converters. The time to simulate a complete large-scale circuit on circuit level can be very long, possible up to days or weeks. Since automatic sizing requires many simulation runs, a flat automatic sizing of such large-scale circuits on the circuit level is not feasible.

Large scale circuits are commonly made up of several functional sub-blocks. Figure 1.4 illustrates exemplarily the block structure of a Charge Pump Phase Locked Loop

(CPPLL). In order to speed up the simulation, the simulation is conducted hierarchically as follows: First, the large-scale circuit is split into its sub-blocks. The blocks are described by the transistor circuit on circuit level and by behavioral models on system level. The behavioral models implement the input/output characteristic of the circuit blocks. The models are parametrized to describe the input/output behavior for different circuit structures and sizings, e.g. VCOs with different gain.

In order to obtain the system performances for a certain sizing of all blocks, the faster hierarchical bottom-up simulation is conducted:

1. Each block is simulated on circuit level to obtain its performances for the given sizing. These circuit performances describe the input/output behavior of the block. The parameters of the behavioral models are extracted directly from the circuit level simulation, such that each block has the same electrical input/output behavior as was found by circuit level simulation.
2. The parametrized behavioral models are used to simulate the performances of the complete circuit much faster.

Automatic hierarchical sizing makes use of this hierarchical description of the circuit. Figure 1.5 shows the hierarchical sizing flow for the CPPLL. The VCO and CP are replaced by behavioral models to speed-up the simulation of the complete PLL. The hierarchical sizing includes a bottom-up characterization of the circuit blocks and a top-down sizing process:

1. In the bottom-up characterization the performance capabilities for each circuit block are obtained by performance space exploration methods. The performance capabilities show which model parameter sets in the parametrized behavioral models can be realized by the underlying circuit structure. The feasible model parameter sets are propagated bottom-up.
2. The sizing process is top-down. First the PLL is optimized for best performance by changing the parameters of the behavioral models. The optimal model parameter set found describes the desired input/output behavior of the blocks. They are propagated down as block specification. It must be assured that the model parameters stay in the feasible ranges calculated in the bottom-up step to avoid unrealistic block specifications. Finally, each block is optimized individually to meet its block specification.

Performance space exploration is a key part of this hierarchical sizing flow. The top-down sizing process can only work if it is known, during system level optimization which block specifications can be realized by each block. For example, if we have no knowledge about the circuit implementation of the VCO, the system level optimization step might generate block specifications for the VCO demanding unrealistic high gain. In order to avoid such unrealistic block specifications, the system-level optimizer must have information about the feasible performance ranges of each sub-block, which are found by performance space exploration.

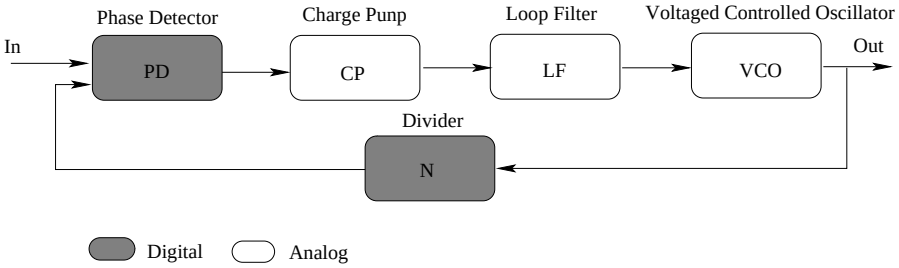


Figure 1.4: Block diagram of a Charge Pump Phase Locked Loop (PLL)

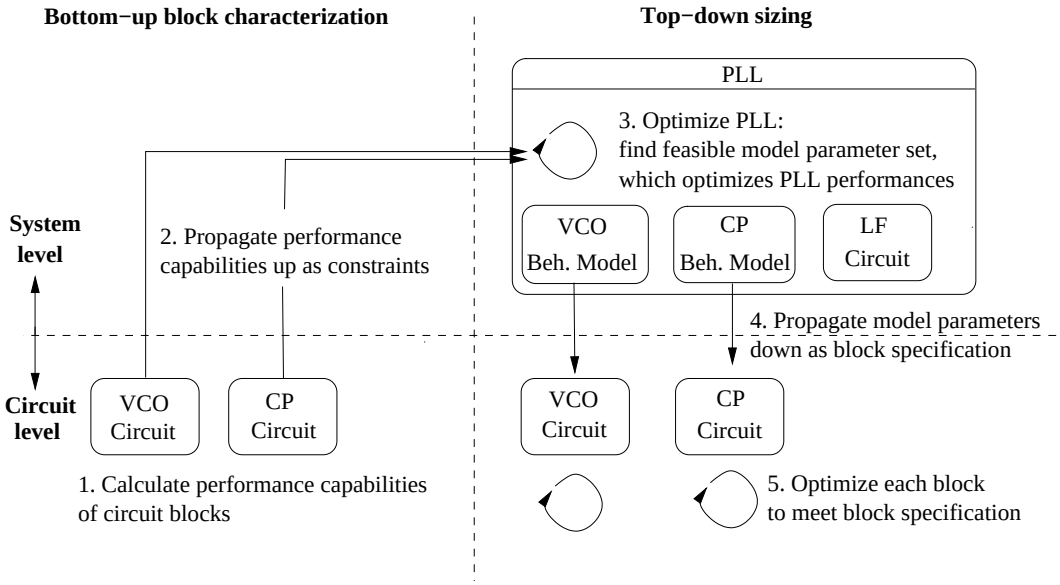


Figure 1.5: Hierarchical sizing flow for a Phase Locked Loop (PLL)

1.3 State-of-the-art

In this section, the state-of-the art in analog performance exploration methods, hierarchical sizing and tolerance analysis is discussed.

1.3.1 Performance Space Exploration Methods

Various methods have been published for calculating the performance capabilities of an analog circuit structure. These methods differ in the way the circuit performances are evaluated, the type of used optimization algorithm, and whether they consider process variations or operating conditions.

In an early work, [HS96], regression and radial basis functions are used to build a performance model. A binary search on the model is used to find the feasible performance space of an analog circuit structure.

Some statistical optimization algorithms , which are applied for the performance exploration of an analog circuit structure, model the biological evolution. They are known as *Genetic Algorithms* (GAs) or *Evolutionary Algorithms* (EAs). These algorithms are based on a so-called population. Each member of the population represents one candidate sizing of the circuit structure. Imitating the biological evolution, new members are generated iteratively by two mechanisms: Cross-over, which is a random combination of two or more members of the population, and mutation, which is a random change of a member of the population. By using a selection mechanism, the algorithm aims at improving the overall quality of the members of the population in each iterations. The selection is based on a fitness criterion, which reflects the quality of each member of the population. In [DG03], a GA is used to generate the nominal Pareto Front. A performance model is constructed based on multi-variate regression. Regression and a GA are applied in [SG03] to generate a Pareto front between a performance and its yield value. In [SCP05], a GA is applied and the performances are evaluated by symbolic expressions. The Strength Pareto Approach [EMG05] and Elitist Non-dominated Sorting algorithm [CA05] make use of the concept of Pareto-optimality to calculate their fitness value. They keep the members of the population in two separate sets. In one set, solutions are kept that are not dominated * by any other member of the population while the other holds the dominated members of the population. The fitness is assigned based on the number of solutions that are dominated or dominate the member. One problem is addressed in both works: The found solutions do not spread evenly along the complete Pareto front. Therefore, the fitness is also assigned such that solutions on sparsely covered areas on the Pareto front are preferred.

In [AV05], a sampling strategy is applied that uses a numerical model of the feasible design region to place the samples. So-called *Analog Platforms* are presented in [DJSV03, BNSV05]. These Analog Platforms include behavioral input/output and performance models as well as a model of the Feasible Performance Space based on a classifier method known as *Support Vector Machines* (SVM). *Simulating Annealing* (SA) is used as optimization method on behavioral level for a hierarchical design flow.

A deterministic optimization method, which has been the focus of some analog performance exploration research, is *Geometric Programming* (GP). GP finds the global optimum for a convex constrained optimization problem. The objective function and constraints must be in the form of *posynomial* functions. Therefore, GP requires a posynomial model of the circuit performances. Due to the fast evaluation of the posynomial performance model, the optimization is very fast. A challenge is located in the model setup. For good model accuracy, the performances must be able to be represented by posynomial functions. In [dMHBL01, dMH03], the nominal Pareto front is generated with GP. In [dMHBL01], additionally, a method, which is based on GP, is shown that generates robust designs †. A so-called robust GP method is presented in [XHL+05] to find the Pareto front between a performance and its yield.

* A solution dominates another, if it is at least in one performance better and in no performance worse.

Other deterministic optimization methods have been studied. In [SGA04, MSGS05, Ste05, SGA07], a method based on a linear performance model is presented. It generates the complete feasible performance space by applying numerical projection methods for polytopes. A deterministic gradient-based optimization method called Sequential Quadratic Programming (SQP) is used in [SGA03, MSGS05, Ste05, SGA07]. The performances are evaluated by simulation. In order to find evenly spread solutions on the Pareto front, a method called Normal-Boundary Intersection (NBI) is used. NBI describes the formulation of objective functions, such that sizings are found that are evenly spread on the Pareto front.

Recently, methods have been presented that consider the influence of the tolerances on the performance capabilities of an analog circuit structure. In [TTR06], a combined GA and SA method is used to generate a nominal Pareto front. By an efficient Monte-Carlo analysis, the so-called yield-aware Pareto front is generated in a subsequent step. The yield-aware Pareto front describes the performance capabilities of the circuit, which is related to a certain minimum yield. In [YL07], another method to generate the yield-aware Pareto front is presented. It uses a numerical performance model known as Kriging model. The Kriging model was originally developed for applications in geostatics.

1.3.2 Hierarchical Sizing Methods

The need for a top-down design flow for large-scale analog and mixed-signal circuits was discussed in [FOK96, VCD+96, CCC+97]. With the upcoming availability of performance exploration methods, more and more approaches for an automatic hierarchical sizing of large-scale analog and mixed-signal circuits are developed. A good overview is given in [RGR07]. In the following, some methods are outlined shortly:

In [DBdMHL01, dMH02], Geometric Programming (GP) and posynomial performance models are used to assign the performances on different stages of a multi-stage circuit. Additionally, a circuit structure is selected for each stage. An optimization approach that does not run a top-down sizing but an simultaneous optimization of circuit block and system performances was presented in [LWP+05]. It targets at optimizing safety margin for a set of block and system specifications based on Geometric Programming.

Deterministic performance exploration methods are applied to realize a top-down hierarchical sizing in [TVR04]. The Pareto front found by Normal-Boundary Intersection (NBI) is used to optimize a Phase Locked Loop (PLL) hierarchically. In [SGA04, Ste05, SGA07], a linear performance model is used to find the block performance capabilities within a hierarchical sizing approach. Due to the inaccuracy of

† Robust design: These robust designs meet the specification for a finite set of process parameters describing variations in the production process. This finite set is found by sampling a given interval for each of the process parameters.

the linear models, an iterative approach is proposed that updates the linearization of the performances repeatedly.

In [EMG05], the Strength Pareto approach is applied to find the performance capabilities of the circuit blocks. Analog platforms, which are discussed in section 1.3.1, are used to realize a hierarchical sizing approach in [DGV+04, BNSV05, BNV06]. Statistical optimization is used. A way to consider process variations in order to generate a robust system [‡] is presented in [BNV06].

A different hierarchical sizing approach is suggested in [ESG+06, GME05]. Instead of propagating the performance capabilities up to the system level and running a top-down sizing process, a bottom-up sizing process is conducted. This is referred to as *Multi-Objective Bottom-Up Methodology* (MUBU). MUBU works as follows: For each block, sizings with different optimal compromises between the performances are generated. On system level, the system performances are evaluated for all combinations of generated optimal compromises of block performances. Assuming a monotonic relationship between the system level performances and block performances, then optimal block performance compromises lead to optimal system performances. By testing all combinations of compromises for all blocks, the complete range of system performances is found. The method to find the sizings with optimal block performance compromises uses an Evolutionary Algorithm (EA).

In [MPGS07, ESG+07], the hierarchical sizing includes a structure selection. In order to find the most suitable structure, the performance capabilities of all structures are generated with an EA.

1.3.3 Tolerance Analysis Methods

Methods for the analysis of process variations [PDML94] can be divided in statistical and deterministic methods. The most common used method is *Monte-Carlo analysis* (MCA) for its very straightforward applicability. This statistical method is used to judge the influence of tolerances on the capabilities of analog circuits in [TTR06, YL07]. The accuracy of MCA depends on the used sample size, which also determines the number of required function evaluations.

Other methods, which are based on so called *worst-case analysis*, have been presented. An elaborate description is given in [Gra07]. The tolerance analysis problem is formulated as optimization problem and can be solved with statistical or deterministic optimization algorithms.

[‡] Robust system: In these systems safety margins are generated systematically to avoid that circuit specifications are violated due to performance degradations caused by changes in the operating conditions such as temperature or by process variations.

1.4 Contributions of this Thesis

This thesis presents several improvements in the field of deterministic methods to generate the Pareto front of analog circuit performances. In the following, the contributions to the state-of-the-art are outlined:

In this thesis an efficient simulation-based deterministic performance space exploration method is presented. Numerical simulators are widely available and accepted in industry, require little setup and offer very good accuracy. The use of efficient gradient-based optimization algorithms offers a way for fast optimization.

A widely used method in this field is Normal-Boundary Intersection (NBI). Already known, NBI has a numerically unfavorable problem formulation with equality constraints when compared to the closely related Goal-attainment method [Ste05, SGA07]. In this thesis, an new approach is present that uses the favorable Goal-attainment method in combination with its equivalent Minmax method. This approach leads to a further improved problem formulation compared to the state-of-the-art because it avoids drawbacks encountered by using solely one of the methods. The Minmax method introduces a discontinuity in the gradient-based optimization due to a discontinuous objective function. The Goal-Attainment method, on the other hand, sometimes flags candidates infeasible that are valid sizings of the circuit due to its additional optimization constraints. These issues are resolved with the presented approach by applying both methods.

Another known drawback of the NBI method is that peripheral regions of the Pareto front are not captured when the trade-off between more than two performances is investigated [MA04, Ste05]. A new iterative approach is presented in this thesis to generate a discretized approximation of the Pareto front that shows the total extent of the Pareto front for any number of performances. The boundary of the Pareto front is defined by so-called trade-off limits, for which one performance can not be further improved at the cost of the others. The presented approach generates the trade-off limits first. Each trade-off limits is equal to the Pareto front of a subset of the performances. By computation of the Pareto fronts of all performance subsets, the boundary of a Pareto front is found, which shows the total extent of the Pareto front. During the computation, the number of performances in the subsets is increased by one in each iteration of this algorithm such that as many iteration steps are required as performances are investigated.

Additionally, analog performance space exploration is a computational expensive, highly nonlinear optimization problem that requires application-specific methods to implement efficient optimization tools. In this thesis, an application-specific implementations to improve the efficiency, robustness and quality of the results of deterministic optimization of analog circuits is presented. A new gradient-based optimization algorithm is introduced, which is referred to as *Wavefront Feasible Sequential Quadratic Programming* (FSQP). It is a feasible SQP algorithm. In contrast to the state-of-the-art, it avoids the violation of constraints of the optimization completely.

This is an important feature in analog sizing, because sizings with constraint violations are usually treated as invalid solutions to the sizing problem and, therefore, are not part of the performance capabilities of analog circuit structures. The Wavefront FSQP method solves several optimization problems simultaneously and features an exchange of solution between these optimizations. This exchange of solutions improves the global convergence. Parallel simulation calls are used to improve the efficiency for multiple CPUs.

Most important, the influence of tolerances such as process variations and operating conditions on the capabilities of analog circuit blocks were not yet addressed for this class of methods. The inclusion of tolerances in deterministic performance space exploration method is presented in this thesis. The influence of process variations and operating conditions is considered in a so-called *Specification Pareto front*. It shows the most ambitious specification on the performances that can be realized by the analog circuit structure, such that a certain minimum yield can be guaranteed. A straightforward computation of this specification Pareto front in reasonable time is infeasible due to the computational costs involved. Therefore, approaches that run an efficient computation of the specification Pareto front in reasonable time are presented in this thesis.

Due to the increasing significance of tolerances in analog circuit design, the importance of these specification Pareto fronts will increase in comparison to the nominal Pareto front for investigating the performance capabilities of analog circuits in future analog design tasks.

1.5 Previous Publications

Parts of the research work on this thesis' topics have been published in [MSGS05, MSGU05, ZMG+05, ZMGS06, MSGS06, MGS06, MGS07b, GZMS07, GMS07, ZMGS07b, ZMGS07a, MGS07a, GMGS08]. In [MGS06, MGS07b], the Wavefront Feasible Sequential Quadratic Optimization algorithm is detailed. The approach to consider tolerances during Pareto optimization is published in [GMS07, MGS07a, GMGS08]. The applicability to hierarchical sizing is shown for a Charge Pump Phase Locked Loop (CPPLL) circuit in [ZMG+05, ZMGS06]. The generation of the system level Pareto front of the CPPLL was shown in [ZMGS07b]. A first approach to consider tolerances in a hierarchical sizing flow is presented in [ZMGS07a].

1.6 Organization of this Thesis

The remainder of this thesis is organized as follows. In chapter 2, basic definitions are given for a mathematical formulation of the sizing step. In chapter 3, the iterative

approach to generate the Pareto front for any number of performances is discussed. In chapter 4, the Wavefront Feasible Sequential Quadratic Programming algorithm is outlined. In chapter 5, the generation of the Specification Pareto front, which also considers the influence of tolerances, is discussed. Chapter 6 describes the experimental results for various circuit examples. Chapter 7 concludes this thesis.

Chapter 2

Description of the Performance Space Exploration Task

This section starts with basic definitions, followed by the description of the performance space exploration task.

2.1 Basic Definitions

2.1.1 Circuit Parameters

The circuit parameter vector $\mathbf{p}$ can be divided into three sub-vectors: The design parameter vector $\mathbf{d}$, the operating parameter vector $\boldsymbol{\theta}$ and the statistical parameter vector $\mathbf{s}$:

$$\mathbf{p} = \begin{bmatrix} \mathbf{d} \\ \boldsymbol{\theta} \\ \mathbf{s} \end{bmatrix}; \qquad \mathbf{p} \in \mathbb{R}^{n_p}; \qquad \begin{array}{l} \mathbf{d} \in \mathbb{R}^{n_d}; \\ \boldsymbol{\theta} \in \mathbb{R}^{n_\theta}; \\ \mathbf{s} \in \mathbb{R}^{n_s}; \end{array} \tag{2.1}$$

The *design parameter vector* $\mathbf{d}$ consists of parameters that have to be be chosen by the designer. Typical design parameters of CMOS circuits include the lengths and widths of the CMOS transistors, bias current values, the capacitance values of compensation capacitors or resistor values. For automatic circuit sizing, lower and upper bound values must be supplied for the design parameters:

$$\mathbf{d}_L \leq \mathbf{d} \leq \mathbf{d}_U \tag{2.2}$$

The *operating parameter vector* $\boldsymbol{\theta}$ models environmental conditions, which influence circuit behavior during its operating time such as temperature or supply voltage. A nominal operating parameter vector $\boldsymbol{\theta}_0$ must be given for the investigation of the nominal circuit behavior, e.g., at room temperature and nominal supply voltage.

During the production of integrated circuits, so-called process variations occur. For example, the geometries of the physical devices of the produced integrated circuits show a statistical behavior. In order to investigate the process variations by circuit simulation, the statistical behavior of the physical device parameters are modeled by distributions of transistor model parameters like oxide thickness, charge mobility or threshold voltage [PDML94].

Additionally, arbitrarily distributed parameters can be transformed into Gaussian distributed parameters [Esh92]. These transformations allow to model arbitrarily distributed process variations by a *statistical parameter vector* $\mathbf{s}$ with n_s Gaussian distributed components. Its mean value vector is denoted as $\mathbf{s}_0$ and its covariance matrix as $\mathbf{C}$. The probability density function (pdf) of the multi-dimensional Gaussian distribution of $\mathbf{s}$ is given as:

$$\text{pdf}(\mathbf{s}) = \frac{1}{\sqrt{2\pi}^{n_s}\sqrt{det(\mathbf{C})}} \cdot e^{-\frac{\beta^2(\mathbf{s})}{2}} \tag{2.3}$$

$$\beta^2(\mathbf{s}) = (\mathbf{s}-\mathbf{s}_0)^T\mathbf{C}^{-1}(\mathbf{s}-\mathbf{s}_0) \tag{2.4}$$

Furthermore, the statistical parameters can be separated into globally varying parameters, which affect all devices of the circuit, and locally varying parameters, which lead to device mismatch [Sch04]. For nominal circuit behavior, the statistical parameter vector is set to its mean vector $\mathbf{s}_0$.

2.1.2 Circuit Performances and Simulation

The performances $\mathbf{f} \in \mathbb{R}^{n_f}$ of a circuit structure such as gain, bandwidth or power consumption depend on the circuit parameters. The circuit performances are obtained by circuit simulation:

$$\mathbf{d}, \boldsymbol{\theta}, \mathbf{s} \overset{Simulation}{\rightarrow} \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}); \quad \mathbb{R}^{n_d}, \mathbb{R}^{n_\theta}, \mathbb{R}^{n_s} \mapsto \mathbb{R}^{n_f} \tag{2.5}$$

In the following, the circuit performances are defined such that the optimal value of the performance corresponds to its minimum value. *

For a so-called nominal investigation of the circuit behavior, the operational and statistical parameters are set to nominal values, given as $\mathbf{s}_0$ and $\boldsymbol{\theta}_0$. For simplicity, we denote the *nominal circuit performances* as:

$$\mathbf{d}, \boldsymbol{\theta}_0, \mathbf{s}_0 \overset{Simulation}{\rightarrow} \mathbf{f}(\mathbf{d}) = \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}_0, \mathbf{s}_0); \quad \mathbb{R}^{n_d} \mapsto \mathbb{R}^{n_f} \tag{2.6}$$

* This can be done without loss of generality. A performance that is optimal at its maximum can be transformed into a performance that is optimal at its minimum by multiplication of the performance values with -1.

2.1.3 Sizing Rules and Valid Parameter Space

Analog circuits usually are composed of so-called basic analog building blocks such as current mirrors, level shifters and differential pairs [GZEA01]. In order for these basic building blocks to work properly, certain geometric and electric requirements must be met. The requirements implement fundamental analog design rules: An example of a geometric requirement is the need of matching transistor geometries in a differential pair. An example of an electrical requirement is the need of transistors to operate in the saturation region.

These requirements, which are referred to in the following as sizing rules, are mentioned under different names in various publications [HRC89, VLv+95, dMHBL98, DNAV99, dMHBL01, VDL+01, GZEA01, MV01, DG03, DSV04]. In [GZEA01, MSG03] a method is presented, which sets up the sizing rules based on a hierarchical recognition of basic analog building blocks in analog CMOS circuits. The method was recently extended to analog Bipolar circuits [MGS08].

Equality sizing rules lead to a reduction of design parameters, since some transistor geometries can not be chosen independently. Additionally, a set of inequality constraints must be fulfilled to assure proper functionality of the basic analog building blocks [†]:

$$\mathbf{c}(\mathbf{d}) \geq \mathbf{0} \tag{2.7}$$

The valid parameter space $\mathcal{D}$ includes all design parameter vectors that fulfill the sizing inequality constraints inside given lower bounds $\mathbf{d}_L$ and upper bounds $\mathbf{d}_U$:

$$\mathcal{D} = \{\mathbf{d} \mid \mathbf{c}(\mathbf{d}) \geq \mathbf{0} \ \wedge \ \mathbf{d}_L \leq \mathbf{d} \leq \mathbf{d}_U\} \tag{2.8}$$

Any sizing that violates the constraints is treated as *not valid*, since it does not follow fundamental analog design rules. Only the design parameter vectors inside $\mathcal{D}$ are taken into account for evaluating the performance capabilities of an analog circuit structure.

2.1.4 Specification

Dependent on the application, analog circuit performances must meet a set of requirements, the so-called specification. Without loss of generality, we define a set of upper bound values on the performances as performance specification [‡]. For nominal

[†] sizing rules are usually checked for nominal operating and statistical parameters. Therefore operating and statistical parameters are not included, even though the sizing rules also depend on these.

[‡] A lower bound value can be transformed into an upper bound value by multiplication with -1.

circuit design, the nominal performance values must meet the performance specification:

$$\mathbf{f}(\mathbf{d}) \leq \mathbf{f}_{spec} \tag{2.9}$$

Usually, the specification also includes ranges for the operating parameters, defining a tolerance region $\mathcal{T}_{\boldsymbol{\theta}}$:

$$\mathcal{T}_{\boldsymbol{\theta}} = \{ \boldsymbol{\theta} \mid \boldsymbol{\theta}_L \leq \boldsymbol{\theta} \leq \boldsymbol{\theta}_U \} \tag{2.10}$$

The circuit works properly, if it meets the performance specification for all operating parameter vectors inside the specified tolerance region:

$$\forall_{\boldsymbol{\theta} \in \mathcal{T}_{\boldsymbol{\theta}}} \quad \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}) \leq \mathbf{f}_{spec} \tag{2.11}$$

The performance values of each individual produced circuit differ statistically from the nominal values due to the process variations. This leads to the definition of the parametric yield.

2.1.5 Parametric Yield

The parametric yield is defined as the percentage of produced circuits that work properly after production according to (2.11). The parametric yield does not consider circuits, which are dysfunctional due to other effects than process variations such as, e.g., spot defects on the wafer.

The acceptance region $\mathcal{A}_s$ includes all the statistical parameters that satisfy the specification according to (2.11):

$$\mathcal{A}_s(\mathbf{d}) = \{ \mathbf{s} \mid \forall_{\boldsymbol{\theta} \in \mathcal{T}_{\boldsymbol{\theta}}} \mathbf{f}(\mathbf{d}, \boldsymbol{\theta}, \mathbf{s}) \leq \mathbf{f}_{spec} \} \tag{2.12}$$

The parametric yield Y for a circuit design parameter vector $\mathbf{d}$ is given as the integral of the probability density function of the statistical parameter vector inside the acceptance region $\mathcal{A}_s$:

$$Y = \int_{\mathcal{A}_s(\mathbf{d})} \ldots \int \mathrm{pdf}(\mathbf{s}) \mathrm{d}\mathbf{s} \tag{2.13}$$

A so-called *yield analysis* solves this integral to obtain the yield of a design parameter vector for a given tolerance region of the operating conditions and a given performance specification:

$$\mathbf{d}, \mathcal{T}_{\boldsymbol{\theta}}, \mathbf{f}_{spec}, \mathrm{pdf}(\mathbf{s}) \overset{\text{Yield analysis}}{\rightarrow} Y \tag{2.14}$$

2.2 The Performance Space Exploration Task

In this section, a problem description for the performance space exploration of analog circuits is derived.

2.2.1 Feasible Performance Space

As was mentioned in section 2.1.3, only those design parameter vectors for which the sizing rules are fulfilled constitute technically meaningful sizings. The performance capabilities of an analog circuit structure are determined by the valid design parameter space $\mathcal{D}$:

$$\mathcal{F} = \{\ \mathbf{f}(\mathbf{d})\ \mid\ \mathbf{d} \in \mathcal{D}\ \} \tag{2.15}$$

This is illustrated graphically in figure 2.1 for two performances and two parameters. The set $\mathcal{F}$ is the *feasible performance space* and includes all performance vectors that can be realized with the analog circuit structure.

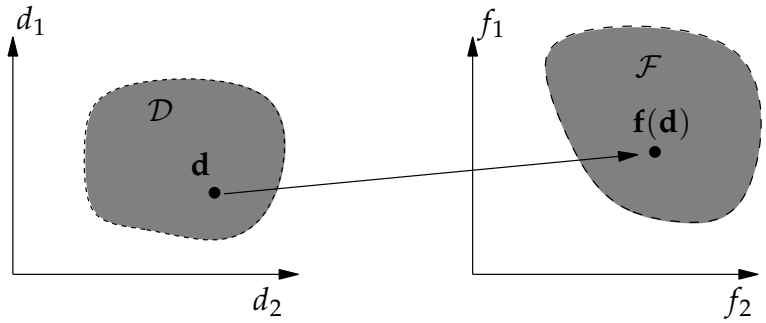


Figure 2.1: Valid design parameter space and feasible performance space

2.2.2 Multi-objective Optimization and Pareto Optimality

A design parameter vector with optimal (minimal) performance values must be generated in order to find an optimal sizing. This requires to solve a multi-objective optimization problem:

$$\min_{\mathbf{d}\in\mathcal{D}} \mathbf{f}(\mathbf{d}) \quad = \quad \min_{\mathbf{d}\in\mathcal{D}} \begin{bmatrix} f_1(\mathbf{d}) \\ \cdot \\ \cdot \\ f_{n_f}(\mathbf{d}) \end{bmatrix} \tag{2.16}$$

Usually, there exists no single design parameter vector, for which all performances become optimal simultaneously. A trade-off situation occurs, such that we have

to decide on a compromise between the performances. This leads to the concept of *Pareto-better* performance vectors and domination. A performance vector $\mathbf{f}_A$ is Pareto-better than or said to dominate a performance vector $\mathbf{f}_B$, if:

$$\forall_i \, f_{A,i} \leq f_{B,i} \quad \wedge \quad \exists_j \, f_{A,j} < f_{B,j} \tag{2.17}$$

In other words: If at least one component of $\mathbf{f}_A$ is better than the same component of $\mathbf{f}_B$ and no component of $\mathbf{f}_A$ is worse than the same component of $\mathbf{f}_B$, then $\mathbf{f}_A$ is Pareto-better than $\mathbf{f}_B$.

This can be further illustrated. We look at the *set of all Pareto-better performance vectors* $\mathcal{M}(\mathbf{f}^+)$ of a given performance vector $\mathbf{f}^+$. It is defined as:

$$\mathcal{M}(\mathbf{f}^+) \; = \; \{\mathbf{f} \neq \mathbf{f}^+ \; | \; \mathbf{f} \leq \mathbf{f}^+\} \; = \; \{\mathbf{f} \; | \; \forall_{i=1,\ldots,n_f} \, f_i \leq f_i^+ \; \wedge \; \exists_j \, f_j < f_j^+\} \tag{2.18}$$

This area is illustrated graphically in figure 2.2 for two performances. It has a box shape and it is unbounded below. The performance vector is equal to the upper right vertex of the upper box boundaries.

A performance vector $\mathbf{f}^*$ is considered Pareto-optimal, if there exists no feasible Pareto-better performance vector. This is the case, if the set of all Pareto-better solutions $\mathcal{M}(\mathbf{f}^*)$ is empty in $\mathcal{F}$:

$$\mathbf{f}^* \in \mathcal{F} \text{ is Pareto-optimal} \quad \Leftrightarrow \quad \mathcal{M}(\mathbf{f}^*) \cap \mathcal{F} = \emptyset \tag{2.19}$$

A Pareto-optimal performance vector $\mathbf{f}^*$ represents an optimal compromise between the individual performances. Each performance can only further be improved at the cost of the others. This is shown in figure 2.2. The illustrated Pareto-optimal performance vector $\mathbf{f}^*$ is located on the lower left boundary of the feasible space. The box-shaped area of Pareto-better solutions is located completely outside the feasible performance space.

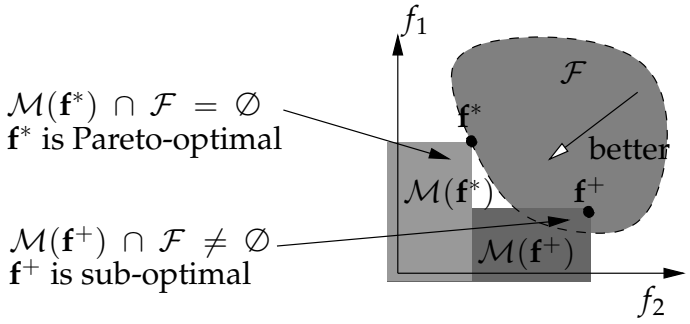


Figure 2.2: Set of Pareto-better solutions and Pareto optimality. The performance vectors Pareto-better than $\mathbf{f}^+$ and $\mathbf{f}^*$ respectively are located in an area, whose upper boundary is defined by $\mathbf{f}^+$ and $\mathbf{f}^*$ respectively. The areas $\mathcal{M}(\mathbf{f}^+)$ and $\mathcal{M}(\mathbf{f}^*)$ are unbounded below.

2.2.3 Pareto Front

The Pareto front $\delta\mathcal{F}$ is the set of all Pareto-optimal performance vectors. It includes all optimal compromises between the performances:

$$\delta\mathcal{F} \quad = \quad \{\mathbf{f} \mid \mathbf{f} \in \mathcal{F} \quad \wedge \quad \mathcal{M}(\mathbf{f}) \cap \mathcal{F} = \emptyset\} \tag{2.20}$$

The Pareto front includes all solutions to the multi-objective optimization problem of (2.16). It shows the ultimate performance capabilities of the analog circuit structure.

The Pareto front concept is illustrated geometrically in figure 2.3 in the two dimensional performance space. In both cases, the so-called utopia performance vector is not part of the feasible performance space such that we can not optimize all performances simultaneously. A compromise between the performances must be made. These optimal compromises are located on the respective Pareto fronts.

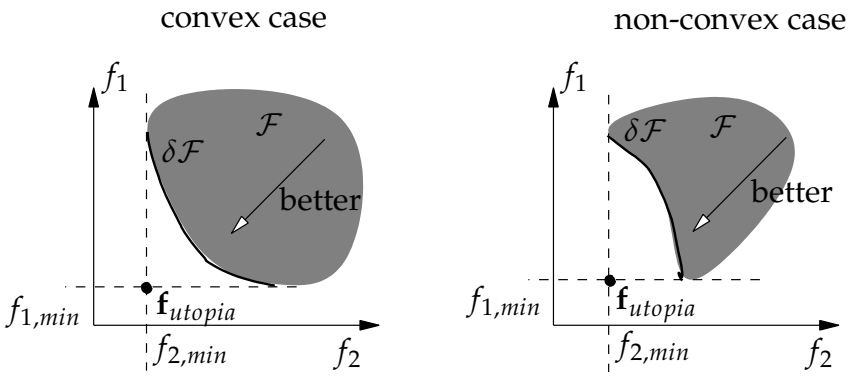


Figure 2.3: Pareto front $\delta\mathcal{F}$ and utopia point $\mathbf{f}_{utopia}$. The minimal performance values are denoted as $f_{1,min}$ and $f_{2,min}$ respectively. The so-called utopia performance vector $\mathbf{f}_{utopia}$ combines all minimal performance values component-wise $f_{utopia,i} = f_{i,min}$. The figure shows the Pareto front, on the *right*, for a convex multi-objective optimization problem and, on the *left*, for a non-convex multi-objective optimization problem.

2.2.4 Weak Pareto Optimality

The condition for weak Pareto optimality is given as:

$$\mathbf{f}^w \in \mathcal{F} \text{ is weakly Pareto} - \text{optimal} \quad \Leftrightarrow \quad \mathcal{N}(\mathbf{f}^w) \cap \mathcal{F} = \emptyset \tag{2.21}$$

with

$$\mathcal{N}(\mathbf{f}^w) = \{\mathbf{f} \mid \mathbf{f} < \mathbf{f}^w\} \tag{2.22}$$

In other words: A feasible performance vector $\mathbf{f}^w$ is weakly Pareto-optimal, if no performance vectors exist that are better (smaller) in all components.

The set $\delta\mathcal{F}_w$ of all weakly Pareto optimal performance vectors is given as:

$$\delta\mathcal{F}_w = \{\mathbf{f} \in \mathcal{F} \mid \mathcal{N}(\mathbf{f}) \cap \mathcal{F} = \emptyset\} \tag{2.23}$$

The Pareto front $\delta\mathcal{F}$ is a subset of the set of all weakly Pareto-optimal performance vectors $\delta\mathcal{F}_w$ [Mie04]. Therefore, there can exist feasible performance vectors that are weakly Pareto-optimal but not Pareto-optimal. This is illustrated in figure 2.4. The boundary of the feasible performance space runs in parallel to the f_1 axis in between the performance vectors $\mathbf{f}^{*W}$ and $\mathbf{f}^{*P}$. The performance f_1 can be improved without a degradation of f_2. The performance vectors along the boundary in between these two points are weakly Pareto-optimal but not Pareto-optimal. They are all dominated by $\mathbf{f}^{*P}$ but no performance vector exists that is better in all components.

If the set of Pareto-optimal performance vectors (Pareto Front) and the set of weakly Pareto-optimal performance vectors are equal, the following holds:

$$\delta\mathcal{F} = \delta\mathcal{F}_w \quad \Rightarrow \quad (\mathbf{f}^* \in \mathcal{F} \text{ is Pareto-optimal} \quad \Leftrightarrow \quad \mathcal{N}(\mathbf{f}^*) \cap \mathcal{F} = \emptyset) \tag{2.24}$$

If it is known that the sets are equal for the investigated feasible performance space, any feasible performance vector that meets the condition for weak Pareto optimality also meets the condition for Pareto optimality. In this case, we only need to check for weak Pareto optimality to see if a feasible performance vector is part of the Pareto front.

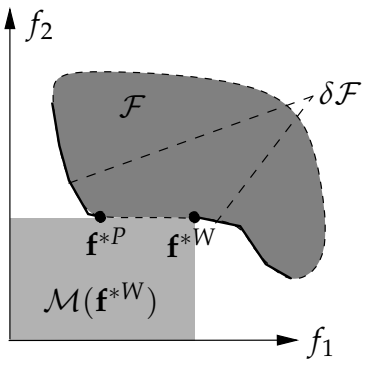


Figure 2.4: Feasible performance space with weakly Pareto-optimal performance vectors. The performance vector $\mathbf{f}^{*W}$ is weakly Pareto-optimal. The performance vector $\mathbf{f}^{*P}$ is Pareto-better than $\mathbf{f}^{*W}$ but not better in all performance values.

2.2.5 Application-Dependent Pareto Front

The *application-dependent Pareto front* describes the performance capabilities for an analog circuit structure with an additional application-dependent performance specification:

$$\mathbf{f}(\mathbf{d}) \leq \mathbf{f}_{spec} \tag{2.25}$$

These performance specifications can be handled like additional sizing rules. The valid parameter space is reduced for the generation of an application-dependent Pareto front to:

$$\mathcal{D}_{ad} = \{\mathbf{d} \mid \mathbf{c}(\mathbf{d}) \geq \mathbf{0} \ \wedge \ \mathbf{f}(\mathbf{d}) \leq \mathbf{f}_{spec} \ \wedge \ \mathbf{d}_L \leq \mathbf{d} \leq \mathbf{d}_U\} \tag{2.26}$$

Application-dependent Pareto fronts are computed, e.g., often for the visual trade-off analysis. It is only possible to visualize the Pareto front for up to three performances. In order to obtain meaningful results, it is required that the remaining circuit performances do not take arbitrary values. For example, we might want to look at the performance trade-offs, but also assure we have a stable circuit. This can be realized by placing a specification that assures stability.

In the following, we will use $\mathcal{D}$ for the valid parameter space, but any application-dependent Pareto front can be generated by exchanging $\mathcal{D}$ with $\mathcal{D}_{ad}$.

2.2.6 Specification Pareto Front

In order to evaluate the capabilities of a circuit under process variations and changing operating conditions, we propose to compute the *specification Pareto front*. The specification Pareto front is defined as the set of most ambitious specifications that can be achieved for a required minimum yield under any operating condition in (2.10). Usually, the yield is calculated for a specification via (2.13). In order to generate the specification Pareto front, we face the inverse problem of mapping a yield requirement on a realizable specification. We refer to this as specification analysis (SpA). The SpA is given as:

$$\mathbf{d}, \mathcal{T}_{\boldsymbol{\theta}}, Y_{min}, \mathrm{pdf}(\mathbf{s}) \quad \overset{\mathrm{SpA}}{\rightarrow} \quad f_{Y,i} \tag{2.27}$$

The resulting specification value $f_{Y,i}$ represents the most ambitious specification bound for the performance f_i, which leads to a yield of $Y = Y_{min}$ (if only this single bound is used to determine the yield).

In order to find the specification Pareto front, the vector of realizable specifications $\mathbf{f}_Y$ has to be the target of the optimization. Specification analysis can be used to calculate

$\mathbf{f}_Y$ for a design parameter vector $\mathbf{d}$. We can write the multi-objective optimization problem to find the specification Pareto front as:

$$\min_{\mathbf{d}\in\mathcal{D}} \mathbf{f}_Y(\mathbf{d}) \quad = \quad \min_{\mathbf{d}\in\mathcal{D}} \begin{bmatrix} f_{Y,1}(\mathbf{d}) \\ \cdot \\ \cdot \\ f_{Y,n_f}(\mathbf{d}) \end{bmatrix} \tag{2.28}$$

Similar to the nominal case in (2.16), there usually exists no single sizing for which all specification values take their minimal (optimal) value. A trade-off situation occurs: To keep the yield above the minimum yield requirement for all performances, the realizable specification on one performance can only be tightened at the cost of the specification on another performance. We define the space of realizable specifications $\mathcal{F}_Y$, which includes all realizable specifications for minimum yield Y_{min} obtainable from valid design parameters:

$$\mathcal{F}_Y = \{\mathbf{f}_Y(\mathbf{d}) \mid \mathbf{d} \in \mathcal{D}\} \tag{2.29}$$

The specification Pareto front $\delta\mathcal{F}_Y$ is composed of all Pareto-optimal specification vectors in the feasible specification space $\mathcal{F}_Y$:

$$\delta\mathcal{F}_Y = \{\mathbf{f}_Y \in \mathcal{F}_Y \mid \mathcal{M}(\mathbf{f}_Y) \cap \mathcal{F}_Y = \emptyset\} \tag{2.30}$$

The specification Pareto front shows the performance capabilities of the analog circuit structure for the given minimum yield requirement.

2.2.7 Performance Space Exploration to Obtain a Discretized Pareto Front

Most performance space exploration methods target at finding the Pareto front, since it describes the optimal performance trade-offs. It is usually not possible to generate the Pareto front analytically. A set of Pareto-optimal performances is generated instead. This set of Pareto-optimal performance vectors represents a discrete description of the Pareto front:

$$\overline{\delta\mathcal{F}} = \{\mathbf{f}_1, \ldots \mathbf{f}_k, \ldots \mathbf{f}_K\} \tag{2.31}$$

The number or density of Pareto-optimal points on the discretized Pareto front $\overline{\delta\mathcal{F}}$ can for instance be chosen by the designer. Figure 2.5 illustrates two discretized Pareto fronts for two performances and seven Pareto-optimal trade-offs graphically.

Pareto optimization targets at finding the design parameter vectors for Pareto-optimal performance vectors systematically. Deterministic gradient-based optimization algorithms minimize a scalar objective function or cost function. We formulate one scalar objective function for each Pareto-optimal performance vector that

we want to generate. Pareto optimization, therefore, requires to solve one optimization problem for each point on the discretized Pareto front. This requires efficient optimization algorithms. The formulation of the objective functions is discussed in chapter 3. In chapter 4, we present an efficient way to run the optimization for all objective functions simultaneously.

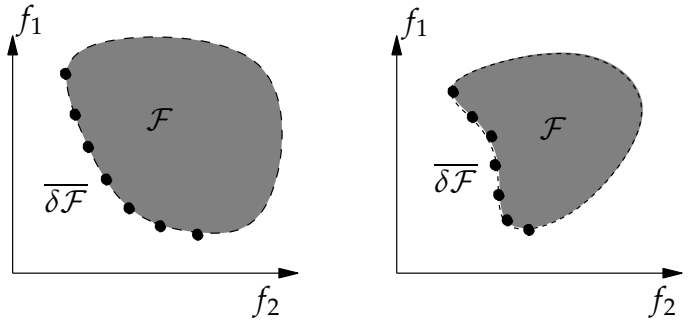


Figure 2.5: Discretized Pareto fronts. *Left*, for the convex case. *Right*, for the non-convex case.

2.3 Summary

The circuit parameters can be divided in design, operating and statistical parameters. Sizing rules guarantee a proper functional behavior of basic analog building blocks. The electrical behavior of a circuit is described by the circuit performances. Specifications on the circuit performances describe the desired functionality of the circuit. Parametric yield is defined as the percentage of proper working circuits after production.

The feasible performance space includes all performance vectors from technically meaningful design parameter vectors of the analog circuit structure. The performance vectors with an optimal compromise between performances are called Pareto-optimal and are located on the boundary of the feasible performance space. For these optimal compromises, one performance can only be improved at the cost of degrading the other performances. A Pareto-optimal point is found by solving a multi-objective optimization problem. The set of all optimal compromises is called Pareto front. It shows the performance capabilities of the circuit.

The specification Pareto front describes the most ambitious specifications that can be obtained for a circuit structure for a given minimum yield requirement. Compared to the nominal Pareto front, it requires to run specification analysis to obtain specifications realizable with the given minimum yield instead of circuit simulation to obtain nominal performance values.

Most performance space exploration methods target at finding a discretized Pareto front. This is called Pareto optimization.

Chapter 3

Pareto Optimization

3.1 Introduction

The task of generating a discretized Pareto front, which shows the performance capabilities of an analog circuit structure, is denoted as Pareto optimization (see Sec. 2.2.7). A discretized Pareto front consists of several Pareto-optimal performance vectors. Pareto optimization consists of solving the multi-objective optimization problem of (2.16) in order to find the design parameter vectors for the Pareto-optimal performance vectors.

Most deterministic Pareto optimization solves one single-objective optimization problem for each Pareto-optimal performance vector. Two aspects are of importance:

1. The applied multi-objective optimization method that combines the different performances in one scalar objective function.
2. The approach to systematically compute Pareto-optimal performance vectors with different performance compromises.

These two aspects of deterministic Pareto optimization are introduced in the following:

3.1.1 Multi-Objective Optimization Methods

Most deterministic optimization algorithms minimize scalar *objective functions*, also known as cost functions [Fle87, NW99]. A multi-objective optimization task must be transformed into such a scalar optimization task. There are different ways to combine the competing performances in a scalar objective function. They are referred to as *multi-objective optimization methods*. A good overview of various multi-objective optimization methods can be found in [MA04] or [Mie04].

Not every one of these multi-objective optimization methods is equally suited for Pareto optimization. The straight-forward weighted-sum method does not guarantee to find every Pareto-optimal performance vectors for non-convex constrained multi-objective optimization problems [DD97]. Alternatives have been suggested for analog circuit sizing and Pareto optimization. In [LD81], *Goal-attainment (GA)* and the *Minmax method* are presented as multi-objective optimization methods. For *Normal-Boundary Intersection (NBI)*, a multi-objective optimization method with characteristic equality constraints was suggested as alternative to the GA method [DD98, Ste05, SGA07]. This method with equality constraints is actually a back-step from the GA method because:

1. The GA method achieves better numerical efficiency than the method with equality constraints for deterministic optimization algorithms. This was shown in [Ste05].
2. The method with equality constraints may compute non-optimal performance vectors for discontinuous Pareto fronts whereas the Minmax and GA method always lead to Pareto-optimal or weakly Pareto-optimal performance vectors [Lin03].

Therefore, the approach in this thesis is based on the Goal-attainment (GA) and the Minmax method. Both methods are well documented but reviewed in this chapter. An understanding of these methods is required as basis for describing the generation of the Pareto front.

Additionally, the equivalence of the two methods is derived in a comprehensive way. This equivalence can be used to increase the efficiency of a deterministic optimization algorithm. This was not yet presented for any other state-of-the-art algorithm. This feature of the GA and Minmax method is discussed in section 4.5 in more detail.

3.1.2 Approaches to find different performance compromises

An approach to compute systematically Pareto-optimal performance vectors with different performance compromises is required. The computed Pareto-optimal performance vectors should show the total extent of the Pareto front and cover it without any clustering. Then, the discretized Pareto front is a good representation of the real Pareto front.

The NBI approach is the state-of-the-art method used often in deterministic Pareto optimization [DD98, Ste05, SGA07]. It can generate the discretized Pareto front for two performances such that an even spread of the Pareto-optimal performance vectors is reached and all parts of the Pareto front are covered. For more than two performances, it fails to generate Pareto-optimal performance vectors in peripheral parts of the Pareto front [MA04].

In this chapter, a novel approach is presented that systematically narrows down the region, in which the Pareto front is located. It is based on the idea that the generated

Pareto-optimal performance vectors always must include solutions on the boundary of the Pareto front, which show the total extent of the Pareto front. It is shown for the first time that the boundary of a Pareto front consists of so-called trade-off limits. A trade-off limit is found by sacrificing one performance to optimize the remaining. This is equal to generating the Pareto front of a subset of the performances. This subset includes all performances except one. A total of n_f such subsets are found by leaving out one of the n_f performances. Therefore, n_f trade-off limits compose the boundary of a Pareto front of n_f performances.

With the insights in the trade-off limits, a new iterative approach to generate the Pareto front for any number of performances is formulated. It allows to find Pareto-optimal performance vectors that show the total extent of the Pareto front. Pareto-optimal performance vectors on the boundary are computed in a first step by using the trade-off limits. Then, the approach populates the inner part of the Pareto front. In this step, the approach aims at achieving a nearly even spread of the generated Pareto-optimal performance vectors. This approach to set up the scalar optimization problems to find nearly evenly spaced Pareto-optimal performance vectors on the inner part of the Pareto front using linear programming.

The approach works iteratively, since the generation of the trade-off limits is a Pareto optimization on its own. The approach computes the discretized Pareto front of n_f performances in exactly n_f subsequent iteration steps. In the n_i-th iteration step, the discretized Pareto front for each performance subset with n_i elements is generated. These discretized Pareto fronts are the trade-off limits for those Pareto fronts that are computed in the following iteration. In the final iteration, the desired discretized Pareto front for all n_f performances is found.

3.1.3 Presence of weakly Pareto-optimal performance vectors

The presence of weakly Pareto-optimal performance vectors that are not Pareto-optimal adds complexity to the Pareto optimization task. For the presented approach, it is assumed that the set of weakly Pareto-optimal performance vectors and Pareto-optimal performance vectors is equal. This allows to change the strict requirement for Pareto-optimality (2.19) to the looser requirement (2.24). All conclusions that are based on this assumption are marked to accent they do not represent the general case. In appendix A, a straight-forward approach to include the case of weakly Pareto-optimal but not Pareto-optimal performance vectors in the presented Pareto optimization approach is outlined.

3.1.4 Structure

This chapter is structured as follows. First, the Minmax and Goal-attainment methods are discussed in section 3.2. Thereafter, the generation of the Pareto front is discussed in section 3.3. In section 3.4, it is shown that its boundary is made of the

so-called trade-off limits. Then, the novel iterative Pareto front generation approach is presented in section 3.5. A way to find evenly spread Pareto-optimal performance vectors on the inner part of the Pareto front is presented in section 3.6. Its use with the new iterative method to generate the Pareto front is shown in section 3.7. The results for a numerical example are compared to the state-of-the-art in section 3.8. Finally, section 3.9 summarizes the chapter.

3.2 Goal-Attainment and Minmax Method

3.2.1 Minmax Method

The Minmax method combines the different performances with the max operator in order to find a design parameter vector $\mathbf{d}^*$ with Pareto-optimal performance vector $\mathbf{f}^* = \mathbf{f}(\mathbf{d}^*)$:

$$\min_{\mathbf{d}\in\mathcal{D}} \max_{i=1..n_f} f_i(\mathbf{d}) \quad \rightarrow \quad \mathbf{d}^*, \mathbf{f}^* = \mathbf{f}(\mathbf{d}^*) \tag{3.1}$$

The scalar objective function value is equal to one target performance, which is the performance with the largest performance value. It can be shown with (2.24) that the resulting performance vector $\mathbf{f}^*$ is always a Pareto-optimal performance vector [*]:

$$\begin{aligned} & \forall_{\mathbf{f}\in\mathcal{F}} \max_{i=1..n_f} f_i \geq \max_{i=1..n_f} f_i^* \\ \Rightarrow \quad & \forall_{\mathbf{f}\in\mathcal{F}} \quad \exists_{f_i, i=1..n_f} f_i \geq f_i^* \\ \Rightarrow \quad & \mathcal{N}(\mathbf{f}^*) \cap \mathcal{F} = \{\mathbf{f} \mid \mathbf{f} < \mathbf{f}^*\} \cap \mathcal{F} = \emptyset \\ \Leftrightarrow \quad & \mathbf{f}^* \text{ is Pareto-optimal} \end{aligned} \tag{3.2}$$

In other words: The Minmax method assigns all performance vectors in $\mathcal{N}(\mathbf{f}^*)$ a better scalar objective function value than $\mathbf{f}^*$. $\mathcal{N}(\mathbf{f}^*)$ includes all performance vectors that are better in all components than a performance vector $\mathbf{f}^*$. If $\mathbf{f}^*$ is the optimum there cannot be a feasible performance vector that is part of $\mathcal{N}(\mathbf{f}^*)$. The optimum $\mathbf{f}^*$ is Pareto-optimal.

The method is illustrated graphically in figure 3.1 for two performances f_1 and f_2. The optimum is reached for the Pareto-optimal performance vector $\mathbf{f}^*$. It is the *feasible* performance vector with minimal performance value f_1^* and f_2^* inside the optimization regions for f_1 and f_2 respectively.

Different Pareto-optimal performance vectors can be found by scaling each performance according to:

$$\hat{f}_i = \frac{f_i - b_i}{v_i} \quad \text{with} \quad v_i > 0, \qquad i = 1, \ldots, n_f; \tag{3.3}$$

[*] The derivation assumes that no weakly Pareto-optimal performance vectors exist.

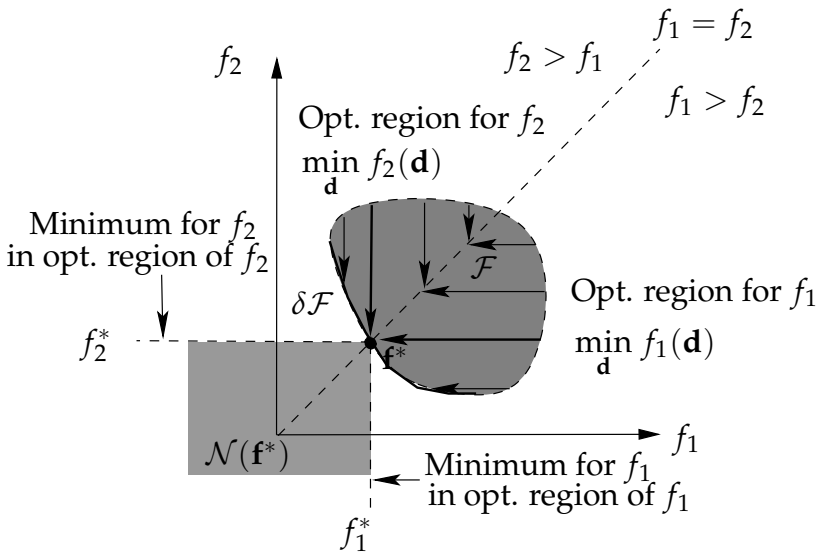


Figure 3.1: Illustration of the Minmax method. Two optimization regions are separated by the line $f_1 = f_2$. In each one of these regions, one the performances is the target performance. Better objective function values are reached by decreasing the target performance value while staying inside the feasible performance space as indicated by the arrows.

The general Minmax method minimizes the maximum of the scaled performance values:

$$\min_{\mathbf{d}\in\mathcal{D}} \max_{i=1..n_f} \frac{f_i(\mathbf{d}) - b_i}{v_i} \quad \rightarrow \quad \mathbf{d}^*, \mathbf{f}^* = \mathbf{f}(\mathbf{d}^*) \tag{3.4}$$

There exists an equivalent Goal-attainment formulation to the Minmax method. It is introduced in the following section.

3.2.2 Equivalent Goal-Attainment Method

This Minmax formulation of (3.4) is not well suited for gradient-based optimization, since the max operator is discrete and non-continuous. The max operator can be transformed into a continuous optimization problem. The largest scaled performance value $\bar{f}_{max}$ can be found by solving a continuous minimization problem:

$$\begin{aligned} \hat{f}_{max} &= \max_{i=1..n_f} \frac{f_i(\mathbf{d}) - b_i}{v_i} \\ \Leftrightarrow \quad \hat{f}_{max} &= \min_{t} t \quad \text{s.t.} \qquad t \geq \frac{f_i(\mathbf{d}) - b_i}{v_i}, \quad i = 1, \ldots, n_f \end{aligned} \tag{3.5}$$

The discrete max operator is replaced with a set of inequality constraints (GA constraints) by introducing the so-called bound parameter t. Such inequality constraints

can be handled by deterministic gradient-based optimization algorithms with so-called active set methods [Fle87, NW99]. We replace the discrete maximization problem in (3.4) by the continuous minimization problem with the additional GA constraints:

$$\min_{\mathbf{d}\in\mathcal{D}} \min_{t} \; t \quad \text{s.t.} \qquad t \geq \frac{f_i(\mathbf{d}) - b_i}{v_i}, \quad i = 1, \ldots, n_f$$

It is possible to merge the two minimization problems:

$$\min_{t,\mathbf{d}\in\mathcal{D}} \; t \quad \text{s.t.} \qquad \underbrace{t \geq \frac{f_i(\mathbf{d}) - b_i}{v_i}, \quad i = 1, \ldots, n_f}_{f_i(\mathbf{d}) \leq v_i\, t + b_i, + \quad i=1, \ldots, n_f}$$

The scaling values can be combined in two vectors $\mathbf{v}$ and $\mathbf{b}$. The GA constraints can then be written in vector notation:

$$\min_{t,\mathbf{d}\in\mathcal{D}} \; t \quad \text{s.t.} \quad \mathbf{f}(\mathbf{d}) \leq \mathbf{v}t + \mathbf{b} \quad \rightarrow \quad \mathbf{d}^*, \mathbf{f}^* = \mathbf{f}(\mathbf{d}^*) \tag{3.8}$$

This formulation is known as Goal-attainment. It leads to the same design parameter vector $\mathbf{d}^*$ as the Minmax problem of (3.4), but is continuous.

3.2.3 Performance Compromise at the Optimum

The performance compromise obtained at the optimum $\mathbf{f}^*$ of the Minmax or Goal-attainment optimization is set by the scaling vectors $\mathbf{v}$ and $\mathbf{b}$. The two scaling vectors $\mathbf{v}$ and $\mathbf{b}$ define a so-called target trajectory in the performance space. This is illustrated for two performances in figure 3.2.

The figure on the left illustrates the Minmax method: One of the performances f_1 or f_2 is the target of the optimization in their two respective optimization regions in the performance space. The regions are separated by the target trajectory. The optimum $\mathbf{f}^*$ has the smallest performance value for f_1 and f_2 inside their respective optimization region. It is located on the target trajectory.

The figure on the right illustrates the Goal-attainment method: The additional GA constraints in (3.8) restrict the performance values $\mathbf{f}(\mathbf{d})$ to a box-shaped area with upper-right vertex given by the expression $\mathbf{v}t + \mathbf{b}$. The position of the vertex on the line is defined by the value of the bound parameter t. The optimization targets at minimizing t inside the feasible region. A decrease in t decreases the box-shaped area by moving the vertex $\mathbf{v}t + \mathbf{b}$ to the lower left along the target trajectory. This is illustrated for two values t_1 and t_2. The optimum is found for the performance vector $\mathbf{f}^*$. It is the only feasible performance vector for $t = t_2$. For smaller t, no feasible performance vectors exist. The optimum $\mathbf{f}^*$ is located on the target trajectory.

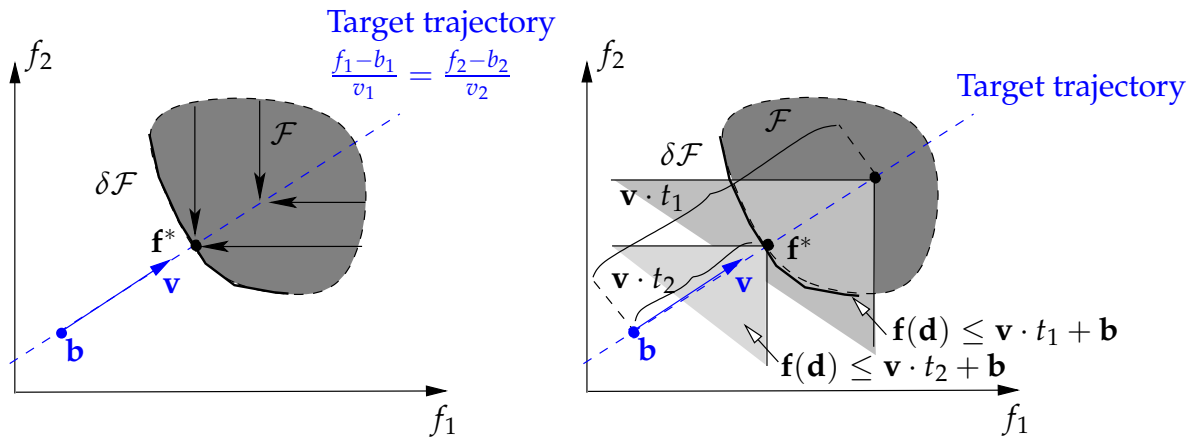


Figure 3.2: Target trajectories for the Minmax and Goal-Attainment method. *Left*, Minmax method. The target trajectory corresponds to the line that separates the two optimization regions for f_1 and f_2. The arrows point in the direction of better objective function values. *Right*, Goal-Attainment method. The performance vector $\mathbf{f}(\mathbf{d})$ is restricted to a box-shaped area with upper-right vertex given by the expression $\mathbf{v}t + \mathbf{b}$. This vertex is located on the target trajectory in the performance space.

Both the Minmax and Goal-Attainment method find an optimal performance compromise on the target trajectory. This optimal compromise is characterized by:

$$\frac{f_i^*(\mathbf{d}) - b_i}{v_i} = \frac{f_j^*(\mathbf{d}) - b_j}{v_j} \quad \text{for} \quad i, j = 1, \ldots, n_f \tag{3.9}$$

In the illustrated example, the target trajectory intersects with the Pareto front. In the following, we investigate the case that the target trajectory does not intersect with the Pareto front.

Two examples are illustrated in figure 3.3. In the left figure, the target trajectory does not cross the feasible performance space $\mathcal{F}$. The optimum $\mathbf{f}^*$ is a Pareto-optimal performance vector. In the right figure, $\mathcal{F}$ has an irregular shape, such that the Pareto front is discontinuous. The intersection point $\mathbf{f}^A$ of the target trajectory and the boundary of $\mathcal{F}$ is not Pareto-optimal, since, e.g., $\mathbf{f}^{*G}$ and $\mathbf{f}^{*L}$ are Pareto-better. Due to the Minmax formulation, $\mathbf{f}^A$ is not the optimum of the optimization. The Minmax can find one of two optima. There is a global optimum at $\mathbf{f}^{*G}$ and a local optimum at $\mathbf{f}^{*L}$. Both optima correspond to Pareto-optimal performance vectors.

The graphical illustrations show that the optima of the Minmax method are always Pareto-optimal unless weakly Pareto-optimal solutions exist that are not Pareto-optimal. If the target trajectory intersects with the Pareto front, the compromise is characterized by (3.9).

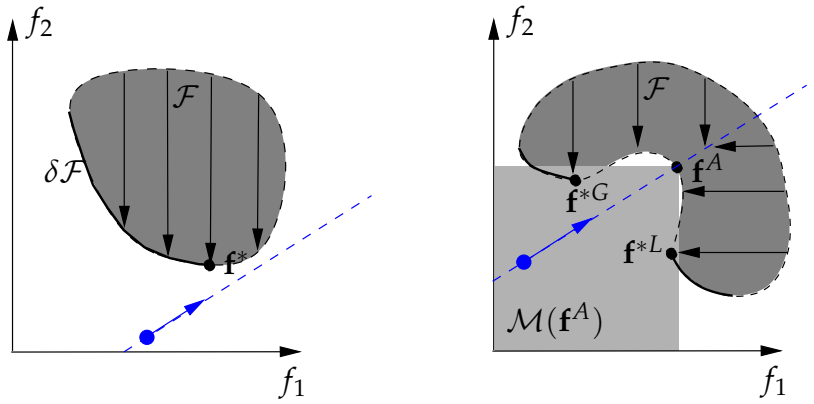


Figure 3.3: Solutions to the Minmax method for the case that the target trajectory does not intersect with the Pareto front. *Left*, target trajectory does not intersect with the feasible performance space. *Right*, discontinuous Pareto front.

3.3 Basics of Pareto Front Generation

The Minmax and Goal-attainment formulation form the basis to compute a Pareto-optimal performance vector. It is located on the target trajectory in the performance space as was shown in the previous section if the target trajectory intersects with the Pareto front. The performance vectors on the target trajectory are characterized by (3.9). We approximate the Pareto front in a discretized way by calculating several Pareto-optimal performance vectors. The art of generating this discretized Pareto front is to define a set of target trajectories intelligently, such that the following holds true:

1. All target trajectories intersect with the Pareto front.
2. The generated Pareto-optimal performance vectors at the intersections show the total extent of the Pareto front.
3. The generated Pareto-optimal performance vectors at the intersections have an even spread.

Solving the scalar optimization problem for a target trajectory is very costly. The number of Pareto-optimal solutions on the discretized Pareto front is limited. The above requirements assure that the generated Pareto-optimal performance vectors capture the Pareto front as efficiently as possible.

In order to define a set of target trajectories, either their direction $\mathbf{v}$ or their base point $\mathbf{b}$ can be changed. Each target trajectory then targets at different Pareto-optimal performance vector. This is illustrated in figure 3.4 for two performances f_1 and f_2.

The feasible performance space is a bounded region for technical applications. One performance cannot be improved unlimitedly on cost of the others. The generation

of the limits of the trade-offs between the performances is investigated in the next section. It is shown that these trade-off limits compose the boundary of the Pareto front. The boundary shows the total extent of the Pareto front. Additionally, target trajectories that intersect with the Pareto front can be defined with knowledge of the boundary.

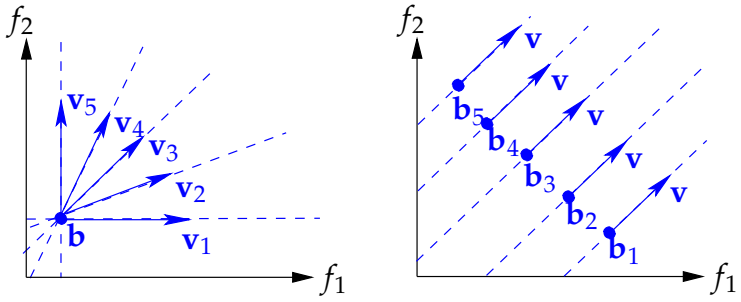


Figure 3.4: Approaches to define a set of target trajectories to find different Pareto-optimal points. *Left*, the approach uses trajectories with different directions $\mathbf{v}_k$ which intersect in one common base point $\mathbf{b}$. *Right*, the approach uses multiple parallel trajectories with same direction $\mathbf{v}$ and different base points $\mathbf{b}_k$.

3.4 Performance Sub-Spaces, Trade-Off Limits and Boundary of the Pareto front

In the following sections, the boundary of the Pareto front is investigated by looking at the trade-off limits on the Pareto front. First, the concept of performance sub-spaces is introduced.

3.4.1 Performance Sub-Spaces

In this section, performance sub-spaces are defined. The set of all performances is given as:

$$S_f = \{f_1, ..., f_{n_f}\} \tag{3.10}$$

If only a subset of the performances $S_m \subseteq S_f$ is considered, then we are working in the corresponding performance sub-space. Elements for performances that are not part of the performance subset must be removed from all vectors. A superscript denotes that the vector is defined in the performance sub-space of S_m, e.g., $\mathbf{b}^{S_m}$, $\mathbf{v}^{S_m}$ and $\mathbf{f}^{S_m}$.

For example in the performance sub-space of the subset $S_m = \{f_1, f_3\}$, the vectors are defined as $\mathbf{b}^{\{f_1,f_3\}} = [b_1\ b_3]^\mathrm{T}$, $\mathbf{v}^{\{f_1,f_3\}} = [v_1\ v_3]^\mathrm{T}$ and $\mathbf{f}^{\{f_1,f_3\}}(\mathbf{d}) = [f_1(\mathbf{d})\ f_3(\mathbf{d})]^\mathrm{T}$.

The same superscript is used to denote the Pareto front $\delta\mathcal{F}^{S_m}$ or set of Pareto-better solution $\mathcal{M}(\mathbf{f}^{S_m})$ in respect to the reduced set of performances S_m:

$$\begin{aligned} \mathcal{M}(\mathbf{f}^{*S_m}) &= \{\mathbf{f} \mid \mathbf{f}^{S_m} \leq \mathbf{f}^{*S_m} \wedge \mathbf{f}^{S_m} \neq \mathbf{f}^{*S_m}\} \\ \delta\mathcal{F}^{S_m} &= \{\mathbf{f} \mid \mathcal{M}(\mathbf{f}^{S_m}) \cap \mathcal{F} = \emptyset\} \end{aligned} \tag{3.11}$$

If the superscript is omitted, the vector or set refers to the complete set of performances S_f.

3.4.2 Trade-Off Limits

For a Pareto-optimal performance vector one performance can only be improved at the cost of the others. The trade-off can only be shifted along a so-called *trade-off direction*. A shift in a trade-off direction is equal to sacrificing one performance f_i to improve the remaining performances. All performance vectors which are located in the i-th trade-off direction from a performance vector $\mathbf{f}^*$ are combined in the following set:

$$\mathcal{T}_i(\mathbf{f}^*) = \{\mathbf{f} \mid f_i > f_i^* \ \wedge \ \underbrace{\mathbf{f}^{S_i} \leq \mathbf{f}^{*S_i} \wedge \mathbf{f}^{S_i} \neq \mathbf{f}^{*S_i}}_{\mathbf{f} \in \mathcal{M}(\mathbf{f}^{*S_i})}\} \quad \text{with} \quad S_i = S_f \setminus f_i \tag{3.12}$$

In other words: The set $\mathcal{T}_i(\mathbf{f}^*)$ includes all those performance vectors, which are worse in the component f_i compared to the same component f_i^*, but Pareto-better in regard to the remaining components. S_i includes all performances except f_i.

Figure 3.5 illustrates the sets $\mathcal{T}_i$ for two performances. For technical applications, the feasible performance space is a bounded region. The Pareto front is, therefore, also bounded. The trade-offs are limited such that one performance cannot be improved unlimitedly at the cost of the other performances. At the trade-off limit $\delta\mathcal{F}_{B,i}$, the compromise between the performances cannot be shifted further in the corresponding trade-off direction along the Pareto front $\delta\mathcal{F}$:

$$\delta\mathcal{F}_{B,i} = \{\mathbf{f} \mid \mathcal{T}_i(\mathbf{f}) \cap \mathcal{F} = \emptyset \ \wedge \ \mathbf{f} \in \delta\mathcal{F}\} \tag{3.13}$$

In other words: The trade-off limit $\delta\mathcal{F}_{B,i}$ includes all Pareto-optimal performance vectors, for which a degradation of f_i cannot be used to improve the remaining performances.

Theorem: The trade-off limit $\delta\mathcal{F}_{B,i}$ is equal to the Pareto front for the performance subset S_i:

$$\delta\mathcal{F}_{B,i} = \delta\mathcal{F}^{S_i}, \quad \text{with} \quad S_i = S_f \setminus f_i \tag{3.14}$$

Proof: Let there be a performance vector $\mathbf{f}_A \in \delta\mathcal{F}^{S_i}$. Applying (3.11) and (3.12), we show that $\mathcal{T}_i(\mathbf{f}_A) \cap \mathcal{F} = \emptyset$:

$$\mathbf{f}_A \in \delta\mathcal{F}^{S_i} \Leftrightarrow \mathcal{M}(\mathbf{f}_A^{S_i}) \cap \mathcal{F} = \emptyset \ \Rightarrow \ \mathcal{T}_i(\mathbf{f}_A) \cap \mathcal{F} = \emptyset$$

In other words: For the performance vectors on the Pareto front $\delta\mathcal{F}^{S_i}$, the performances in the subset $S_i = S_f \setminus f_i$ can not be further improved at the cost of f_i because $\delta\mathcal{F}^{S_i}$ already calculated the ultimate performance capabilities in regard to S_i.

With (3.11) and (2.24) we show that $\mathbf{f}_A \in \delta\mathcal{F}$ †

$$\begin{aligned} &\mathbf{f}_A \in \delta\mathcal{F}^{S_i} \Leftrightarrow \mathcal{M}(\mathbf{f}_A^{S_i}) \cap \mathcal{F} = \emptyset \\ \Rightarrow &\forall_{\mathbf{f}\in\mathcal{F}, \mathbf{f}^{S_i} \neq \mathbf{f}_A^{S_i}} \mathbf{f}^{S_i} \not\leq \mathbf{f}_A^{S_i} \Rightarrow \forall_{\mathbf{f}\in\mathcal{F}} \mathbf{f} \not\prec \mathbf{f}_A \\ \Rightarrow &\mathcal{N}(\mathbf{f}_A) \cap \mathcal{F} = \emptyset \Leftrightarrow \mathbf{f}_A \in \delta\mathcal{F} \end{aligned}$$

In other words: If there exists no feasible performance vector that is in every component of the subset S_i better or equal than $\mathbf{f}_A$, there can also not exist a feasible performance vector that is better than $\mathbf{f}_A$ in all n_f components.

The relation $\mathbf{f}_A \in \delta\mathcal{F}^{S_i} \Rightarrow \mathbf{f}_A \in \delta\mathcal{F}_{B,i}$ holds. Now let there be a performance vector $\mathbf{f}_B \in \delta\mathcal{F}_{B,i}$. We use (2.18), (3.11) and (3.12) to show that $\mathbf{f}_B \in \delta\mathcal{F}_{B,i} \Rightarrow \mathbf{f}_B \in \delta\mathcal{F}^{S_i}$ also holds:

$$\begin{aligned} \mathbf{f}_B \in \delta\mathcal{F}_{B,i} &\Leftrightarrow \mathcal{T}_i(\mathbf{f}_B) \cap \mathcal{F} = \emptyset \quad \wedge \quad \mathbf{f}_B \in \delta\mathcal{F} \\ \Leftrightarrow \{\mathbf{f} \mid f_i > f_{B,i} \wedge \mathbf{f}^{S_i} \leq \mathbf{f}_B^{S_i} \wedge \mathbf{f}^{S_i} \neq \mathbf{f}_B^{S_i}\} \cap \mathcal{F} = \emptyset &\quad \wedge \quad \{\mathbf{f} \mid \mathbf{f} \leq \mathbf{f}_B \wedge \mathbf{f} \neq \mathbf{f}_B\} \cap \mathcal{F} = \emptyset \\ \Rightarrow \{\mathbf{f} \mid \mathbf{f}^{S_i} \leq \mathbf{f}_B^{S_i} \wedge \mathbf{f}^{S_i} \neq \mathbf{f}_B^{S_i}\} \cap \mathcal{F} = \emptyset &\quad \Rightarrow \quad \mathcal{M}(\mathbf{f}_B^{S_i}) \cap \mathcal{F} = \emptyset \Rightarrow \mathbf{f}_B \in \delta\mathcal{F}^{S_i} \end{aligned}$$

such that the sets $\delta\mathcal{F}_{B,i}$ and $\delta\mathcal{F}^{S_i}$ are equal.

□

The trade-off limits are generated by computing the Pareto fronts of the performance subsets $S_i = S_f \setminus f_i$. Table 3.1 and table 3.2 illustrate the trade-off directions and the generation of the trade-off limits for two and three performances.

Table 3.1: Trade-off directions and trade-off limits for two performances

Trade-off direction	Trade-off limit generation
Improvement of f_1 at the cost of f_2	$\min_{\mathbf{d}} f_1 \rightarrow \mathbf{d}^{*1}, \delta\mathcal{F}^{\{f_1\}} = \mathbf{f}^{*1} = \mathbf{f}(\mathbf{d}^{*1})$
Improvement of f_2 at the cost of f_1	$\min_{\mathbf{d}} f_2 \rightarrow \mathbf{d}^{*2}, \delta\mathcal{F}^{\{f_2\}} = \mathbf{f}^{*2} = \mathbf{f}(\mathbf{d}^{*2})$

† This derivation is only valid if the set of weakly Pareto-optimal performance vectors and Pareto-optimal performance vectors is equal.

Table 3.2: Trade-off directions and trade-off limits for three performances

Trade-off direction	Trade-off limit generation
Improvement of f_1 and f_2 at the cost of f_3	$\min_{\mathbf{d}} \begin{bmatrix} f_1 \\ f_2 \end{bmatrix} \rightarrow \delta\mathcal{F}^{\{f_1,f_2\}}$
Improvement of f_1 and f_3 at the cost of f_2	$\min_{\mathbf{d}} \begin{bmatrix} f_1 \\ f_3 \end{bmatrix} \rightarrow \delta\mathcal{F}^{\{f_1,f_3\}}$
Improvement of f_2 and f_3 at the cost of f_1	$\min_{\mathbf{d}} \begin{bmatrix} f_2 \\ f_3 \end{bmatrix} \rightarrow \delta\mathcal{F}^{\{f_2,f_3\}}$

3.4.3 Boundary of the Pareto Front

The trade-off limits compose the boundary $\delta\mathcal{F}_B$ of the Pareto front:

$$\delta\mathcal{F}_B = \bigcup_{i=1...n_f} \delta\mathcal{F}_{B,i} = \bigcup_{i=1...n_f} \delta\mathcal{F}^{S_i}, \quad \text{with} \quad S_i = S_f \setminus f_i \tag{3.16}$$

On the boundary, the trade-off cannot be further shifted in one of the trade-off directions. The Pareto front for each performance subset $S_i = S_f \setminus f_i$, $i = 1...n_f$ must be generated in order to generate the boundary.

Figure 3.5 illustrates the boundary for the case of two performances. In this case, the boundary consists of the two so-called Individual Performance Minima (IPM) $\mathbf{f}^{*1}$ and $\mathbf{f}^{*2}$ (See table 3.1). At the IPM, the respective performance has its overall smallest value. For each IPM, the set $\mathcal{T}_i$ of the corresponding trade-off direction is empty inside the feasible performance space.

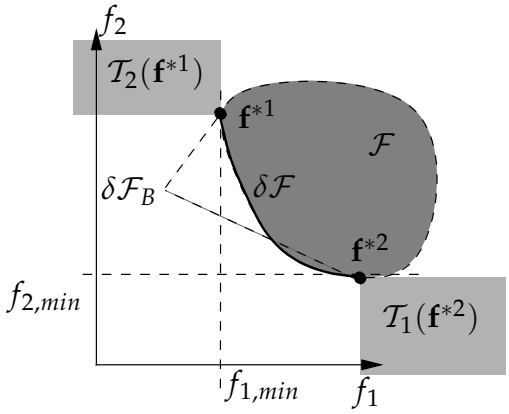


Figure 3.5: Boundary of the Pareto front for two performances.

For three performances, the trade-off limits are equal to the Pareto front of each performance pair respectively, denoted as $\delta\mathcal{F}^{\{f_1,f_2\}}$, $\delta\mathcal{F}^{\{f_1,f_3\}}$ and $\delta\mathcal{F}^{\{f_2,f_3\}}$ (See table 3.2). Figure 3.6 illustrates the boundary of the Pareto front for three performances in the performance space. Each Pareto front of a performance pair is a curved edge. Every two edges share an IPM as common vertex. The three Pareto fronts of the performance pairs form the closed boundary of a face. The bounded face is the Pareto front for the three performances.

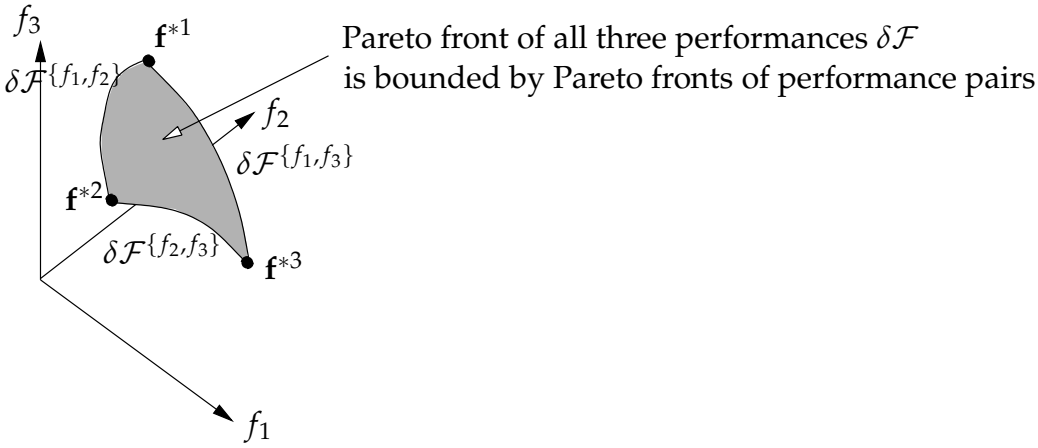


Figure 3.6: Boundary of the Pareto front for three performances.

3.5 Iterative Pareto Front Generation Approach

The idea of the iterative approach to generate the Pareto front will be illustrated in the following for two and three performances. Then, its general structure for any number of performances is presented.

3.5.1 Generation of the Discretized Pareto Front for Two Performances

Target trajectories that intersect with the Pareto front can only be defined with knowledge about the location of the Pareto front. The region, in which the Pareto front must be located, can be found by generating its boundary first.

The discretized Pareto front for two performances can be generated in a two-step process as is illustrated in figure 3.7. The boundary consists of the two IPM $\delta\mathcal{F}_B = \{\mathbf{f}^{*1}, \mathbf{f}^{*2}\}$ (see table 3.1). The IPM can be generated by single-objective optimization in a first step. The target trajectories are defined in a second step. Two ways to define the target trajectories are illustrated: For the left approach, the base point is the utopia

point and the directions of the target rays are chosen to run in between the IPM. In the right approach, multiple base points are set on the connecting line between the IPM. A common direction of the target trajectories is chosen, which is normal to the connecting line of the IPM. This approach was suggested and name-giving for the NBI method [DD98, Ste05, SGA07].

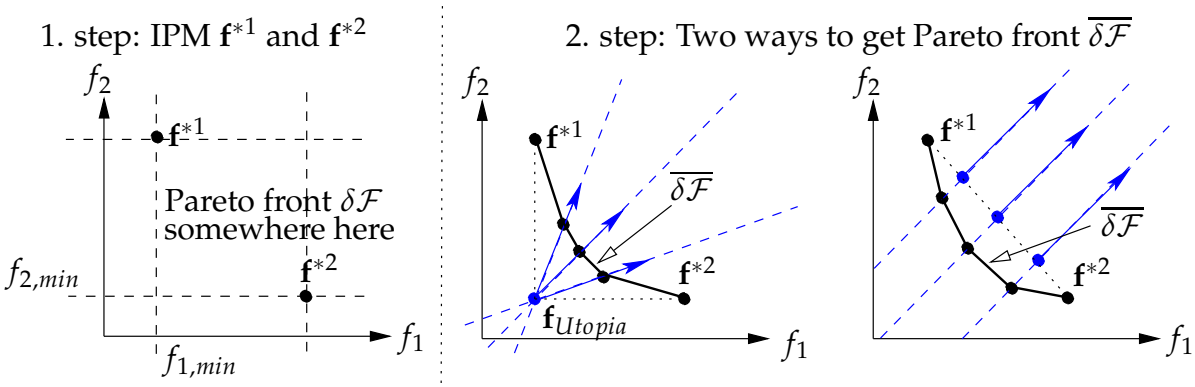


Figure 3.7: Generation of the discretized Pareto front for two performances. *Left*, IPM generation to locate Pareto front. *Middle*, multiple direction approach. *Right*, multiple base point approach.

3.5.2 Generation of the Discretized Pareto Front for Three Performances

As was shown, the Pareto front for two performances can be generated through its boundary that is defined by the two IPM. The complexity of calculating the boundary of the Pareto front increases for three performances. The trade-off directions are defined by improving two of the three performances at the cost of the third (See table 3.2). The boundary consists of the Pareto fronts of each performance pair respectively: $\delta\mathcal{F}^{\{f_1,f_2\}}$, $\delta\mathcal{F}^{\{f_1,f_3\}}$ and $\delta\mathcal{F}^{\{f_2,f_3\}}$. The boundary of the Pareto front is approximated by generating the discretized fronts of the performance pairs:

$$\overline{\delta\mathcal{F}}_B = \overline{\delta\mathcal{F}}^{\{f_1,f_2\}} \cup \overline{\delta\mathcal{F}}^{\{f_1,f_3\}} \cup \overline{\delta\mathcal{F}}^{\{f_2,f_3\}} \tag{3.17}$$

The generation of the Pareto front for three performances requires three steps. This is illustrated in figure 3.8 for an approach with parallel search trajectories. The IPM are generated by single-objective optimization in the first step. In the second step, the discretized Pareto fronts of the performance pairs are generated as described in the previous section. The discretized boundary is found by joining the discretized Pareto fronts according to (3.17). In the final third step, target trajectories are defined that run *'inside'* the boundary to populate the inner parts of the Pareto front with

Pareto-optimal performance vectors. A method to define fitting target trajectories is introduced in section 3.6. The resulting Pareto-optimal performance vectors on the inner part are joined with the points on the discretized boundary. Together they construct the discretized Pareto front of all three performances.

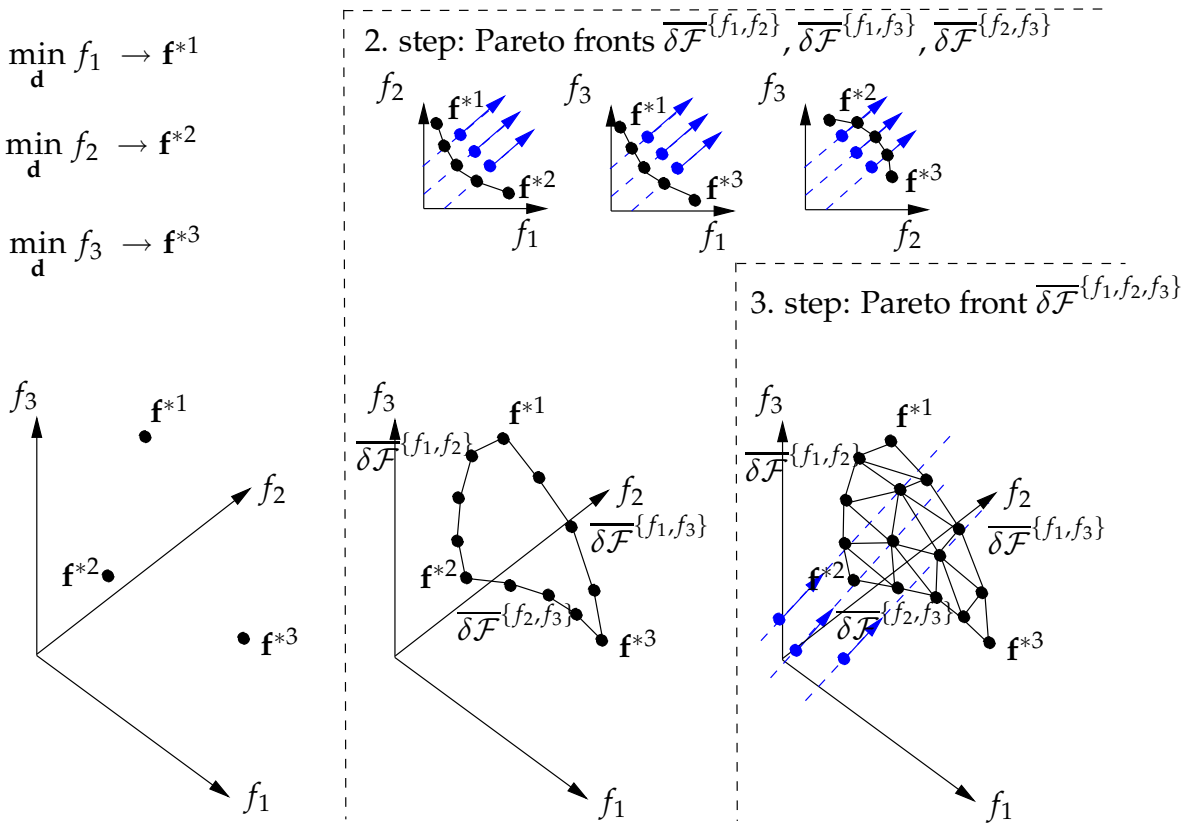


Figure 3.8: Generation of the discretized Pareto front for three performances with parallel target trajectories. *Left*, IPM generation *Middle*, generation of discretized Pareto fronts of performance pairs *Right*, generation of the discretized Pareto front of all three performances.

3.5.3 Generation of the Pareto front for Any Number of Performances

The Pareto front for any number of performances n_f can be found systematically in n_f subsequent iteration steps. The main idea of the iterative approach is to assure that the boundary of the Pareto front is known before it is generated. The boundary must be built up iteratively starting from the Individual Performance Minima. During the iterative process, the Pareto front for every combination of the n_f performances is generated.

The approach is illustrated in figure 3.9 for four performances. In the 1st iteration step, the Individual Performance Minima (IPM) are generated by single-objective optimization. The IPM define the boundaries of the Pareto fronts of all performance pairs or 2-performance combinations completely, such that their location is known. The Pareto fronts of the performance pairs are generated in the 2nd iteration step. They, again, can be used to estimate the boundaries of the Pareto fronts of the performance triples. Those are generated in the 3rd iteration step. Finally, the Pareto fronts of all performance triples define the boundary of the Pareto front of all four performances, which can then be generated in the final iteration step.

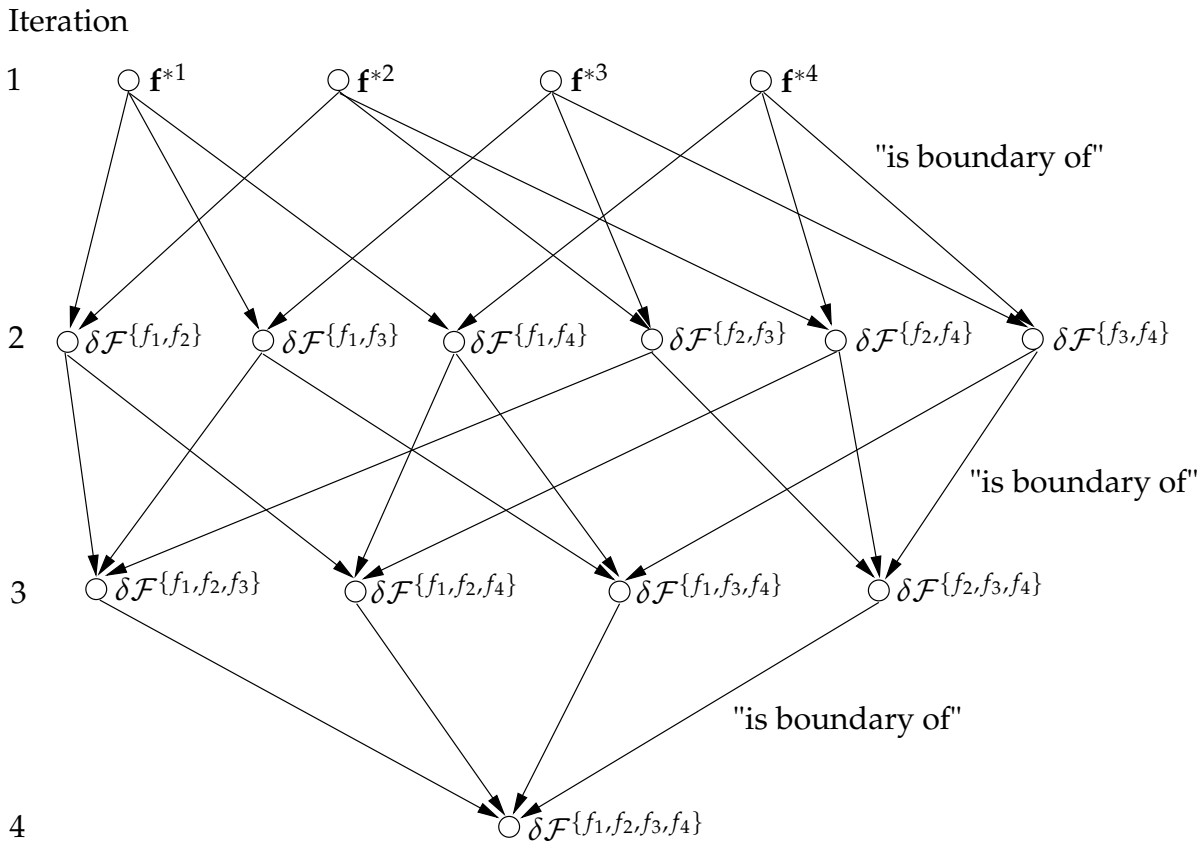


Figure 3.9: Illustration of the iterative Pareto front generation approach.

3.5.4 Structure of the Iterative Pareto Front Generation Approach

The structure diagram of this iterative Pareto front generation approach is shown in table 3.3. We define the following family set to keep track of all performance combinations and their boundaries:

$$C^{(n_i)}(S_f) : \text{Family set of all subsets of } S_f \text{ with } n_i \text{ elements}$$

For example, we obtain for the performance set $S_f = \{f_1, f_2, f_3, f_4\}$:

$$\begin{aligned} C^{(1)}(S_f) &= \{\{f_1\}, \{f_2\}, \{f_3\}, \{f_4\}\} \\ C^{(2)}(S_f) &= \{\{f_1, f_2\}, \{f_1, f_3\}, \{f_1, f_4\}, \{f_2, f_3\}, \{f_2, f_4\}, \{f_3, f_4\}\} \\ C^{(3)}(S_f) &= \{\{f_1, f_2, f_3\}, \{f_1, f_2, f_4\}, \{f_1, f_3, f_4\}, \{f_2, f_3, f_4\}\} \end{aligned}$$

The boundary of any performance combination $S_m^{(n_i)}$ is given as:

$$\delta\mathcal{F}_B^{S_m^{(n_i)}} = \bigcup_{j=1\ldots n_i} \delta\mathcal{F}^{S_j^{(n_i-1)}} \quad \text{with} \quad S_j^{(n_i-1)} = S_m^{(n_i)} \setminus f_h \tag{3.19}$$

The performance f_h is the j-th element in the set $S_m^{(n_i)}$ [‡].

As can be seen from the structure diagram in table 3.3, the IPM are generated first. In the main while loop, the number of performances n_i in the performance subsets is increased by one iteratively.

In the n_i-th iteration step, the discretized Pareto front for each performance combination $S_m^{(n_i)}$ with n_i elements is generated in four steps:

1. The discretized boundary of a Pareto front of the n_i-performance combination $S_m^{(n_i)}$ is looked up. It consists of discretized Pareto fronts that were generated in the previous iteration step.
2. Target trajectories are defined that intersect evenly with the Pareto front of the n_i-performance combination $S_m^{(n_i)}$.
3. Pareto-optimal performance vectors are computed on the inner part of the Pareto front for the combination $S_m^{(n_i)}$. One scalar optimization problem must be solved for each target trajectory.
4. The Pareto-optimal performance vectors on the discretized boundary are joined with those on the inner part of the Pareto front.

In the n_f-th iteration step, the performance set $S_m^{(n_i)}$ is equal to the complete performance set S_f. The final front for the set S_f is the desired discretized Pareto front for all n_f performances.

[‡] Assuming we have chosen some order in the elements of $S_m^{(n_i)}$.

Table 3.3: Structure diagram of the iterative Pareto front generation approach

Choose set of all performances $S_f = \{f_1, ...f_{n_f}\}$
for each $f_m \in S_f$
Run single-objective optimizations to find IPM $\mathbf{f}^{*m}$
Set $\overline{\delta\mathcal{F}}^{\{f_m\}} = \{\mathbf{f}^{*m}\}$;
Set n_i=2
while $n_i \leq n_f$
for each n_i-combination $S_m^{(n_i)} \in C^{(n_i)}(S_f)$
Look up boundary $\delta\mathcal{F}_B^{S_m^{(n_i)}}$ from the results of the previous iteration.
Define target trajectories to populate inner parts of Pareto front $\delta\mathcal{F}^{S_m^{(n_i)}}$ (See sec. 3.6)
Compute performance vectors on inner parts of Pareto front $\delta\mathcal{F}^{S_m^{(n_i)}}$
Join Pareto-optimal performance vectors on boundary and inner parts
n_i=n_i+1

3.6 Definition of Target Trajectories to Populate Inner Parts of the Pareto Front

The step of populating the inner parts of the Pareto front with Pareto-optimal performance vectors was not detailed in the previous section. The iterative approach computes Pareto-optimal performance vectors on the boundary first. Therefore, a method is required that spreads the remaining Pareto-optimal performance vectors as evenly as possible inside the region defined the boundary. The method presented in the following uses parallel target trajectories. These are advantageous because the problem can be described on a projection plane normal to the direction of the target trajectories. A set of Pareto-optimal performance vectors is found by defining a set of evenly spaced so-called compromise weight vectors. These describe the relative importance of each performance. The compromise weight vectors are mapped on the base points by solving a linear program.

3.6.1 Problem Description for Parallel Target Trajectories

The target trajectory must run *'inside'* the boundary in order to intersect with the Pareto front. For parallel target trajectories, this problem can be best described by looking for a hyperplane which is normal to the direction of the target trajectories $\mathbf{v}$. We project the boundary and base point on this hyperplane. This is illustrated in figure 3.10 for three performances. In this case the hyperplane is a two-dimensional plane.

The target trajectory intersects with the Pareto front if the projected base point $\mathbf{b}^{'}$ is part of the projected Pareto front $\delta\mathcal{F}^{'}$. The area of the projected Pareto front, $\delta\mathcal{F}^{'}$, is enclosed by the boundary $\delta\mathcal{F}^{'}_{B}$. A discretized approximation of the boundary, $\overline{\delta\mathcal{F}}^{'}_{B}$, is available by using the presented iterative method. The problem of finding equally spaced Pareto-optimal performance vectors can then be simplified to:

Find K base points, $\mathbf{b}_k$, whose projections, $\mathbf{b}^{'}_{k}$, are evenly spread inside the known approximation of the boundary $\overline{\delta\mathcal{F}}^{'}_{B}$.

3.6.2 Calculation of the Direction of the Target Trajectories

The approach uses parallel search trajectories. Since the Minmax optimization tries to find a point in the opposite direction of $\mathbf{v}$ on the target trajectory, $\mathbf{v}$ must point towards worse performance values:

$$v_i = f_{i,max} - f_{i,min}, \quad \text{with} \quad f_{i,max} = \max_{j=1\ldots n_f} f_i^{*j}, \quad f_{i,min} = f_i^{*i} \tag{3.20}$$

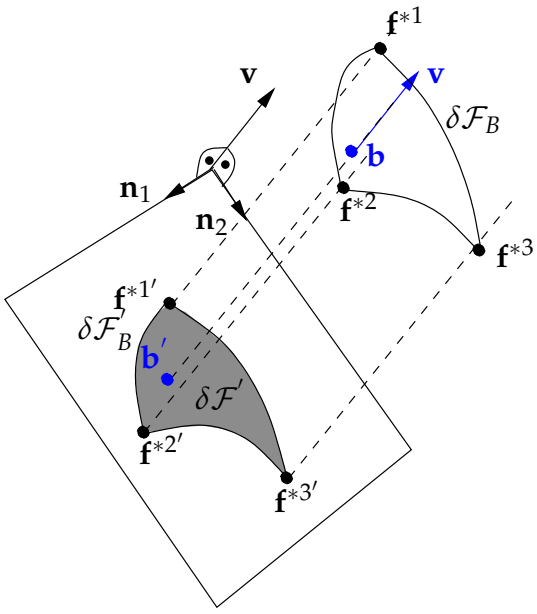


Figure 3.10: Projection to hyperplane normal to the direction of the target trajectories for three performances. The target trajectory defined by the base point $\mathbf{b}$ and direction $\mathbf{v}$ intersects with the Pareto front, because the projection of the base point $\mathbf{b}'$ on the plane normal to the search direction $\mathbf{v}$ is inside the projection of the boundary. The area enclosed by the boundary, highlighted in gray, is the projection of the Pareto front $\delta\mathcal{F}'$.

The direction vector is equal to the diagonal line of a hyperbox that contains the Individual Performance Minima. This formulation is independent of the used normalization of the performances. This means, a normalization of the performances does not change the optimization result in the unnormalized performance space.

3.6.3 Compromise Weight Vectors

We start from an abstract representation of the desired optimal compromises. All optimal compromises on the Pareto front are represented by an element of the following set of compromise weight vectors:

$$\mathcal{W} = \{\mathbf{w} \mid \mathbf{w} \geq 0 \wedge \sum_{i=1\ldots n_f} w_i = 1\} \tag{3.21}$$

Each so-called compromise weight vector $\mathbf{w} \in \mathcal{W}$ shows the relative importance of each performance in the desired compromise. This set of optimal compromises is mapped on actual Pareto-optimal performance vectors $\mathcal{W} \rightarrow \delta\mathcal{F}$.

We choose a set of evenly distributed weight vectors $\overline{\mathcal{W}}$, for which we compute the corresponding Pareto-optimal performance vectors:

$$\overline{\mathcal{W}} = \{\mathbf{w}_1 ... \mathbf{w}_k ... \mathbf{w}_K\} \rightarrow \overline{\delta\mathcal{F}} = \{\mathbf{f}_1^* ... \mathbf{f}_k^* ... \mathbf{f}_K^*\} \tag{3.22}$$

The density D of the weight vectors can be chosen by the designer. For a known density D, the compromise weight vectors are generated in a grid-like fashion. Table 3.4 illustrates the compromise weight vector set $\overline{\mathcal{W}}$ for three performances and the density D=3.

Table 3.4: $\overline{\mathcal{W}}$ for three performances $n_f = 3$ and density D=3

w_1	1	0	0	.75	.5	.25	0	0	0	.75	.5	.25	.5	.25	.25
w_2	0	1	0	.25	.5	.75	.75	.5	.25	0	0	0	.25	.5	.25
w_3	0	0	1	0	0	0	.25	.5	.75	.25	.5	.75	.25	.25	.5
	$\overline{\mathcal{W}}^{\{f_1\}}$	$\overline{\mathcal{W}}^{\{f_2\}}$	$\overline{\mathcal{W}}^{\{f_3\}}$	$\overline{\mathcal{W}}^{\{f_1,f_2\}}$			$\overline{\mathcal{W}}^{\{f_2,f_3\}}$			$\overline{\mathcal{W}}^{\{f_1,f_3\}}$			$\overline{\mathcal{W}}^{\{f_1,f_2,f_3\}}$		

3.6.4 Mapping of Compromise Weight Vectors on Base Points

The iterative Pareto front generation approach computes the Pareto-optimal performance vectors systematically in n_f steps. In the n_i-th step, the discretized Pareto front of each performance combinations $S_m^{(n_i)}$ with n_i elements is generated. To each combination belongs a subset $\overline{\mathcal{W}}^{S_m^{(n_i)}}$ of the compromise weight vectors:

$$\mathbf{w} \in \overline{\mathcal{W}}^{S_m^{(n_i)}} \quad \text{if} \quad w_i \begin{cases} \neq 0, & , f_i \in S_m^{(n_i)} \\ = 0, & , f_i \notin S_m^{(n_i)} \end{cases} \quad \text{with} \quad i = 1...n_f \tag{3.23}$$

Table 3.4 shows the distribution of weight vectors among the performance combinations. In the following, we will look at the generation of one Pareto front $\delta\mathcal{F}^{S_m^{(n_i)}}$ for a combination $S_m^{(n_i)}$. For simplicity, we omit the superscript $S_m^{(n_i)}$.

Each $\mathbf{w}_k$ is first mapped to a base point $\mathbf{b}_k$. The base point $\mathbf{b}_k$ defines the target trajectory together with the fixed direction $\mathbf{v}$ according to figure 3.2. It is used to calculate the Pareto-optimal performance vector $\mathbf{f}_k^*$ by the Minmax method:

$$\mathbf{w}_k \mapsto \mathbf{b}_k \mapsto \mathbf{f}_k^* \tag{3.24}$$

In the following, we look at the mapping of the compromise weight vectors $\mathbf{w}_k$ to the base points $\mathbf{b}_k$: The projected base point $\mathbf{b}_k'$ must be located inside the projection

of the boundary $\overline{\delta\mathcal{F}}_B$ on the plane normal to the direction of the search trajectory. The discrete boundary $\overline{\delta\mathcal{F}}_B$ was already calculated in previous iterations. The compromise weight vectors for points on $\overline{\delta\mathcal{F}}_B$ are combined in a set $\overline{\mathcal{W}}_B$. The mapping is illustrated in figure 3.11 for the case of three performances and density D=3 (See table 3.4).

The boundary compromise weight vectors in $\overline{\mathcal{W}}_B$ and the boundary Pareto-optimal performance vectors in $\overline{\delta\mathcal{F}}_B$ are combined column-wise in two matrices $\mathbf{W}_B$ and $\mathbf{F}_B$ with same ordering:

$$\underbrace{\overline{\mathcal{W}}_B = \{\mathbf{w}_{B,1}, ...\mathbf{w}_{B,l}, ...\mathbf{w}_{B,L}\}}_{\mathbf{W}_B=[\mathbf{w}_{B,1}...\mathbf{w}_{B,l}...\mathbf{w}_{B,L}]}; \quad \underbrace{\overline{\delta\mathcal{F}}_B = \{\mathbf{f}^*_{B,1}, ...\mathbf{f}^*_{B,l}, ...\mathbf{f}^*_{B,L}\}}_{\mathbf{F}_B=[\mathbf{f}^*_{B,1}...\mathbf{f}^*_{B,l}...\mathbf{f}^*_{B,L}]} \tag{3.25}$$

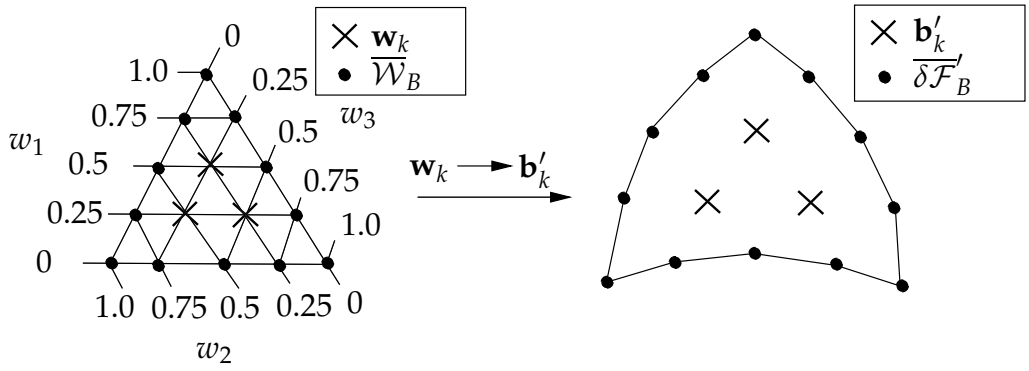


Figure 3.11: Mapping of weight vectors on base points. *Left*, illustration of the grid-like structure of the weight vectors for the case n_f=3 and D=3. The lines correspond to a constant value of one weight component. *Right*, base points and boundary in the plane normal to the direction of the target trajectories.

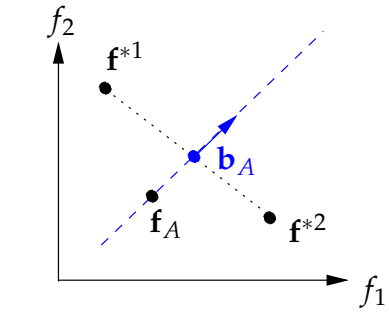
A new base point, $\mathbf{b}_k$, is generated as a linear combination of the boundary Pareto-optimal performance vectors in $\overline{\delta\mathcal{F}}_B$ by representing its compromise weight vector $\mathbf{w}_k$ as a linear combination of the boundary weight vectors in $\overline{\mathcal{W}}_B$:

$$\mathbf{w}_k = \mathbf{W}_B\,\boldsymbol{\rho}_k = \sum_l \mathbf{w}_{B,l}\,\rho_{k,l} \quad \mapsto \quad \mathbf{b}_k = \mathbf{F}_B\,\boldsymbol{\rho}_k = \sum_l \mathbf{f}^*_{B,l}\,\rho_{k,l} \tag{3.26}$$

The vector $\boldsymbol{\rho}_k$ holds the weights of the linear combination. The same linear combination is applied to $\mathbf{F}_B$ to obtain $\mathbf{b}_k$. Figure 3.12 illustrates the relation of the vectors $\boldsymbol{\rho}_k$, $\mathbf{w}_k$ and $\mathbf{b}_k$. In the example with two performances, the boundary consists of the two known IPM. A base point $\mathbf{b}_A$ is placed in the middle of both IPM, which is equal to the linear combination of both IPM weighting each with a factor $\rho_{A,1} = a_{A,2} = 0.5$. The corresponding vector of relative importance $\mathbf{w}_A$ is found by forming the linear combination over the vectors $\mathbf{w}_{*1}$ and $\mathbf{w}_{*2}$ of the IPM. The resulting weight vector

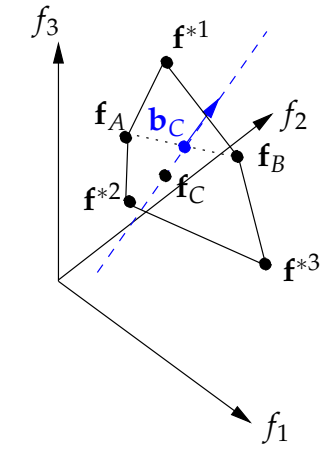
$\mathbf{w}_A$ that corresponds to $\mathbf{b}_A$ shows, that both performances are given equal relative importance. Clearly, the target trajectory aims at such a compromise, since it includes performance vectors, which have about the same distance from the two IPM.

In the example with three performances, the base point $\mathbf{b}_C$ is now placed in between two points $\mathbf{f}_A$ and $\mathbf{f}_B$ of the boundary. Forming the linear combination over their weight vectors $\mathbf{w}_A$ and $\mathbf{w}_B$, we obtain $\mathbf{w}_C$. The generated base point $\mathbf{b}_C$ corresponds to a double as high relative importance of f_1 compared to f_2 and f_3.



$$\mathbf{b}_A = \rho_{A,1}\mathbf{f}^{*1} + \rho_{A,2}\mathbf{f}^{*2}$$
$$\mathbf{b}_A = 0.5\mathbf{f}^{*1} + 0.5\mathbf{f}^{*2}$$
$$\mathbf{w}_A = \rho_{A,1}\mathbf{w}_{*1} + \rho_{A,2}\mathbf{w}_{*2}$$
$$\mathbf{w}_A = 0.5\mathbf{w}_{*1} + 0.5\mathbf{w}_{*2}$$

$$\mathbf{w}_{*1} = \begin{bmatrix} 1 \\ 0 \end{bmatrix} \quad \mathbf{w}_{*2} = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \quad \mathbf{w}_A = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}$$



$$\mathbf{b}_C = \rho_{C,1}\mathbf{f}_A + \rho_{C,2}\mathbf{f}_B$$
$$\mathbf{b}_C = 0.5\mathbf{f}_A + 0.5\mathbf{f}_B$$
$$\mathbf{w}_C = \rho_{C,1}\mathbf{w}_A + \rho_{C,2}\mathbf{w}_B$$
$$\mathbf{w}_C = 0.5\mathbf{w}_A + 0.5\mathbf{w}_B$$

$$\mathbf{w}_A = \begin{bmatrix} 0.5 \\ 0.5 \\ 0 \end{bmatrix} \quad \mathbf{w}_B = \begin{bmatrix} 0.5 \\ 0 \\ 0.5 \end{bmatrix} \quad \mathbf{w}_C = \begin{bmatrix} 0.5 \\ 0.25 \\ 0.25 \end{bmatrix}$$

Figure 3.12: Relation between the vectors $\boldsymbol{\rho}$, $\mathbf{w}$ and $\mathbf{b}$. *Upper*, example with two performances. *Lower*, example with three performances.

3.6.5 Calculation of the Base Points by Linear Programming

The system of linear equalities $\mathbf{w}_k = \mathbf{W}_B\,\boldsymbol{\rho}_k$ in the mapping in (3.26) becomes underdetermined for more than two performances. There are degrees of freedom for finding a weight vector $\mathbf{w}_k$ as linear combination of the boundary weight vectors in $\overline{\mathcal{W}}_B$. A vector $\mathbf{a}_k$ can be chosen by selecting a set of basis vectors from $\overline{\mathcal{W}}_B$ to represent $\mathbf{w}_k$. A similar problem is described in [JP07, DT05]. In these works, the target is to obtain a solution for $\boldsymbol{\rho}_k$ with minimal number of non-zero entries (sparsest vector) by solving a linear program that minimizes the L1 norm of the vector $\boldsymbol{\rho}_k$. In our case the

target is to select the vectors $\boldsymbol{\rho}_k$ such that the resulting base points are evenly spread inside the boundary.

One solution is to chose the vector $\boldsymbol{\rho}_k$ in the following way: Those Pareto-optimal performance vectors on the boundary are preferred in the linear combination that have a *'similar'* performance compromise compared to the compromise defined by $\mathbf{w}_k$. The similarity of the compromise weight vector $\mathbf{w}_k$ and a compromise weight vector $\mathbf{w}_{B,l}$ of a boundary point is measured with the L_1 norm of the difference:

$$d_{k,l} = || \mathbf{w}_{B,l} - \mathbf{w}_k ||_1 = \sum_{i=1}^{n_f} | w_{l,i}^{B_n} - w_{k,i} | \tag{3.27}$$

The value $d_{k,l}$ is also a measurement for the distance of two points in the performance space. The more similar the compromise, the closer the two Pareto-optimal performance vectors will be located on the discretized Pareto front. The main idea of this approach is the following:

By preferring *'close'* points on the boundary, the base points are placed according to the *'nearest'* boundary. This is advantageous, because it will automatically assure, that target trajectories for compromises, which are near to a concave boundary are shifted to the interior of the Pareto front. Compromises near convex parts are shifted to the exterior, which leads finally to an even spread of the base points.

The vector $\boldsymbol{\rho}_k^*$, which prefers *'close'* points on the boundary, is found by solving the following linear program (LP):

$$\boldsymbol{\rho}_k^* = \underset{\boldsymbol{\rho}_k}{\operatorname{argmin}} \, \mathbf{d}_k^{\mathrm{T}} \, \boldsymbol{\rho}_k \quad \text{s.t.} \quad \mathbf{W}_B \, \boldsymbol{\rho}_k = \mathbf{w}_k \ \wedge \ \boldsymbol{\rho}_k \geq \mathbf{0} \tag{3.28}$$

The objective function of the linear program is the L1 norm of the vector $\boldsymbol{\rho}_k$ weighted by the distances in the vector $\mathbf{d}_k$. The optimal vector $\boldsymbol{\rho}_k^*$ has large values $\boldsymbol{\rho}_{k,l}^*$ for *'close'* Pareto-optimal performance vectors $\mathbf{f}_{B,l}$ with small difference value $d_{k,l}$. The linear program does not always have a unique optimal vector $\boldsymbol{\rho}_k^*$. Each of the alternative optima is equally suited.

The base point $\mathbf{b}_k$ is found by the linear combination of all Pareto-optimal performance vectors $\mathbf{f}_{B,l}$ using the optimal vector $\boldsymbol{\rho}_k^*$ that is resulting from the LP:

$$\mathbf{b}_k = \mathbf{F}_B \, \boldsymbol{\rho}_k^* \tag{3.29}$$

The resulting base points are shown for different densities D and three performances in figure 3.13. The linear combination vectors $\boldsymbol{\rho}_k^*$ are illustrated by the dashed lines. A connection with a dashed line shows that the boundary point has a non-zero in the linear combination to calculate the base point.

This linear programming based approach can be easily implemented. Its main advantage is that the base points are obtained with relatively low computational costs. Considering the computational effort to find a Pareto-optimal performance vector,

the computational cost to solve the linear program is insignificant. The solutions $\boldsymbol{\rho}_k^*$ to (3.28) lead to a sufficiently even spread of the base points inside the area enclosed by the projected boundary.

Calculating base points with a perfect spread, e.g., by maximizing the minimal distances in between all base points and all boundary points is a complex optimization task. The complexity is increased since the base points must be located inside the projected area enclosed by the boundary. This area can be concave and, therefore, not described by a set of linear inequalities. Especially in high dimensions, a mathematical description of the projected area becomes difficult. In contrast, the LP-based method can be used to determine base points for Pareto fronts with more than three performances. In the investigated cases, the linear combination vector $\boldsymbol{\rho}_k^*$ mapped the boundary in the compromise weight vector space to the discretized boundary in the performance space. This shows empirically that the method creates no base points outside the boundary. All target trajectories intersect with the Pareto front.

The complete algorithm using the iterative Pareto generation approach is given in section 3.7 after comparing the approach to the state-of-the-art NBI approach.

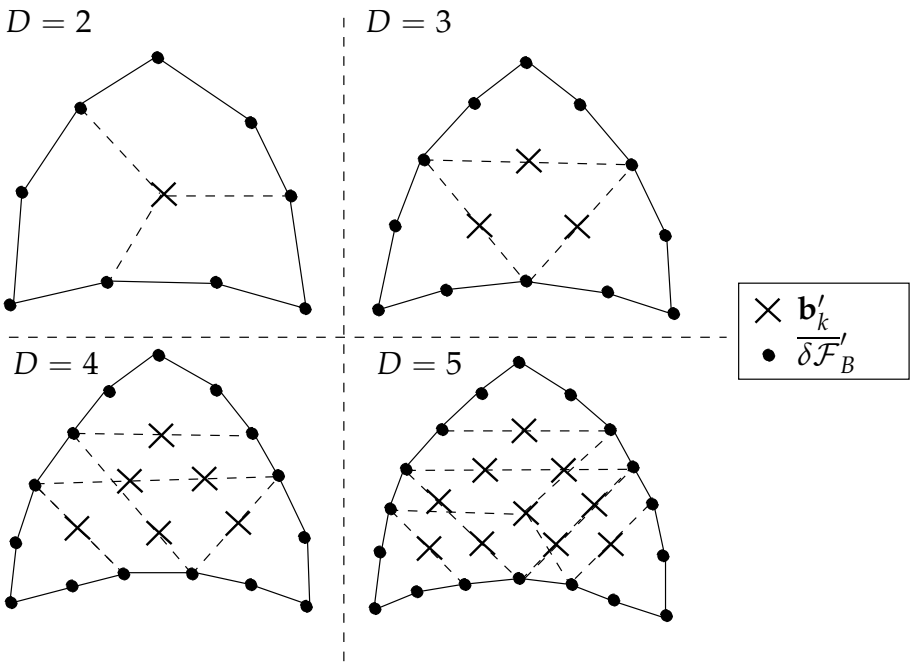


Figure 3.13: Linear combinations to generate base points found with linear programming. Base points and boundary are shown in the plane normal to the direction of the target trajectories.

3.6.6 Comparison to Normal-Boundary Intersection

It can be shown that the NBI approach is a special case of the presented approach. If the elements of the discrete boundary are solely the Individual Performance Minima (IPM), the matrices that describe the boundary are given as

$$\mathbf{F}_B = [\mathbf{f}^{*1} ... \mathbf{f}^{*n_f}]; \quad \mathbf{W}_B = \mathbf{I} \tag{3.30}$$

whereas $\mathbf{I}$ is the identity matrix. Looking at (3.26), the unique solution to $\mathbf{w}_k = \mathbf{W}_B \, \boldsymbol{\rho}_k$ is $\boldsymbol{\rho}_k^* = \mathbf{w}_k$. This means a base point $\mathbf{b}_k$ can be directly calculated to

$$\mathbf{b}_k = \mathbf{F}_B \, \boldsymbol{\rho}_k^* = \mathbf{F}_B \mathbf{w}_k = \sum_{i=1 ... n_f} w_{k,i} \, \mathbf{f}^{*i} \tag{3.31}$$

without solving the linear program. This is equal to the generation of the base points on the convex hull of individual minima (CHIM) as it is described for the NBI approach [DD98, Ste05, SGA07].

The presented approach is equal to the NBI approach if the discretized boundary only includes IPM. This is only the case for the generation of Pareto fronts for two performances. For a higher number of performances, the presented approach considers the location of more Pareto-optimal performance vectors on the boundary than just the IPM.

3.7 Iterative Pareto Front Generation Approach with Parallel Target Trajectories

3.7.1 Structure

The structure diagram of the approach for the iterative Pareto front generation with parallel search trajectories is shown in table 3.5. It combines the iterative Pareto front generation approach with the method to define parallel search trajectories by linear programming.

3.7.2 Complexity

The number of Pareto fronts that must be found to generate the trade-off limits grows fast with the number of performances n_f, since the number of performance combinations increases combinatorically. The number M of n_i-performance combinations out of n_f performances is given as:

$$M(n_i, n_f) = \begin{pmatrix} n_f \\ n_i \end{pmatrix} \tag{3.32}$$

The overall number of Pareto-optimal performance vectors to be generated depends on the desired density D of Pareto-optimal performance vectors. The higher the density, the more Pareto-optimal performance vectors are used to describe the discretized Pareto front.

The total number of Pareto-optimal performance vectors K, which will be generated, depends on the number of performances n_f and the density D:

$$K = \begin{pmatrix} n_f + D \\ n_f - 1 \end{pmatrix} = \frac{(n_f + D)!}{(n_f - 1)!(D + 1)!} \tag{3.33}$$

One scalar optimization problem is solved for each Pareto-optimal performance vector. The number of scalar optimization runs K increases fast with the number of considered objectives n_f and the density of the solutions D. Therefore, an efficient optimization algorithm to solve the scalar optimization problems is required. Such an algorithm is described in chapter 4.

Table 3.5: Structure diagram of the iterative Pareto front generation approach with LP approach to populate inner parts of the Pareto front

Choose set of all performances $S_f = \{f_1, ...f_{n_f}\}$ and density D
Generate set of all compromise weight vectors $\overline{\mathcal{W}}$ (Sec. 3.6.3)
for each $f_m \in S_f$
Run single-objective optimizations to find IPM $\mathbf{f}^{*m}$
Set $\overline{\delta\mathcal{F}}^{S_m^{(1)}} = \{\mathbf{f}^{*m}\}$;
n_i=2;
while $n_i \leq n_f$
for each n_i-performance combination $S_m^{(n_i)} \in C^{(n_i)}(S_G)$:
Calculate direction vector of the target trajectories $\mathbf{v}^{S_m^{(n_i)}}$; (Eq. (3.20))
Look up boundary $\overline{\delta\mathcal{F}}_B^{S_m^{(n_i)}}$ and $\overline{\mathcal{W}}_B^{S_m^{(n_i)}}$ from the results of the previous iteration; (Eq. (3.25))
Look up compromise weight vector subset $\overline{\mathcal{W}}^{S_m^{(n_i)}}$ (Eq. (3.23))
for each compromise weight vector $\mathbf{w}_l^{S_m^{(n_i)}} \in \overline{\mathcal{W}}^{S_m^{(n_i)}}$:
Calculate $\boldsymbol{\rho}_l^{*S_m^{(n_i)}}$ via LP (Eq. (3.28))
Calculate base point $\mathbf{b}_l^{S_m^{(n_i)}} = \boldsymbol{\rho}_l^{*S_m^{(n_i)}} \mathbf{F}_B^{S_m^{(n_i)}}$ (Eq. (3.29))
Run scalar optimization with target trajectory $\mathbf{v}_l^{S_m^{(n_i)}}$ and $\mathbf{b}_l^{S_m^{(n_i)}}$ (Sec. 3.2)
Insert found solution $\mathbf{f}_l^{S_m^{(n_i)}}$ to $\overline{\delta\mathcal{F}}^{S_m^{(n_i)}}$
Insert boundary points $\overline{\delta\mathcal{F}}_B^{S_m^{(n_i)}}$ in $\overline{\delta\mathcal{F}}^{S_m^{(n_i)}}$ to construct Pareto front $\delta\mathcal{F}^{S_m^{(n_i)}}$ with boundary
n_i=n_i+1

3.8 Numerical example

The difference of the presented approach compared to the state-of-the-art can be illustrated with a numerical example. We solve the following multi-objective optimization problem:

$$\begin{aligned} &\min_{\mathbf{p}} [f_1 = d_1 \quad f_2 = d_2 \quad f_3 = d_3]^T \\ s.t. \quad &0 \leq d_1 \leq 10, \quad 0 \leq d_2 \leq 10, \quad 0 \leq d_3 \leq 10 \\ &d_1^2 + d_2^2 + d_3 \geq 119, \quad d_1 d_2 d_3 \geq 300 \end{aligned} \tag{3.34}$$

The IPM of the performances are:

$$\mathbf{f}^{*1} = [3\ 10\ 10]^{\mathrm{T}} \quad \mathbf{f}^{*2} = [10\ 3\ 10]^{\mathrm{T}} \quad \mathbf{f}^{*3} = [10\ 10\ 3]^{\mathrm{T}} \tag{3.35}$$

The Pareto fronts of the performance pairs can be calculated analytically to:

$$\delta f^{\{f_1,f_2\}} = \{\, \mathbf{f} \mid \quad f_1 = \sqrt{109 - f_2};\ f_3 = 10; \quad 3 \leq f_2 \leq 10\} \tag{3.36}$$

$$\delta f^{\{f_1,f_3\}} = \{\, \mathbf{f} \mid \quad f_1 = 30/f_3;\ f_2 = 10; \quad 3 \leq f_3 \leq 10\} \tag{3.37}$$

$$\delta f^{\{f_2,f_3\}} = \{\, \mathbf{f} \mid \quad f_2 = 30/f_3;\ f_1 = 10; \quad 3 \leq f_3 \leq 10\} \tag{3.38}$$

We compare the novel iterative approach with the NBI approach and with the NBI approach with added boundary that is suggested in [Ste05, SGA07]. Figure 3.14 shows the base points and available boundary points in the projection plane normal to the direction of the search trajectories for all three methods.

The NBI approach generates the Individual Performance Minima (IPM) first. The base points are defined based on the location of the IPM. Some of the base points are located outside the concave boundary in the projection plane. Additionally, no base points are generated on the convex boundaries. Figure 3.15 shows the resulting discretized Pareto front. The Pareto-optimal performance vectors do not cover the convex boundaries and their spread is uneven near the concave boundary.

The NBI approach with added boundary calculates the performance vectors on the boundary by generating the IPM and dicretized Pareto fronts of the performance pairs first. Further base points are defined equally to the NBI approach based on the location of the IPM. Equally to the NBI approach, some base points are located outside the concave boundary. The convex boundary is captured by the previously generated discretized Pareto front of the performance pairs. The resulting discretized Pareto front is shown in figure 3.16. The points cluster around the boundaries. The points on the concave boundary are obtained twice such that some optimization runs are unnecessary. More optimization runs are required than for the other methods, since the added boundary points do not replace any of the standard NBI points.

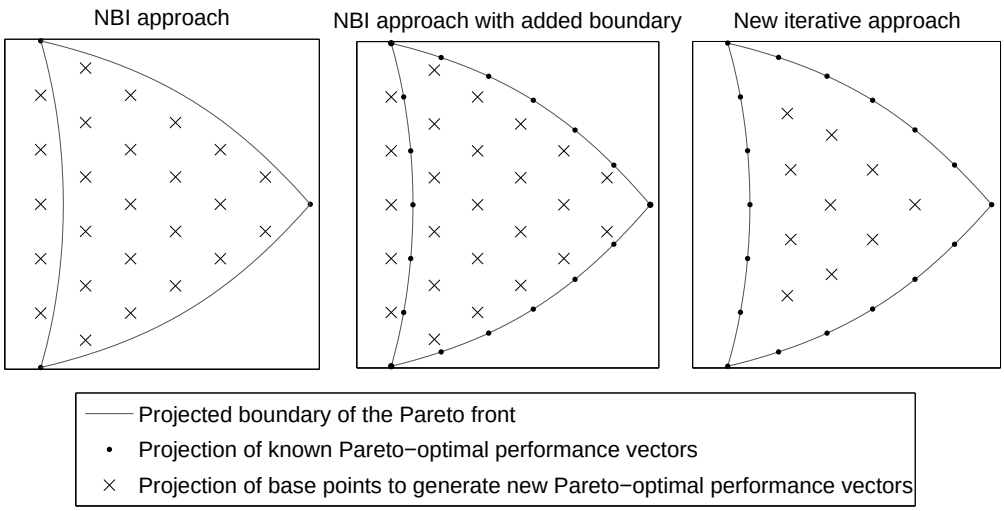


Figure 3.14: Boundary, base points and known Pareto-optimal performance vectors projected on the plane normal to the search direction for the numerical example.

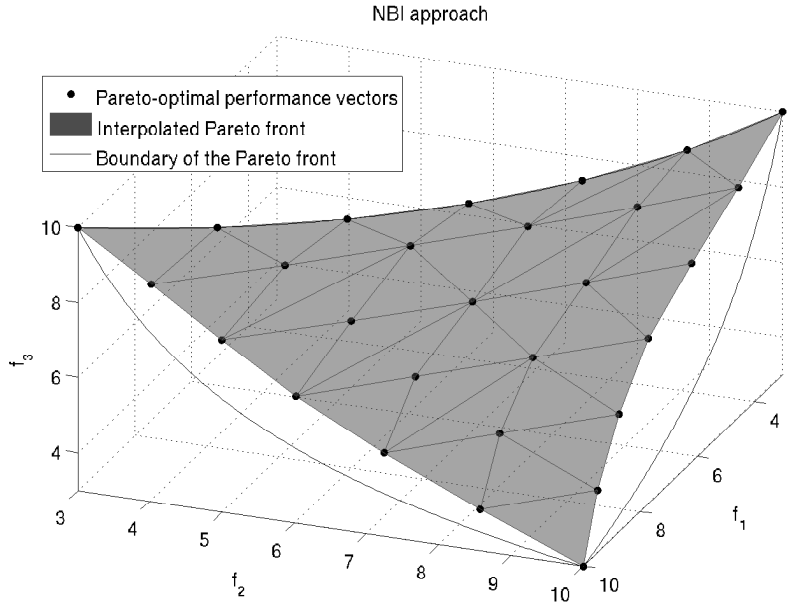


Figure 3.15: Discretized Pareto front obtained by NBI approach

The novel iterative approach generates the IPM and dicretized Pareto fronts of the performance pairs first. The location of the base points is set by a linear combination of the *'nearest'* boundary points. The base points cover the interior of the Pareto front evenly. The obtained discretized Pareto front is shown in figure 3.17. It covers all parts inside the boundaries and features an even spread of the Pareto-optimal performance vectors.

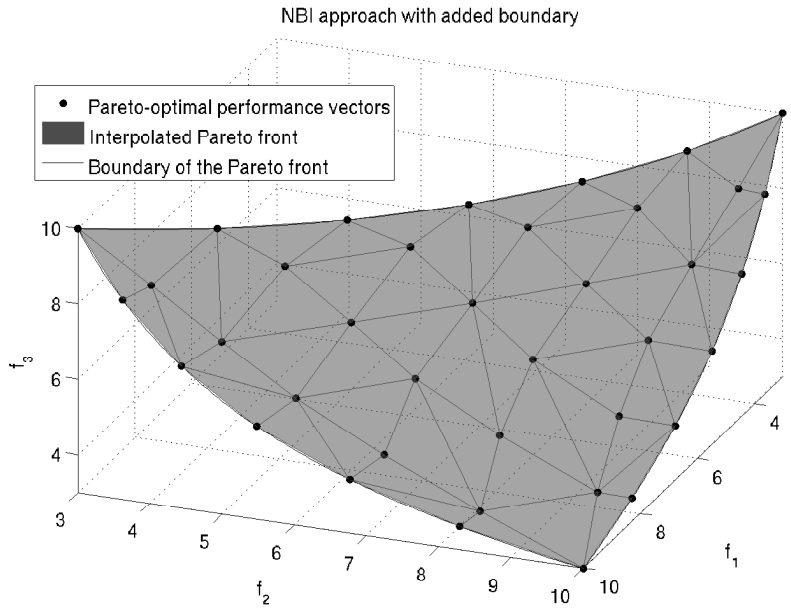


Figure 3.16: Discretized Pareto front obtained by NBI approach with boundary added

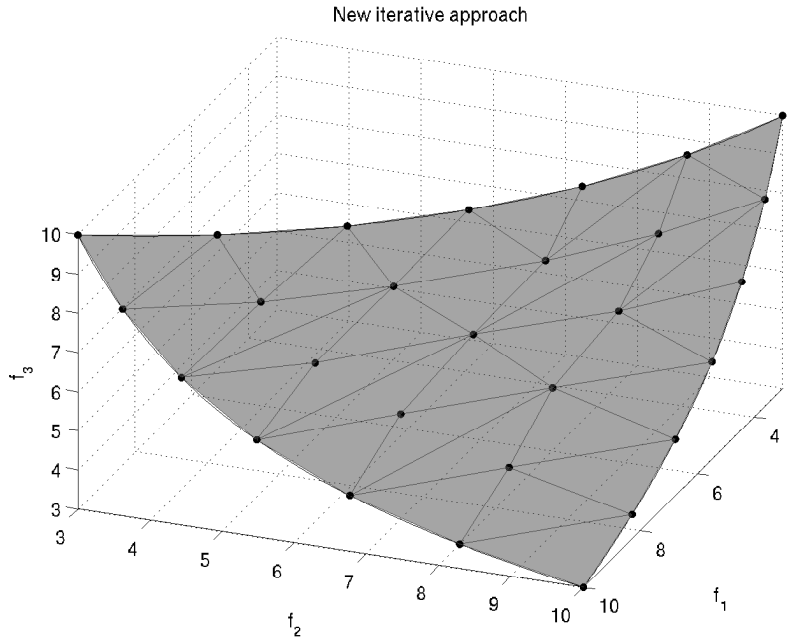


Figure 3.17: Discretized Pareto front obtained by new iterative approach

3.9 Summary

This section summarizes the chapter. The multi-objective optimization problem is transformed into a scalar optimization problem using the Minmax- and Goal-attainment method. These methods are equivalent formulations. The Minmax and GA methods target at a compromise, which is located on the target trajectory in the performance space if the target trajectory intersects with the Pareto front.

The trade-offs in technical applications are limited. No performance can be improved unlimitedly at the cost of the others. The Pareto front is bounded by the so-called trade-off limits. An iterative approach was shown that was able to generate Pareto-optimal performance vectors on the boundary of the Pareto front. These show the total extent of the Pareto front. The approach calculates the Pareto front of any performance subset (combination) and increases the number of performances in the combinations by one in each iteration. A way to define parallel target trajectories by solving a linear program was shown. It targets at an nearly even spread of the Pareto-optimal performance vectors inside the region defined by the boundary.

Chapter 4

Wavefront Feasible Sequential Quadratic Programming

4.1 Introduction

It was shown in chapter 3 that Pareto optimization consists of systematically solving a set of scalar constrained nonlinear optimization problems (CNOP). Each CNOP targets at a different Pareto-optimal performance vector. In this chapter, a gradient-based, deterministic optimization algorithm called *Wavefront Feasible Sequential Quadratic Programming* algorithm is presented that computes the solutions to these scalar CNOPs.

4.1.1 Standard Form of the Scalar Constrained Nonlinear Optimization Problem (CNOP)

The scalar CNOP is considered in the following standard form:

$$\min_{\mathbf{x}} o(\mathbf{x}) \quad \text{s.t.} \quad \mathbf{a}_{ineq}(\mathbf{x}) \leq \mathbf{0} \tag{4.1}$$

The candidate vector $\mathbf{x}$ includes the parameters of the optimization, $o(\mathbf{x})$ is the objective function and $\mathbf{a}_{ineq}(\mathbf{x})$ are inequality constraint functions.

4.1.2 Minmax and GA Formulation in the Standard CNOP Form

For the Pareto optimization of analog circuits, one CNOP is solved to obtain one Pareto-optimal performance vector. The multi-objective optimization problem in

(2.16) was transformed to a scalar CNOP using the Minmax and Goal-attainment formulations that are given in (3.4) and (3.8). For the Minmax formulation, we obtain the standard form of the CNOP by setting:

$$\mathbf{x} = \mathbf{d}; \qquad o(\mathbf{x}) = \max_{i=1..n_f} \frac{f_i(\mathbf{d}) - b_i}{v_i}; \qquad \mathbf{a}_{ineq}(\mathbf{x}) = \begin{bmatrix} \mathbf{d}_L - \mathbf{d} \\ \mathbf{d} - \mathbf{d}_U \\ -\mathbf{c}(\mathbf{d}) \end{bmatrix}; \tag{4.2}$$

For the GA method, we obtain the standard form of the CNOP by setting:

$$\mathbf{x} = \begin{bmatrix} \mathbf{d} \\ t \end{bmatrix}; \qquad o(\mathbf{x}) = t; \qquad \mathbf{a}_{ineq}(\mathbf{x}) = \begin{bmatrix} \mathbf{d}_L - \mathbf{d} \\ \mathbf{d} - \mathbf{d}_U \\ -\mathbf{c}(\mathbf{d}) \\ \mathbf{f}(\mathbf{d}) - \mathbf{v}t - \mathbf{b} \end{bmatrix}; \tag{4.3}$$

The inequality constraints include the design parameter bounds, see (2.2), sizing rules, see (2.7), and for the GA method the GA constraints, see (3.8).

4.1.3 SQP Optimization Algorithms

A numerical optimization algorithm should be able to compute the solution of the CNOP reliable and fast. A very effective, standard numerical optimization algorithm to solve a scalar CNOP is Sequential Quadratic Programming (SQP). The various different implementations of SQP can be roughly classified in trust-region based and line search based algorithms [Fle87, Spe98, NW99].

The optimization time in analog circuit sizing depends mainly on the number of candidates that are evaluated during the optimization and the duration of the required simulations to evaluate the circuit performances for one candidate. For example, circuit performances that are evaluated with a transient (TR) analysis simulate usually longer than performances evaluated with a AC analysis. The optimization can be speeded up by either reducing the total number of candidates required to compute the solution or by running simulations in parallel on different CPUs.

The reliability of the optimization algorithm is an important issue in analog circuit sizing. The optimization returns the candidate vector $\mathbf{x}^*$ that has the smallest objective function value compared to all other candidates encountered during the optimization. For a reliable optimization algorithm it should hold:

- The returned candidate $\mathbf{x}^*$ must be feasible with no violations in the constraints $\mathbf{a}_{ineq}(\mathbf{x}^*)$.
- The objective function value $o(\mathbf{x}^*)$ should be the global minimum of the objective function.
- The optimization should find the candidate $\mathbf{x}^*$ in a limited number of iterations.

Standard SQP implementations feature the following three main issues when applied to the Pareto optimization of analog circuits:

1. Many standard SQP algorithms accept violations in the inequality constraints for a sufficient decrease of the objective function. This causes great reliability issues in analog sizing because the performances become very nonlinear outside of the valid design parameter space, e.g., caused by transistors running out of saturation [LGXP07]. This nonlinearity can cause the optimization algorithm to get stuck at an infeasible candidate with constraint violations. In this case, the solution is no valid sizing of the circuit.
2. The speed of standard SQP implementations is limited because the candidates are evaluated sequentially. Parallel simulations to obtain the performances for a set of candidates is usually not supported.
3. Another reliability issue is caused by the fact that the standard SQP method is a local optimizer converging to local optima.

A new *Wavefront Feasible Sequential Quadratic Optimization (FSQP)* algorithm is presented in this chapter. It addresses the above issues and features the following improvements:

1. It implements a feasible SQP (FSQP) algorithm. It does not accept any candidates with constraint violations. A so-called feasibility filter removes infeasible candidates.
2. Computation time is speeded up by making strong use of parallel simulations on multiple CPUs. This is motivated by the great availability of multi-core PCs and multi-PC networks.
3. The algorithm is a so-called *Wavefront* FSQP algorithm, because the CNOPs to obtain different Pareto-optimal performance vectors are computed simultaneously. The global convergence of the optimization algorithm is improved by exchanging candidates between the different CNOP. If one CNOP converges to a local minimum, it is supplied with alternative candidates from the remaining CNOPs.

Different FSQP approaches have been published, e.g., in [LT00, WT04]. The presented method is a line search based method. It implements a modified version of the algorithm presented in [LT00]. As was suggested in [LT00], the presented approach features two measures to improve the feasibility of the candidates: A tilting of the search direction as well as second-order correction.

Additionally, the presented Wavefront FSQP optimization algorithm makes use of the equivalence of the Minmax and GA formulation of the CNOP, shown in (4.2) and (4.3). Applying solely the Minmax formulation may cause an issue during the calculation of the search direction due to the discontinuous objective function. Applying solely the GA method may lead to numerical problems concerning the feasibility of candidates. Some candidates that constitute valid sizings of the circuit may be disregarded because of violations in the additional GA constraints. This may be caused

by a poor choice of the additional parameter t due to the inaccuracy of the quadratic model created to compute the search direction in a SQP algorithm. These issues can be avoided by applying both formulations during the optimization.

4.1.4 Structure

The chapter is structured as follows: First, basics of sequential quadratic programming are given in section 4.2. Then, the FSQP algorithm is introduced in section 4.3 with a discussion of the tilting of the search direction and the second-order correction. The Wavefront FSQP algorithm that features the exchange of solutions is presented in section 4.4. Finally, in section 4.5, the application of the equivalence of the Minmax and GA method is discussed. Section 4.6 summarizes the chapter.

4.2 Basics of Sequential Quadratic Programming

The flow chart of a line search sequential quadratic programming algorithm is shown in figure 4.1. SQP is an iterative optimization method. The algorithm starts from an initial candidate $\mathbf{x}^{(0)}$. Three major steps are conducted in each iteration:

1. A local quadratic model is derived around a currently selected candidate $\mathbf{x}^{(r)}$.
2. A search direction $\Delta\mathbf{x}^{*(r)}$ to look for better candidates is computed by solving a quadratic programming (QP) problem.
3. A line search along $\Delta\mathbf{x}^{*(r)}$ is conducted to find a better candidate. The quality of a candidate is measured by a merit function, which considers the objective function and constraint values. If a better candidate is found, it is set to be the selected candidate $\mathbf{x}^{(r+1)}$ for the next iteration.

The optimization stops if no better candidate could be found during the line search or the maximum number of iterations was exceeded. The final, selected candidate is returned as the solution $\mathbf{x}^*$ for the CNOP. These three steps are discussed in more detail in the following.

4.2.1 Initialization and Update of the Quadratic Model

For the selected candidate $\mathbf{x}^{(r)}$ of the current iteration, the quadratic model is given as:

$$\mathbf{g}(\mathbf{x}^{(r)}) \quad : \quad \text{Estimate of } \nabla_{\mathbf{x}}\, o(\mathbf{x}) \mid_{\mathbf{x}=\mathbf{x}^{(r)}}; \tag{4.4}$$

$$\mathbf{A}_{ineq}(\mathbf{x}^{(r)}) \quad : \quad \text{Estimate of } \nabla_{\mathbf{x}}\, \mathbf{a}_{ineq}(\mathbf{x}) \mid_{\mathbf{x}=\mathbf{x}^{(r)}}; \tag{4.5}$$

$$\mathbf{H}^{(r)} \quad : \quad \text{Estimate of the Hessian matrix of the Lagrange function} \tag{4.6}$$

The gradient of the objective function $\mathbf{g}(\mathbf{x}^{(r)})$, and the Jacobi matrix of the inequality constraints $\mathbf{A}_{ineq}(\mathbf{x}^{(r)})$ can be estimated by finite difference methods, see [NW99]. The Lagrange function is a function that includes the constraints and objective function. Its Hessian matrix is estimated by Quasi-Newton methods such as the SR1 or BFGS update [NW99, Fle87, Spe98]. The presented approach will apply the BFGS update that guarantees a positive definite Hessian matrix.

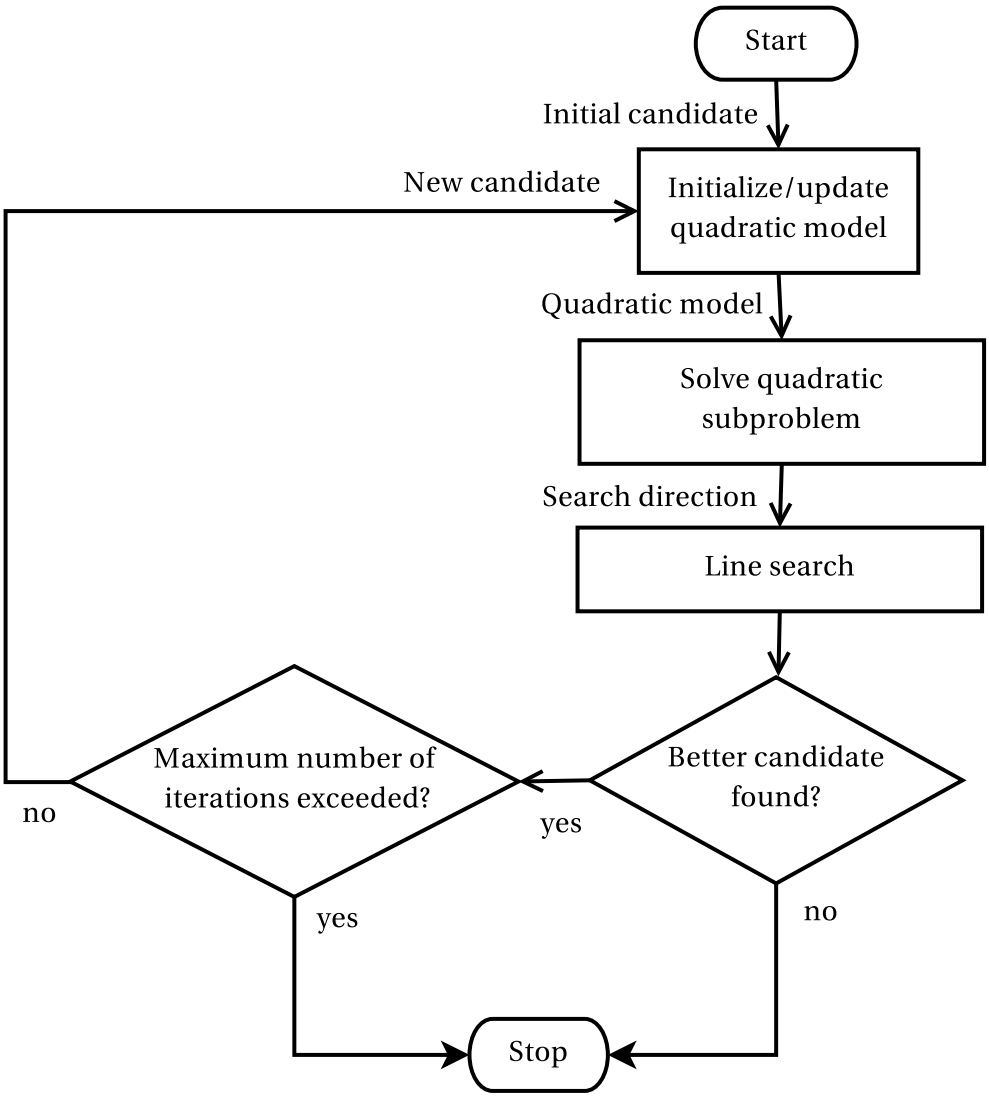


Figure 4.1: Flow chart of a line search sequential quadratic programming algorithm

4.2.2 The Quadratic Program (QP)

The search direction $\Delta\mathbf{x}^{*(r)}$ for the line search is obtained by solving the following quadratic programming (QP) problem:

$$\begin{aligned} \min_{\Delta\mathbf{x}^{(r)}} \quad & \mathbf{g}^{\mathrm{T}}(\mathbf{x}^{(r)})\ \Delta\mathbf{x}^{(r)} \quad + \quad \Delta\mathbf{x}^{(r)\mathrm{T}}\ \mathbf{H}^{(r)}\ \Delta\mathbf{x}^{(r)} \\ \text{s.t.} \quad & \mathbf{A}_{ineq}(\mathbf{x}^{(r)})\ \Delta\mathbf{x}^{(r)} \quad + \quad \mathbf{a}_{ineq}(\mathbf{x}^{(r)}) \quad \leq \quad \mathbf{0} \end{aligned} \tag{4.7}$$

A QP with inequality constraints can be solved by so-called active set methods [Fle87, Spe98, NW99].

4.2.3 Backtracking Line Search

The flow diagram of a backtracking line search method is shown in figure 4.2. It starts with a step size $\alpha = 1$. A new candidate in the search direction is found as:

$$\mathbf{x}^{(r+1)} = \mathbf{x}^{(r)} + \alpha \cdot \Delta\mathbf{x}^{*(r)} \tag{4.8}$$

The objective function and constraints are evaluated for the new candidate. This requires to simulate the circuit to obtain the performances and sizing rules. Finally, a so-called merit function is calculated for the new candidate $\mathbf{x}^{(r+1)}$ and compared to the merit function value of the old candidate $\mathbf{x}^{(r)}$. The merit function includes the objective function value and constraint values. Depending on the merit function, violations of the constraints might be accepted if enough improvement in the objective function was made. If sufficient improvement in the merit function was made, $\mathbf{x}^{(r+1)}$ is the selected candidate for the next iteration. Sufficient improvement is often checked by applying the Wolfe-condition [Fle87, NW99]. In this work, every improvement in the objective function is accepted. If there was no improvement, the step size α is reduced. The reduction can for example be made by a fixed factor, e.g. 0.5 in this work. A new candidate is generated using (4.8). Finally, a termination criteria assures the line search does not end up in a infinite loop if no improvement in the search direction is possible. It terminates as soon as the reduction of the step size leads to the step size being smaller than a given minimum step size.

4.3 Feasible SQP Algorithm

The flow diagram for the feasible sequential quadratic programming (FSQP) algorithm is shown in figure 4.3. The optimization algorithm solves two quadratic programming problems in each iteration: The standard QP in (4.7) and a tilted QP introduced in (4.9). The search direction obtained by the tilted QP differs from the one obtained from the standard QP. Both search directions are used to conduct a line

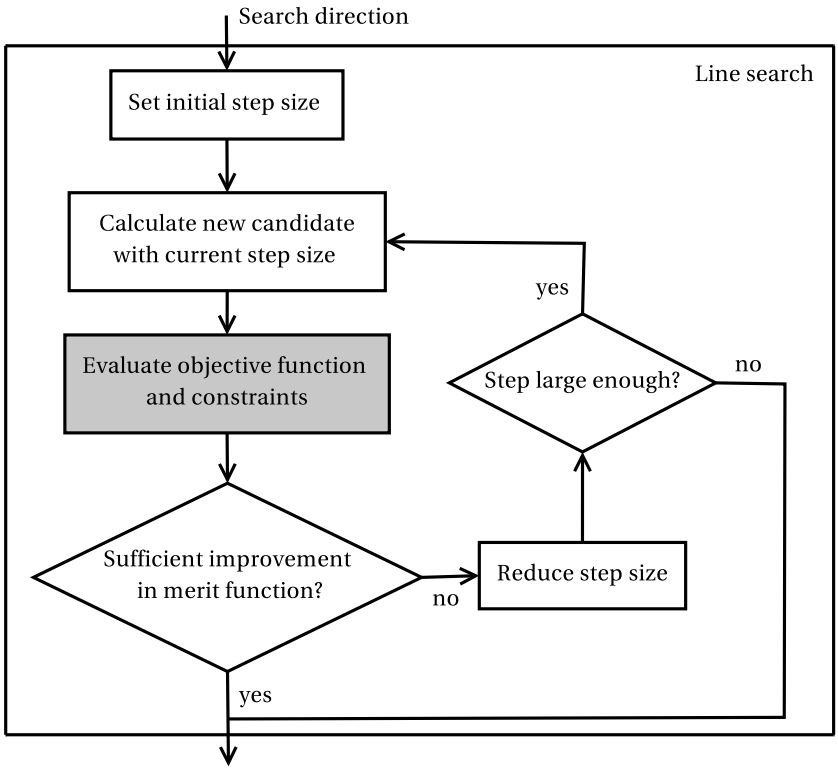


Figure 4.2: Flow chart of the line search of a SQP algorithm

search and look for better candidates. Additionally, a second-order correction step is applied to find new candidates if infeasible candidates are encountered during the optimization. The BFGS update is used as quasi-Newton method ([Spe98] p. 507). The FSQP algorithm is discussed in the following in more detail.

4.3.1 Tilted Quadratic Program

The tilting of the search direction applies a safety margin to those inequality constraints that are violated for the standard search direction. This safety margin leads to a shift or tilt of the search direction such that the obtained tilted search direction, $\Delta\mathbf{x}_t^{*(r)}$, differs from the standard search direction $\Delta\mathbf{x}^{*(r)}$. The tilt emphasizes feasibility in the inequality constraints at the cost of descent in the objective function.

The tilted search direction $\Delta\mathbf{x}_t^{*(r)}$ is computed by solving the following tilted quadratic programming problem [LT00]:

$$\begin{aligned} \min_{\Delta\mathbf{x}_t^{(r)},\gamma} \quad & \gamma + \quad \Delta\mathbf{x}_t^{(r)\mathrm{T}} \ \mathbf{H}^{(r)} \ \Delta\mathbf{x}_t^{(r)} \\ \text{s.t.} \quad & \mathbf{g}^{\mathrm{T}}(\mathbf{x}^{(r)}) \ \Delta\mathbf{x}_t^{(r)} \quad \leq \gamma \\ \wedge \quad & \mathbf{A}_{ineq}(\mathbf{x}^{(r)}) \ \Delta\mathbf{x}_t^{(r)} \quad + \quad \mathbf{a}_{ineq}(\mathbf{x}^{(r)}) \quad \leq \quad \gamma\,\boldsymbol{\eta}^{(r)} \end{aligned} \tag{4.9}$$

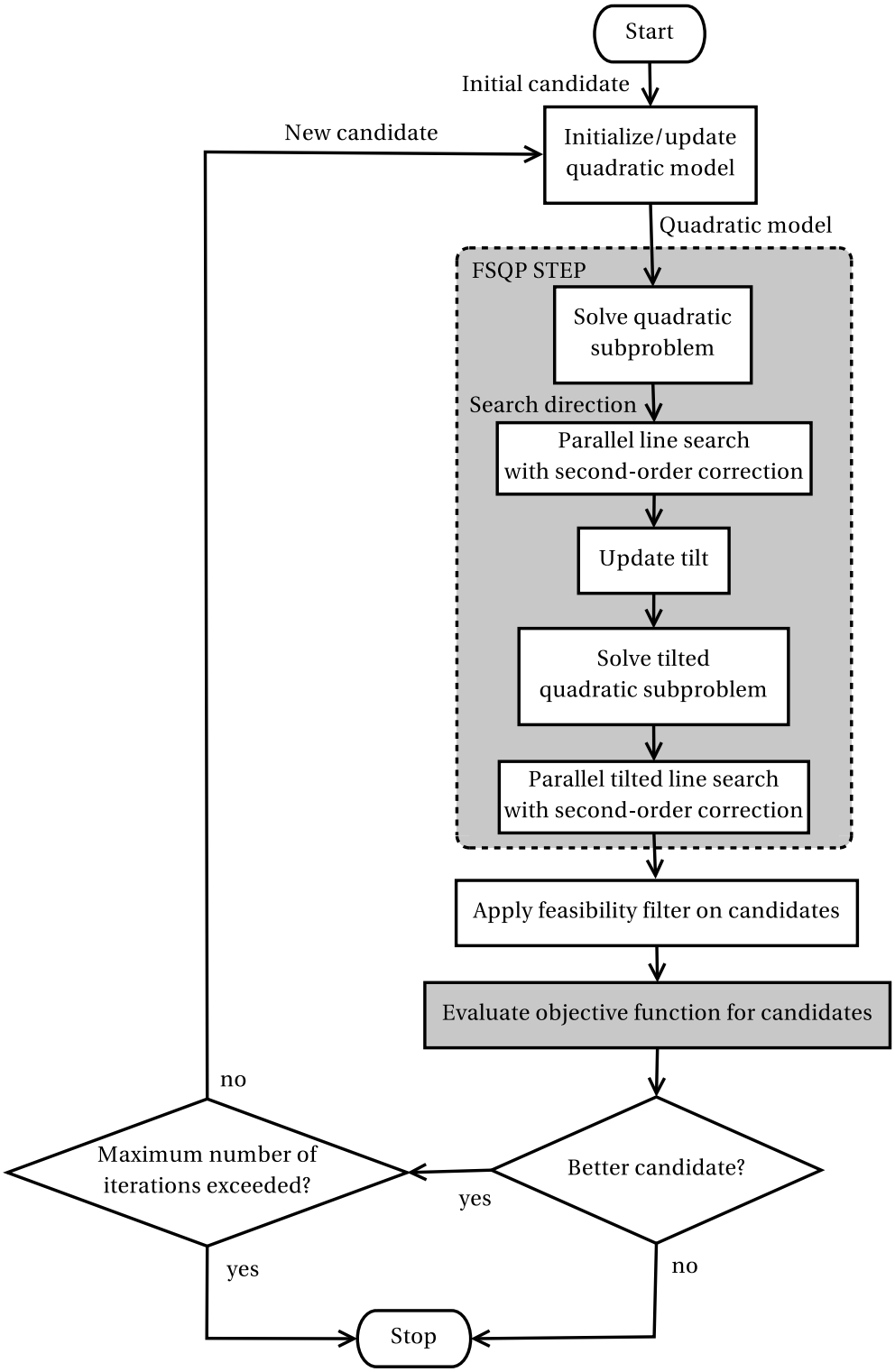


Figure 4.3: Flow chart of a line search feasible sequential quadratic programming algorithm (FSQP)

This quadratic programming problem features an additional scalar optimization parameter γ. It is part of the objective function and must be minimized. The safety margins are set with a tilting vector $\boldsymbol{\eta}$ with one positive component for each inequality constraint: $\eta_j^{(r)} > 0$.

The parameter γ takes negative values, if $\Delta\mathbf{x}_t^{(r)}$ is a descent direction. This is assured by the constraint:

$$\mathbf{g}^{\mathrm{T}}(\mathbf{x}^{(r)})\,\Delta\mathbf{x}_t^{(r)} \quad \leq \gamma \tag{4.10}$$

The parameter γ is bounded below by the descent in the search direction $\Delta\mathbf{x}_t^{(r)}$. In order to minimize the parameter γ, a direction $\Delta\mathbf{x}_t^{(r)}$ with large descent must be found.

The tilting vector $\boldsymbol{\eta}^{(r)}$ includes one positive value for each inequality constraint. This value η_j is multiplied with the parameter γ to obtain a safety margin for the corresponding constraint:

$$\mathbf{A}_{ineq}(\mathbf{x}^{(r)})\,\Delta\mathbf{x}_t^{(r)} \quad + \quad \mathbf{a}_{ineq}(\mathbf{x}^{(r)}) \quad \leq\ \gamma\,\boldsymbol{\eta}^{(r)} \tag{4.11}$$

The safety margins increase for smaller, negative values of parameter γ and larger, positive values of the components of the tilting vector $\boldsymbol{\eta}^{(r)}$. In order to minimize the parameter γ, a direction $\Delta\mathbf{x}_t^{(r)}$ must be found that satisfies increasingly strict inequality constraints. The tilted search direction $\Delta\mathbf{x}_t^{*(r)}$ does not point right on the boundary of the inequality constraints due to the safety margins.

In conclusion, the constraints (4.10) and (4.11) relate to opposite goals: (4.10) looks for descent in the objective function and (4.11) for feasibility in the inequality constraints. The trade-off between these two goals is controlled with the tilting vector $\boldsymbol{\eta}^{(r)}$. A large value of a component of the tilting vector $\boldsymbol{\eta}^{(r)}$ is equal to sacrificing descent in order to emphasize feasibility in the corresponding inequality constraint. The tilting vector $\boldsymbol{\eta}^{(r)}$ is adaptively updated during each iteration of the optimization. This update is described in the next section.

4.3.2 Update of the Tilting Vector

The tilt is controlled by the tilting vector $\boldsymbol{\eta}^{(r)}$. This tilting vector is updated in each iteration. The update checks if a constraint violation occurred at the candidate found by taking the full step size $\alpha = 1$ in the standard search direction $\Delta\mathbf{x}^{*(r)}$ (See eq. 4.8).

The components of the vector of tilt factors $z_j^{(r)}$ is updated as follows [LT00]:

$$\begin{aligned} a_{ineq,j}(\mathbf{x}^{(r)} + \Delta\mathbf{x}^{*(r)}) > 0 \quad &\rightarrow \quad z_j^{(r)} = \beta\, z_j^{(r-1)} \\ a_{ineq,j}(\mathbf{x}^{(r)} + \Delta\mathbf{x}^{*(r)}) \leq 0 \quad &\rightarrow \quad z_j^{(r)} = \frac{1}{\beta}\, z_j^{(r-1)} \end{aligned} \tag{4.12}$$

Feasibility is emphasized for those constraints that are violated in the full step by increasing the tilt factor $z_j^{(r)}$ by a factor $\beta > 1$ compared to the value in the last iteration. For the other constraints, the feasibility requirements are relaxed by decreasing the tilt factor $z_j^{(r)}$ by the factor $1/\beta$.

The tilting vector $\boldsymbol{\eta}$ for the tilted QP problem in (4.9) is found by normalizing the tilt factors with the L2-norm of the untilted search direction [LT00]:

$$\eta_j^{(r)} = z_j^{(r)} \mid \Delta \mathbf{x}^{*(r)} \mid_2 \tag{4.13}$$

The presented method uses a tilt vector with one component for each inequality constraint. This is an improvement in the original FSQP algorithm presented in [LT00], which suggests a single common tilt for all constraints. Additionally, the tilting factors are kept in fixed ranges to avoid extremely large and small values:

$$z_j^{(r)} \in [z_{low} \quad z_{high}] \tag{4.14}$$

The tilt factors are initialized with the lower bound $z_j^{(0)} = z_{low}$. The update increases the tilt factors during the optimization for those inequality constraints that are violated for the standard search direction. The following, suited choices of the parameters for the tilt update were found empirically: $\beta = 10$, $z_{low} = 0.001$ and $z_{high} = 1000$.

4.3.3 Parallel Line Search with Second-Order Correction

The flow chart of the parallel line search is shown in figure 4.4. The line search is conducted once for the standard search direction and once for the tilted search direction as shown in figure 4.3. This is an additional modification of the original FSQP algorithm presented in [LT00], which suggests to conduct a line search only in the tilted search direction. The line search is presented for the standard search direction in the following:

First, a set of Q candidates is generated along the search direction for different step sizes in parallel:

$$\mathbf{x}_q^{(r+1)} = \mathbf{x}^{(r)} + \alpha_q \cdot \Delta \mathbf{x}^{*(r)} \quad \text{with} \quad \alpha_q = 0.5^q; \quad q = 0, ..., Q-1 \tag{4.15}$$

For example for $Q = 5$, the algorithm uses five step sizes $\alpha_1 = 1, \alpha_2 = 0.5, \alpha_3 = 0.25, \alpha_4 = 0.125$ and $\alpha_5 = 0.0625$. The constraints are evaluated for each of the candidates in the set. For infeasible candidates with violated constraints, the following system of linear equalities is solved to obtain a so called second-order correction step $\Delta \mathbf{x}_{q,SOC}^{(r)}$:

$$\mathbf{A}_{act} \;\; \Delta \mathbf{x}_{q,SOC}^{(r)} = \mathbf{a}_{act} \tag{4.16}$$

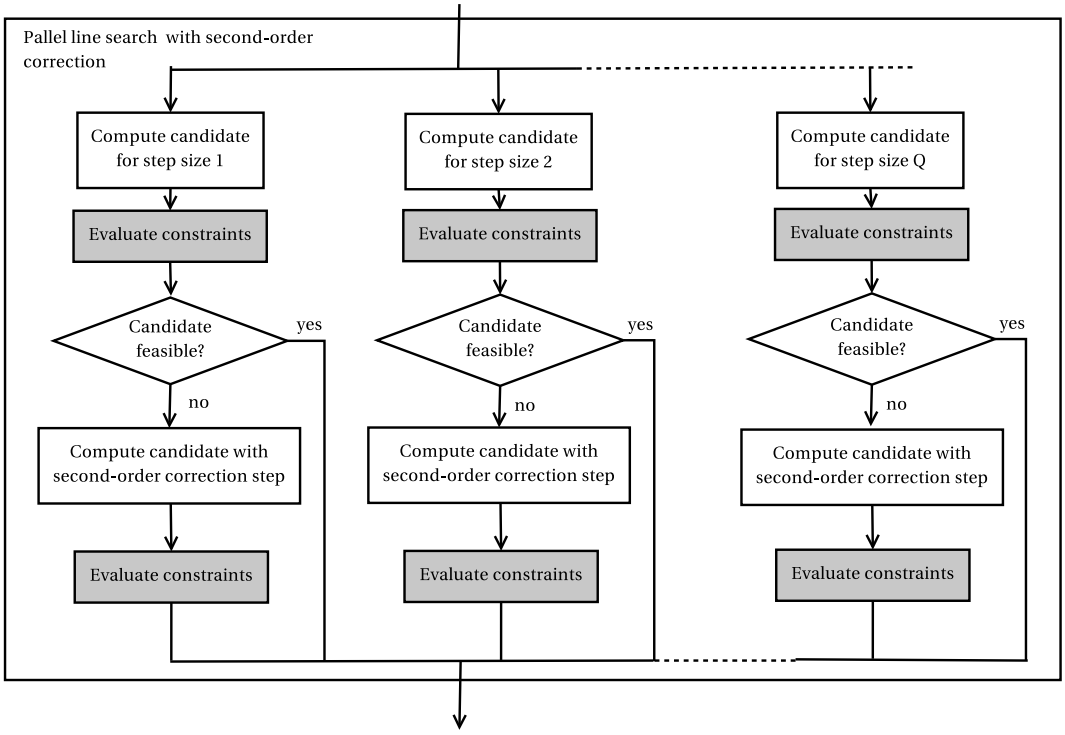


Figure 4.4: Flow chart of the parallel line search with second-order correction

A list of so-called active constraints is kept during the SQP optimization. A constraint is active if it can not be removed from the optimization problem without changing its solution. The matrix $\mathbf{A}_{act}$ is the Jacobi matrix of the active inequality constraints, evaluated at the candidate $\mathbf{x}^{(r)}$. It is available from the QP. The vector $\mathbf{a}_{act}$ equals those components of $\mathbf{a}_{ineq}$ that correspond to active constraints. These are evaluated at the candidate $\mathbf{x}_q^{(r+1)} = \alpha_q \Delta\mathbf{x}^{(r)} + \mathbf{x}^{(r)}$. The values are available since the constraint vector $\mathbf{a}_{ineq}(\mathbf{x}_q^{(r+1)})$ of the new candidates were already simulated.

The second-order correction step is calculated with the Pseude-inverse as shown in [NW99] p. 570:

$$\Delta\mathbf{x}_{q,SOC}^{(r)} = -\mathbf{A}_{act}^{\mathrm{T}} \left(\mathbf{A}_{act}\, \mathbf{A}_{act}^{\mathrm{T}}\right)^{-1} \mathbf{a}_{act} \tag{4.17}$$

An additional candidate is computed by adding the second-order correction step to the infeasible candidate:

$$\mathbf{x}_{q,SOC}^{(r+1)} = \mathbf{x}^{(r)} + \alpha_q \cdot \Delta\mathbf{x}^{*(r)} + \Delta\mathbf{x}_{q,SOC}^{(r)} \tag{4.18}$$

The constraint values of these new candidates $\mathbf{x}_{q,SOC}^{(r+1)}$ are also simulated. All computed candidates and their constraint vectors are handed over to the feasibility filter. The tilted parallel line search can be implemented identically. It uses the tilted search direction instead of the standard search direction to compute the candidates with different step sizes.

4.3.4 Feasibility Filter and Candidate Selection

The constraints are evaluated for all candidates during the parallel line search. The following feasibility filter deletes all candidates with constraint violations. The candidate with smallest (best) objective value is selected from the remaining candidates. It is checked whether is has a smaller objective function value than the selected candidate from the last iteration. If an improvement was made, the generated candidate becomes the selected candidate for the next iteration. Otherwise, the optimization terminates.

A problem arises if the initial candidate has some constraint violations. Sometimes, no new feasible candidates are found in the first iteration step. In this case, all new candidates are deleted and the optimization terminates. In order to start from an infeasible candidate, the feasibility filter is adapted, such that it first looks at the number of constraints violated for all known candidates. It keeps those candidates that have a minimal number of constraint violations.

For example, if the initial candidate has two constraint violations, the filter accepts all candidates with two violations until a candidate with one constraint violation is found. Then, it only accepts candidates with one constraint violation until a candidate with no constraint violations is found. At this point, the filter returns to its original state and only accepts candidates with no constraint violations. This adaption allows the optimization algorithm to converge to feasible candidates starting from an infeasible initial candidate.

4.4 Wavefront Approach

4.4.1 Simultaneous Optimization

An optimization algorithm for Pareto optimization can make use of the special nature of the task to compute the Pareto front. The Pareto-optimal performance vectors generated in each iteration of the iterative approach, presented in chapter 3, can be computed in parallel. For this, one CNOP in the form of (4.2) or (4.3) must be solved for each Pareto-optimal performance vector. The CNOPs according to (4.2) and (4.3) are geometrically represented by a target trajectory in the performance space as detailed in sections 3.2 and 3.3.

The Wavefront approach optimizes the corresponding CNOPs simultaneously. In each iteration of the Wavefront optimization, the solutions of all CNOPs move like a wave towards the Pareto front. Each CNOP targets at a different optimal compromise but looks to minimize the different competing performances. *'Neighboring'* target trajectories optimize for a *'similar'* goal. Therefore, the candidates of one CNOP can also be good candidates for another CNOP. The Wavefront approach shares the candidates between the CNOPs.

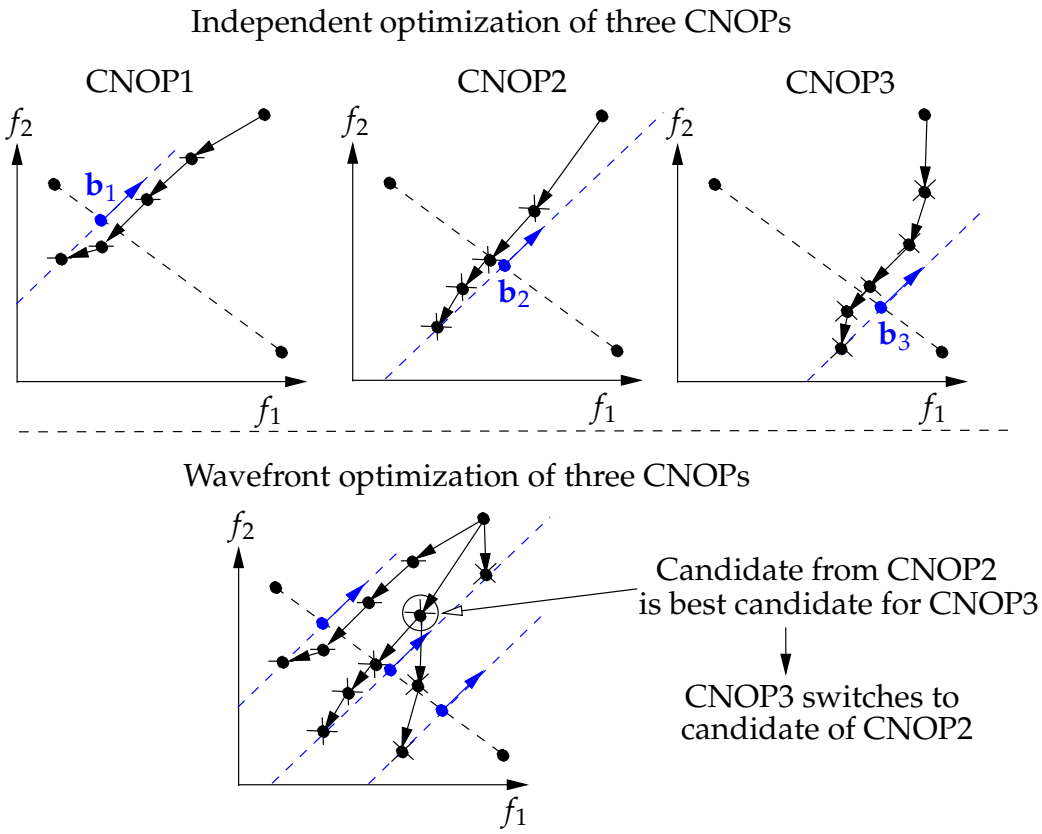


Figure 4.5: Illustration of the Wavefront approach, *lower*, compared to an independent optimization, *upper*.

This is illustrated in figure 4.5 for the case of two performances (See sec. 3.5.1). There are three CNOPs with different base points $\mathbf{b}_k$. For simplicity, only one candidate per CNOP is shown. The upper part of the figure shows the progress of an independent optimization of the CNOPs. The lower part of the figure shows the Wavefront approach. During the optimization, a candidate of CNOP2 is better than the corresponding candidate of CNOP3 in regard to the progress in CNOP3. Therefore, CNOP3 uses this candidate instead of the one created by its own line search. This sharing of candidates leads to a larger improvement of the objective in CNOP3 in this iteration step, which can lead to an overall faster convergence of the optimization.

Additionally, it may occur that one CNOP converges to a local minimum. If one of the other CNOP contributes a candidate near the global minimum, then this CNOP may *jump* to this solution near the global minimum and converge from there. The exchange of solutions can lead to an improved global convergence.

4.4.2 Wavefront FSQP Algorithm

The flow chart of the Wavefront FSQP algorithm is shown in figure 4.6. Each CNOP selects one candidate. It has the minimum objective function value for this CNOP compared to all candidates encountered so far. The standard search direction and tilted search direction are calculated by solving the quadratic programs in (4.4) and (4.9) respectively. New candidates are computed for each CNOP. At this stage, the candidates of all CNOPs are combined and sent to the feasibility filter. The objective function is evaluated for the remaining feasible candidates. A parallel evaluation of all candidates on several CPUs is possible. From the feasible candidates, each CNOP selects the best candidate with minimum objective function value regardless which CNOP the candidate came from.

4.4.3 Stopping Criteria and Activeness of CNOPs

This simultaneous computation requires more sophisticated criteria to terminate the optimization. Each CNOP is either active or inactive during each iteration step. If a CNOP is active, the quadratic model is updated and new candidates are generated. If the CNOP is inactive, these steps are skipped. The CNOP is set active or inactive at the end of an iteration: A CNOP stays active as long as a candidate with smaller objective function value is generated. A CNOP can become active after being inactive if another CNOP supplies a better candidate for it. This can, for example, occur, if the CNOP converged to a local minimum and becomes inactive. If another CNOP supplies a better candidate, the CNOP is set active again. It restarts the optimization from the new candidate. The optimization terminates if all CNOP are inactive because no new candidates are generated, or if the maximum number of iterations is exceeded.

4.5 Application of the Minmax and Goal-Attainment Formulation

The optimization can be made more effective by using the equivalence of the Minmax and Goal-attainment formulation. In the following, the individual application of both formulations is discussed. Then, an approach is presented that applies both formulations during the optimization.

4.5.1 Optimization with the Minmax Formulation

The Minmax formulation has a discontinuous objective function $o(\mathbf{x})$, as shown in (4.2). The gradient $\mathbf{g}(\mathbf{x})$ is required to set up the quadratic model (See 4.4). The gradient has a discontinuity if the Minmax operator switches from one performance term

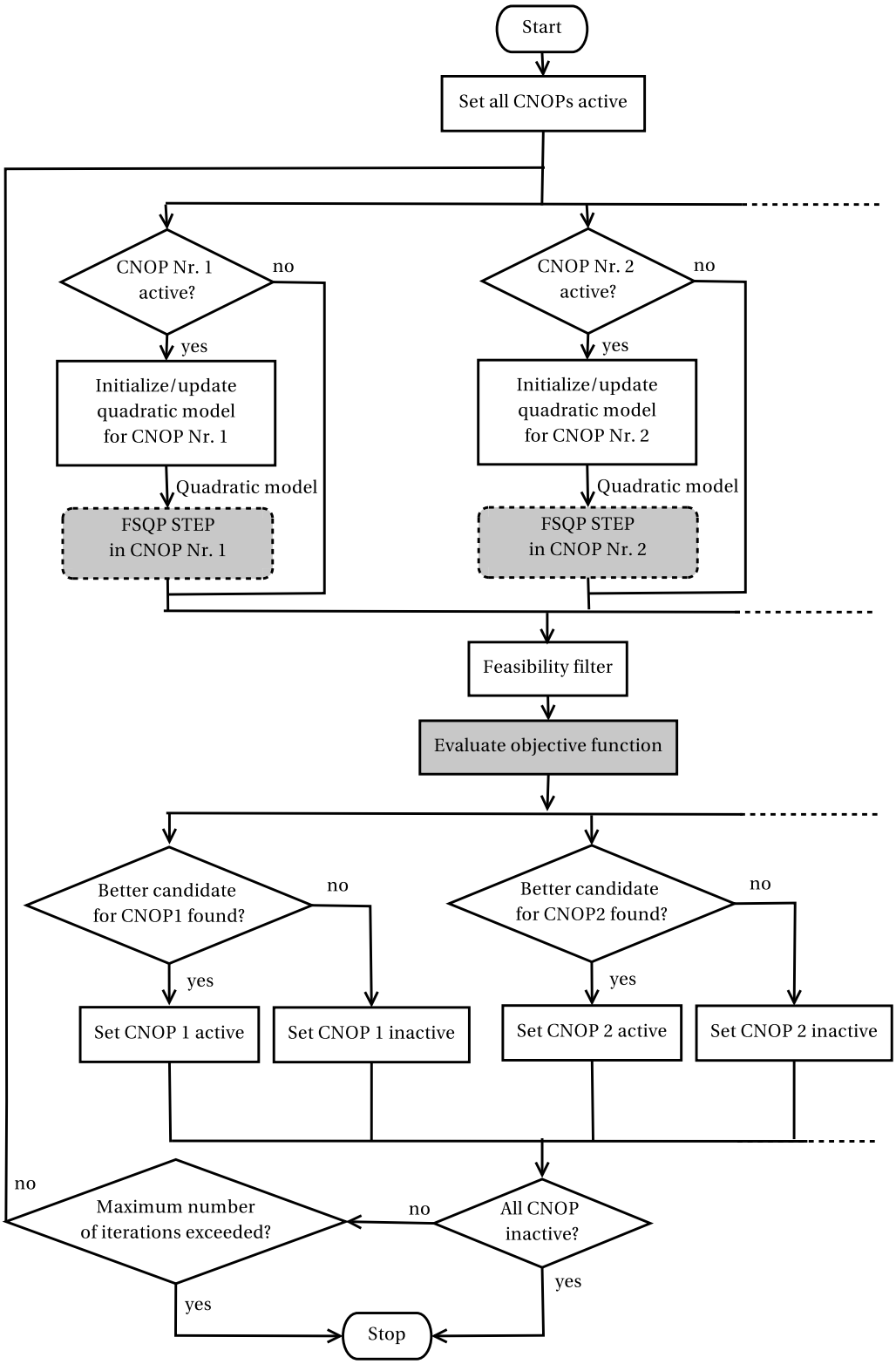


Figure 4.6: Flow chart of the Wavefront FSQP algorithm

to another. This leads to *'jumps'* in the gradient that introduce wrong information in a quadratic model. Therefore, the Minmax method is not well suited to set up a quadratic model with Quasi-Newton methods or determine the search directions solving the QP or tilted QP in (4.4) and (4.9) respectively.

4.5.2 Optimization with the GA Formulation

The CNOP for the GA formulation is shown in (4.3). The inequality constraints of the CNOP $\mathbf{a}_{ineq}(\mathbf{x})$ include the sizing rules $\mathbf{c}(\mathbf{d})$ as well as the design parameter bounds. The remaining inequality constraints are based on the GA constraints. The GA constraints depend on the performances $\mathbf{f}(\mathbf{d})$ and the bound variable t.

These additional GA constraints effect the evaluation of newly generated candidates during the line search negatively. Some candidates, which have no violation in the sizing rules or bounds, are deleted by the feasibility filter because of violated GA constraints. These violations are caused either by the approximations in the quadratic model or by the exchange of candidates between different CNOP. The effects are discussed in the following in more detail:

The computation of the search direction in (4.4) and tilted search direction in (4.9) is based on a quadratic model of the optimization problem. This model consists of the linearized inequality constraints of the QP as well as the Hessian matrix $\mathbf{H}$ of the Lagrangian function. The performances and sizing rules are implicitly approximated in this model by model functions, that are denoted in the following as $\mathbf{f}_m(\mathbf{d})$ and $\mathbf{c}_m(\mathbf{d})$. The optimization looks for a search direction that yields an improvement in the objective function t. It requires to find a step in the design parameter vectors $\Delta\mathbf{d}$ such that:

1. the constraint vector $\mathbf{c}_m(\mathbf{d}+\Delta\mathbf{d})$ in the quadratic model is feasible.
2. the performance vector $\mathbf{f}_m(\mathbf{d}+\Delta\mathbf{d})$ in the quadratic model allows to make the GA constraints more strict by changing the value of t by Δt.

The upper part of figure 4.7 shows the generation of such a search direction $\Delta\mathbf{x}^*$ for the GA method. The search direction is composed of two parts $\Delta\mathbf{d}$ and Δt:

$$\Delta\mathbf{x}^* = \begin{bmatrix} \Delta\mathbf{d} \\ \Delta t \end{bmatrix}; \tag{4.19}$$

During the line search, the sizing rules and circuit performance values of the new candidate are computed by simulation. The approximation in the quadratic model can cause four different cases that are illustrated in the lower part of figure 4.7:

- Cases BC and BD: In these cases, sizing rules are violated. The design parameter vector is located outside the valid parameter space. The feasibility filter removes the candidate correctly, since it is not a valid sizing of the circuit.

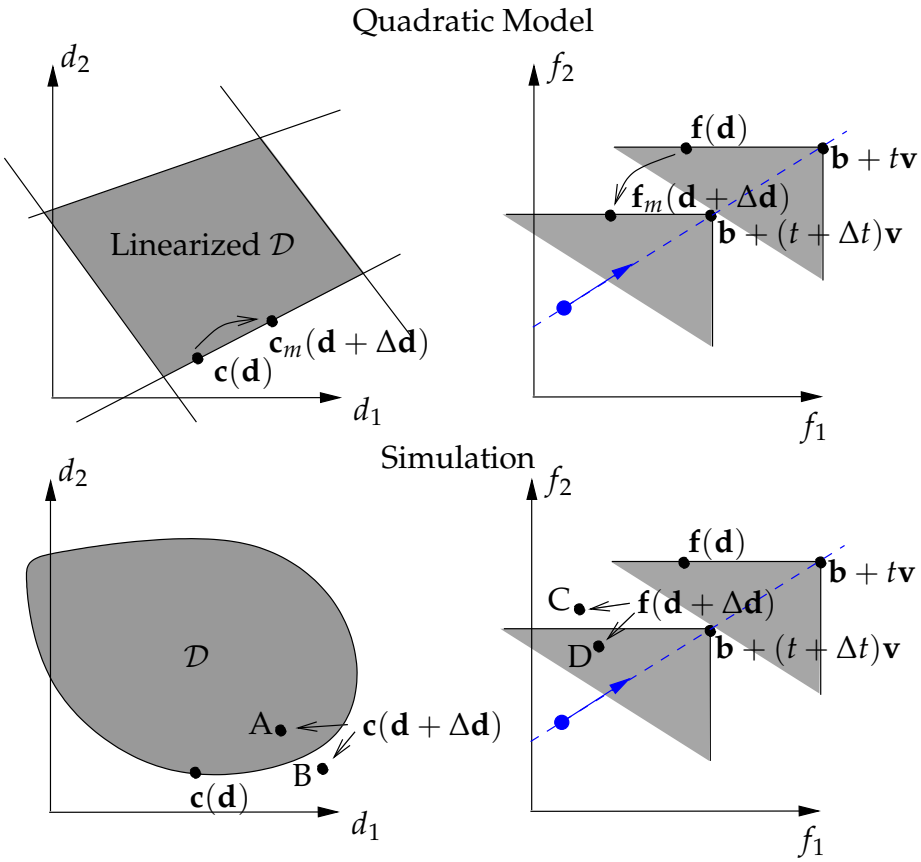


Figure 4.7: Sizing rules, GA constraints and circuit performances obtained inside the quadratic model and by simulation during the optimization of a CNOP using the Goal-attainment formulation

- Case AD: The sizing rules and GA constraints are met. The feasibility filter keeps this candidate correctly.
- Case AC: The design parameter vector constitutes a valid sizing of the circuit since it is inside the feasible design parameter space. Still, the performance vector of this candidate violates the GA constraints. Such a candidate is removed by the feasibility filter although it is a valid sizing of the circuit. The case is created if the real improvement in the performances $\mathbf{f}(\mathbf{d}+\Delta\mathbf{d})$ did not match the improvement in the quadratic model $\mathbf{f}_m(\mathbf{d}+\Delta\mathbf{d})$. The GA constraints were made too strict by adding the step Δt to the parameter t. This case occurs regularly since the GA constraint's dependency on the performances is only approximated in the quadratic model.

A similar effect occurs, if we use the GA formulation and exchange solutions between CNOPs during the optimization. This effect is illustrated in figure 4.8. The CNOPs have different base points as part of their GA constraints. The base points define different target trajectories (See section 3.3). Due to the different target trajectories,

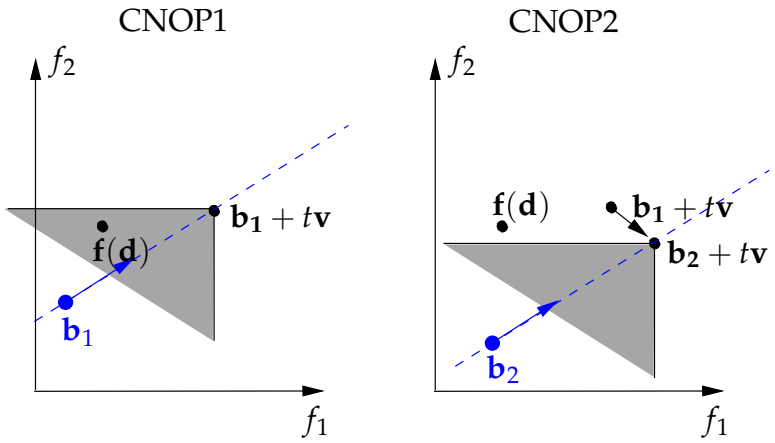


Figure 4.8: GA constraints during an exchange of a candidate between two CNOP using the Goal-attainment formulation

the GA constraint values differ between CNOPs for one candidate $\mathbf{x} = [\mathbf{d}^T \; t]^T$. In figure 4.8, the GA constraints are fulfilled for CNOP1. For the same candidate, the GA constraints are violated for CNOP2. Therefore, the exchange of solutions, as done in the Wavefront approach, would lead to an undetermined state for the candidate. The feasibility filter deletes the candidate for one CNOP but keeps it for another.

Concluding, the GA formulation is not well suited for the line search phase of the optimization, since the GA constraints may flag candidates infeasible that constitute valid sizings of the circuit.

4.5.3 Optimization with the GA and Minmax Formulation

The limitations of the Minmax and GA method can be eliminated by using both formulations during the optimization as follows:

- The GA formulation is used to set up the quadratic model and solve the QP problems in order to obtain the search directions.
- The Minmax method is used during the line search, the feasibility filter and the selection of candidates.

This setup has some similarity to applying the Minmax formulation as a merit function for the GA formulated CNOP. The difference is that one component must be added and removed from the candidate vector $\mathbf{x}$. This is shown in the following in more detail:

The selected candidate only includes circuit design parameters $\mathbf{x}^{(r)} = \mathbf{d}^{(r)}$ according to the Minmax formulation of the CNOP given in (4.2). The bound parameter $t^{(r)}$ is appended to the candidate to obtain the candidate vector $\mathbf{x}_{QP}^{(r)}$ for the GA formulation of the CNOP given in (4.3). It is used for the quadratic model and QP problem. The

bound variable t is initialized, such that the GA constraints are not violated and t is minimal *. This leads to the following GA candidate vector $\mathbf{x}_{QP}^{(r)}$:

$$\mathbf{x}_{QP}^{(r)} = \begin{bmatrix} \mathbf{d}^{(r)} \\ t^{(r)} \end{bmatrix}; \quad \text{with} \quad t^{(r)} = \max_{i=1..n_f} \frac{f_i(\mathbf{d}^{(r)}) - b_i}{v_i} \tag{4.20}$$

The candidate $\mathbf{x}_{QP}^{(r)}$ is used with the GA formulation in (4.3) to set up the quadratic model and solve the QP problems in order to obtain the search direction $\Delta\mathbf{x}_{QP}^{*(r)}$ and tilted search direction $\Delta\mathbf{x}_{QP,t}^{*(r)}$:

$$\Delta\mathbf{x}_{QP}^{*(r)} = \begin{bmatrix} \Delta\mathbf{d}^{(r)} \\ \Delta t^{(r)} \end{bmatrix}; \quad \Delta\mathbf{x}_{QP,t}^{*(r)} = \begin{bmatrix} \Delta\mathbf{d}_t^{(r)} \\ \Delta t_t^{(r)} \end{bmatrix}; \tag{4.21}$$

We are only interested in the part of the search direction that relates to the actual design parameters. For the search directions for the line search, the component for Δt is removed: $\Delta\mathbf{x}^{*(r)} = \Delta\mathbf{d}^{(r)}$ and $\Delta\mathbf{x}_t^{*(r)} = \Delta\mathbf{d}_t^{(r)}$. At this point, the algorithm switches to the Minmax formulation and determines the constraints and objective function value according to (4.2). Only invalid sizings are removed by the feasibility filter and the exchange of solutions can easily be conducted.

4.6 Summary

The Wavefront FSQP algorithm was presented in this chapter. A feasibility filter removes infeasible candidates. An adaption of the filter, that allows to find feasible candidates from an infeasible initial candidate was shown. The FSQP algorithm features tilting and second-order correction as methods to improve the feasibility of the candidates. The tilt is updated in each iteration.

The Wavefront approach optimizes the CNOP to find different Pareto-optimal performance vectors simultaneously. Candidates are exchanged between the CNOP to improve global convergence. Parallel simulation of the candidates of different CNOPs are possible.

The Wavefront FSQP algorithm applies both the Minmax and Goal-attainment formulation of the CNOP. This allows a more effective optimization than applying just one of the formulations.

* The initial value for t is found by setting the bound parameter t as the solution of the continuous optimization problem that replaced the Max operator in (3.5).

Chapter 5

Pareto Optimization With Tolerances

5.1 Introduction

The previous chapters 3 and 4 have focused on the generation of the Pareto front for the circuit performances under nominal process and operating conditions. Usually, the specification on the performance values of a circuit does not only have to be met for the nominal case but for any operating condition inside specified ranges. These ranges define a box-shaped tolerance region for the operating conditions that is given in (2.11). Additionally, process variations occur during the production of the integrated circuit. These process variations influence the performance values of the circuit. The yield value is equal to the percentage of circuits that meet the specification after production. The yield is defined as an integral over the probability density function of the statistical parameters inside an acceptance region as shown in (2.13).

The generation of the specification Pareto front is discussed in this chapter. It considers the influence of tolerances such as changing operating conditions and process variations on the performance capabilities of an analog circuit.

The specification Pareto front is defined in (2.30) as the set of most ambitious specifications on the performances that can be implemented with a given minimum yield requirement. The multi-objective optimization problem to calculate the specification Pareto front is given in (2.28). In order to generate the specification Pareto front, a specification analysis (SpA) is required that was introduced in (2.27). It calculates the most ambitious specification that leads to the given minimum yield for a given sizing. So-called geometric worst-case analysis offers an efficient method to conduct a SpA. It formulates the specification analysis as an optimization problem.

This optimization based SpA is used in the presented approach to generate the specification Pareto front. The resulting scalar GA and Minmax optimization problems feature two nested optimization loops. Solving these nested optimization problems in reasonable time is infeasible due to extremely high computational costs. It requires to run a full optimization based SpA for each performance of each candidate that is

generated during the Pareto optimization. Approaches are needed that reduce the number of required SpA optimizations in order to compute the specification Pareto front in reasonable time. Two approaches are presented: The first approach runs the SpA only once after a nominal Pareto optimization. The second approach runs the SpA and Pareto optimization in alternation. A single step of the SpA is conducted for each performance of each candidate instead of a full SpA optimization. The second approach can be further speeded up by running the SpA optimization steps only for the most promising candidates. The applicability of both approaches to different circuit performances is discussed.

The chapter is structured as follows: In section 5.2, the optimization based specification analysis is reviewed. The optimization problem to generate the specification Pareto front is presented in section 5.3. The approaches to compute the specification Pareto front in reasonable time are discussed in section 5.4. Finally, section 5.5 summarizes the chapter.

5.2 Optimization Based Specification Analysis (SpA)

The optimization based specification analysis is introduced in the following sections.

5.2.1 Optimization Problem of the Specification Analysis

The specification analysis of (2.27) can be conducted by solving the following optimization problem [Gra07]:

$$\max_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_{\mathbf{s}}} f_i(\mathbf{d},\boldsymbol{\theta},\mathbf{s}) \quad \rightarrow \quad \boldsymbol{\theta}_{Y,i},\ \mathbf{s}_{Y,i},\ f_{Y,i} = f_i(\mathbf{d},\boldsymbol{\theta}_{Y,i},\mathbf{s}_{Y,i}) \tag{5.1}$$

The performance value $f_{Y,i}$ is the worst (maximum) value of the performance inside a given tolerance region, $\mathcal{T}_{\boldsymbol{\theta}}$, of the operating conditions and a tolerance region,$\mathcal{T}_{\mathbf{s}}$, of the statistical parameters that is given as:

$$\mathcal{T}_{\mathbf{s}} = \{\mathbf{s} \mid\ (\mathbf{s}-\mathbf{s}_0)^T\mathbf{C}^{-1}(\mathbf{s}-\mathbf{s}_0)\ \leq\ \beta_Y^2\} \tag{5.2}$$

This formula describes an ellipsoid-shaped area in the statistical parameter space. The worst performance value $f_{Y,i}$ inside the tolerance regions is the required specification value, with a corresponding yield value of Y_0 in regard to $f_{Y,i}$:

$$Y_0: \quad \text{yield value for specification} \quad f_i \leq f_{Y,i} = f_i(\mathbf{d},\boldsymbol{\theta}_{Y,i},\mathbf{s}_{Y,i}) \tag{5.3}$$

Due to the Gaussian distribution of the statistical parameters given in (2.3), the yield value Y_0 is equal to the cumulative density function (cdf) of the one-dimensional standard Gaussian distribution (SGD) for a given β_Y [Gra07]:

$$Y_0 = \mathrm{cdf}_{SGD}(\beta_Y) \tag{5.4}$$

Values of Y_0 are given in table 5.1 for some values of β_Y. The specification value $f_{Y,i}$ is found by computing the so-called specification parameter vectors $\boldsymbol{\theta}_{Y,i}$ and $\mathbf{s}_{Y,i}$ for β_Y. The specification parameter vectors are obtained by solving the optimization problem given in (5.1). The value of $f_{Y,i}$ is equal to the worst (maximum) value of the performance f_i inside the given tolerance regions. This problem formulation is also known as geometric worst-case analysis [Gra07]. The specification parameter vectors $\boldsymbol{\theta}_{Y,i}$ and $\mathbf{s}_{Y,i}$ usually depend

- strongly on the selected performance f_i. The specification parameter vectors must be calculated separately for each performance.
- weakly on the design parameter vector $\mathbf{d}$. The specification parameter vectors must be recalculated for each sizing $\mathbf{d}$ of the circuit. Due to the weak dependency, the specification parameter vectors for one design parameter vector usually constitute a good approximation of the specification parameter vectors for another design parameter vector. This weak dependency is used later for the efficient approach that runs the Pareto optimization and SpA in alternation as discussed in section 5.4.2.

β_Y	1	2	3
Y_0	84.1%	97.7%	99.9 %

Table 5.1: Relation between worst-case distance and yield

5.2.2 SpA and the Minimum Yield Requirement

One input parameter of the SpA is the worst-case distance β_Y that sets the size of the tolerance region of the statistical parameters. It is chosen such that the minimum yield requirement Y_{min} is met for all Pareto-optimal specification vectors on the specification Pareto front as follows:

The optimization based SpA computes the specification value $f_{Y,i}$ for each performance individually based on β_Y. Each β_Y is related to a yield value Y_0 as given in equation (5.4). The total yield Y_g for the complete specification vector $\mathbf{f}_Y$ is lower than Y_0, because the yield loss for the individual specifications on different performances might add up. The relation between the total yield Y_g and yield value Y_0 is illustrated in figure 5.1 for two performances. A lower bound for the total yield Y_g in dependency of the yield for each single performance Y_0 is given as:

$$Y_g \geq Y_{g,lb} \quad \text{with} \quad Y_{g,lb} = 1 - n_f(1 - Y_0) \tag{5.5}$$

The minimum yield requirement $Y_g \geq Y_{min}$ is definitely met for:

$$Y_{min} = Y_{g,lb} = 1 - n_f(1 - Y_0) \tag{5.6}$$

The minimal yield requirement Y_0 for each individual performance specification is, therefore, determined for a given minimum yield requirement $Y_g \geq Y_{min}$ by:

$$Y_0 = 1 - \frac{1 - Y_{min}}{n_f} \tag{5.7}$$

To find a specification value $f_{Y,i}$ that leads to a yield value Y_0, a SpA is conducted with the worst-case distance β_Y equal to:

$$\beta_Y = \mathrm{cdf}_{SGD}^{-1}(Y_0) = \mathrm{cdf}_{SGD}^{-1}(1 - \frac{1 - Y_{min}}{n_f}) \tag{5.8}$$

For example, for two performances and a minimum yield requirement of $Y_{min} = 95\%$, the individual yield requirement Y_0 is 97.5%, which leads to $\beta_Y = 1.96$.

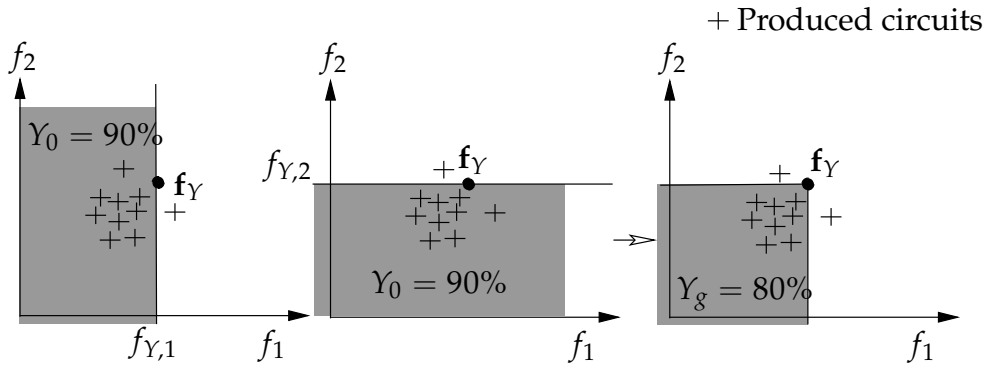


Figure 5.1: Relation between yield for single performance specifications and total yield

5.3 Generation of the Specification Pareto Front

The specification Pareto front shows the performance capabilities of the analog circuit structure for the given minimum yield requirement. In the following, the corresponding multi-objective optimization problem of (2.28) is formulated to apply specification analysis.

5.3.1 Multi-Objective Problem Formulation Considering Tolerances

The multi-objective optimization problem in (2.28) can be rewritten with the optimization based specification analysis to:

$$\min_{\mathbf{d}\in\mathcal{D}} \mathbf{f}_Y(\mathbf{d}) \quad = \quad \min_{\mathbf{d}\in\mathcal{D}} \begin{bmatrix} f_{Y,1}(\mathbf{d}) \\ \cdot \\ \cdot \\ f_{Y,n_f}(\mathbf{d}) \end{bmatrix} = \quad \min_{\mathbf{d}\in\mathcal{D}} \begin{bmatrix} \max\limits_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_\mathbf{s}} f_1(\mathbf{d},\boldsymbol{\theta},\mathbf{s}) \\ \cdot \\ \cdot \\ \max\limits_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_\mathbf{s}} f_{n_f}(\mathbf{d},\boldsymbol{\theta},\mathbf{s}) \end{bmatrix} \tag{5.9}$$

This multi-objective optimization problem can be transformed to a scalar optimization problem by applying the Minmax and GA formulation in the same way as in the nominal case given in (3.4) and (3.8).

5.3.2 GA and Minmax Formulation Considering Tolerances

The scalar optimization problem using the Minmax formulation is given as:

$$\min_{\mathbf{d}\in\mathcal{D}} \max_{i=1..n_f} \frac{\max\limits_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_\mathbf{s}} f_i(\mathbf{d},\boldsymbol{\theta},\mathbf{s}) - b_i}{v_i} \quad \rightarrow \quad \mathbf{d}^*, \mathbf{f}_Y^* = \mathbf{f}_Y(\mathbf{d}^*) \tag{5.10}$$

The scalar optimization problem using the GA formulation is given as:

$$\min_{t,\mathbf{d}\in\mathcal{D}} t \text{ s.t. } \max_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_\mathbf{s}} f_i(\mathbf{d},\boldsymbol{\theta},\mathbf{s}) \leq v_i\, t + b_i,\ i = 1,\ ...,\ n_f \quad \rightarrow \quad \mathbf{d}^*, \mathbf{f}_Y^* = \mathbf{f}_Y(\mathbf{d}^*) \tag{5.11}$$

The resulting vector $\mathbf{f}_Y^*$ is a Pareto-optimal specification vector. It constitutes the most ambitious performance specifications that can be implemented under the condition that the given minimum yield requirement is met. Trade-offs in the performances also result in trade-offs in the obtainable specification values. Different compromises between the specification values can be obtained by selecting suitable scaling vectors $\mathbf{b}$ and $\mathbf{v}$ in the same way as was shown for the nominal case in section 3.2.3.

5.3.3 Relation between the Specification Pareto Front and the Nominal Pareto Front

The specification values are inferior to the values obtained by nominal Pareto optimization. This is illustrated in figure 5.2 for two performances. The difference between the nominal performance value $f_i(\mathbf{d})$ and and specification value $f_{Y,i}(\mathbf{d})$ is given as:

$$\Delta f_i(\mathbf{d}) = f_{Y,i}(\mathbf{d}) - f_i(\mathbf{d}) = f_i(\mathbf{d},\boldsymbol{\theta}_{Y,i},\mathbf{s}_{Y,i}) - f_i(\mathbf{d},\boldsymbol{\theta}_0,\mathbf{s}_0) \tag{5.12}$$

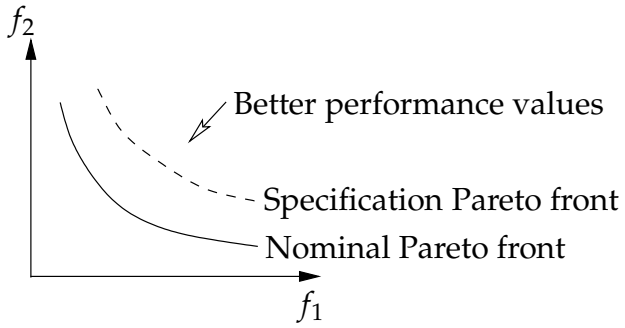


Figure 5.2: Illustration of the specification Pareto front compared to the nominal Pareto front.

The difference $\Delta f_i(\mathbf{d})$ is a measure for the performance degradation due to process variations and operating conditions. It depends on the given ranges on the operating parameters and the demanded minimum yield. Its value is also a measure of the sensitivity of the circuit performance to changes in the operating parameters and statistical parameters. Usually, the sensitivity of a performance to the operating parameters and statistical parameters changes for different design parameter vectors $\mathbf{d}$. This dependency of $\Delta f_i(\mathbf{d})$ on the design parameter vector $\mathbf{d}$ can lead to two cases. They are illustrated in figure 5.3 assuming there is only one design parameter d:

- Left case: The specification value $f_{Y,i}(d)$ is always improved by improving the nominal performance value $f_i(d)$. A design parameter vector that has an optimal nominal performance value f_i^* also has an optimal specification value $f_{Y,i}^*$.
- Right case: The specification value $f_{Y,i}(d)$ does not always improve with the nominal performance value $f_i(d)$. For these performances, a design parameter vector that has an optimal nominal performance value f_i^* might have a sub-optimal specification value $f_{Y,i}^+$.

The classification of the performances in these two cases requires circuit knowledge. If performances are identified that have a *'common'* minimum for the nominal and specification performance value, then this knowledge can be used to speed-up the simulation. It is possible to target at the nominal values during the optimization and save computational time because the specification values do not need to be calculated. This approach is suggested in section 5.4.1.

5.3.4 Challenges in the Computation of the Specification Pareto Front

The GA and Minmax formulations in (5.10) and (5.11) feature two nested optimization loops:

- The Pareto optimization loop that minimizes the performance values for the design vector $\mathbf{d}$.

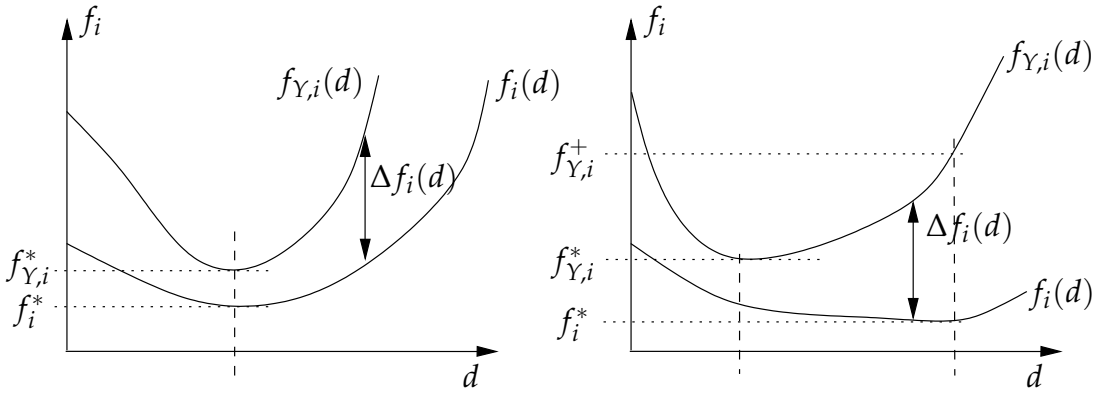


Figure 5.3: Illustration of the dependency of Δf_i on one design parameter vector d. *Left*, common minimum for nominal and specification performance value. *Right*, different Minima for nominal and specification performance value.

- The SpA loop that maximizes the performance values for the operating parameters $\boldsymbol{\theta}$ and statistical parameters $\mathbf{s}$.

Solving this optimization problem with deterministic optimization algorithms imposes two major challenges:

- The Pareto optimization requires the gradient of the specification values towards the design parameter vectors.
- The SpA loop must be conducted for each candidate, making it impossible to conduct the computation in reasonable time.

This is illustrated in the following in more detail. For a computation with SQP, the gradient of the specification values on each individual performance is required for the quadratic model (See (4.4) for details):

$$\nabla_{\mathbf{d}}\, f_{Y,i}(\mathbf{d}) \mid_{\mathbf{d}=\mathbf{d}^{(r)}} = \nabla_{\mathbf{d}} \max_{\boldsymbol{\theta}\in\mathcal{T}_{\boldsymbol{\theta}},\mathbf{s}\in\mathcal{T}_{\mathbf{s}}} f_i(\mathbf{d}^{(r)},\boldsymbol{\theta},\mathbf{s}); \quad i=1,...,n_f \tag{5.13}$$

This gradient is computed using the specification parameter vectors:

$$\nabla_{\mathbf{d}}\, f_{Y,i}(\mathbf{d}) \mid_{\mathbf{d}=\mathbf{d}^{(r)}} = \nabla_{\mathbf{d}}\, f_i(\mathbf{d}^{(r)},\boldsymbol{\theta}_{Y,i},\mathbf{s}_{Y,i}); \quad i=1,...,n_f \tag{5.14}$$

The optimization problem of the SpA is also solved deterministically. This requires the gradients of the performances for the operating and process parameter vectors:

$$\nabla_{\boldsymbol{\theta}} f_i(\mathbf{d},\boldsymbol{\theta},\mathbf{s}), \quad \nabla_{\mathbf{s}} f_i(\mathbf{d},\boldsymbol{\theta},\mathbf{s}); \quad i=1,...,n_f \tag{5.15}$$

The progression of the computation of the specification Pareto front with the applied gradients is illustrated in table 5.2. For simplicity, the SpA is only shown for the statistical parameters leaving out the operating parameters. It is also assumed that

Progression	It. step	Perf. gradients	Result
Initial	PO 0	-	$\mathbf{d}^{(0)} = \mathbf{d}_{initial};\ \mathbf{s}_0$
SpA for all f_i	SpA 1	$\nabla_{\mathbf{s}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_0}$	$\mathbf{s}_i^{(0,1)};\quad i = 1, ..., n_f$
	SpA 2	$\nabla_{\mathbf{s}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_i^{(0,1)}}$	$\mathbf{s}_i^{(0,2)};\quad i = 1, ..., n_f$
	$\vdots$		
	SpA S	$\nabla_{\mathbf{s}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_i^{(0,S-1)}}$	$\mathbf{s}_{Y,i}^{(0)} = \mathbf{s}_i^{(0,S)};\quad i = 1, ..., n_f$
Spec. value sim.	-	-	$\mathbf{f}_Y(\mathbf{d}^{(0)}) = \begin{bmatrix} f_1(\mathbf{d}^{(0)}, \mathbf{s}_{Y,1}^{(0)}) \\ \cdot \\ \cdot \\ f_{n_f}(\mathbf{d}^{(0)}, \mathbf{s}_{Y,n_f}^{(0)}) \end{bmatrix}$
Pareto opt. step	PO 1	$\nabla_{\mathbf{d}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_{Y,i}^{(0)}}$	$\mathbf{d}^{(1)};\quad i = 1, ..., n_f$
SpA for all f_i	SpA 1	$\nabla_{\mathbf{s}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(1)},\ \mathbf{s}=\mathbf{s}_0}$	$\mathbf{s}_i^{(1,1)};\quad i = 1, ..., n_f$
	SpA 2	$\nabla_{\mathbf{s}} f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(1)},\ \mathbf{s}=\mathbf{s}_i^{(1,1)}}$	$\mathbf{s}_i^{(1,2)};\quad i = 1, ..., n_f$
$\vdots$	$\vdots$		

Table 5.2: Progress of the computation of the specification Pareto front

no line search is conducted such that a single candidate is generated in each iteration of the Pareto optimization.

Computing the specification Pareto front in this way is practically infeasible in terms of computational costs. The SpA optimization is nested into several other loops. This is depicted in figure 5.4: The Pareto optimization must be conducted once for each target trajectory. In each iteration, one sensitivity analysis for the design parameters $\mathbf{d}$ is required and a set of new candidates is generated. For each performance of each candidate, one iterative SpA optimization must be conducted. In each iteration of the SpA, one sensitivity analysis for the operating parameters $\boldsymbol{\theta}$ and statistical parameters $\mathbf{s}$ as well as one performance simulation at the computed specification parameter vectors is required.

For simplicity, it is assumed that the number of iterations of the Pareto and SpA optimization is constant and that the number of generated candidates is also constant. Neglecting the sensitivity analysis for the design parameter vectors, the required number of simulations is given as:

$$\#\,\text{simulations (spec. front, general)} = K \cdot R \cdot Q \cdot n_f \cdot S \cdot (n_s + n_\theta + 1) \tag{5.16}$$

As comparison, the generation of the nominal Pareto front requires one nominal simulation for each performance of each candidate:

$$\#\,\text{simulations (nom. front)} = K \cdot R \cdot Q \cdot n_f \tag{5.17}$$

A fast computation of the nominal Pareto front already requires heavy use of parallel simulations. Running a Pareto optimization to obtain the specification Pareto front becomes quickly infeasible as the simulation costs are multiplied with $S \cdot (n_s + n_\theta + 1)$. This factor is equal to the number of simulations required for one SpA. Therefore, approaches are needed that allow to compute the Specification Pareto front in reasonable time. Two such approaches are presented in section 5.4.

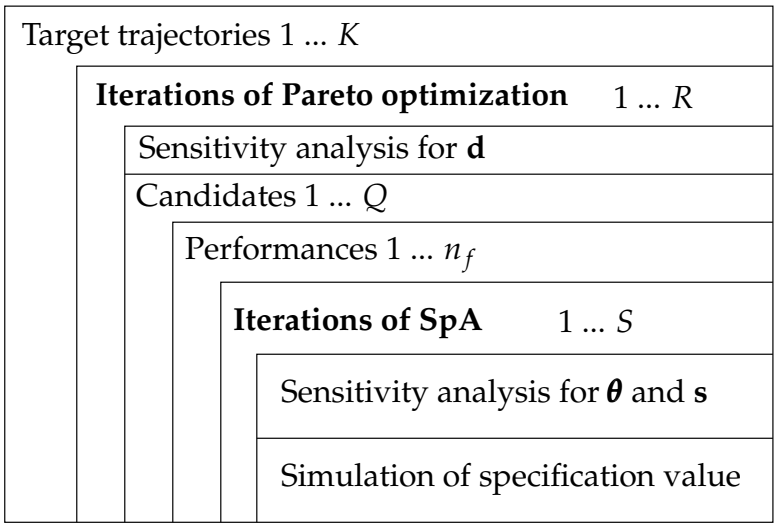


Figure 5.4: Illustration of the nested optimization loops for generating the specification Pareto front.

5.4 Efficient Computation of the Specification Pareto Front

5.4.1 SpA after Pareto Optimization

For some circuit performances, an improvement in the nominal value may simultaneously improve the specification value as was mentioned in section 5.3.3. A design parameter vector with optimal nominal performance then also has an optimal specification value. If the specification Pareto front must be generated for a set of these performances, then a design parameter vector that has a Pareto-optimal nominal performance vector also has a Pareto-optimal specification vector.

It is sufficient to generate the nominal Pareto front for such a set of performances. The SpA are conducted subsequently to find the specification vector with the given minimum yield for the generated solutions. This approach is illustrated in figure 5.5. During the Pareto optimization, only nominal performance values are considered. One SpA is run subsequently for each performance of the solution candidate of each target trajectory.

This method speeds up the computation of the specification Pareto front significantly. The SpA must only be conducted for one candidate of each target trajectory. The approach can not be applied to performances, that have no shared optimum for the nominal and specification value because for such performances a sub-optimal specification vector is calculated.

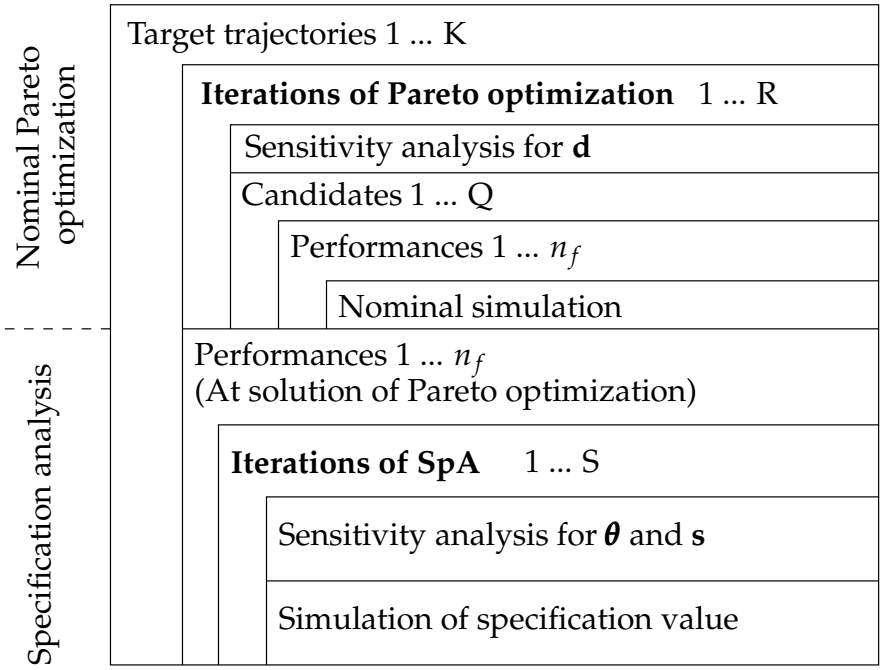


Figure 5.5: Illustration of the approach to speed-up the optimization by conducting the SpA after the Pareto optimization.

5.4.2 Alternating SpA and Pareto Optimization

As was noted in section 5.2, the specification parameter vectors $\boldsymbol{\theta}_{Y,i}$ and $\mathbf{s}_{Y,i}$ usually depend only weakly on the design parameter vector $\mathbf{d}$. Therefore the specification parameter vectors that are obtained for one candidate are a good first approximation for another candidate. This is used to implement an efficient SpA that is conducted in an alternating way with the Pareto optimization. A similar approach was already successfully applied in [SSPG02] for analog circuit sizing. The approach is outlined in the following:

The selected candidate in the r-th iteration of the Pareto optimization has the design parameter vector $\mathbf{d}^{(r)}$. The approximated specification parameter vectors for the performance f_i are $\boldsymbol{\theta}_{Y,i}^{(r)}$ and $\mathbf{s}_{Y,i}^{(r)}$. In the iteration $(r+1)$ the Pareto optimization generates a new candidate with design parameter vector $\mathbf{d}^{(r+1)}$. Its specification parameter set is found by taking the specification parameter vectors $\boldsymbol{\theta}_{Y,i}^{(r)}$ and $\mathbf{s}_{Y,i}^{(r)}$ of the previous iteration and conducting a single SpA iteration step. The progression of the computation of the specification Pareto front is illustrated in table 5.3. For simplicity, the SpA is only shown for the statistical parameters.

Progression	It. step	Perf. gradients	Result
Initial	PO 0	-	$\mathbf{d}^{(0)} = \mathbf{d}_{initial};\ \mathbf{s}_0$
SpA step for all f_i	SpA 1	$\nabla_{\mathbf{s}}\, f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_0}$	$\mathbf{s}_{Y,i}^{(0)};\quad i=1,...,n_f$
Spec. value sim.	-	-	$\mathbf{f}_Y(\mathbf{d}^{(0)}) = \begin{bmatrix} f_1(\mathbf{d}^{(0)}, \mathbf{s}_{Y,1}^{(0)}) \\ \vdots \\ f_{n_f}(\mathbf{d}^{(0)}, \mathbf{s}_{Y,n_f}^{(0)}) \end{bmatrix}$
Pareto opt. step	PO 1	$\nabla_{\mathbf{d}}\, f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(0)},\ \mathbf{s}=\mathbf{s}_{Y,i}^{(0)}}$	$\mathbf{d}^{(1)};\quad i=1,...,n_f$
SpA step for all f_i	SpA 2	$\nabla_{\mathbf{s}}\, f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(1)},\ \mathbf{s}=\mathbf{s}_{Y,i}^{(0)}}$	$\mathbf{s}_{Y,i}^{(1)};\quad i=1,...,n_f$
Spec. value sim.	-	-	$\mathbf{f}_Y(\mathbf{d}^{(1)}) = \begin{bmatrix} f_1(\mathbf{d}^{(1)}, \mathbf{s}_{Y,1}^{(1)}) \\ \vdots \\ f_{n_f}(\mathbf{d}^{(1)}, \mathbf{s}_{Y,n_f}^{(1)}) \end{bmatrix}$
Pareto opt. step $\vdots$	PO 2	$\nabla_{\mathbf{d}}\, f_i\big\vert_{\mathbf{d}=\mathbf{d}^{(1)},\ \mathbf{s}=\mathbf{s}_{Y,i}^{(1)}}$	$\mathbf{d}^{(2)};\quad i=1,...,n_f$

Table 5.3: Progress of the computation of the specification Pareto front by running the SpA and Pareto optimization in alternation

This approach conducts the optimization by alternately taking one step in the Pareto optimization and the SpA for each performance instead of running a full SpA in each iteration of the Pareto optimization. The design parameter vector changes in each iteration of the SpA. The SpA converges during the Pareto optimization due to the weak dependency of its solution (the specification parameter vector) on the design parameter vector.

The approach reduces the simulation costs of the computation of the specification Pareto front. The nested loops for this approach are shown in figure 5.6. Only one

sensitivity analysis for the operating and statistical parameters is required for each candidate. The simulation cost can be estimated to:

$$\#\text{ simulations (spec. front, alternating)} = K \cdot R \cdot Q \cdot n_f \cdot (n_s + n_\theta + 1) \tag{5.18}$$

It is a factor S lower compared to the general calculation of the specification Pareto front and a factor $(n_s + n_\theta + 1)$ higher than the calculation of the nominal Pareto front.

In contrast to the subsequent SpA optimization approach, the sensitivities of the performances towards operating and statistical parameters are considered during the optimization. The performances are evaluated at approximated specification parameter vectors. The specification Pareto front is obtained correctly for performances, whose nominal and specification value does not share a common optimum, as discussed in section 5.3.3. This is shown with experimental results in section 6.8 and 6.9.

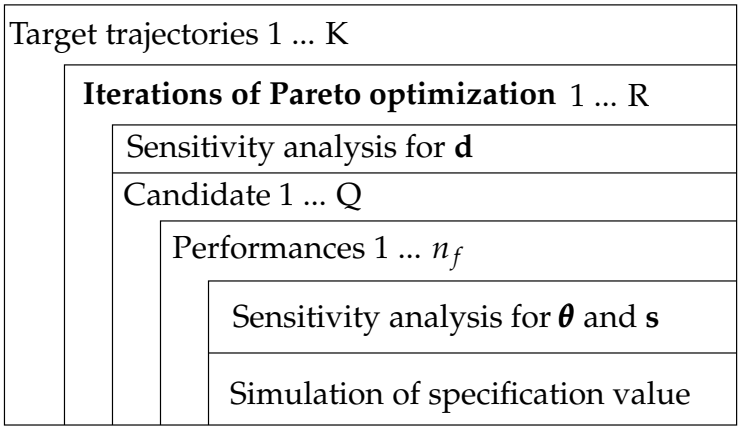


Figure 5.6: Illustration of the approach to speed-up the optimization by running alternating SpA and Pareto optimization.

Another speed-up can be reached by conducting the SpA step only for the most promising candidates that are generated during an Pareto optimization step. This approach is illustrated in figure 5.7. The decision on the best or most promising candidate can not be made based on the specification vector $\mathbf{f}_Y$. This would require to take the SpA step for all candidates first, which is exactly what is attempted to be avoided in this approach. The best candidate can be chosen by looking either at nominal performance values or an approximation of $\mathbf{f}_Y$ using specification parameter vectors of a candidate from the previous iteration. This approach speeds up the alternating SpA and Pareto optimization approach to obtain a very efficient method to compute the specification Pareto front.

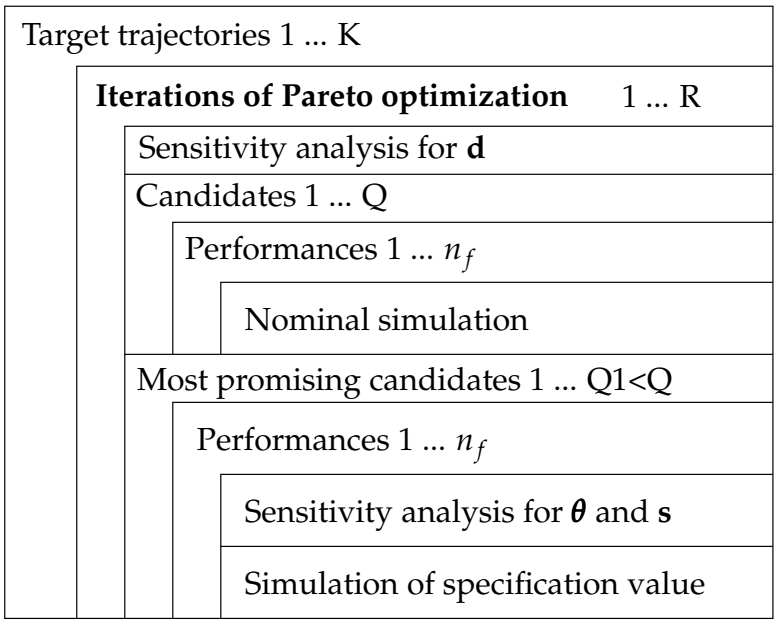


Figure 5.7: Illustration of the approach to speed-up the optimization by running SpA only for the most promising candidates.

5.5 Summary

The specification Pareto front shows the most ambitious specification vectors for a given minimum yield requirement. It requires to conduct specification analysis (SpA) during the Pareto optimization. An optimization based SpA is applied that is derived from geometric worst-case analysis. The application of the optimization based SpA leads to a nested optimization problem. Finding its solution is too costly in terms of computational effort, since it requires to solve the SpA optimization problem for each candidate and performance generated during the Pareto optimization. Two approaches to generate the specification Pareto front with reasonable computational effort were presented: Subsequent SpA after Pareto optimization as well as alternating SpA and Pareto optimization, which can further be speeded up by applying the SpA for most promising candidates only. Both approaches target at reducing the numbers of SpA runs during the computation of the specification Pareto front.

Chapter 6

Experimental Results

This chapter presents experimental results. First, the used CMOS test circuits are described in section 6.1 and implementation details are given in section 6.2. Then, the following experimental results are presented:

- The iterative Pareto generation approach is compared to the NBI approach for an Operational Transconductance Amplifier in section 6.3 and a Voltage Controlled Oscillator in section 6.4.
- The effectiveness of the features of the Wavefront FSQP method are investigated for three Operational Amplifiers in section 6.5 and for a Voltage Controlled Oscillator in section 6.6. The Wavefront FSQP algorithm is compared to a general purpose SQP algorithm for the three Operational Amplifiers in section 6.7.
- The specification Pareto front is computed for a Voltage Controlled Oscillator in section 6.8 and one Operational Amplifier in section 6.9.

Additionally three application examples for the generated Pareto fronts are given:

- The Pareto fronts of three Operational Amplifiers are used to conduct a structure selection in section 6.10.1.
- The Pareto front of the Voltage Controlled Oscillator is used to implement a behavioral model that includes a description of the realizable performance compromises of the circuit structure in section 6.10.2.
- The Pareto front of a Phased Lock Loop is used for a trade-off analysis in section 6.10.3.

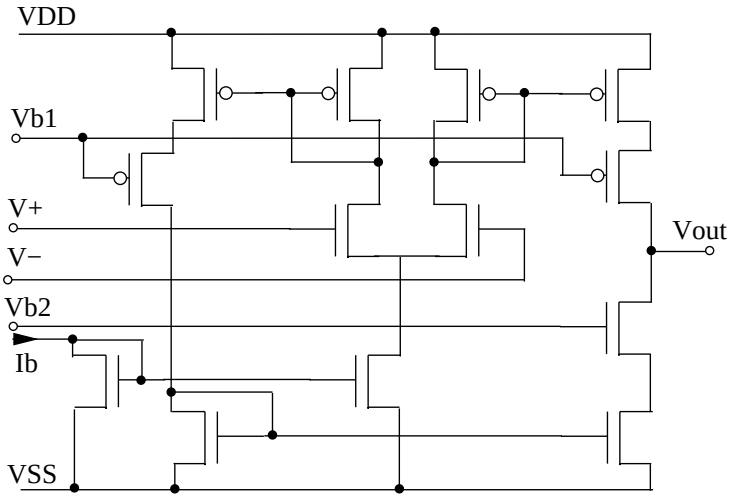


Figure 6.1: Structure of the CMOS Operational Transconductance Amplifier

6.1 Test Circuits

This section gives an overview of the used test circuits.

6.1.1 Operational Transconductance Amplifier (OTA)

Figure 6.1 shows the schematic of a CMOS Operational Transconductance Amplifier (OTA). The OTA is implemented in an industrial 0.8 μm technology with a supply voltage of 3.3 V. It features 11 tunable design parameters and 97 sizing rules.

6.1.2 CMOS Operational Amplifiers (OpAmps)

Table 6.1 shows an overview over three different CMOS operational amplifier (OpAmp) circuits. All circuits are implemented in an industrial 180 nm technology with a supply voltage of 2.2 V. The process variations are modeled with 13 globally as well as 2 locally varying statistical parameters for each transistor. Changes in the supply voltage and temperature are modeled with two operating parameters.

6.1.3 Voltaged Controlled 5-Stage Ring Oscillator (VCO)

Figure 6.2 shows the schematic of a voltaged controlled 5-stage ring oscillator (VCO). The VCO is implemented in a non-industrial artificial technology that models a 180 nm process. It has a nominal supply voltage of v_{DD}=1.8 V. The controlling voltage has an input range of $v_{c,min}$=0.6 V to $v_{c,max}$=1.4 V.

Table 6.1: Overview: Operational Amplifiers

Circuit	Nr. transistors	Nr. design parameters	Nr. operating parameters	Nr. statistical parameters	Nr. sizing rules
Opamp A	8	11	2	29	62
Opamp B	20	14	2	53	162
Opamp C	26	14	2	65	206

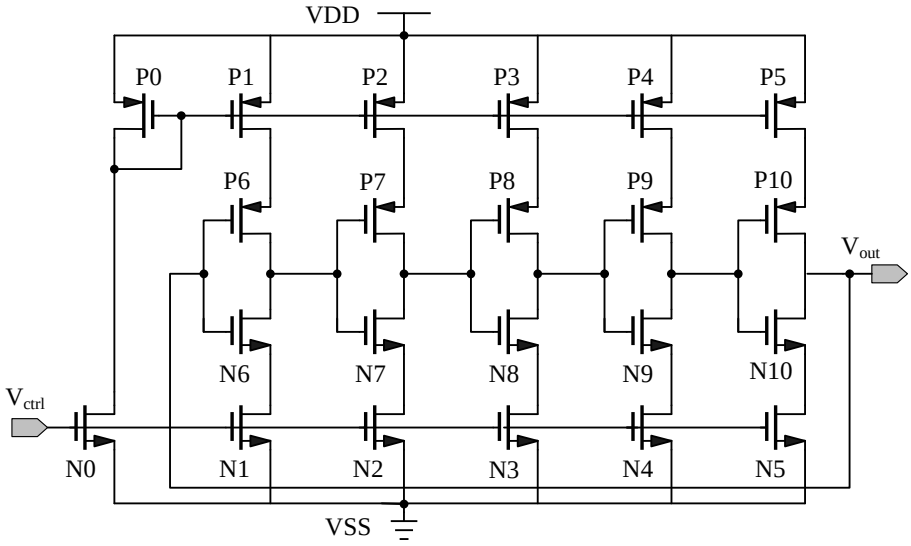


Figure 6.2: Structure of the voltage controlled 5-stage ring oscillator (VCO)

The operating parameters considered are supply voltage v_{DD}, which ranges from 1.6 V to 2.0 V and and the temperature T (in degree Celsius), with a nominal T_{nom}=27^o and a specified temperature range from -20^o to 100^o. Global variations of the threshold voltage $v_{th,n}$ and $v_{th,p}$, oxide thickness t_{ox} and carrier mobility $\mu_{0,p}$ and $\mu_{0,n}$ are considered as process parameters.

6.1.4 Charge Pump Phased Locked Loop (CPPLL)

Figure 1.4 shows the block diagram of a charge pump phased locked loop. All blocks are implemented with Verilog-A behavioral models to allow a fast simulation of the complete circuit. A 2-stage hierarchical simulation approach is used for the Pareto optimization of the circuit. First each block is simulated on transistor level. The performances are used to determine the behavioral model parameters for a simulation of the complete circuit on behavioral level. The circuit features 22 design parameters, 8 for the VCO, 9 for the charge pump and 3 for the loop filter.

6.2 Implementation Details

All computations were run on a dual-quadcore 1.86 GHz Xeon PC with 4 GB RAM. The Wavefront FSQP algorithm was implemented in C++. The tool WiCkeD [MAW07] from MunEDA was used as simulation server. Titan [FWZ+92] from Infineon was used as analog simulator for the OTA and OpAmp circuits and Spectre [Kun95] from Cadence [Cad] for the VCO circuit.

6.3 Comparison of Novel Iterative and NBI Approach for the OTA

The iterative Pareto front generation approach, which was presented in section 3.7, is compared with the state-of-the-art Normal Boundary Intersection approach for the OTA circuit. Both approaches have been discussed in chapter 3.

Setup: The trade-off between the three performances transconductance, bandwidth and power consumption of the OTA is analyzed. The density of the Pareto optimal performance vectors on the discretized Pareto front was chosen to $D = 8$ (See sec. 3.6.3). This leads to 55 Pareto-optimal performance vectors (See eq. (3.33)).

Discussion of the resulting Pareto fronts: Figure 6.3 shows the generated discretized Pareto fronts for both approaches. The Pareto-optimal performance vectors are triangulated to draw a surface approximation for a better visualization. For comparison, the Pareto front of the iterative approach is shown shaded in the plot of the NBI approach. As can be seen, peripheral regions of the Pareto front, which are covered by the novel iterative approach, are not covered by the NBI approach.

Computational time: The Wavefront FSQP algorithm presented in chapter 4 was used to generate both Pareto fronts. The computation of the Pareto front for the novel iterative approach took 7,2 hours. The computation of the discretized Pareto front for the NBI approach took 4,3 hours.

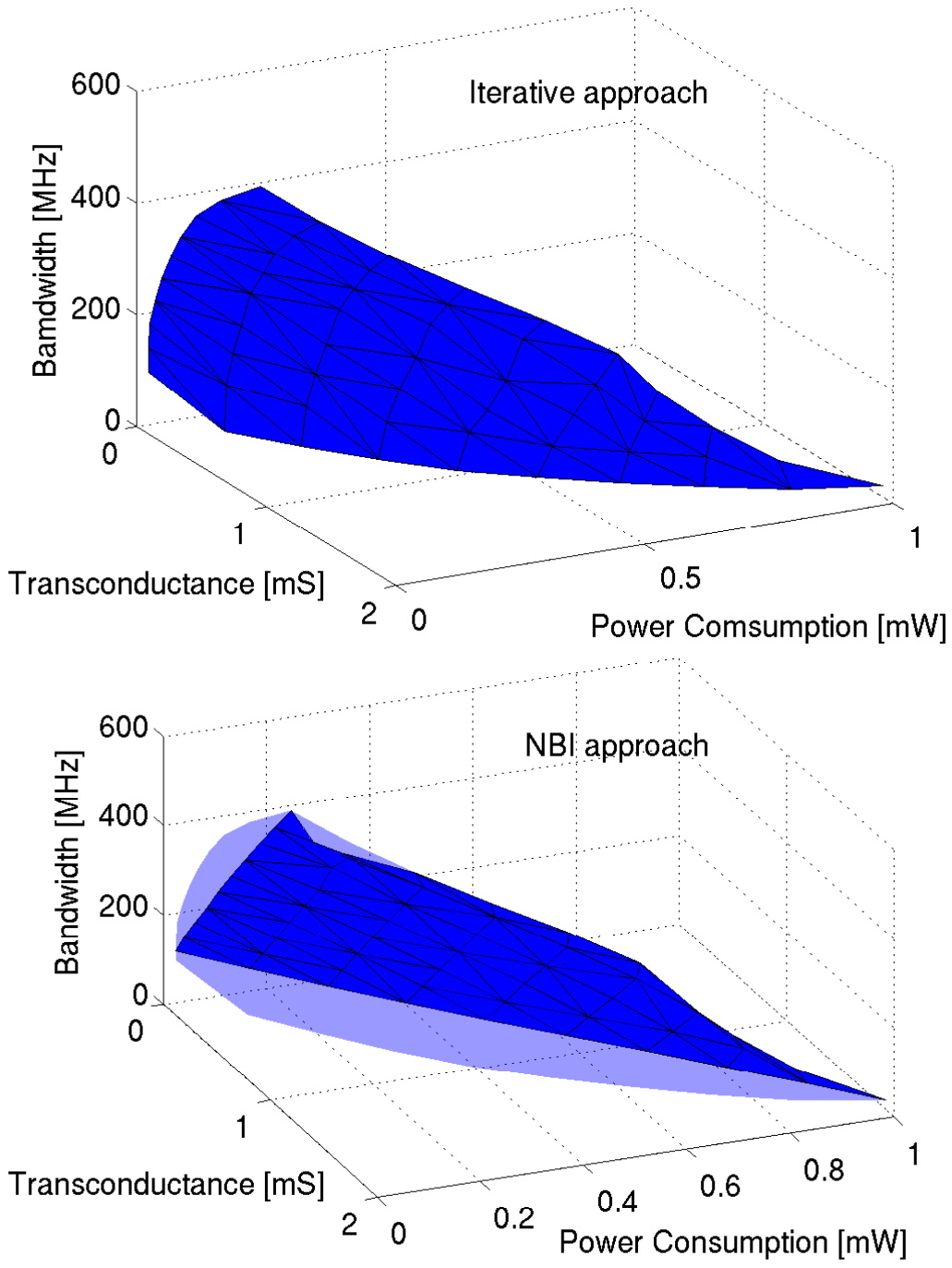


Figure 6.3: Pareto Fronts for a Operational Transconductance Amplifier for the performances Transconductance, Bandwidth and Power Consumption. For a better visualisation, a surface is shown that is generated by triangulation of the Pareto-optimal performance vectors. *Upper*, the triangulated surface obtained by the iterative approach. *Lower*, the triangulated surface obtained by the NBI approach and, shaded, the triangulated surface obtained by the iterative approach for comparison.

6.4 Comparison of Novel Iterative and NBI Approach for the VCO

The experiment is repeated for the VCO circuit. Again, the iterative Pareto front generation approach is compared with the state-of-the-art Normal Boundary Intersection approach.

Setup: The trade-off between the three performances VCO gain, supply current and VCO jitter is analyzed. The density of the Pareto optimal performance vectors on the discretized Pareto front was again chosen to $D = 8$. 55 Pareto-optimal performance vectors are generated for both fronts.

Discussion of the resulting Pareto fronts: Figure 6.4 shows the generated discretized Pareto fronts for both approaches. The iterative approach again covers large regions of the Pareto front that are not covered by the NBI approach.

Computational time: The computation of the Pareto front for the novel iterative approach took around 25 hours. The computation of the discretized Pareto front for the NBI approach around 28 hours.

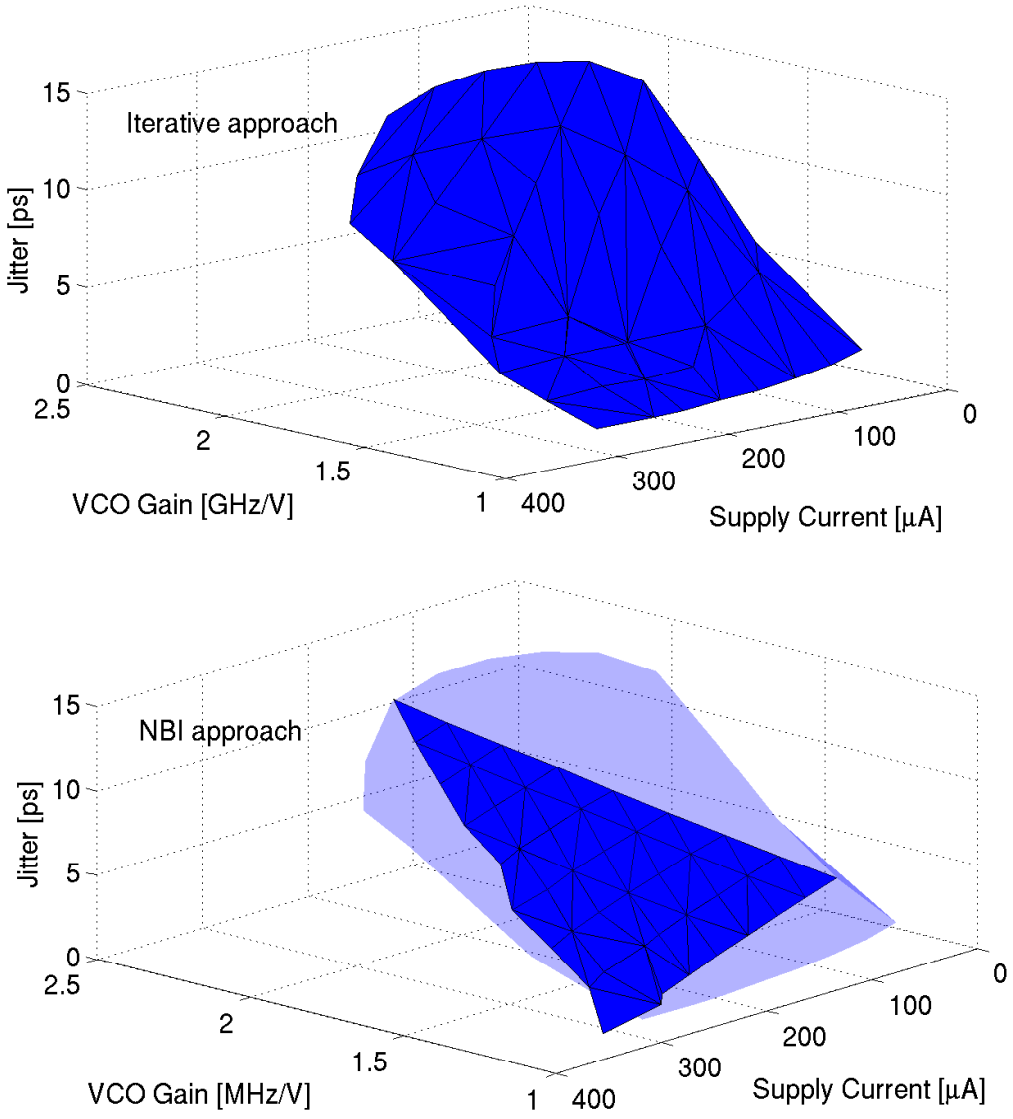


Figure 6.4: Pareto Fronts for the VCO Circuit for the performances VCO gain, supply current and VCO jitter. For a better visualization, a surface is shown that is generated by triangulation of the Pareto-optimal performance vectors. *Upper*, the triangulated surface obtained by the iterative approach. *Lower*, the triangulated surface obtained by the NBI approach and, shaded, the triangulated surface obtained by the iterative approach for comparison.

6.5 Evaluation of Wavefront FSQP Features for the Three OpAmps

In this experiment, the effectiveness of the different Wavefront FSQP features that were introduced in chapter 4 is investigated.

Setup: The nominal Pareto fronts for all three OpAmp circuits are computed for the performances slew rate and power consumption. The optimization is conducted and the features of the Wavefront FSQP algorithm are activated one-by one as follows:

1. STD: The optimization algorithm only looks for better candidates along the standard search direction.
2. SOC: The optimization algorithm looks for better candidates along the standard search direction. For infeasible candidates, a second-order correction step is applied.
3. TILT: The optimization algorithm looks for better candidates along the standard search direction and the tilted search direction. For infeasible candidates, a second-order correction step is applied.
4. WAV: The optimization algorithm looks for better candidates along the standard search direction and the tilted search direction. For infeasible candidates, a second-order correction step is applied. Additionally, the Wavefront approach is used: The CNOPs exchange solutions.

A discretized Pareto front with 16 Pareto-optimal performance vectors is computed for each configuration of the algorithm. A set of specifications, which are shown in table 6.9, is set on the remaining performances.

Discussion of resulting Pareto fronts: The results are shown in figure 6.5. As can be seen, the quality of resulting discretized approximation of the Pareto front improves with the activation of additional features of the Wavefront FSQP algorithm. Without tilting (STD and SOC), the computed discretized Pareto fronts show only parts of the performance capabilities of the circuit. Additionally sub-optimal performance vectors are computed, especially for OpAmp B. The optimization with tilting (TILT) results in a discretized Pareto front that shows a wider range of the performance capabilities of the OpAmp circuits. The activation of the exchange of solutions (WAV) shows again an improvement in the resulting discretized approximation of the Pareto front compared to the case without exchange (TILT), but not as drastically as between with tilting (TILT) and without tilting (STD and SOC).

Computational costs: Tables 6.2 shows the computational time and number of simulations for all optimization runs. One simulation includes the evaluation of all

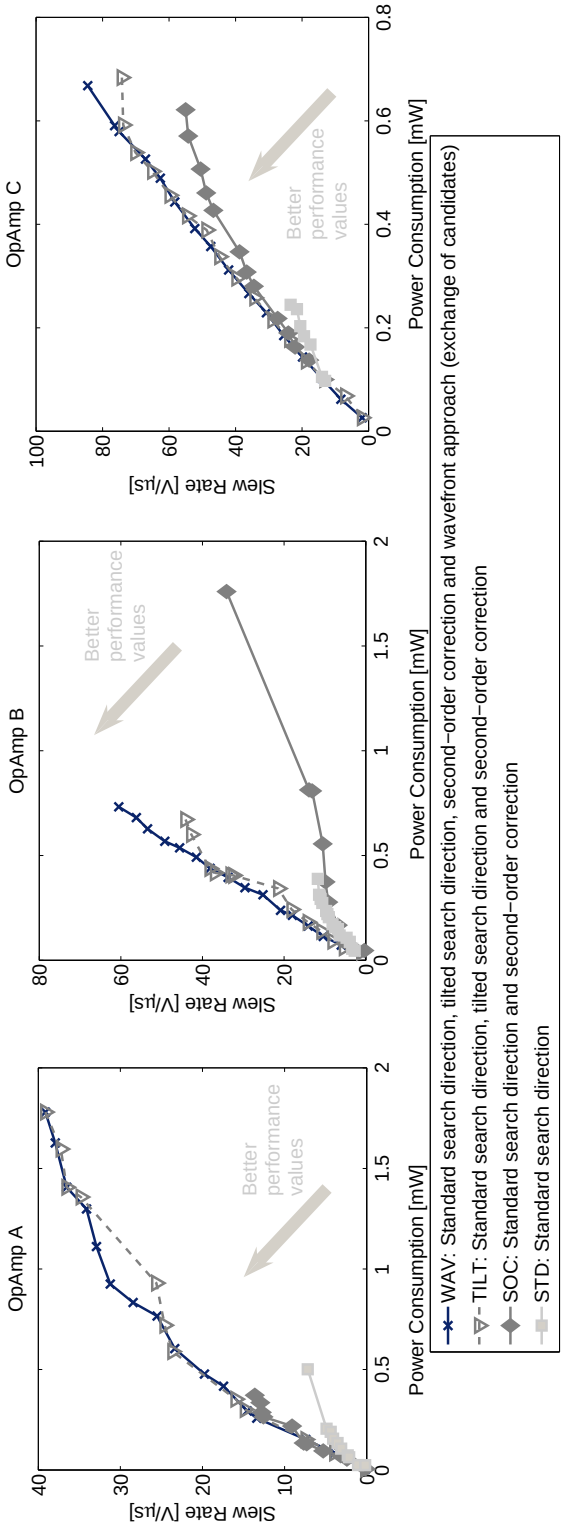


Figure 6.5: Evaluation of Wavefront FSQP features for OpAmp Circuits

performances for one candidate. The activation of the different features of the Wavefront method leads in all cases to increased simulation costs and longer computational time. This is caused by additional candidates that are computed by second-order correction or for the tilted search direction. Secondly, the overall number of iterations increases. With additional features the optimization terminates after a higher number of iteration steps, which is shown in tables 6.3, 6.4 and 6.5.

Table 6.2: Computational time and number of simulations of the Wavefront FSQP algorithm with different features activated.

Opt.	OpAmp A		OpAmp B		OpAmp C	
	Comp. time	Nr. of simulations	Comp. time	Nr. of simulations	Comp. time	Nr. of simulations
STD	24 min	428	50 min	1039	1 h 13 min	943
SOC	1 h 31 min	1802	1 h 05 min	823	2 h 18 min	1680
TILT	2 h 32 min	3692	5 h 29 min	5341	6 h 57 min	5558
WAV	4h 37 min	6226	11h 30 min	11088	10 h 16 min	7858

Discussion of the significance of the features: In each iteration, the optimization selects one new candidate with the largest improvement in the objective function value compared to the best candidate of the last iteration. The candidates can be classified according to the following cases:

1. SSD: Candidate that was found with the standard search direction.
2. SSD+SOC Candidate that was found with the standard search direction after application of a second-order correction step.
3. TSD: Candidate that was found with the tilted search direction.
4. SSD+SOC Candidate that was found with the tilted search direction after application of a second-order correction step.

A feature that generates a certain class of candidates does not contribute to the progress of the optimization, if it never generates the candidate with the largest improvement in the objective function value.

Tables 6.3, 6.4 and 6.5 give an overview over the class of candidates that had the largest improvement in the objective function value in one of the iteration steps. The progress in all 16 CNOPs is shown together. As can be seen, every class of candidates contributes to the progress of the optimization. All features are significant for the Pareto optimization of the three OpAmp test circuits.

For the optimization with the Wavefront approach, the number of exchanges of solutions is also shown. As can be seen, the exchange of solutions occurs in a significant number of iteration steps. Candidates from one CNOP regularly create the largest

improvement in the objective function of another CNOP. For the OpAmp B, an exchange occurred in 132 of 435 iteration steps.

Table 6.3: Nr. of selected candidates encountered during optimization by different features to generate new candidates for OpAmp A

Opt.	Nr. total it. steps	Nr. selected candidates generated by SSD	SSD+SOC	TSD	TSD+SOC	Nr. Exchanges
STD	41	41	-	-	-	-
SOC	124	21	103	-	-	-
TILT	135	41	19	17	58	-
WAV	254	46	27	101	80	49

Table 6.4: Nr. of selected candidates encountered during optimization by different features to generate new candidates for OpAmp B

Opt.	Nr. total it. steps	Nr. selected candidates generated by SSD	SSD+SOC	TSD	TSD+SOC	Nr. Exchanges
STD	80	80	-	-	-	-
SOC	125	108	17	-	-	-
TILT	202	66	29	84	23	-
WAV	435	94	35	272	34	132

Table 6.5: Nr. of selected candidates encountered during optimization by different features to generate new candidates for OpAmp C

Opt.	Nr. total it. steps	Nr. selected candidates generated by SSD	SSD+SOC	TSD	TSD+SOC	Nr. Exchanges
STD	120	120	-	-	-	-
SOC	142	61	81	-	-	-
TILT	229	87	41	92	9	-
WAV	307	117	46	128	16	39

6.6 Evaluation of Wavefront FSQP Features for the VCO

In this experiment, the effectiveness of the different Wavefront FSQP features that were introduced in chapter 4 is investigated for another circuit class. The setup is the same as described in section 6.5. The Pareto fronts are generated for the VCO circuit for the performances VCO Gain and Jitter. The density is D=8, such that 10 Pareto-optimal performance vectors are generated for each configuration of the Wavefront FSQP algorithm.

Discussion of resulting Pareto fronts: Figure 6.6 shows the resulting discretized Pareto fronts for the four configurations of the Wavefront FSQP algorithm that are described in section 6.5. As can be seen, the activation of the second-order correction (SOC) generates a discretized Pareto front that shows almost the complete range of performance capabilities. The tilting (TILT) and exchange of solutions (WAV) can achieve slightly superior solutions at the *'knee'* of the discretized Pareto front.

It can be noted that the optimization for some circuits or performances requires more sophisticated optimization strategies as for others. The discretized Pareto fronts of the OpAmps shown in section 6.5 required second-order correction, tilting and the Wavefront approach in order to show the complete trade-off range of the performances. In contrast, the VCO requires only second-order correction to gain reasonable insight in the performance capabilities.

Computational time and significance of FSQP features: Table 6.6 shows the computational time and number of candidates of the different classes introduced in section 6.5. The tilting doubles the simulation costs, caused by conducting two line searches instead of a single one. As can be seen, the tilting is not required as often as for the OpAmp circuits. The standard search direction with/without second-order correction generate the largest part of the candidates with the highest improvement in the objective function.

Table 6.6: Overview: Iteration steps and computational time for the VCO

Opt.	Nr. total it. steps	Nr. selected candidates found by SSD	SSD+SOC	TSD	TSD+SOC	Nr. Exchanges	Comp. Time
STD	98	98	-	-	-	-	2 h 11 min
SOC	46	32	14	-	-	-	2 h 42 min
TILT	83	42	24	15	2	-	5 h 16 min
WAV	90	65	14	10	2	9	4 h 30 min

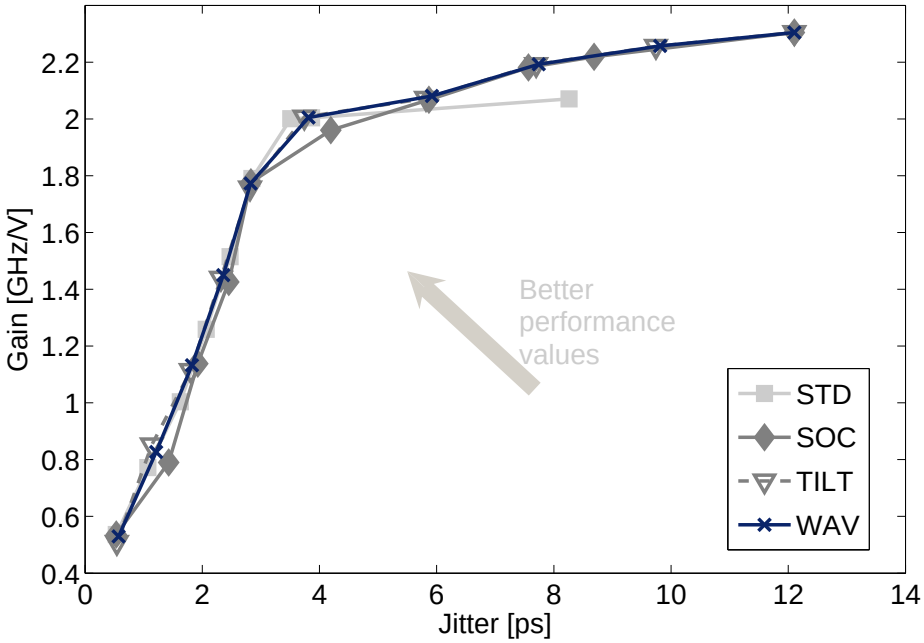


Figure 6.6: Evaluation of Wavefront FSQP features for VCO Circuit

6.7 Comparison of Wavefront FSQP to General Purpose SQP for the Three OpAmps

In this experiment, the Pareto fronts of the OpAmp circuits that were generated by the Wavefront FSQP algorithm are compared to the Pareto fronts found by a general purpose SQP algorithm. It is a line search based SQP algorithm that uses an L1 cost function, which accepts constraint violations.

A meaningful comparison of different optimization algorithms for Pareto optimization is very difficult. It would require to investigate different circuit classes as well as different performance trade-offs. Additionally, each investigation must be conducted several times, since the results are usually also depending on the initial sizing supplied to the optimizer. Therefore, this experiment can only give a basic impression of the efficiency of the Wavefront FSQP algorithm compared to the standard SQP optimizers. The criteria used for comparison are:

- The quality of the solutions. Are the solutions of the Wavefront FSQP algorithm Pareto-better or Pareto-worse than the solutions found by the general purpose SQP method?
- The spread of the solutions. Is the complete trade-off range covered or does the general purpose SQP algorithm generate compromises between the performances that are not visible in the discretized Pareto front found by the Wavefront FSQP algorithm?
- Computational time and required number of simulations.

Setup: The three OpAmps are used as test circuits. The Pareto front found by the setup WAV according to section 6.5 is used for comparison. In this setup all features of the Wavefront FSQP algorithm are used. A number of 16 Pareto-optimal performance vectors are computed with the general purpose SQP method. The general purpose SQP algorithm regularly returned solutions that featured violations in the constraints. These infeasible solutions are removed subsequently from the discretized Pareto front because they are no valid sizings of the circuit.

Comparison of resulting Pareto fronts: Figure 6.7 shows the resulting discretized Pareto fronts. The general purpose SQP algorithm only returns two feasible solutions for OpAmp A. It accepts constraint violations during the optimization and later fails in 14 CNOPs to converge to a feasible solution. For OpAmp B, the general purpose method generates ten feasible solutions. These cover the complete trade-off range but one solution is clearly non-optimal. The others are slightly inferior to the solutions found by the Wavefront FSQP method. Finally, for OpAmp three, the SQP method generates twelve feasible solutions. Again these solutions are slightly inferior compared to the Pareto front generated by the Wavefront FSQP method and, additionally, do not cover the complete range of trade-offs.

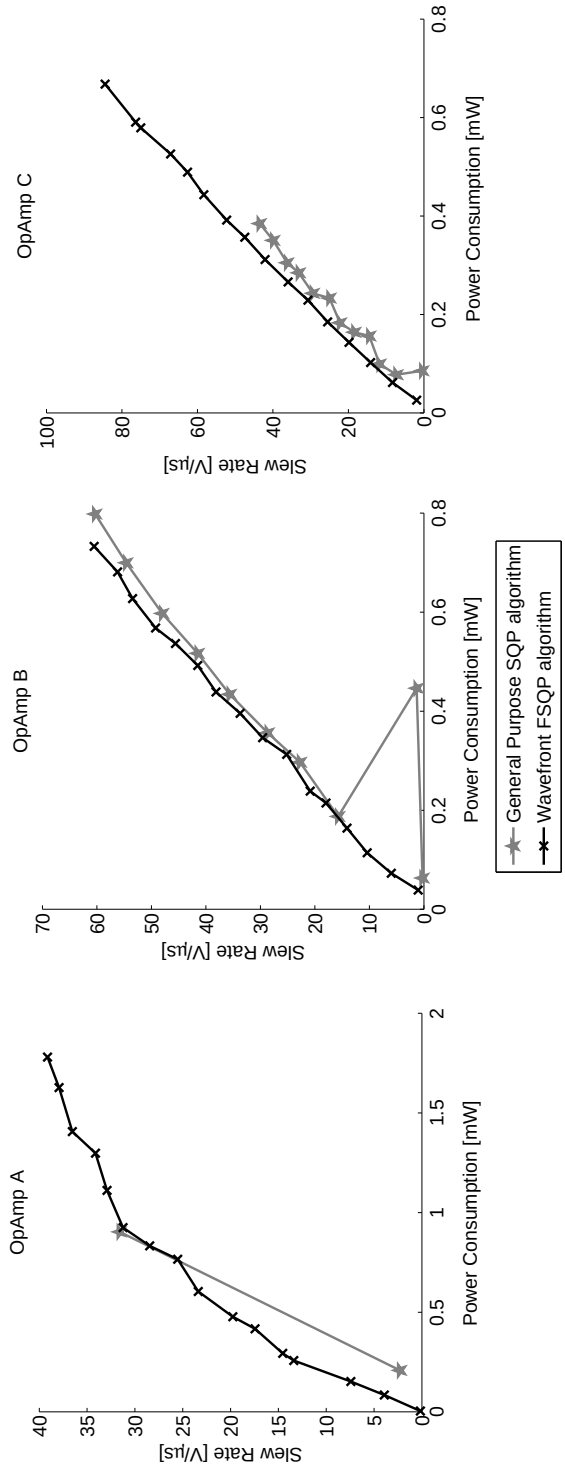


Figure 6.7: Comparison of Wavefront FSQP to general-purpose SQP algorithm for three OpAmp circuits

Comparison of computational time and costs: The number of required simulations is significantly higher for the Wavefront FSQP algorithm. This is caused by the need of two line searches and the parallel generation of new candidates. Due to the efficient use of parallelism, the computational time is shorter for the Wavefront FSQP method for the generation of two of the three Pareto fronts.

Table 6.7: Overview: Comparison of computational time for Wavefront FSQP algorithm (WAV) and a general purpose SQP algorithm (SQP)

Opt.	OpAmp A		OpAmp B		OpAmp C	
	Comp. time	Nr. of simulations	Comp. time	Nr. of simulations	Comp. time	Nr. of simulations
WAV	4 h 37 min	6226	11 h 30 min	11088	10 h 16 min	7858
SQP	8 h 31 min	2059	7 h 26min	1074	11 h 19 min	1797

6.8 Specification Pareto Front for the VCO

In this experiment, the discretized specification Pareto front of the VCO circuit is computed with the approaches presented in chapter 5.

Setup: The discretized Pareto fronts are generated for the VCO gain and the supply current. First, the nominal Pareto front is generated. Then a SpA is conducted at each nominal Pareto-optimal performance vector to obtain the most ambitious specification vector that can be realized for a minimum yield requirement of $Y_g \geq Y_{min} = 99.73\%$. This minimum total yield requires an individual yield for each performance of $Y_0 = 99.865\%$. The SpA is conducted with a tolerance region defined by $\beta_Y = 3.0$. See section 5.2.2 for further details. The resulting specification vectors compose the specification Pareto front as found by the approach - SpA after Pareto optimization - that was discussed in section 5.4.1.

In a second run, the specification Pareto front is computed with the approach that conducts an alternating SpA and Pareto optimization, see sec. 5.4.2. The approach additionally uses the method that runs a SpA for the most promising candidates only.

Computational time: The computational time for the generating of the discretized Pareto fronts were 3h 52min for the SpA after Pareto optimization approach and 8h 14 min for the alternating SpA and Pareto optimization.

Discussion of resulting Pareto fronts: The resulting discretized nominal Pareto front and discretized specification Pareto fronts are shown in figure 6.8. The nominal performance vectors are, as expected, superior to the specification vectors. The specification vectors consider the degradations to the performances due to changing operating conditions and process variations. The specification Pareto fronts found by the two approaches are composed of equally optimal specification vectors. It can be followed, that the nominal values and specification values share a common optimum for these performances as discussed in section 5.3.3. It would have been sufficient to compute the nominal Pareto front and conduct the SpA subsequently.

Verification of minimum yield requirement with Monte Carlo analysis: A Monte-Carlo analysis (MCA) with 15 000 sample elements is conducted for the alternating SpA and Pareto optimization approach. In order to verify that the minimum yield requirement is met, the yield is measured in regard to the corresponding specification parameter vectors that are shown in figure 6.8. One design parameter vector was computed for each point of the specification Pareto front. Eight Monte-Carlo samples are required at the eight different design parameter vectors, each with 15 000 sample elements.

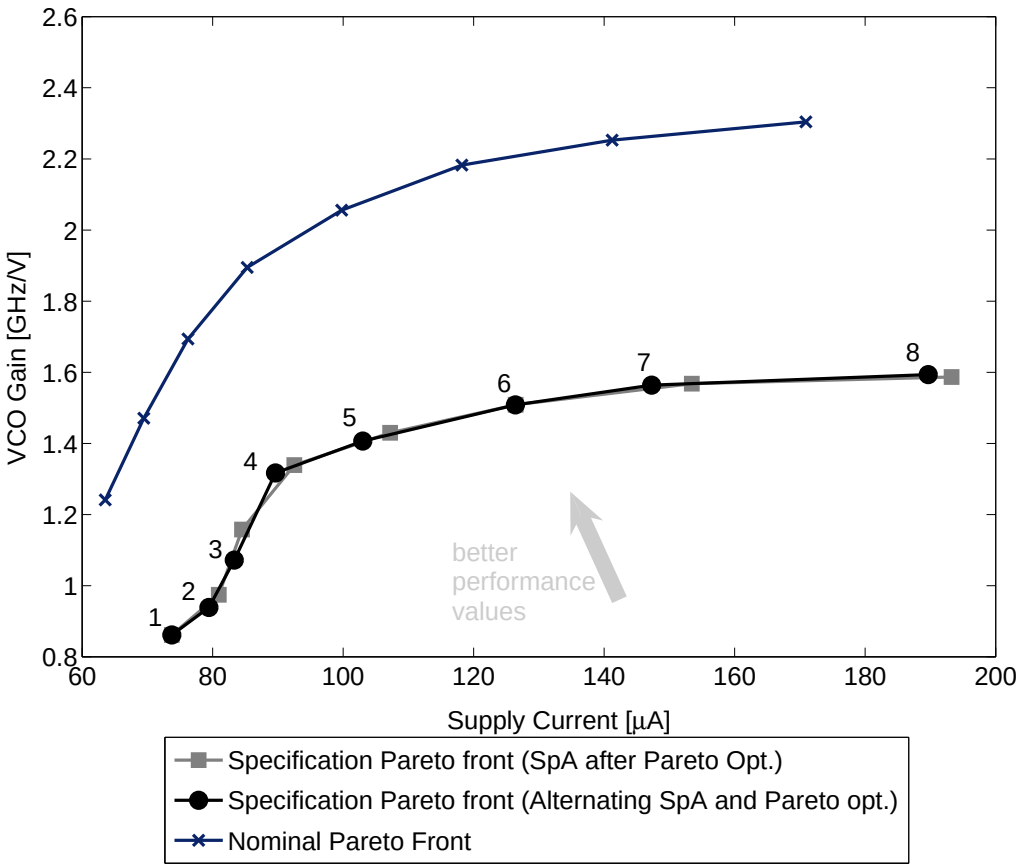


Figure 6.8: Nominal and Specification Pareto Fronts for VCO circuit

The demanded total minimum yield is 99.73%. This lead to individual yield requirements of 99.865% in regard to each individual performance specification. This is a lower bound consideration, such that the total yield should be equal or higher to the demanded minimum yield. Table 6.8 shows the specification values that are equal to the Pareto-optimal specification vectors, which compose the specification Pareto front. The estimated yield in regard to each performance specification as well as the total yield is also listed for each point on the specification Pareto front. The total yield meets for all solutions the minimum yield requirement. The MCA also shows that the specification analysis is slightly too pessimistic in the calculation of the VCO supply current specification values. This high accuracy Monte-Carlo analysis to verify the results took about 76 hours compared to about 8h for the computation of the complete specification Pareto front.

6.9 Specification Pareto front for one OpAmp

Setup: The OpAmp A is used as test circuit. The considered performances are power source rejection ratio (PSRR) and the transit frequency. There is an inherent

Table 6.8: Yield values for specification vectors on specification Pareto front

Specification values			Yield [%]		
Nr.	Current [μA]	Gain [MHz/V]	(Current)	(Gain)	Total
1	73.85	853.9	99.92	99.85	99.77
2	80.19	939.4	99.97	99.85	99.82
3	82.69	1074	99.95	99.88	99.84
4	90.49	1317	99.95	99.88	99.83
5	104.4	1406	99.98	99.87	99.85
6	128.3	1508	99.96	99.88	99.84
7	150.0	1564	99.98	99.89	99.87
8	189.6	1590	99.95	99.89	99.84

trade-off between these two performances: A high transit frequency requires small transistor sizes. High PSRR is reached by better matching of transistors, which requires large transistor sizes. In addition to the saturation constraints, a minimum DC Gain of 70 dB and a minimum Phase Margin of 60 degree is demanded.

The specification Pareto front for a minimum yield requirement of $Y_g \geq Y_{min} = 99.73\%$ is computed. This requires an individual yield requirement for both performances of $Y_0 = 99.865\%$ or $\beta_Y = 3.0$. See section 5.2.2 for details. Such a design with $\beta_Y = 3.0$ is also known as 3-sigma design.

The specification Pareto front is computed once with the approach that runs the SpA subsequently after the Pareto optimization, see sec. 5.4.1, and once with the approach with alternating SpA and Pareto optimization, see sec. 5.4.2. The approach with alternating SpA/Par. Opt. is further speeded up by applying the SpA only for the most promising candidates.

Discussion of resulting specification Pareto fronts: Figure 6.9 illustrates the resulting Specification Pareto fronts for the two approaches. The specification Pareto front found by SpA after Pareto optimization shows inferior specification values compared to the Specification Pareto front found by the alternating approach.

The PSRR value depends heavily on the matching of some transistors. These transistors must be large to decrease the mismatch due to local process variations. The *'SpA after Pareto optimization'* approach only optimizes nominal performance values. As can be seen in figure 6.9, the nominal performance values of the solution of the *'SpA after Pareto optimization approach'* show optimal performance values. In the nominal case, the transistors are perfectly matching, such that the influence of the local process variations is not visible. As can be seen, the resulting solutions are very sensitive to these local process variations since the computed specification values for 99.73% yield only reach PSRR values of about 72dB.

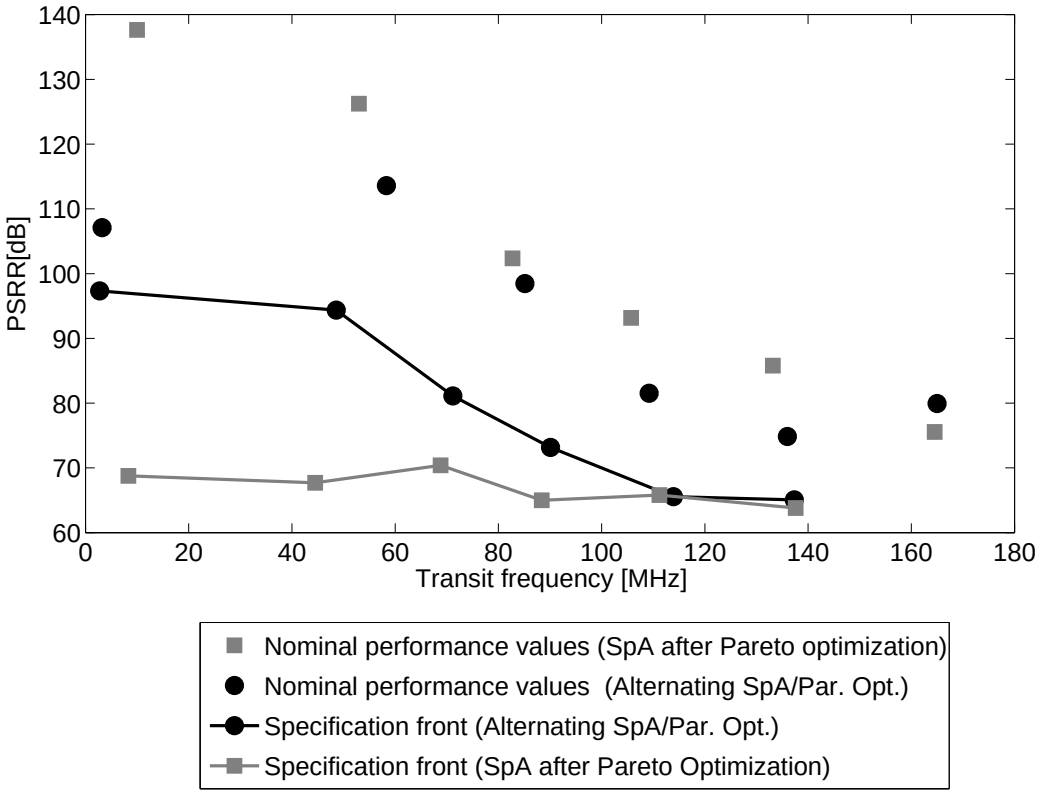


Figure 6.9: Specification Pareto Fronts for OpAmp A

The *'alternating SpA and Pareto optimization approach'* considers the influence of the process variations during the Pareto optimization. It computes solutions that are less sensitive to these variations and show superior specification values with PSRR values up to 98dB. These solutions have inferior nominal performance values. This shows that the performance PSRR does not share a common optimum for the specification value and nominal value as was discussed in section 5.3.3. The alternating approach is required to generate the specification Pareto front.

Investigation with Monte-Carlo analysis: The difference in the Specification Pareto fronts is further investigated by running a Monte Carlo analysis (MCA) with 5000 sample elements for each obtained Pareto-optimal solution. The resulting performance distributions are shown together with the specification Pareto fronts in figure 6.10.

The spread of the PSRR performance values is larger for the solutions obtained by the SpA after Pareto optimization approach. The sensitivity of the PSRR values on the process variations is higher. The alternating approach generates solutions that have less sensitive PSRR values. This results in the superior specification values compared to the SpA after Pareto optimization approach.

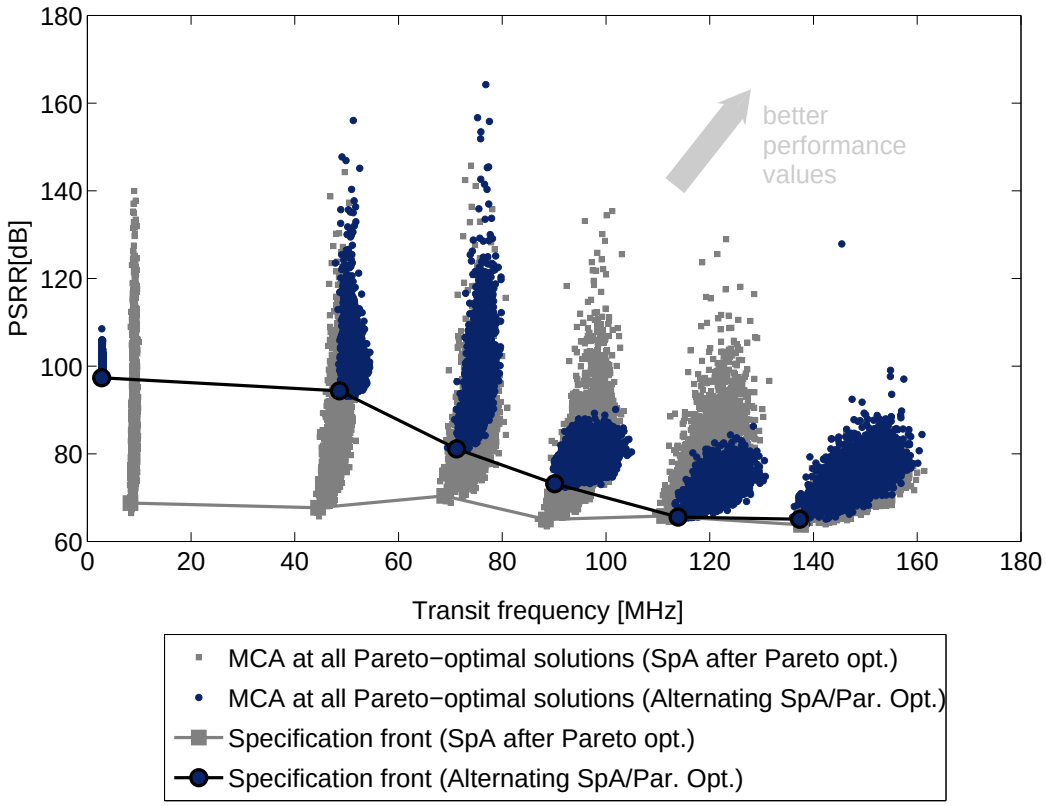


Figure 6.10: Monte-Carlo analysis for specification Pareto Fronts for OpAmp A

6.10 Application Examples

In the following, two examples for the application of the Pareto front of analog circuit structures are given. Firstly, a trade-off analysis and structure selection for the three OpAmp circuits is described. Secondly, the development of a VCO behavioral model that considers the trade-off between VCO gain and supply current is presented.

6.10.1 Selection of OpAmp Structure

Motivation: One application of the Pareto front is the selection of a suitable circuit structure by looking at the performance capabilities of possible alternatives. This is illustrated for the three OpAmp circuit structures. As can be seen in table 6.1, the OpAmps differ in number of transistors.

Generation of the Pareto fronts: For each of the OpAmp circuits, the Pareto front for the performances slew rate and power consumption is computed. An application-dependent set of specifications is used for the remaining performances, which is shown in table 6.9. It assures stability, a minimum gain, and appropriate rejection ratios.

Trade-off analysis and structure selection: The Pareto fronts of the three OpAmp structures are shown in figure 6.11. The figure shows that the power consumption is decreased by switching From OpAmp A to B and from OpAmp B to C for a fixed slew rate value. A comparison with table 6.1 reveals that this is equal to choosing a structure with more transistors. In this case, the designer has two degrees of freedom. He can implement a structure with higher slew rate either by accepting a higher power consumption or by switching to a structure with higher number of transistors.

Table 6.9: Application dependent specifications for the OpAmp circuits

Performance	Spec
DC gain	> 65 dB
Phase Margin	> 56^o
CMRR	> 80 dB
PSRR	> 70 dB

6.10.2 Development of a Trade-off Behavioral Model For the VCO

Motivation: For large-scale circuits, the time to simulate the complete circuit on transistor level becomes too long. Behavioral models are used to represent analog

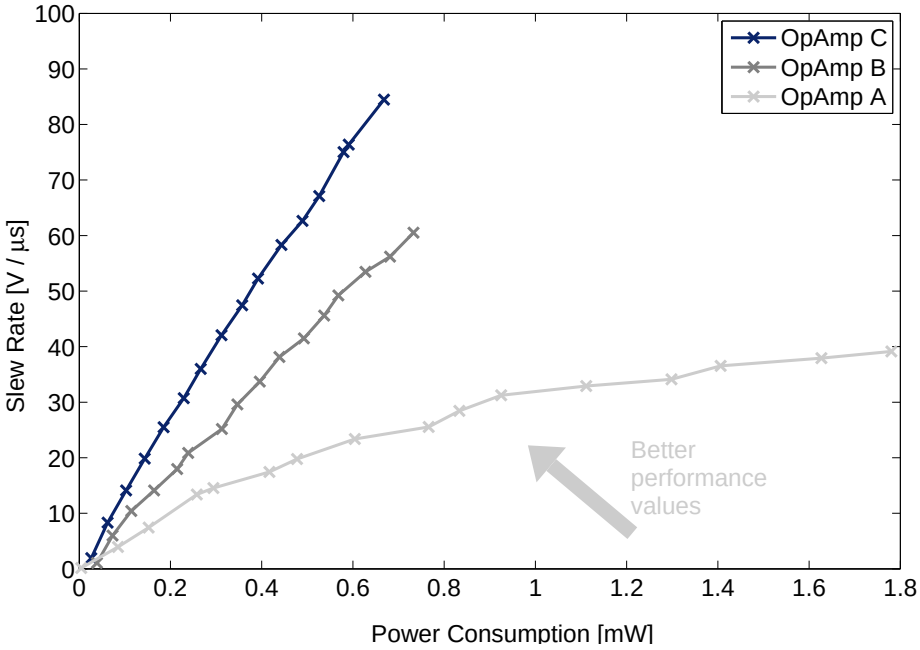


Figure 6.11: Topology Selection for OpAmp Circuits

circuit blocks in such circuits. These behavioral models allow a fast simulation of the circuit but at the expense of loosing implementation details. The behavioral models are, usually, parametrized to represent different circuit implementations. The parameters of such behavioral models are circuit performances such as gain or bandwidth.

Pareto fronts can be used to extract realizable ranges as well as trade-offs between the behavioral model parameters. This allows to apply realistic values to these model parameters and assures that a circuit can be implemented with the properties described by the behavioral model. This is shown in the following for the VCO circuit.

Trade-off extraction of VCO performances: The extraction of the trade-offs and realizable ranges is illustrated for the VCO circuit. The VCO output frequency f_{out} is modeled as a linear function of the controlling input voltage V_{in} and the VCO gain K_{vco}:

$$f_{out} = (V_{in} - V_{min}) \cdot K_{vco} + f_{min} \tag{6.1}$$

The VCO gain depends on the circuit implementation. In order to assure that the VCO gain can be reached by a given circuit structure, all realizable values of the VCO gain must be computed. Additionally, higher gain will also lead to a higher required supply current I_{vco}. This trade-off can be considered in the behavioral model by generating the Pareto front between the supply current and gain of the VCO, which is shown in figure 6.12.

In order to implement the trade-off in the behavioral model, the relation between the current and gain is modeled by fitting a 4-th order polynomial function to the Pareto

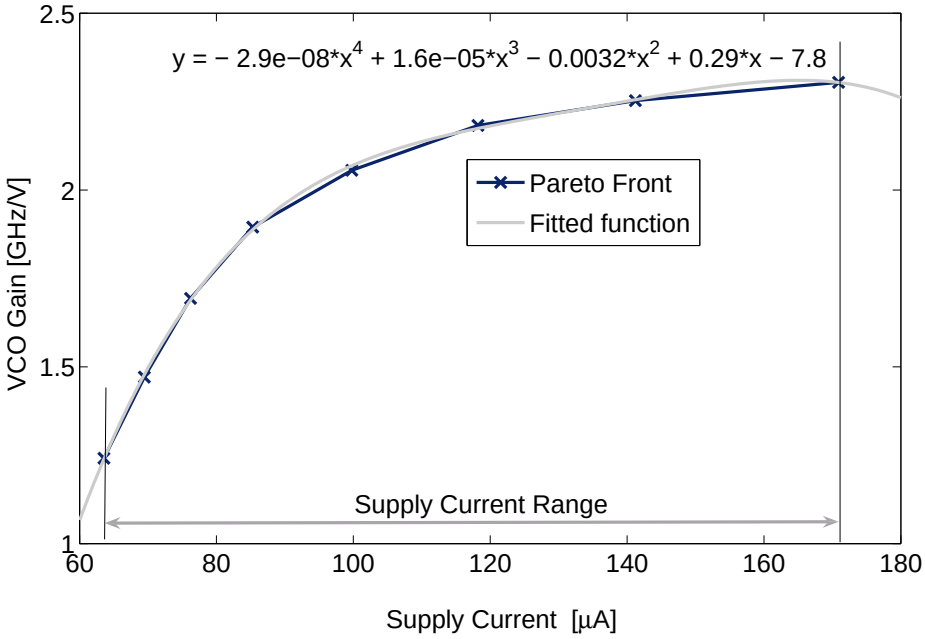


Figure 6.12: Polynomial fit of the Pareto front for the VCO

front as shown in figure 6.12. It describes the gain as a function of the supply current: $K_{vco} = p(I_{vco})$. Additionally, the possible ranges of the supply current that lead to different gain values can be read from the Pareto front. The range for the supply current is about 63 μA to 171 μA as can be seen from figure 6.12. With this function, the behavioral model can be set up that considers the extracted trade-off between VCO gain and supply current.

6.10.3 Trade-off analysis for a CPPLL

Motivation: The Pareto front illustrates the possible optimal compromises between the performances. It is of interest to see, how much of one performance must be sacrificed to improve another. This can be investigated by looking at the slope the Pareto front. This is especially interesting if the Pareto front has a *'knee'*, at which the trade-off rate of improving one performance at the cost of another changes quickly. This is shown in the following for the CPPLL circuit.

Trade-off analysis: Figure 6.13 shows the Pareto front of the charge pump phase locked loop for the performances locking time and power consumption. The Pareto front has a clear *'knee'* at a compromise of about 2.3 μs for 0.8 mW. To obtain other compromises, one either has to sacrifice a large amount in terms of settling time to

reduce the power consumption or increase the power consumption by a large amount to reduce the settling time.

Computational time: The computation of the Pareto front of the CPPLL took 6 h and 52 min.

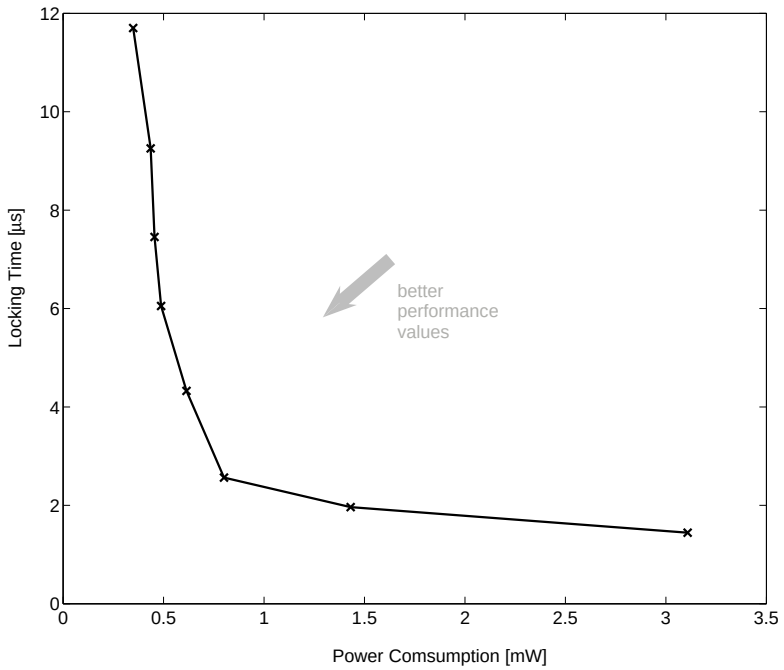


Figure 6.13: Pareto front for the CPPLL

Chapter 7

Conclusion

Despite the trend to implement functionality by means of digital hardware, there is also a growing need for analog circuitry, e.g., digital circuits need analog interfaces to communicate with the *'analog'* real world.

Analog circuit design requires new methodologies for an increased automation of the design flow. Many tasks are still done manually by analog designers. Recent years have shown a trend towards an automatic optimization of the circuit sizing step, known as automatic sizing or analog optimization. The sizing step is challenging due to the many inherent trade-offs between analog circuit performances. An optimal compromise between the performances must be found that fits the application.

This leads to the task of computing the utmost performance capabilities of the circuit, which supplies knowledge about all possible optimal performance compromises. The computation of the performance capabilities is denoted as performance space exploration. Knowledge of the performance capability allows to

- select a circuit structure from alternatives based on the different performance capabilities, which was shown for three alternative OpAmp circuits in section 6.10.1 ,
- compute realizable ranges and the trade-offs in the model parameters for parametrized behavioral models as shown for a VCO behavioral model in section 6.10.2,
- and realize an automatic hierarchical sizing process for large-scale circuits as described in section 1.2.3.

Performance space exploration is a challenging field due to the computational costs involved to evaluate the performances by circuit simulation. Additionally, the performance capabilities are degraded due to changing environmental conditions during the operating time of the circuit and process variations that occur during the production of the circuit. With decreasing device sizes in modern technologies, there is a relative increase in the influence of such tolerances on the electrical properties of

analog circuits. Therefore, these tolerances can not be ignored when computing of the performance capabilities of an analog circuit structure.

The approaches to solve the performance space exploration task presented in this thesis apply simulation-based deterministic Pareto optimization. This methodology is advantageous due to the great availability of simulation tools in industry and the efficiency of deterministic optimization algorithms. A complete methodology for the Pareto optimization of analog circuits has been described: In chapter 2, the fundamentals of the performance space exploration task and the concepts required to generate the Pareto front are discussed. It is shown that the performance space exploration task requires to solve a multi-objective optimization problem. A single solution to this multi-objective optimization problem is a Pareto-optimal performance vector. The set of all Pareto-optimal performance vectors is called the Pareto front. A discrete approximation of the Pareto front that consists of several Pareto-optimal performance vectors is computed by the presented approaches. Several advancements in the methodology of Pareto optimization and implementation with deterministic optimization algorithms are presented in this thesis:

Iterative Pareto front generation approach: In chapter 3, the methodology to transform the multi-objective optimization problem into a set of scalar optimization problems is shown. The solution of one scalar optimization problems leads to a single Pareto-optimal performance vector. The great suitability of the Minmax and equivalent Goal-attainment multi-objective optimization methods are discussed. A comprehensive geometric illustration of the selection of different compromises is derived by looking at the target trajectory. An iterative approach is shown that allows to capture the total extent of the Pareto front for more than two performances for the first time. It is based on so-called trade-off limits that are equal to the Pareto front of a subset of the performances. These trade-off limits define the boundary of a high dimensional Pareto front. The boundary shows the total extent of the Pareto front.

The remainder of chapter 3 describes an approach to select the target trajectories. A set of so-called compromise weight vectors that show the relative importance of each competing performance is mapped on a set of base points. Each base point defines one of several parallel target trajectories. The resulting Pareto-optimal performance vectors are nearly evenly spread over the trade-off range.

The presented contributions to the formulation of suitable scalar optimization problems lead to an improved representation of the Pareto front by the set of computed Pareto-optimal performance vectors. It establishes a fundamental methodology to assure, that the complete range of trade-offs are covered nearly evenly for the computation of the Pareto front for more than two performances with deterministic Pareto optimization methods. The improved coverage of the trade-off ranges compared to the state-of-the-art is shown for a numerical example in section 3.8, an operational transconductance amplifier (OTA) in section 6.3 as well as a voltage-controlled oscillator (VCO) in section 6.4.

Wavefront Feasible Sequential Quadratic programming algorithm: The Wavefront Feasible Sequential Quadratic programming algorithm (FSQP) is described in chapter 4. It is called a feasible optimization algorithm because it allows no violation in any inequality constraint. This is advantageous since the optimization constraints in analog sizing include so-called sizing rules that are based on basic circuit design knowledge. If sizing rules are violated, the circuit behavior often becomes very nonlinear, e.g., due to a transistor leaving the saturation operating region. Such sizings constitute no valid designs of the circuit and, therefore, must be disregarded when computing the performance capabilities. Feasible optimization algorithms require additional features to allow an efficient optimization. Two such features were presented, the tilting of the search direction and second-order correction. Additionally, the algorithm makes heavy use of parallel simulations to speed up the computation of the Pareto front.

The Wavefront approach was also introduced in chapter 4. It features an exchange of solutions between the scalar optimization problems in order to improve global convergence. If one scalar optimization is stuck at a local minimum during the iterative optimization process, it is supplied with alternative candidates from the other scalar optimization problems.

Finally, it is shown in chapter 4 that it is advantageous to use both the Minmax and Goal-attainment formulation during the optimization compared to using solely one of both formulations. The Goal-attainment method is suited to set up and solve the quadratic optimization problem and the Minmax method is suited for the line search and Wavefront approach.

The described algorithm is developed with the special requirements for the application in the Pareto optimization of analog circuits in mind. It constitutes a novel efficient deterministic optimization algorithm for this task. The clear significance of the different Wavefront FSQP features to obtain a representative set of Pareto optimal performance vectors is shown for the optimization of three OpAmp circuits in section 6.5 and a VCO circuit in section 6.6. The efficiency of the Wavefront FSQP algorithm compared to other optimization algorithms is shown for the three OpAmp circuits in section 6.7.

Specification Pareto front: The computation of the so-called specification Pareto front is discussed in chapter 5. The specification Pareto front of an analog circuit shows the most ambitious specifications on the performances that can be set to achieve a given minimum yield. The specification Pareto front considers the degradation of the performance values due to changing operating conditions and process variations.

Known worst-case methods are used to compute the specification values for a circuit sizing, known as specification analysis. It allows to formulate the optimization problem to find the specification Pareto front. The straight-forward computation of the

specification Pareto front in reasonable time is not possible. Alternative approaches are presented in this thesis. The applicability of the approaches to different circuit performances is discussed. It is shown that it is relevant whether the optimum for the nominal performance and specification value is reached for the same sizing of the circuit. This is shown for two circuit examples, a VCO circuit in section 6.8 and an OpAmp circuit in section 6.9.

The specification Pareto front allows to consider operating conditions and process variations during the deterministic Pareto optimization of analog circuits. Since the influence of such tolerances increases with modern technologies, these specification Pareto fronts might become more relevant to describe the performance capabilities of analog circuit structures in comparison to the nominal Pareto fronts.

In conclusion, the presented methods contribute to the development of Pareto optimization methods for analog integrated circuits from a research field in analog EDA towards an application in industrial design flows and EDA tools in order to support designers in future analog circuit design tasks.

Appendix A

Pareto Optimization with Weakly Pareto-Optimal Performance Vectors

The presence of weakly Pareto-optimal performance vectors that are not Pareto-optimal has two consequences:

1. The performance vector generated with the Minmax or Goal-Attainment method is not guaranteed to be Pareto-optimal. It can be weakly Pareto-optimal only. Derivation (3.2) does not hold.
2. A performance vectors that is Pareto-optimal in regard to a subset of the performances does not need to be Pareto-optimal in regard to the complete set of performances. It can be weakly Pareto-optimal only. Derivation (3.16) does not hold.

The presence of weakly Pareto-optimal performance vectors that are not Pareto-optimal cannot be tested easily. Be $\mathbf{f}^*$ the optimum of a Minmax optimization. The performance vector $\mathbf{f}^*$ is weakly Pareto-optimal but not Pareto-optimal if an improvement in one performance f_i is possible without degrading the remaining. The following second optimization problem must be solved in order to find a guaranteed Pareto-optimal performance vector:

$$\min_{\mathbf{d}} f_i \quad \text{s.t.} \quad \mathbf{f}^{S_i} = \mathbf{f}^{*S_i}; \quad S_i = S_f \setminus f_i \tag{A.1}$$

This second optimization must be conducted for all performances $f_i \in S_f$ to assure that the final result is Pareto-optimal. This is illustrated in figure A.1 for the case of two performances. The intersection point of the target trajectory and Pareto front is weakly Pareto-optimal but not Pareto-optimal. The optimum of the Minmax method could be located anywhere along the boundary that runs parallel to the f_1 axis. For example, the optimum could be the weakly Pareto-optimal solution $\mathbf{f}^{*W}$. By optimizing f_1 without allowing a degradation of f_2, the Pareto-optimal performance vector $\mathbf{f}^{*P}$ is found. Another optimization in f_2 leads to no improvement.

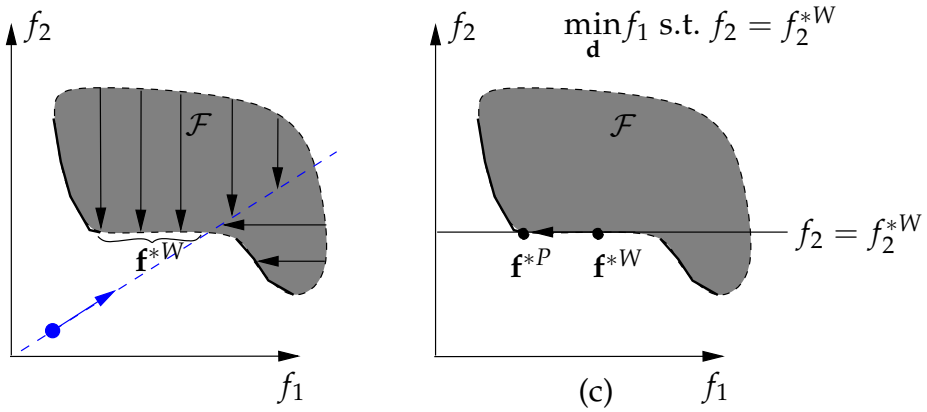


Figure A.1: Pareto optimization with weakly Pareto-optimal performance vectors that are not Pareto-optimal. *Left*, The Minmax method calculates the weakly Pareto-optimal performance vector $\mathbf{f}^{*W}$ somewhere in on the embraced part of the boundary. *Right*, a subsequent minimization of f_1 finds the Pareto-optimal performance vector $\mathbf{f}^{*P}$.

To avoid weakly Pareto-optimal performance vectors, n_f subsequent optimizations are required for each Pareto-optimal performance vector. This increases the complexity of the optimization setup but the finally obtained performance vectors are guaranteed to be truly Pareto-optimal.

Bibliography

[ABD03] G. Alpaydin, S. Balkir, G. Dundar: *An evolutionary approach to automatic synthesis of high-performance analog integrated circuits*, IEEE Transactions on Evolutionary Computation, volume 7(3), pages 240–252, June 2003.

[AEG+00] K. Antreich, J. Eckmueller, H. Graeb, M. Pronath, F. Schenkel, R. Schwencker, S. Zizala: *WiCkeD: Analog circuit synthesis incorporating mismatch*, in: *IEEE Custom Integrated Circuits Conference (CICC)*, pages 511–514, May 2000.

[APT02] P. Ashenden, G. D. Peterson, D. A. Teegarden: *The System Designer's Guide to VHDL-AMS*, Morgan Kaufmann, September 2002.

[AV05] A. Agarwal, R. Vemuri: *Hierarchical performance macromodels of feasible regions for synthesis of analog and rf circuits*, ICCAD 2005, 2005.

[BHT+03] D. M. Binkley, C. E. Hopper, S. D. Tucker, B. C. Moss, J. M. Rochelle, D. P. Foty: *A cad methodology for optimizing transistor current and sizing in analog cmos design*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2003.

[BNSV05] F. D. Bernardinis, P. Nuzzo, A. Sangiovanni-Vincentelli: *Mixed signal design space exploration through analog platforms*, in: *ACM/IEEE Design Automation Conference (DAC)*, 2005.

[BNV06] F. D. Bernardinis, P. Nuzzo, A. S. Vincentelli: *Robust systemlevel design with analog platforms*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 334–341, San Jose, November 2006.

[CA05] M. Chu, D. J. Allstot: *Elitist nondominated sorting genetic algorithm based rf ic optimizer*, IEEE Transactions on Circuits and Systems CAS, volume 52(3), pages 535–545, March 2005.

[Cad] Cadence: *Cadence framework*, www.cadence.com.

[CCC+97] H. Chang, E. Charbon, U. Choudhury, A. Demir, E. Felt, E. Liu, E. Malavasi, A. Sangiovanni-Vincentelli, I. Vassiliou: *A Top-Down, Constraint-Driven Design Methodology for Analog Integrated Circuits*, Kluwer Academic Publishers, 1997.

[CGRS96] L. R. Carley, G. G. E. Gielen, R. A. Rutenbar, W. M. C. Sansen: *Synthesis tools for mixed-signal ICs: Progress on frontend and backend strategies*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 298–303, 1996.

[DBdMHL01] J. L. Dawson, S. P. Boyd, M. del Mar Hershenson, T. H. Lee: *Optimal allocation of local feedback in multistage amplifiers via geometric programming*, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, volume 48(1), pages 1–11, January 2001.

[DD97] I. Das, J. Dennis: *A closer look at drawbacks of minimizing weighted sums of objectives for pareto set generation in multicriteria optimization problems*, Structural Optimization, volume 14(1), pages 63–69, 1997.

[DD98] I. Das, J. E. Dennis: *Normal-boundary intersection: A new method for generating the Pareto surface in nonlinear multicriteria optimization problems*, SIAM Journal on Optimization, volume 8(3), pages 631–657, August 1998.

[De87] M. Degrauwe, et al: *IDAC: An interactive design tool for analog CMOS circuits*, IEEE Journal of Solid-State Circuits SC, volume 22, pages 1106–1116, 1987.

[DG03] B. De Smedt, G. 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, volume 22(2), pages 213–223, February 2003.

[DGS03] W. Daems, G. Gielen, W. Sansen: *Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 22(5), pages 517–, May 2003.

[DGV+04] F. De Bernardinis, S. Gambini, F. Vincis, F. Svelto, R. Castello, A. Sangiovanni Vincentelli: *Design space exploration for a UMTS front-end expointing analog platforms*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 923–930, 2004.

[DJSV03] F. De Bernardinis, M. I. Jordan, A. Sangiovanni-Vincentelli: *Support vector machines for analog circuit performance representation*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 964–969, 2003.

[DLG+98] G. Debyser, F. Leyn, G. Gielen, W. Sansen, M. Styblinski: *Efficient statistical analog IC design using symbolic methods*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, page VI/21, 1998.

[dMH02] M. del Mar Hershenson: *Design of pipeline analog-to-digital converters via geometric programming*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 317–324, November 2002.

[dMH03] M. del Mar Hershenson: *Efficient description of the design space of analog circuits*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 970–973, 2003.

[dMHBL98] M. del Mar Hershenson, S. P. Boyd, T. H. Lee: *GPCAD: A tool for CMOS op-amp synthesis*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 1998.

[dMHBL01] M. del Mar Hershenson, S. P. Boyd, T. H. Lee: *Optimal design of a CMOS Op-Amp via geometric programming*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 20(1), pages 1–21, January 2001.

[DNAV99] N. Dhanwada, A. Nunez-Aldana, R. Vemuri: *Hierarchical constraint transformation using directed interval search for analog system synthesis*, in: *Design, Automation and Test in Europe (DATE)*, 1999.

[DSV04] F. De Bernardinis, A. Sangiovanni-Vincentelli: *A methodology for system-level analog design space exploration*, in: *Design, Automation and Test in Europe (DATE)*, pages 676–677, February 2004.

[DT05] D. L. Donoho, J. Tanner: *Sparse nonnegative solution of underdetermined linear equations by linear programming*, in: *Proceedings of the National Academy of Sciences*, volume 102, pages 9446–9451, 2005.

[DV03] A. Doboli, R. Vemuri: *Behavioral modeling for high-level synthesis of analog and mixed-signal systems from vhdl-ams*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, 2003.

[DV05] M. Ding, R. Vemuri: *A combined feasibility and performance macromodel for analog circuits*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 63 – 68, Annaheim, June 2005.

[EMG05] T. Eeckelaert, T. McConaghy, G. Gielen: *Efficient multiobjective synthesis of analog circuits using hierarchical pareto-optimal performance hypersurfaces*, in: *Design, Automation and Test in Europe (DATE)*, 2005.

[ESG+06] T. Eeckelaert, R. Schoofs, G. Gielen, M. Steyaert, W. Sansen: *Hierarchical bottom-up analog optimization methodology validated by a delta–sigma a/d converter design for the 802.11a/b/g standard*, in: *ACM/IEEE Design Automation Conference (DAC)*, San Francisco, July 2006.

[ESG+07] T. Eeckelaert, R. Schoofs, G. Gielen, M. Steyaert, W. Sansen: *An efficient methodology for hierarchical synthesis of mixed-signal systems with fully integrated building block topology selection*, in: *Design, Automation and Test in Europe (DATE)*, Nizza, April 2007.

[Esh92] K. S. Eshbaugh: *Generation of correlated parameters for statistical circuit simulation*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 11(10), pages 1198–1206, October 1992.

[ETP89] F. El-Turky, E. Perry: *BLADES: An artificial intelligence approach to analog circuit design*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 8, pages 680–692, 1989.

[Fle87] R. Fletcher: *Practical Methods of Optimization*, John Wiley & Sons, 1987.

[FOK96] A. Fuad Mas'ud, T. Ohtsuka, H. Kunieda: *Translation of specifications in hierarchical analog LSI design*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 735–738, 1996.

[FWZ+92] U. Feldmann, U. Wever, Q. Zheng, R. Schultz, H. Wriedt: *Algorithms for modern circuit simulation*, Archiv für Elektronik und Übertragungstechnik (AEÜ), volume 46, pages 274–285, 1992.

[GESSL95] G. J. Gomez, S. H. K. Embabi, E. Sanchez-Sinencio, M. C. Lefebvre: *A nonlinear macromodel for CMOS OTAs*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, volume 2, pages 920–923, 1995.

[GME05] G. Gielen, T. McConaghy, T. Eeckelaert: *Performance space modeling for hierarchical synthesis of analog integrated circuits*, in: *ACM/IEEE Design Automation Conference (DAC)*, 2005.

[GMGS08] H. Graeb, D. Mueller-Gritschneder, U. Schlichtmann: *Pareto optimization of analog circuits considering variability*, International Journal of Circuit Theory and Applications, 2008.

[GMS07] H. Graeb, D. Mueller, U. Schlichtmann: *Pareto optimization of analog circuits considering variability*, in: *European Conference on Circuit Theory and Design (ECCTD)*, August 2007.

[Gra07] H. Graeb: *Analog Design Centering and Sizing*, Springer, 2007.

[GWS89] G. Gielen, H. C. Walscharts, W. C. Sansen: *ISAAC: A symbolic simulation for analog integrated circuits*, IEEE Journal of Solid-State Circuits SC, volume 24, pages 1587–1597, December 1989.

[GWS90] G. Gielen, H. C. Walscharts, W. C. Sansen: *Analog circuit design optimization based on symbolic simulation and simulated annealing*, IEEE Journal of Solid-State Circuits SC, volume 25, pages 707–713, June 1990.

[GZEA01] H. Graeb, S. Zizala, J. Eckmueller, K. Antreich: *The sizing rules method for analog integrated circuit design*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 343–349, 2001.

[GZMS07] H. Graeb, J. Zou, D. Mueller, U. Schlichtmann: *Hierarchische optimierung einer phasenregelschaltung*, in: U. Bremen (ed.), *ASIM/GI-Fachgruppentagung Simulation technischer Systeme/Grundlagen und Methoden in Modellbildung und Simulation*, February 2007.

[HEL92] J. Harvey, M. Elmasry, B. Leung: *STAIC: An interactive framework for synthesizing CMOS and BiCMOS analog circuits*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 11, pages 1402–1417, 1992.

[HRC89] R. Harjani, R. Rutenbar, L. Carley: *OASYS: A framework for analog circuit synthesis*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 8, pages 1247–1266, 1989.

[HS96] R. Harjani, J. Shao: *Feasibility and performance region modeling of analog and digital circuits*, Analog Integrated Circuits and Signal Processing, volume 10(1), pages 23–43, June 1996.

[ITR06] ITRS: *The international technology roadmap for semiconductors*, http://www.itrs.net/, 2006.

[JP07] S. Jokar, M. Pfetsch: *Exact and approximate sparse solutions of underdetermined linear equations*, ZIB-Report 07-05, Konrad-Zuse-Zentrum für Informationstechnik Berlin, March 2007.

[KCJ+00] K. Kundert, H. Chang, D. Jefferies, G. Lamant, E. Malavasi, F. Sendig: *Design of mixed-signal systems-on-a-chip*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 19(12), pages 1561–1571, December 2000.

[KL95] W. Kruiskamp, D. Leenaerts: *DARWIN: CMOS opamp synthesis by means of a genetic algorithm*, in: *ACM/IEEE Design Automation Conference (DAC)*, 1995.

[KPRC99] M. Krasnicki, R. Phelps, R. A. Rutenbar, L. R. Carley: *MAELSTROM: Efficient simulation-based synthesis for custom analog cells*, in: *ACM/IEEE Design Automation Conference (DAC)*, 1999.

[KSG90] H. Koh, C. Sequin, P. Gray: *OPASYN: A compiler for CMOS operational amplifiers*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 9, pages 113–125, 1990.

[Kun95] K. S. Kundert: *The Designer's Guide to SPICE & SPECTRE*, Kluwer Academic Publishers, 1995.

[LD81] M. R. Lightner, S. W. Director: *Multiple criterion optimization for the design of electronic circuits*, IEEE Transactions on Circuits and Systems CAS, volume 28(3), pages 169–179, March 1981.

[LGXP04] X. Li, P. Gopalakrishnan, Y. Xu, L. T. Pileggi: *Robust analog rf circuit design with projection-based posynomial modeling*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 855 – 862, San Jose, November 2004.

[LGXP07] X. Li, P. Gopalakrishnan, Y. Xu, L. T. Pileggi: *Robust analog/rf circuit design with projection-based performance modeling*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 26(1), January 2007.

[Lin03] J. G. Lin: *On min-norm and min-max methods of multi-objective optimization*, Mathematical Programming, volume 103, pages 1–33, September 2003.

[LT00] C. Lawrence, A. Tits: *A computationally effcient feasible quadratic programming algorithm*, SIAM Journal on optimization, volume 11(4), 2000.

[LWP+05] X. Li, J. Wang, L. T. Pileggi, T.-S. Chen, W. Chiang: *Performance-centering optimization for system-level analog design exploration*, ICCAD 2005, 2005.

[MA04] R. Marler, J. Arora: *Survey of multi-objective optimization methods for engineering*, Structural and Multidisciplinary Optimization, volume 26, pages 369–395, 2004.

[Mar01] B. Martin: *Automation comes to analog*, ieeespectrum, pages 70–75, June 2001.

[MAW07] *WiCkeD - Design for Manufacturability and Yield, MunEDA GmbH Munich, www.muneda.com*, 2007.

[MC91] P. C. Maulik, L. R. Carley: *Automating analog circuit design using constrained optimization techniques*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 390–393, November 1991.

[MFDCRV94] F. Medeiro, F. V. Fernandez, Dominguez-Castro, A. Rodriguez-Vasquez: *A statistical optimization-based approach for automated sizing of analog cells*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 1994.

[MGS06] D. Mueller, H. Graeb, U. Schlichtmann: *Optimierung analoger schaltungsbloecke mittels pareto-wellenfront-optimierung*, in: *9. GMM/ITG-Fachtagung Analog, Dresden*, September 2006.

[MGS07a] D. Mueller, H. Graeb, U. Schlichtmann: *Generation of robust pareto fronts for analog circuits*, TU Bericht TUM-LEA-07-1, TU Muenchen, June 2007.

[MGS07b] D. Mueller, H. Graeb, U. Schlichtmann: *Trade-off design of analog circuits using goal attainment and "wave front" sequential quadratic programming*, in: *Design, Automation and Test in Europe (DATE)*, April 2007.

[MGS08] T. Massier, H. Graeb, U. Schlichtmann: *Sizing rules for bipolar analog circuit design*, in: *Design, Automation and Test in Europe (DATE)*, March 2008.

[Mie04] K. M. Miettinen: *Nonlinear Multiobjective Optimization*, Internatinal series in operations research & management science, Kluwer Academic Publishers, 101 Philip Srive, Assinippi Park, Norwell, Massachusetts, 02061, USA, 4th edition, 2004.

[MPGS07] T. McConaghy, P. Palmers, G. Gielen, M. Steyaert: *Simultaneous multi-topology multi-objective sizing across thousands of analog circuit topologies*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 944–947, San Diego, June 2007.

[MSG03] T. Massier, G. Stehr, H. Gräb: *Ein Beitrag zur Automatisierung der Strukturanalyse und der impliziten Spezifikation von analogen integrierten Schaltungen*, 7. GMM/ITG Diskussionssitzung Entwurf von Analogschaltungen (ANALOG '03), page 6, June 2003.

[MSGS05] D. Mueller, G. Stehr, H. Graeb, U. Schlichtmann: *Deterministic approaches to analog performance space exploration (PSE)*, in: *ACM/IEEE Design Automation Conference (DAC)*, June 2005.

[MSGS06] D. Mueller, G. Stehr, H. Graeb, U. Schlichtmann: *Fast evaluation of analog circuit structures by polytopal approximations*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, 2006.

[MSGU05] D. Mueller, G. Stehr, H. Graeb, U.Schlichtmann: *Eigenschaftsraumexploration bei der hierarchischen dimensionierung analoger integrierter schaltungen*, Informatik 2005, Beiträge der 35 Jahrestagung der Gesellschaft für Informatik, volume 1, pages 334–338, September 2005.

[MV01] P. Mandal, V. Visvanathan: *CMOS Op-Amp sizing using a geometric programming formulation*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 20(1), pages 22–38, January 2001.

[Nag75] L. Nagel: *SPICE2: A computer program to simulate semiconductor circuits*, Ph. D. dissertation, Univ. of California, Berkeley, 1975.

[NBHB04] L. Naethke, V. Burkhay, L. Hedrich, E. Barke: *Hierarchical automatic behavioral model generation of nonlinear analog circuits based on nonlinear symbolic techniques*, in: *Design, Automation and Test in Europe (DATE)*, volume 1, Paris, February 2004.

[NRSVT88] W. Nye, D. Riley, A. Sangiovanni-Vincentelli, A. Tits: *DELIGHT.SPICE: An optimization-based system for the design of integrated circuits*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 7, pages 501–519, 1988.

[NW99] J. Nocedal, S. J. Wright: *Numerical Optimization*, Springer, 1999.

[ORC96] E. S. Ochotta, R. A. Rutenbar, L. R. Carley: *Synthesis of high-performance analog circuits in ASTRX/OBLX*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 15(3), pages 273–294, March 1996.

[PDML94] J. Power, B. Donellan, A. Mathewson, W. Lane: *Relating statistical mosfet model parameter variabilities to ic manufacturing process fluctuations enabling realist worst-case design*, IEEE Transactions on Semiconductor Manufacturing (SM), volume 7(3), pages 306–318, August 1994.

[PKR+00] R. Phelps, M. Krasnicki, R. A. Rutenbar, L. R. Carley, J. R. Hellums: *Anaconda: Simulation-based synthesis of analog circuits via stochastic pattern search*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 19(6), pages 703–717, June 2000.

[RGR07] R. A. Rutenbar, G. G. E. Gielen, J. Roychowdhury: *Hierarchical modeling, optimization, and synthesis for system-level analog and rf designs*, in: *Proceedings of the IEEE*, volume 95, IEEE, March 2007.

[Sch02] R. Schwencker: *Zur Dimensionierung analoger integrierter Schaltungen unter Berücksichtigung struktureller Nebenbedingungen*, Ph.D. thesis, 2002.

[Sch04] F. C. Schenkel: *Tolerance Analysis and Design Centering of Analog Circuits, with Consideration of Mismatch*, Ph.D. thesis, 2004.

[SCP05] A. Somani, P. P. Chakrabarti, A. Patra: *Mixing global and local competition in genetic optimization-based design space exploration of analog circuits*, in: *Design, Automation and Test in Europe (DATE)*, 2005.

[SG03] B. D. Smedt, G. Gielen: *HOLMES: capturing the yield-optimized design space boundaries of analog and RF integrated circuits*, in: *Design, Automation and Test in Europe (DATE)*, pages 256–261, March 2003.

[SGA03] G. Stehr, H. Graeb, K. Antreich: *Performance trade-off analysis of analog circuits by normal-boundary intersection*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 958–963, 2003.

[SGA04] G. Stehr, H. Graeb, K. Antreich: *Analog performance space exploration by Fourier-Motzkin elimination with application to hierarchical sizing*, in:

IEEE/ACM International Conference on Computer-Aided Design (ICCAD), November 2004.

[SGA07] G. Stehr, H. Graeb, K. Antreich: *Analog performance space exploration by normal-boundary intersection and by fourier-motzkin elimination*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 26(10), pages 1733–1748, October 2007.

[Spe98] P. Spellucci: *Numerische Verfahren der nichtlinearen Optimierung*, Birkhäuser, 1998.

[SSP87] B. Sheu, J. H. Shieh, M. Patil: *BSIM: Berkeley short-channel IGFET model for MOS transistors*, IEEE Journal of Solid-State Circuits SC, volume 22, pages 558–566, August 1987.

[SSPG02] R. Schwencker, F. Schenkel, M. Pronath, H. Graeb: *Analog circuit sizing using adaptive worst-case parameter sets*, in: *Design, Automation and Test in Europe (DATE)*, 2002.

[Ste05] G. Stehr: *On the Performance Space Exploration of Analog Integrated Circuits*, Ph.D. thesis, 2005.

[TTR06] S. K. Tiwary, P. K. Tiwary, R. A. Rutenbar: *Generation of yield-aware pareto surfaces for hierarchical circuit design space exploration*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 31– 36, July 2006.

[TVR04] S. K. Tiwary, S. Velu, R. A. Rutenbar: *Pareto optimal modeling for efficient PLL optimization*, in: *NSTI Nanotech*, volume 2, pages 195–198, 2004.

[VCD+96] I. Vassiliou, H. Chang, A. Demir, E. Charbon, P. Miliozzi, A. Sangiovanni-Vincentelli: *A video driver system designed using a top-down, constraint-driven methodology*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 463 – 468, 1996.

[VDL+01] G. Van der Plas, G. Debyser, F. Leyn, K. Lampaert, J. Vandenbussche, G. Gielen, W. Sansen, P. Veselinovic, D. Leenaerts: *AMGIE–A synthesis environment for CMOS analog integrated circuits*, IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, volume 20(9), pages 1037–1058, September 2001.

[VLR] *Verilog-AMS Language Reference Manual: Analog & Mixed-Signal Extensions to Verilog-HDL - version 2.1.*, Available at www.verilog-ams.com.

[VLv+95] P. Veselinovic, D. Leenaerts, W. van Bokhoven, F. Leyn, G. Gielen, W. Sansen: *A flexible topology selection program as part of an analog synthesis system*, in: *European Design and Test Conference (ED&TC)*, pages 119–123, 1995.

[WD06] Y. Wei, A. Doboli: *Systematic development of nonlinear analog circuit macromodels through successive operator composition and nonlinear model decoupling*, pages 1023–1028, 2006.

[WH06] X. Wang, L. Hedrich: *An approach to topology synthesis of analog circuits using hierarchical blocks and symbolic analysis*, in: *Asia and South Pacific Design Automation Conference*, 2006.

[WT04] S. J. Wright, M. J. Tenny: *A feasible trust-region sequential quadratic programming algorithm*, volume 14(4), pages 1074–1105, 2004.

[XHL+05] Y. Xu, K.-L. Hsiung, X. Li, I. Nausieda, S. Boyd, L. Pileggi: *Opera: optimization with ellipsoidal uncertainty for robust analog ic design*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 632 – 637, Annaheim, June 2005.

[YL07] G. Yu, P. Li: *Yield-aware analog integrated circuit optimization using geostatistics motivated performance modeling*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 464–469, November 2007.

[Ziz01] S. Zizala: *Numerische Verhaltensmodellierung analoger CMOS-Schaltungen unter Berücksichtigung von Dimensionierungsregeln*, Ph.D. thesis, 2001.

[ZMG+05] J. Zou, D. Mueller, H. Graeb, U. Schlichtmann, E. Hennig, R. Sommer: *Fast automatic sizing of a charge pump phase-locked loop based on behavioral models*, Proceeding of IEEE International Behavioral Modeling and Simulation workshop 2005, page 6, September 2005.

[ZMGS06] J. Zou, D. Mueller, H. Graeb, U. Schlichtmann: *A CPPLL hierarchical optimization methodology considering jitter, power and locking time*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 19–24, 2006.

[ZMGS07a] J. Zou, D. Mueller, H. Graeb, U. Schlichtmann: *Optimization of SC ΣΔ modulators based on worst-case-aware pareto-optimal fronts*, in: *IEEE Custom Integrated Circuits Conference (CICC)*, September 2007.

[ZMGS07b] J. Zou, D. Mueller, H. Graeb, U. Schlichtmann: *Pareto-front computation and automatic sizing of CPPLLs*, in: *IEEE International Symposium on Quality Electronic Design*, March 2007.

Nomenclature

General Conventions

$\mathbf{A}$	Matrix $\mathbf{A}$	15
a_{ij}	Component of matrix $\mathbf{A}$ at i^{th} row and j^{th} column	15
$\mathbf{b}$	Vector $\mathbf{b}$	15
b_k	k^{th} component of vector $\mathbf{b}$	15
x	Scalar x	15

Basic quantities, vectors and sets

$\mathbf{p}$	Vector of circuit parameters, n_p-dimensional	16
$\mathbf{d}$	Vector of design parameters, n_d-dimensional	16
$\boldsymbol{\theta}$	Vector of operating parameters, n_θ-dimensional	16
$\mathbf{s}$	Vector of statistical parameters, n_s-dimensional	16
$\mathbf{f}(..)$	Vector of circuit performances, n_f-dimensional	16
$\mathbf{c}(..)$	Vector of sizing rules, n_c-dimensional	17
$\mathcal{D}$	Valid parameter space	17
$\mathbf{f}_{spec}$	Upper bound performance specification	18
$\mathcal{T}_{\boldsymbol{\theta}}$	Tolerance region of operating conditions	18
$\mathcal{A}_s$	Acceptance region of statistical parameters	18
Y	Parametric yield	18
$\mathcal{F}$	Feasible performance space	19
$\mathcal{M}(\mathbf{f})$	Set of all Pareto-better performance vectors	20
$\delta\mathcal{F}$	Pareto front	21
$\mathcal{N}(\mathbf{f})$	Set of performance vectors that are better in all performance values	22
$\mathbf{f}_Y$	Specification vector for a given yield	24
$\mathcal{F}_Y$	Space of realizable specifications	24
$\delta\mathcal{F}_Y$	Specification Pareto front	24
$\overline{\delta\mathcal{F}}$	Discretized Pareto front	24

Pareto Optimization

$\mathbf{b}$	Base point vector of target trajectory	32
$\mathbf{v}$	Direction vector of target trajectory	32
t	Bound variable of Goal-attainment method	32
S_f	Set of performances	35
$\mathcal{T}_i(\mathbf{f})$	Set of performance vectors in the i-th trade-off direction	36
$\delta\mathcal{F}_{B,i}$	trade-off limit for the i-th trade-off direction	36
$\delta\mathcal{F}_B$	Boundary of the Pareto front	38
$C^{(n_i)}(S_f)$	Family set of all combinations of n_i performances out of S_f	43
W	Set of compromise weight vectors	46
$\mathbf{w}$	Compromise weight vector	47
D	Density of Pareto-optimal performance vectors on the Pareto front	47
$\mathbf{W}_B$	Matrix of compromise weight vectors of boundary points	48
$\mathbf{F}_B$	Matrix of performance vectors of boundary points	48
$\boldsymbol{\rho}$	Vector with weights of linear combination of boundary points	48
d	L_1 norm of difference of compromise weight vectors	50

Wavefront FSQP algorithm

$\mathbf{x}$	Candidate	59
$o(\mathbf{x})$	Objective function	59
$\mathbf{a}_{ineq}(\mathbf{x})$	Inequality constraints	59
$\Delta\mathbf{x}$	Search direction	62
$\mathbf{g}$	Estimate of gradient of objective function	63
$\mathbf{A}_{ineq}$	Estimate of Jacobi matrix of inequality constraints	63
$\mathbf{H}$	Estimate of Hessian matrix of Lagrange function	63
α	Step size	64
$\Delta\mathbf{x}_t$	Tilted search direction	67
γ	Additional parameter for tilted quadratic program	67
$\boldsymbol{\eta}$	Tilt vector	67
z_j	Tilt factor	68

Specification Pareto front

$\boldsymbol{\theta}_{Y,i}$	Specification operating parameter vector	80
$\mathbf{s}_{Y,i}$	Specification statistical parameter vector	80
$\mathcal{T}_{\mathbf{s}}$	Tolerance region of statistical parameters	80
β_Y	Worst-case distance	80
Y_0	Yield for specification for single performance	80
Y_g	Total yield	81
Y_{min}	Required minimum yield	81
Δf_i	Difference between specification and nominal performance value	84

List of Figures

List of Tables