Lehrstuhl für Entwurfsautomatisierung
der Technischen Universität München

On the Performance Space Exploration of Analog Integrated Circuits

Guido Stehr

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

Doktor-Ingenieurs

genehmigten Dissertation.

Die Dissertation wurde am 13. 5. 2005 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 10. 9. 2005 angenommen.

A paperback version of this thesis was published by Verlag Dr. Hut, Munich, in 2005.
ISBN 3-89963-256-7.

When the tree is born, it is not big right away.
When the tree is big, it does not blossom right away.
When it bears fruit, they are not ripe right away.
When they are ripe, you do not eat them right away.

Aegiduis of Assisi

Acknowledgements

From October 1999 to February 2005, I had the opportunity to work as a research and teaching assistant at the Institute for Electronic Design Automation, Technische Universität München. This thesis comprises of the results of my research activities in the analog CAD group.

First of all, I would like to thank Professor Kurt Antreich and Professor Ulf Schlichtmann for giving me the chance to work at this institute. I am particularly grateful for this opportunity because Professor Antreich agreed to advise me even beyond his time as head of the institute.

A grateful word of thanks also to the committee members Professor Lars Hedrich and Professor Georg Färber for their interest in my work.

I am very grateful to my "analog" colleagues. I am particularly indebted to Dr. Helmut Gräb who, as the "senior" of the analog CAD research group, never got tired of discussing my ideas and who patiently proofread my publications. Dr. Stephan Zizala brought my attention to the area of performance space exploration. Tobias Massier provided expertise in the automatic setup of sizing constraints (cf. Section 2.1.3). Michael Pronath was a brilliant colleague to discuss with, and he came up with essential algorithmic ideas for initial sizing (cf. Appendix A). Dr. Volker Glöckel, Dr. Frank Schenkel and Dr. Robert Schwencker encouraged me very much, which was especially important in the early stages of my project. I would like to thank Daniel Müller who completed a lot of the setup required for the hierarchical optimization demonstrator circuit (cf. Section 5.4.2). I am also grateful to all the other colleagues at the institute for providing a truly cooperative and creative environment.

I would like to express my gratitude to my contact persons in the industry, especially Dr. Josef Eckmüller and Roland Schwenk.

I had the pleasure to supervise a number of students who supported me a lot, everyone in their particular way: Thomas Kränzler and Bernhard Christ helped me to demonstrate the potential of automatic hierarchical simulation of large mixed-signal circuits (cf. [SGA03b]). Shantanu Gupta examined different versions of the Normal-Boundary Intersection algorithm (cf. Section 3.2.4). Matthias Oster implemented a first version of the Fourier-Motzkin algorithm (cf. Section 4.2). Hugo Caballero Figueroa contributed to the setup of the demonstrator circuit for hierarchical optimization (cf. Section 5.4.2). Alois Unterholzner evaluated the behavior of digital implementation alternatives to traditional mixed-signal PLLs.

Furthermore, I am obliged to the following mathematicians who provided essential hints: Dr. Thorsten Theobald, Technische Universität Berlin, pointed out the Fourier-Motzkin elimination method to me (cf. Section 4.2). Professor Michael Ulbrich, Universität Hamburg, told me that the Normal-Boundary Intersection method (Section 3.2) was superior to a similar method I had considered. Professor Günter Ziegler, Technische Universität Berlin, helped me to better understand the Chernikov redundancy criterion (Section 4.2.3.1).

I am happy that my friends Maria Sekertzi and Heidi Liesch proofread this thesis and patiently improved my English.

Last, but definitely not least, I want to express my deep gratitude to my parents Cäcilia and Gerhard, to my sisters Petra and Vera and in particular to my wife Maria. They eagerly supported me in times of disappointment or pressing deadlines and cordially shared my happiness upon success.

Munich, May 2005

Guido Stehr

Contents

Chapter 1

Introduction

Undoubtedly, today's integrated electronic systems owe their remarkable performance primarily to the rapid advancements of digital technology since the 1970s. The key advantage of digital circuit design is its abstraction from the physical details of the actual circuit implementation.

Digital functionality lends itself to a description based on Boolean algebra. This was the basis for the development of automatic synthesis tools which cover the path from a behavioral description at RTL level to the final layout. Furthermore, digital circuitry is comparatively insensitive with respect to variations in the manufacturing process and the operating conditions. Consequently, digital circuits frequently offer a more robust behavior than their analog counterparts, albeit often with area, power and speed drawbacks. Last but not least, digital designs allow functional complexity that would not be possible based on analog technology. Due to these and other benefits, analog functionality has increasingly been replaced by digital implementations.

In spite of the trends discussed above, analog components are far from obsolete [GR00, Gil01, ITRS03]. In fact, a closer look reveals that they are key components of modern electronic systems. There is a definite trend toward pervasive and ubiquitous use of electronic circuits in everyday life. Wearable electronics [JLSW03], wireless communications [GK03] and the widespread application of RF tags [Bri03] are just some examples of current developments. While all of these electronic systems are based on digital circuitry, they heavily rely on analog components as interfaces to the "real", i.e. analog, world. In fact, many modern designs combine powerful digital systems and complementary analog components on a single chip for cost and reliability reasons [KCJ$^+$00]. Unfortunately, the design of such systems-on-chip (SOCs) suffers from the vastly different design styles of analog and digital components. While mature synthesis tools are readily available for digital designs [Cad, Syn], there is hardly any support for analog designers apart from well-established SPICE-like circuit simulators. Just recently, some simulation-based automatic sizing tools have been introduced in the industrial environment [Mun, Neo]. Consequently, although the analog part usually only occupies a small fraction of the entire die area of an

SOC, its design often constitutes a major bottleneck within the entire development process [ORC96, ITRS03].

The rapid improvement of circuit functionality has only been possible due to a dramatic rise of the achievable integration densities. The corresponding permanent shrink of realizable circuit structures, however, is a mixed blessing: While it is desirable from the integration point of view, it promotes more and more nonlinear physical phenomena which have only had minor impact so far. Therefore, many simplifying assumptions no longer hold, which complicates the design of electronic circuits. In fact, not only the analog domain is affected, but digital design is also increasingly becoming aware of physical effects [KCJ+00, CDSS02, ITRS03, Kon04]. Unfortunately, the consumer marketplace does not acknowledge these challenges. On the contrary: Missing the market window can render a good product idea worthless. With analog design automation at a low level, increasingly nonlinear circuit behavior, and tightening time-to-market pressures, is becomes obvious that improved analog design tools are urgently needed [GR00, ITRS03].

1.1 Motivation

1.1.1 Systems on Chip

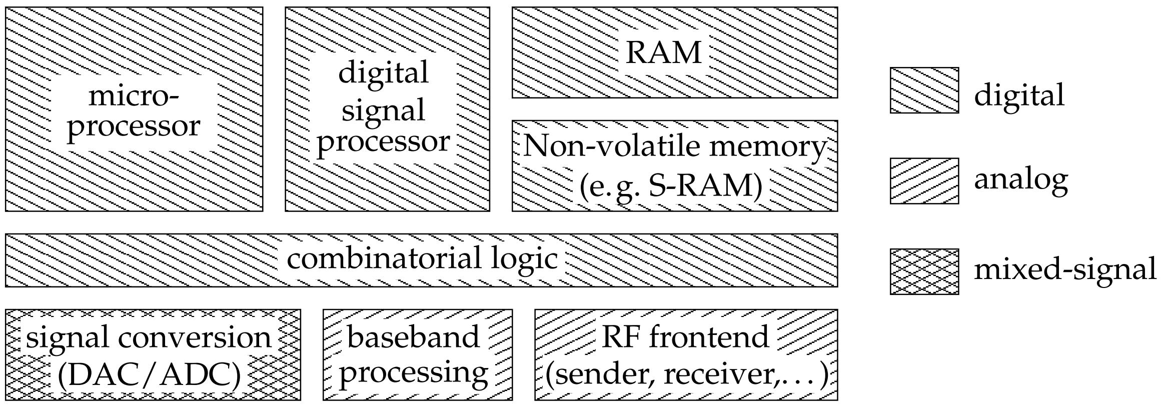


Figure 1.1: System on Chip

In early stages of the SOC design process, the entire system functionality is partitioned into a number of functional subsystems. The assignment of these subsystems to digital, analog and mixed-signal circuit realizations is done according to “hard” function-specific constraints such as the physical nature of the signal to be processed, or “soft” considerations including power, cost and area trade-offs. Figure 1.1 shows an SOC for a telecommunication device [KCJ+00]. It contains an analog RF frontend, analog baseband signal processing, mixed-signal circuitry for signal conversion, as well as digital units for signal processing and general device management. The work at hand focuses on the performance space exploration of analog circuitry as it is used in baseband processing, for instance.

The following section gives an overview of different analog design styles and puts the application areas of performance space exploration into perspective.

1.1.2 Design Process of Analog Circuits

Overview of Analog Design Steps In general, the analog design process can be partitioned into three major steps [HEL92,CGRS96,Sch02]:

1. Topology selection / generation:
   A circuit structure has to be found, which has enough potential to finally meet the given circuit specifications. To this end, an existing topology is chosen if possible, or a new circuit structure is created otherwise.
2. Component sizing:
   Actual values are assigned to the designable circuit parameters such as the transistor widths and lengths or resistance and capacitance values. The goal is to adjust the circuit performances in such a way that they meet or exceed the specified goals, even with tolerances in the fabrication process and the operating conditions. Examples of circuit performance specifications are the minimum transit frequency or the maximum power consumption of an operational amplifier.
3. Layout generation:
   Geometric structures are created which reflect the parameter values obtained in the component sizing step, and which are suited for the actual implementation in silicon.

In [CGRS96], the first two steps are referred to as the *frontend* of the analog design process, while the third step is called the *backend*. At this point it should also be noted that sporadic attempts have been made to intertwine these steps, e. g. [MCR95].

There are a number of influences which affect the circuit performance, but which are beyond the control of the designer:

- The actual performances of a circuit do not only depend on the designable circuit parameters, but also on a number of parameters, which may vary during the operation of the circuit. These parameters collectively describe the *operating conditions*. Examples are the ambient temperature and the supply voltage, among others. For each of them, an admissible value range is specified. The designer then has to guarantee the specified circuit performances within the entire region of tolerable operating conditions.
- The manufacturing process is subject to inevitable random process variations, which can be described by probability density functions. This affects *technological* parameters such as oxide thickness or doping. Besides, the actual values of the *designable* parameters deviate from the intended ones.

Especially analog circuits are sensitive with respect to these variations. Nevertheless, their behavior must meet the specifications under all circumstances. Therefore, the component sizing step is often separated into two distinct design tasks [Sch02]:

2 a. Nominal sizing:
The performances are adjusted under the assumption that both variations in the operating conditions and in the manufacturing process can be neglected. To do this, typical fixed values are assigned to the technological parameters. Furthermore, it is common to assume worst-case values for the operating conditions [Sch02]. Finally, there is no random deviation of the designable parameters.

2 b. Design centering:
The inevitable variations in both the operating conditions and the manufacturing process affect the circuit performances. Given a stochastic characterization of the manufacturing process and a certain range of admissible operating conditions, the designer has to make sure that the circuit meets the specifications under all conditions. A circuit which does not meet this requirement is considered dysfunctional and cannot be sold to the customer. This obviously raises the effective production costs. Therefore, a prominent goal of the design process is a good production yield, i.e. a high percentage of fully functional circuits. Accordingly, the circuits have to be designed in such a way that they are robust. This means that they have to be insensitive to variations in the operating and manufacturing conditions and that they should also have enough safety margins as regards the specifications. The design steps which are required to achieve these goals are usually very simulation-intensive. Thus, the result of the nominal sizing step usually serves as a starting point for a subsequent yield optimization [AEG$^+$00, GR00, Sch02].

Due to the high simulation costs, design centering is primarily used in the last phases of a design. In earlier stages, especially in the conceptual and exploratory phases, nominal sizing is performed. Hence, efficient nominal sizing techniques are urgently required. They are needed to support designer creativity, and to help in exploring different realization alternatives. Furthermore, a good nominal sizing facilitates successful design centering. Considering the great rewards in this area, this work focuses on nominal sizing techniques.

Performance Evaluation During the sizing process, the performance values have to be determined for given parameter values. This is predominantly done using *numerical simulators*, such as Eldo [Ana92], Saber [VVS87], SPICE [Nag75], Spectre [Kun95], or Titan [FWZ$^+$92]. These tools use sophisticated transistor models and consequently yield accurate results. On the other hand, depending on the type of simulation, they may require a considerable amount of computation power.

As an alternative, *symbolic equations* can be evaluated faster by orders of magnitude. While there are automatic tools to build symbolic equations for a given circuit, they are usually restricted to DC and AC performances [GWS89, VDL+01]. Furthermore, the complexity of the resulting equations is hard to keep under control [GRFRV02].

Response surface models try to combine the accuracy of numerical simulation with the fast evaluation of symbolic equations. Based on comprehensive simulations, numerical models can be built [HS96, Ziz01]. It depends on the actual application whether the initial consumption of simulation time can be compensated by the subsequent use of the model.

Sizing Method For a given set of parameter values, the performance values can uniquely be determined using simulation or symbolic equations. The inverse map in the opposite direction, i.e. from performances to parameters, is usually not unique and also unknown. The parameters mostly outnumber the performances, which causes ambiguity in the inverse map. The designer can exploit these degrees of freedom for performance optimization. Hence, the sizing task can be interpreted mathematically as an optimization problem.

One way to eliminate the ambiguity is the use of design knowledge: The sizing problem is simplified heuristically until the resulting mathematical problem has a unique solution. A final sizing result can then be obtained from the execution of a fixed sequence of symbolic calculation steps, which are defined in a so-called *design plan*. Note that this knowledge-based approach is no optimization process in the strict sense.

Other approaches exploit the available degrees of freedom mathematically, either *deterministically* or *stochastically*. Numerical *deterministic techniques* are mostly based on gradient information and tend to find (local) optima efficiently. In order to escape local optima, *stochastic approaches* allow a degradation in the objective function at a certain probability. Many of these approaches mimic natural phenomena. *Simulated annealing* is based on the observation that perfect crystals result from an infinitely slow cooling process of a melted mass. *Evolutionary* techniques rely on the improvement of circuit features by operations which copy the natural processes of mutation, crossover, and selection. Yet, the ability to escape local minima comes at the price of a large number of performance evaluations.

Classification of Automatic Circuit Sizing Techniques The two criteria concerning performance evaluation and sizing method can be used to categorize automatic sizing tools from the literature, as shown in Table 1.1.

The performance space exploration techniques presented in this work are simulation-based and deterministic for the following reasons: Especially in the industry, simulation is the prevailing method for performance evaluation because it combines accuracy and ease of use. Due to the high cost of simulation, advanced deterministic algorithms are used here in order to keep the overall execution times down.

	Heuristic	Deterministic Optimization	Stochastic Optimization
Numerical Simulation		[NRSVT88, AEG+00]	[MFDCRV94, ORC96] [KPRC99, PKR+00]
Symbolic Equations	[De87, ETP89] [HRC89]	[KSG90, MC91] [HEL92, dMHBL98] [dMHBL01, VDL+01]	[GWS90, ORC96] [DLG+98, VDL+01]
Numerical Models		[Ziz01, DGS03]	[ABD03]

Table 1.1: Classification of Automatic Sizing Tools according to Performance Evaluation Method (vertical) and Sizing Method (horizontal)

Role of Performance Space Exploration within Sizing Process In the sizing process, usually a number of performances have to be optimized simultaneously. Typically, there is not a single solution that optimizes all performances at the same time. Instead, there is a trade-off situation where one performance can only be improved at the cost of another. Most automatic sizing techniques, however, yield only a single solution where it is not obvious to the designer why this particular solution was chosen. Performance space exploration tools help to examine the trade-offs and to pick a certain solution deliberately. In addition, the resulting information can help to evaluate the performance capabilities of a given circuit topology, hence facilitating topology selection [SG03,SGA04]. That way, performance space exploration provides a link between the two design steps of topology selection and circuit sizing.

1.1.3 Hierarchical Circuit Sizing

For digital circuits, a hierarchical *top-down* design style has widely been adopted: Starting from an integral view of the system, the latter is partitioned into subsystems, which in turn are decomposed into functional blocks. In this way, implementation details are added until the final realization is found. Accordingly, the original specifications are successively broken down to elementary specifications for basic cells which are available in predefined libraries. This pure top-down design style used to be feasible thanks to the abstract digital view. It has led to a remarkable degree of automation and a high productivity in the area of digital design. Yet, especially in modern platform-based digital designs, the non-ideal behavior of the available building blocks has to be taken into account already at higher levels of abstraction [CDSS02]. In fact, this consideration of non-ideal effects already in early design stages has always been characteristic for analog design.

Figure 1.2 shows the hierarchical decomposition of an analog system as exemplified by a phase-locked loop (PLL). The entire circuit is partitioned into three functional blocks, which in turn consist of a large number of transistors. In a top-down siz-

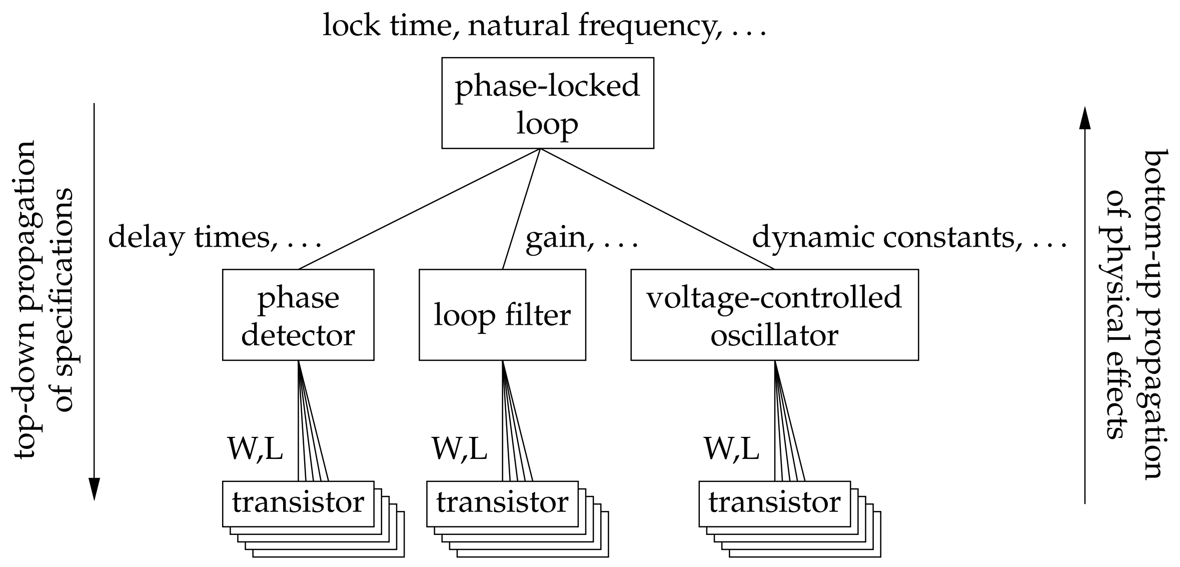

lock time, natural frequency, . . .
phase-locked loop
delay times, . . .
gain, . . .
dynamic constants, . . .
phase detector
loop filter
voltage-controlled oscillator
W,L
W,L
W,L
transistor
transistor
transistor
top-down propagation of specifications
bottom-up propagation of physical effects

Figure 1.2: Hierarchical Sizing of an Analog System

ing process, the PLL specifications such as lock time or natural frequency are broken down into specifications for the individual blocks. These block specifications are finally mapped onto transistor specifications which amount to channel widths and lengths (W, L). Unfortunately, such a top-down refinement of specifications may easily produce overambitious block specifications if the performance capabilities of the underlying analog circuit implementations are not taken into account. Consequently, low-level physical effects have to be propagated bottom-up: At each level of abstraction, there has to be a description of the performance capabilities of the respective functional blocks. In [CCH+99,CDSS02,DJSV03,DSV04,DGV+04], the availability of such an explicit block characterization is identified as a fundamental prerequisite to any reuse-oriented platform-based design of SOCs. Thus, performance space exploration techniques turn out to be a cornerstone of modern SOC design.

In summary, hierarchical analog design is characterized by a *top-down* propagation of specifications and a *bottom-up* propagation of physical constraints. Therefore, performance space exploration techniques provide the indispensable link between the abstraction levels. They can also help to select a suitable circuit topology if several implementation alternatives are available.

1.2 State of the Art

The essential role of bottom-up component characterization techniques in hierarchical sizing approaches has widely been acknowledged [HS96, KCJ+00, CDSS02, DJSV03,DSV04]. Nevertheless, not all top-down sizing techniques do this component characterization by means of performance space exploration in the strict sense. Instead, some of them model the relation of component parameters and performances.

In [FOK96], numerical response surface models are generated based on *design of experiment* techniques. In contrast to this quantitative model, a qualitative model is employed in [DNAV99]: The parameter space is partitioned into subspaces in which component parameters and performances either change in the same direction, in the opposite direction or are independent. As results of the accompanying sizing techniques, both approaches aim at finding entire parameter intervals instead of unique solutions. Thus, they focus on a top-down propagation of the specifications rather than on the bottom-up propagation of physical effects. The explicit characterization of a component entirely in the performance space is not their intent. In other words, they do not perform the abstraction step of fully decoupling the designable parameters at the distinct abstraction levels. Instead, they provide an efficient link between the component parameters and performances.

In this thesis the explicit identification of boundaries in the performance space is regarded as an essential feature of any performance space exploration technique in a narrower sense [DJSV03]. As a common denominator, all such exploration techniques map bounds from the circuit *parameter* space into the circuit *performance* space. Hence, they rely on an efficient performance evaluation. Especially for mixed-signal circuits, this often requires a considerable amount of computational resources [SGA03b]. Consequently, performance space exploration techniques were proposed, which were customized to certain circuit types [dMHHM+99,dMH02,VG03,BGH04]. The approach in [BGH04], for example, particularly aims at delta-sigma analog-to-digital converters: To reduce costly time-domain simulations to the minimum, approximate linear formulae are used for performance evaluation in a first stage. Only promising circuit realizations are further examined in detail using simulation. To explore the entire performance space, this approach relies on the fact that for delta-sigma analog-to-digital converters, there is only a limited number of configurations. Especially for challenging mixed-signal circuits such a customized approach is justified. Nevertheless, this thesis focuses on general approaches which allow the examination of a broad range of analog circuits.

1.2.1 Accurate Partial Exploration

The achievable performance values of a circuit topology are described by a region in the performance space. Generally speaking, performance space exploration techniques examine the boundary of this region, which is typically nonlinear. Accordingly, nonlinear approaches are required if accuracy is important. Since it is rarely possible to determine the boundary analytically, nonlinear performance space exploration methods sample the performance space and determine distinct points which characterize the boundary. Unfortunately, the resulting computational requirements grow exponentially with the number of examined performances. In practical cases, virtually all approaches aiming at an accurate description of the nonlinear performance space boundary can only be applied to a few performances of particular interest. Nevertheless, this is often sufficient, e.g. for visualization purposes.

While in an interactive environment a set of sampling points might already convey some insight, often a more systematic mathematical representation of the performance boundaries is desired. In [HS96], specifications for an operational amplifier are systematically varied and passed to the analog synthesis environment OASYS [HRC89]. This tool tries to create an according circuit by executing a predefined design plan based on symbolic equations. Response surface modeling techniques then relate the given specifications and the obtained performance values.

In many applications it is enough to examine those parts of the performance space in which the different performances become "simultaneously optimal" in some sense. This leads to a trade-off situation which can be described mathematically by the concept of *Pareto optimality*. For this reason, many performance space exploration approaches focus on this particularly interesting region, the so-called *Pareto front* [dMHBL98, dMHBL01, DG03, WKWC03]. There are different methods to actually identify the Pareto front.
The authors of [DG03] use a genetic algorithm for this task. Such an optimization algorithm mimics the natural evolution process including gene mutation and crossover. In this context, the numerical values of the designable circuit parameters represent the genes. Thus, a circuit is characterized by its genome of parameter values. In [DG03], the circuit fitness is evaluated based on circuit simulation. The result is used to decide on the reproductive success of this circuit instance. The proposed performance space exploration technique starts with a random set of genomes. In the course of the evolution process, the circuit population is drawn to the Pareto front. Unfortunately, like all stochastic optimization techniques, this approach requires a great number of performance evaluations resulting in serious simulation costs. As an alternative, the genetic algorithm in [WKWC03] uses symbolic equations for a quick performance estimation.
Similarly, in [dMHBL98, dMHBL01], the circuit behavior is described by means of *posynomial* symbolic models. This means that a point on a performance boundary can be determined by solving a convex optimization problem. For this type of problems, there is a unique global optimum which can be found extremely fast using deterministic optimization techniques. It is therefore possible to simply sweep some performance specifications and to solve the resulting optimization problems. Since even a large number of boundary points can be found without great effort, no particular strategy for an efficient selection of sample points is required in [dMHBL98, dMHBL01].

1.2.2 Approximate Comprehensive Exploration

For efficiency reasons, all exploration techniques discussed above restrict themselves in two ways: First, they select only a few important performances in order to escape the curse of dimensionality. Second, they focus on particularly interesting parts of the associated feasible performance space and just examine a fraction of its entire boundary. There are a number of approaches in the literature which are different in the second aspect: They examine the entire performance space boundary and yield a description of the feasible performance space as a closed region. Nevertheless, this particular problem has gained far less attention than the low-dimensional performance space exploration.

The authors of [DJSV03] focus on efficient nonlinear performance space modeling for a large number of dimensions. It is shown that a numerical modeling technique using *support vector machines* can capture strong nonlinearities in high-dimensional spaces accurately and efficiently. Instead of quantitatively describing relations between different performances, this numerical model separates feasible points in the performance space from infeasible ones. Based on a large number of training samples, the numerical parameters of the model are adjusted to allow a good amount of generalization while minimizing the danger of misclassifications. Even though the modeling works efficiently even in high-dimensional spaces, the applicability of this approach is limited to a few performances due to the large amount of required training data.

At the price of a lower accuracy, performance spaces with a larger number of dimensions can be described using linear approximations. A work based on [dMHBL98, dMHBL01] is presented in [dMH03]. Again, the circuit behavior is approximated by posynomial models. The optimizations, however, do not only focus on the Pareto front, but involve the entire performance space. A convex combination of the solution points yields a polyhedral approximation to the feasible performance space. Although this exploration technique is based on nonlinear analysis methods, it yields a linear description of the feasible performance space.

By the same token, symbolic equations are the basis of a performance space exploration approach described in [VLv$^+$95, VDL$^+$01], which uses early work presented in [Lee90,LH91]. In a first step, the individual performances are examined separately, and lower and upper limits are determined for each of them. The resulting feasible hypercube is often too optimistic since it does not consider correlations between the different performances. That is the reason why in a second step, linear performance dependencies are considered [LH91] and the hypercube approximations are refined by additional faces. Again, nonlinear symbolic analysis yields a linearized description of the feasible performance space.

In contrast to that, the approach presented in [MV89] relies on linearizations entirely. Although this work primarily aims at the test of analog circuits, its techniques are applicable to performance space exploration as well. To model random parameter variations, value ranges are assigned to the individual designable parameters. In this way, a hyperbox is defined in the parameter space. The relation of the parameters and the performances is described by a linear approximation which is obtained via nonlinear circuit simulation. The parameter hyperbox is then projected into the circuit performance space, geometrically resulting in a polytope.

1.2.3 Reasons for Performance Limitations

The performance boundaries of a given topology are ultimately caused by technological and topological constraints. Therefore, any performance space exploration technique relies on a concise definition of these constraints. Surprisingly, many publications only contain little or vague information on this matter.

Almost any performance space exploration technique defines lower and upper bounds for the individual designable parameters of the circuit. With this restriction, the authors in [FOK96, DJSV03] assume that any circuit instance which can be simulated successfully is acceptable. Going beyond that, most authors agree on keeping transistors in saturation without going into detail [VLv+95,DNAV99,DG03]. A few publications, especially on knowledge-based or symbolic approaches, discuss additional constraints like symmetry, matching or maximum currents [HRC89, dMHBL98]. However, even these approaches are far from a systematic or even automatic derivation of performance-limiting constraints.

1.3 Objectives of the Work

The following sections give account for general considerations which led to the development of the algorithms presented in this dissertation.

1.3.1 Systematic Derivation of Performance-Limiting Constraints

In spite of the outstanding importance of the performance-limiting constraints, Section 1.2.3 states a surprising lack of systematics in their derivation. As an answer, Section 2.1.3 systematically reviews prior work presented in [GZEA01]. The significance of this work to performance space exploration is discussed in Sections 2.2 and 2.3.

1.3.2 Choice of Simulation-Based Deterministic Algorithms

Resource requirements are a major concern for any automatic circuit design tool. To keep the computational costs low, many approaches resort to symbolic performance descriptions [HRC89, dMHBL98, VDL+01]. Their unquestioned advantage is that they allow extremely fast performance evaluations. They have a number of disadvantages, though: First, symbolic equations usually only apply to DC and AC behavior. Second, they often require a considerable amount of setup effort. Third, their accuracy is limited. Fourth, industrial circuit designers predominantly rely on numerical circuit simulation rather than symbolic techniques. In line with these arguments, a general trend toward *simulation-based* techniques has been observed [GR00].

Unfortunately, circuit simulations often require considerable computational resources. That is the reason why a high efficiency of the proposed algorithms is of particular importance. Advocates of stochastic algorithms usually emphasize their ability to escape local minima. Yet, this feature comes at the price of extremely high computational costs. In contrast, deterministic algorithms tend to consume far less computational resources, but they might get stuck in local optima. Fortunately, it could be shown that the behavior of a circuit is usually well-natured as soon as it works in the correct region of operation [SEGA99]. Hence, *deterministic* methods are used in the new algorithms in order to keep the execution times in reasonable limits.

1.3.3 Application Scenarios

The main challenge for any performance space exploration technique is the vastness of the space to be examined. Consequently, some simplification has to be applied in virtually all practical situations. As indicated previously, there are two major simplification schemes:

- *Reduce the examined portions of the performance space to the most interesting ones.*
  First, a small number of performances is selected, typically two or three. Second, the examination of the performance boundaries is focused on those areas where the performances become "simultaneously optimal". With these restrictions, an accurate nonlinear analysis becomes practical.
- *Allow a higher number of targeted circuit performances, but limit the accuracy of the exploration algorithm.*
  With this restriction, a high-dimensional exploration at reasonable accuracy levels is viable. With a highly efficient exploration algorithm, the region of achievable circuit performance values can even be approximated in its entirety.

It is obvious that none of the above two strategies is superior to the other one. Instead, there is a trade-off between accuracy and efficiency. The actual choice of a particular algorithm depends on the particular application. The goal of this work is to provide advanced performance space exploration techniques for the two scenarios described above. Both of them offer a combination of efficiency and accuracy well beyond the state of the art. Since the industrial applicability is a major requirement, both algorithms were implemented based on the commercial design tool WiCkeD [Mun] and haven proven their strength for different example circuits.

1.3.4 Proposed Algorithms

1.3.4.1 Accurate Low-Dimensional Nonlinear Performance Space Exploration

In Section 1.2.1, a number of low-dimensional performance trade-off analysis techniques were reviewed. None of them combines the accuracy and flexibility of simulation with an efficient deterministic exploration technique. The algorithm presented in Chapter 3 closes this gap. Additionally, the new performance space exploration technique allows the analysis of those constraints which inhibit a further performance improvement. The application area of this algorithm includes performance trade-off visualization as an interactive design aid and topology selection. The presented approach is the first one to use the algorithm presented in [DD98] for trade-off analysis of analog circuits.

1.3.4.2 Approximate High-Dimensional Linear Performance Space Exploration

Section 1.2.2 discussed performance space exploration techniques which aim at a larger number of circuit performances. However, there is no simulation-based technique that convincingly deals with a high dimensionality. The algorithm from [Lee90] could be used albeit not very efficiently. The approach presented in [MV89] could be adapted for linearized performance space exploration but it only allows hyperboxes as feasible parameter spaces.

This dissertation presents a technique which allows a fast exploration of high-dimensional performance spaces. Due to the complexity of the problem, it relies on a simulation-based sensitivity analysis to linearize the circuit behavior. It applies a well-established algorithm [DE73] which so far has not been used for performance space exploration. The approximate performance space exploration technique covered in Chapter 4 allows a consistent bottom-up propagation of physical effects in high-dimensional performance spaces at a reasonable accuracy. The targeted application area is primarily automatic hierarchical optimization, as outlined in Section 5.4.

1.3.5 Previous Publications

A total number of twelve publications arose from this work. The particular role of high-dimensional performance space exploration for hierarchical optimization is subject of [SGA01, Ste01, SGA02, SGA03a, MSGS05]. The performance potential of hierarchical simulation of large mixed-signal systems is highlighted in [SGA03b]. A low-dimensional nonlinear performance space exploration is described in [SGA03c, SGA03d]. A high-dimensional linear performance space exploration technique is presented in [SGA04, SGA05]. The automatic derivation of performance-limiting constraints from a given circuit topology is covered in [MSG03]. [SPS+03] shows how these constraints can be used to find a beneficial starting point for any type of circuit optimization technique.

1.3.6 Organization of this Dissertation

This thesis is organized as follows: Chapter 2 introduces the process of nominal circuit sizing and describes the impact of sizing constraints on the achievable circuit performances. Chapter 3 describes a low-dimensional nonlinear performance space exploration technique, and Chapter 4 a full-dimensional linear exploration method. Experimental results for both methods are presented in Chapter 5. Chapter 6 summarizes the main ideas of this work. Appendix A details how the sizing constraints can be utilized to find a good starting point for a subsequent circuit performance space exploration. Alternative linear performance space exploration approaches are reviewed in Appendix B, while the main ideas of stochastic optimization algorithms are summarized in Appendix C.

1.4 Summary

A large fraction of today's integrated circuits includes analog functionality. While mature automatic digital design tools are available, analog design automation is still in an early stage. The work at hand offers tools for the analog design frontend including topology selection and circuit sizing. In both flat and hierarchical design, the performance capabilities of a given circuit topology are of particular interest. To this end, two performance space exploration techniques have been developed. One of them accurately analyzes a low-dimensional subspace of interest using nonlinear optimization techniques. The other approach yields approximations to high-dimensional performance spaces based on linear techniques. Both procedures are simulation-based for accuracy and deterministic for efficiency.

Chapter 2

Problem Description

This section introduces the terminology and the fundamental concepts which are required to mathematically describe the sizing process and to formalize the task of performance space exploration.

2.1 Nominal Circuit Sizing

As stated in Section 1.1.2, this work mainly focuses on nominal circuit sizing which is usually referred to as *circuit sizing* or simply *sizing* in the following sections. In this thesis, these terms are used in a very broad sense: Any assignment of parameter values in such a way that the nominal circuit performances meet certain optimization criteria is called sizing. Since the outcome of the sizing procedure is commonly also called *a sizing*, the terms *sizing process* and *sizing result* are used for clarity where appropriate.

2.1.1 Circuit Parameters

Analog circuit sizing is usually done at a comparatively low level of abstraction where a circuit is described by a transistor netlist. For a fixed topology and production technology, the circuit behavior is controlled by the values of its *parameters*. Basically, there are three types of parameters [Ziz01]:

- *Designable parameters* are under full control of the designer. They comprise of, for instance, CMOS channel widths and lengths, or capacitor and resistor values.
- *Statistical parameters* describe the variations in the manufacturing process. They are beyond the control of the circuit designer because they are given by the production technology.

- *Operational parameters* take into account the variability of the operating conditions, such as the ambient temperature, the supply voltage, or the load conditions. The ranges of the operational parameters are given as part of the specifications and cannot be controlled by the designer.

For nominal design, fixed values are assigned to the statistical and operational parameters. Thus, solely the designable parameters $\mathbf{p}$ with[*]

$$\mathbf{p} \in \mathbb{R}^m \tag{2.1}$$

are considered in this context. The terms designable parameters and circuit parameters are hence interchangeable.

2.1.2 Circuit Performances

From a customer point of view, a certain circuit implementation is characterized by its *performances* $\mathbf{f}$. For a digital component such as a pad driver, important performances are the power consumption and the propagation delay. As an example of an analog block, an operational amplifier is described by its DC gain and its transit frequency among others. In this thesis, the term performances refers to the input/output behavior of a circuit in a black box fashion. Furthermore, the circuit behavior is determined via circuit simulation and the actual performance values result from a postprocessing of the raw simulation data.

For a certain technology, a given circuit topology maps its parameters $\mathbf{p}$ to its performances $\mathbf{f}$. Consequently, the evaluation of the performances, including simulation and postprocessing, can formally be written as a function evaluation:

$$\mathbf{f} = \mathbf{f}(\mathbf{p})\,, \quad \mathbf{f} \in \mathbb{R}^n\,. \tag{2.2}$$

Note that usually the parameters outnumber the performances, i.e. $m > n$.

2.1.3 Sizing Constraints

Analog circuits are built up hierarchically: Individual transistors form transistor pairs which constitute elementary building blocks such as current mirrors or differential pairs. These transistor pairs are combined again to obtain larger building blocks such as cascode current mirrors. Hence, a complete circuit can be interpreted as a hierarchical combination of basic building blocks.

Most of these structures imply particular restrictions which have to be met for a proper circuit operation. Examples are matching or saturation. These restrictions can

[*] In this thesis, regular lower case letters denote scalars. Vectors are written in bold lower case. Matrices are bold capitals.

be captured systematically by means of *sizing constraints* which frequently appear in the literature, albeit with different names [HRC89, VLv^{+}95, dMHBL98, DNAV99, dMHBL01, VDL^{+}01, GZEA01, MV01, DG03, DSV04]. Nevertheless, most authors take them as granted and do not systematically describe their derivation.

The approach presented in [GZEA01, MSG03] allows an *automatic* setup of the sizing constraints for a given circuit topology. It consists of two main steps: First, circuit substructures are identified bottom-up in a hierarchical fashion as described in Section 2.1.3.1. Based on the information thus obtained, sizing constraints are assigned to the individual transistors in a second step as exemplified in Section 2.1.3.2.

2.1.3.1 Topology Analysis

For CMOS technology, a number of fundamental analog building blocks was identified based on [GZEA01, Ziz01]. There are five levels of hierarchy as depicted in Table 2.1. Note that blocks at a certain hierarchy level are *pairs* of blocks taken from lower levels. The atomic building blocks at level 0 are single transistors. Level 1 comprises of all meaningful pairs of transistors. Level 2 covers pairs of structures from levels 0 and 1, and so on.

Based on this classification of building blocks, a circuit topology can be analyzed in a bottom-up fashion. The concept shall be explained using an example. For a more rigorous treatment, see [GZEA01, MSG03].

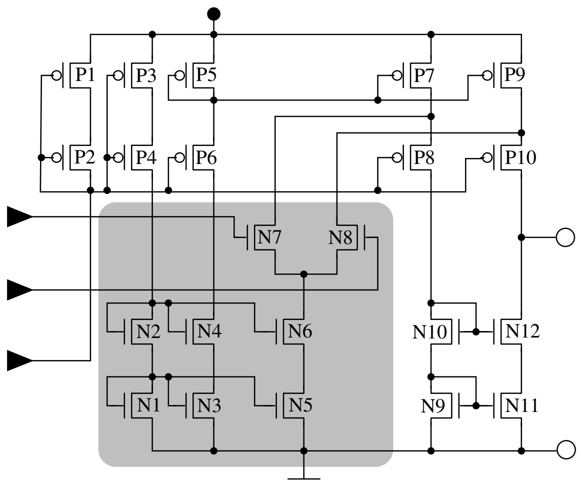


Figure 2.1: Folded Cascode Operational Amplifier

A schematic of a folded cascode operational amplifier is given in Figure 2.1. Although it consists of only 22 transistors, a number of 47 building blocks can be identified. Owing to the hierarchical method, the results of the analysis can be represented by means of a topology tree as depicted in Figure 2.2. It shows the detected building blocks in distinct hierarchy levels according to Table 2.1. The tree is a complete representation of the shaded area in Figure 2.1.

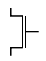
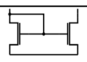
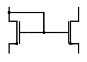
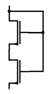
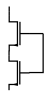
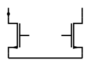
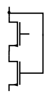
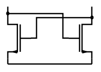
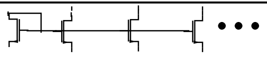
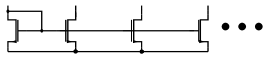
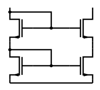
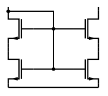
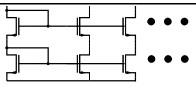
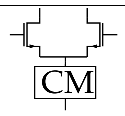
Function		Schematic (NMOS) (PMOS analogously)	Hierarchy Level
Voltage-controlled resistor Voltage-controlled current source	(res) (cs)		0
Simple current mirror	(cm)		1
Level shifter	(ls)		
Voltage reference 1	(vr1)		
Current mirror load	(cml)		
Differential Pair	(dp)		
Voltage reference 2	(vr2)		
Flip-flop	(ff)		
Level shifter bank	(LSB)		2
Current mirror bank	(CMB)		
Cascode current mirror	(CCM)		
4-Transistor current mirror	(4TCM)		
Cascode current mirror bank	(CCMB)		3
Differential stage (CM ∈ { cm, CCM, 4TCM, CCMB })	(DST)		4

Table 2.1: Basic Analog Building Blocks

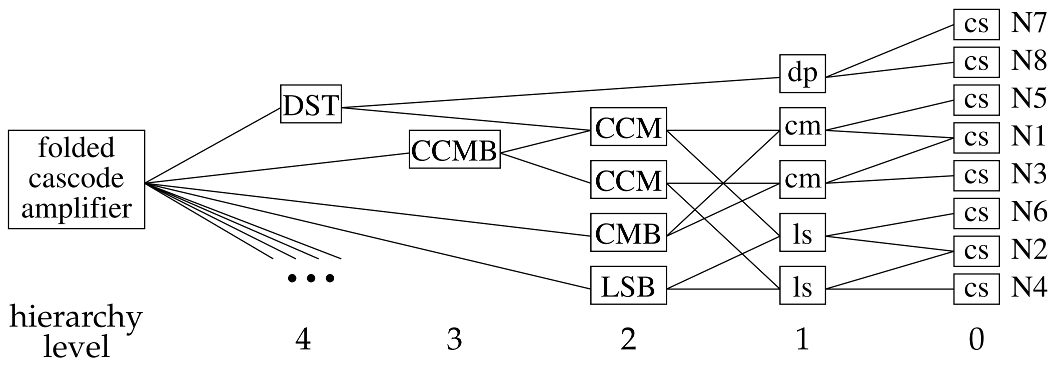


Figure 2.2: Topology Tree of Folded Cascode Amplifier

For example, it is evident that the transistors N7 and N8 operate as voltage-controlled current sources (cs, level 0) and form a differential pair (dp, level 1). It constitutes a differential stage (DST, level 4) together with a cascode current mirror (CCM, level 2). The latter in turn is made up of a current mirror (cm, level 1) and a level shifter (ls, level 1) with according transistors working as current sources (cs, level 0). In addition to being the basis of automatic sizing constraint generation, the hierarchical decomposition and its representation as a tree is a valuable tool to visualize a given circuit topology.

2.1.3.2 Constraint Assignment

A given building block only exhibits the expected behavior if the sizing constraints are met. Three latter can be classified by three categories:

1. Geometric / electric: *Geometric* sizing constraints directly refer to transistor geometries. *Electric* constraints need to be evaluated based on circuit simulation. At the current stage of development, the sizing constraints specify the DC operating point only. This is sufficient for circuits operating in the AC domain and for biasing circuitry. The temporary violation of these constraints in transient operation, however, cannot be prevented in all cases by this approach.
2. Function / robustness: *Functional* constraints have to be met unconditionally in order to allow a building block to fulfill the desired function. If, for example, the transistors of a differential pair do not work in saturation, then the entire circuit might not even exhibit the intended fundamental functionality (e. g. constant signal from an "oscillator"). *Robustness* constraints account for both variations in the manufacturing process and in the operating conditions already in the nominal design phase.
3. Inequality / equality: *Inequality* sizing constraints require that electric or geometric circuit quantities exceed or remain below certain thresholds. *Equality* constraints exist only for geometric quantities and thus demand that circuit parameters may only differ by a constant factor.

Usually, a transistor is part of a multitude of building blocks at different levels of abstraction. Eventually, all sizing constraints refer to transistor quantities, either geometric or electric. Nevertheless, the constraints which are assigned to a certain transistor reflect the entire path from the respective level-0 node to the top node.

The sizing constraints for transistor N7 are listed below based on the results from Figure 2.2. For each sizing constraint, the type is given according to the classification above.

$$v_{ds,N7} - (v_{gs,N7} - V_{th}) \geq V_{sat,min} \qquad \text{electric / function / inequality} \tag{2.3}$$
$$v_{ds,N7} \geq 0 \qquad \text{electric / function / inequality} \tag{2.4}$$
$$v_{gs,N7} - V_{th} \geq 0 \qquad \text{electric / function / inequality} \tag{2.5}$$
$$W_{N7} \cdot L_{N7} \geq A_{min} \qquad \text{geometric / robustness / inequality} \tag{2.6}$$
$$W_{N7} \geq W_{min} \qquad \text{geometric / robustness / inequality} \tag{2.7}$$
$$L_{N7} \geq L_{min} \qquad \text{geometric / robustness / inequality} \tag{2.8}$$
$$L_{N7} = L_{N8} \qquad \text{geometric / function / equality} \tag{2.9}$$
$$W_{N7} = W_{N8} \qquad \text{geometric / function / equality} \tag{2.10}$$
$$|v_{ds,N7} - v_{ds,N8}| \leq \Delta v_{ds,max} \qquad \text{electric / function / inequality} \tag{2.11}$$
$$v_{gs,N7} - V_{th} \leq V_{gs,max} \qquad \text{electric / robustness / inequality} \tag{2.12}$$

Transistor N7 has to work as a voltage-controlled current source (cs). Therefore, (2.3)–(2.5) provide for saturation while (2.6)–(2.8) make sure that the transistor is less sensitive to local parameter variations.

Since N7 forms a differential pair (dp) together with N8, both transistors have to be sized identically to avoid mismatch, cf. (2.9) and (2.10). The difference between the drain-source voltages must not be too large in order to avoid a systematic current mismatch, cf. (2.11). The effective gate-source voltage is limited by (2.12) because the differential pair has to provide accurate current differences.

Finally, N7 is part of a differential stage (DST). Although no generic sizing constraints are assigned to this building block, this affiliation is mandatory for the assignment of the differential pair sizing constraints. In other words, structures that look like differential pairs, but are not part of a differential stage are not subject to the respective sizing constraints. Hence, the entire path from the lowest to the highest hierarchy level has to be taken into consideration when the sizing constraints are assigned.

The limit values like A_{min} in (2.6) or $V_{sat,min}$ in (2.3) can be chosen by the designer as safety margins. In this way, some heuristic account is already taken of manufacturing and operational variations in the nominal sizing step already.

As shown above, a large number of sizing constraints is assigned to each transistor. While transistors are the key components in electronic circuits, there may certainly be additional components such as capacitors and resistors. For their component values, there are usually only explicit constraints which are comparable to the geometric sizing constraints for transistors. For simplicity, the term *geometric sizing constraint* will be used for any type of explicit constraint hereafter. Table 2.2 lists the total number of equality and inequality constraints for six operational amplifiers. The first one is the folded cascode architecture from above. Especially for larger designs, a manual constraint assignment would be tedious and prone to errors.

Amplifier	# Transistors	# Equality Constraints	# Inequality Constraints
1	22	34	189
2	22	22	182
3	23	16	176
4	23	30	184
5	27	20	204
6	31	28	255

Table 2.2: Number of Sizing Constraints for Different Operational Amplifiers

In summary, the sizing constraints capture elementary design knowledge in the form of mathematical expressions. They can be evaluated directly or based on DC simulations. From a formal point of view, inequality sizing constraints bear resemblance to circuit performance specifications. Yet, circuit performances usually describe the external black-box behavior of a circuit, while sizing constraints refer to the internal behavior. Moreover, circuit specifications are explicitly given by the customer, whereas sizing constraints implicitly result from technological and topological necessities.

2.2 Feasible Parameter Space

A particular vector of designable parameter values $\tilde{\mathbf{p}} \in \mathbb{R}^m$ can be interpreted geometrically as a point in the m-dimensional parameter space. Within this space, the sizing constraints describe a subspace of *feasible* parameter values. The latter represent technically meaningful sizings according to good design practice. The different types of constraints restrict the parameter values in particular ways.

Equality constraints are always given as explicit algebraic equations which directly correlate the circuit parameters, cf. (2.9) and (2.10). Consequently, they can be used to eliminate parameters algebraically. Thus, each of them effectively reduces the dimension of the parameter space by one. The actual sizing problem then has to be solved in an m'-dimensional space with $m' < m$. Since the algebraic parameter elimination based on equality sizing constraints is trivial, only the reduced parameter space $\mathbb{R}^{m'}$ is considered throughout this thesis. For ease of notation, the tick indicating the reduction is omitted in the remainder of this thesis.

Within the reduced parameter space, only *inequality constraints* have to be considered. They exclude additional portions of the parameter space without further reducing its dimension. With elementary algebraic transformations, all the sizing constraints can be combined into a single nonlinear vector inequality which is interpreted elementwise:

$$\mathbf{c}(\mathbf{p}) \geq \mathbf{0} \;\Leftrightarrow\; \underset{i \in \{1 \ldots q\}}{\forall} c_i(\mathbf{p}) \geq 0 \,. \tag{2.13}$$

Here, the index i denotes the i^{th} entry of the constraint vector function. The total number of sizing constraints is q. Even for small circuits there is a large number of constraints (cf. Section 2.1.3.2), i.e. $q \gg m$.

Figure 2.3 interprets (2.13) graphically for two constraints. Each of them describes the positive halfspace of the associated coordinate axis. The intersection of all these halfspaces yields the feasible constraint space $\mathcal{C}$.

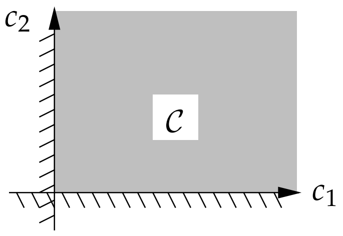


Figure 2.3: Feasible Constraint Space $\mathcal{C}$

Since the constraints are functions of $\mathbf{p}$, each of them implicitly defines a hypersurface in the parameter space corresponding to $c(\mathbf{p}) = 0$. This hypersurface separates the region where the associated constraint is violated ($c(\mathbf{p}) < 0$) from the region where it is satisfied ($c(\mathbf{p}) \geq 0$). For a geometric inequality constraint, the implicit representation of the associated hypersurface can be transformed into an explicit one analytically. After all, such a constraint is a simple algebraic expression in $\mathbf{p}$, cf. (2.6)–(2.8). An electric constraint, however, requires a nonlinear simulation, cf. (2.3)–(2.5). This means that in this case it is usually not possible to analytically derive an explicit representation of the associated hypersurface.

All sizing constraints are fulfilled in the feasible parameter space $\mathcal{P} \subset \mathbb{R}^m$:

$$\mathcal{P} = \{\mathbf{p} | \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\} \,, \quad \mathbf{c}(\mathbf{p}) \in \mathbb{R}^q \,, \quad \mathbf{p} \in \mathbb{R}^m \,. \tag{2.14}$$

The region $\mathcal{P}$ is bounded since the sizing constraints always include upper and lower bounds for each parameter. For CMOS channel sizes, for example, minimum lengths are ultimately given by the production technology if no larger bounds are required due to matching considerations. Upper bounds make sure that the devices do not become excessively large.

Not all constraints effectively restrict the feasible parameter space. Some of them are redundant, which means that their satisfaction is implied in the fulfillment of some other constraints. For a five-dimensional constraint space with one redundant constraint, the feasible parameter space is illustrated in Figure 2.4.

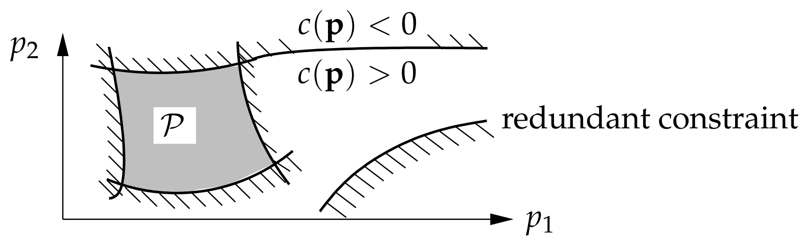


Figure 2.4: Nonlinear Inequality Constraints Defining Feasible Parameter Space $\mathcal{P}$

2.3 Feasible Performance Space

As discussed above, the sizing constraints implicitly define the feasible parameter space $\mathcal{P}$. According to Figure 2.5, there exists a corresponding feasible performance space $\mathcal{F} \subset \mathbb{R}^n$, which is the image of $\mathcal{P}$ under the map $\mathbf{f}(\cdot)$. Note that in general the boundary of $\mathcal{F}$, $\partial\mathcal{F}$, is not the image of $\partial\mathcal{P}$ [†].

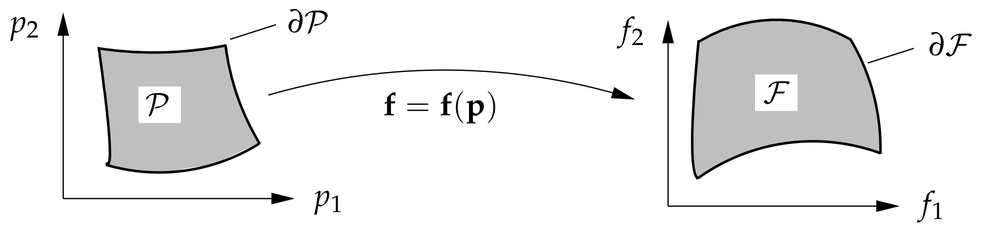


Figure 2.5: Nonlinear Relation of Feasible Parameter Space $\mathcal{P}$ and Feasible Performance Space $\mathcal{F}$

Unfortunately, both the constraints and the performances can only be simulated pointwise. Therefore, there is no way to determine $\mathcal{F}$ in its entirety. Yet, an implicit definition of the feasible performance space can be given easily: It comprises of all performance values which result from a set of feasible parameter values by simulation. In mathematical notation, this corresponds to

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{f} = \mathbf{f}(\mathbf{p}) \wedge \mathbf{p} \in \mathcal{P}\}, \quad \mathbf{p} \in \mathcal{P} \Leftrightarrow \mathbf{c}(\mathbf{p}) \geq \mathbf{0}. \tag{2.15}$$

[†] If $\mathbf{f}(\cdot)$ is a homeomorphism (not to be confused with homomorphism), then the boundaries are images of each other. A homeomorphism is a continuous bijective map with a continuous inverse. It maps neighborhoods to neighborhoods and thus preserves orientation and boundary components [CG78].

2.4 Performance Space Exploration

In (2.15), $\mathcal{F}$ is still coupled to the parameter space. The goal of performance space exploration is to find a description of $\mathcal{F}$ entirely in the performance space. This allows the evaluation of a given circuit without having to consider the actual implementation details. In general, performance space exploration identifies the boundary of $\mathcal{F}$ entirely (cf. Chapter 4) or in parts (cf. Chapter 3).

Knowledge of the feasible performance space is extremely useful for a designer:

- The boundary of the feasible performance space *illustrates the performance capabilities* of a given circuit topology: On $\partial\mathcal{F}$, the performances cannot further be improved without violating sizing constraints. Thus, topological and technological constraints ultimately define the performance limits.
- Based on $\partial\mathcal{F}$, different topologies can be evaluated and compared. Therefore, performance space exploration enables *topology selection*.
- Traditional optimization techniques yield only a single point on $\partial\mathcal{F}$. Its location heavily depends on the algorithmic details. Yet, the intricate impact of these details is often not obvious to the designer. With knowledge of $\partial\mathcal{F}$, the designer can choose among "equally optimal" solution alternatives deliberately.
- Larger electronic systems are usually composed of several functional blocks. When the performance capabilities of each of these blocks are known, then the overall system specifications can be broken down to realistic block specifications without having to consider the actual block implementations in full detail. In fact, a description of $\mathcal{F}$ entirely in the performance space yields sizing constraints at a higher level of abstraction.

2.5 Summary

This chapter introduced the terminology and concepts which are the basis of the discussion of different performance space exploration techniques in the following chapters. Circuit parameters and circuit performances were defined, and the particularities of nominal sizing were described. Owing to their fundamental importance for performance space exploration, the systematic derivation of sizing constraints was explained and exemplified. It was discussed how they define the feasible parameter space and how the latter has an image in the performance space. Finally, the interrelationship of the feasible performance space and performance space exploration was explained together with the role of performance space exploration within the design process.

Chapter 3

Nonlinear Performance Space Exploration

Performance space exploration is a computationally expensive process. Due to resource limitations there is a trade-off between accuracy on the one hand and the number of performances which can be examined simultaneously on the other hand. Therefore, this thesis presents two exploration techniques aiming at different application scenarios: This chapter presents a nonlinear technique aiming at high accuracy for low-dimensional performance subspaces. Chapter 4 introduces a method, which relies on linearizations in order to deal with high-dimensional performance spaces.

Typically, both the sizing constraints $\mathbf{c}(\mathbf{p})$ and the circuit performances $\mathbf{f}(\mathbf{p})$ are nonlinear functions in $\mathbf{p}$. Accordingly, nonlinear techniques are required to accurately identify the boundary $\partial\mathcal{F}$ of the feasible performance space $\mathcal{F}$. In the following section, a number of such performance space exploration techniques are discussed. To keep the computational costs within reasonable bounds, all of them have to reduce the problem complexity by two restrictions:

1. Only those parts of $\mathcal{F}$ are identified, where the performances become "simultaneously optimal".
2. Out of the entirety of performances only the most important ones are selected for exploration while minimum requirements are given for the remaining performances.

3.1 Multi-Objective Optimization

3.1.1 Problem Formulation

Usually, the sizing process tries to simultaneously obtain values for a number of performances that can be considered optimal in some sense. At the same time, the sizing constraints have to be met. Such a *multi-objective* or *multi-criteria* optimization problem can be stated as

$$\underset{\mathbf{p}}{\text{“optimize”}}\ \mathbf{f}(\mathbf{p}) = \begin{bmatrix} f_1(\mathbf{p}) \\ \vdots \\ f_n(\mathbf{p}) \end{bmatrix} \quad \text{s.t.} \quad \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,, \quad n \geq 2\,. \tag{3.1}$$

It is rarely possible to find a unique set of parameter values that optimizes all the performances at the same time. Instead, there is usually a trade-off situation where it is only possible to improve one performance at the cost of another. This leads to the concept of *Pareto optimality* [HM79,LTZ87,Sta88,EKO90,Ehr97,DD98,Deb01].

Without loss of generality, optimization means minimization[*] in the remainder of this chapter . In multi-objective optimization, a vector $\mathbf{a} = [a_1 \dots a_n]^T$ [†] is considered more optimal than a vector $\mathbf{b} = [b_1 \dots b_n]^T$ if it *dominates* $\mathbf{b}$:

$$\mathbf{a} \prec \mathbf{b} :\Leftrightarrow \underset{i\in\{1\dots n\}}{\forall} (a_i \leq b_i) \quad \wedge \underset{i\in\{1\dots n\}}{\exists} (a_i < b_i)\,. \tag{3.2}$$

A vector $\mathbf{f}^*$ [‡] is *Pareto optimal* within $\mathcal{F}$ if it is *non-dominated* in $\mathcal{F}$:

$$\neg \underset{\mathbf{f}\in\mathcal{F}}{\exists} \mathbf{f} \prec \mathbf{f}^*\,. \tag{3.3}$$

In Figure 3.1, point $\mathbf{f}^\diamond$ is dominated, but all the points on the arc between $\mathbf{f}^{*a}$ and $\mathbf{f}^{*b}$ are non-dominated. This portion of the boundary $\partial\mathcal{F}$ is denoted as $\partial\mathcal{F}^\prec$ here, and it is called *Pareto optimal front*. This part of $\mathcal{F}$ is especially interesting for a designer since it characterizes the ultimate performance capabilities of a topology and the trade-offs involved. The points $\mathbf{f}\in\partial\mathcal{F}^\prec$ are also called *efficient* points. The goal of low-dimensional performance space exploration is to compute $\partial\mathcal{F}^\prec$ efficiently and accurately.

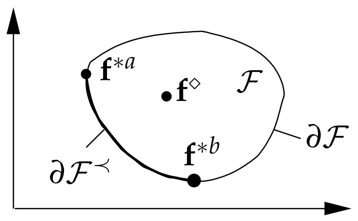


Figure 3.1: Feasible Performance Space $\mathcal{F}$ with Pareto Optimal Front $\partial\mathcal{F}^\prec$

The simultaneous exploration of numerous performances using nonlinear techniques easily exhausts the available computing resources. Yet, in many situations only a few

[*] For maximization, the objective function can simply be multiplied by (-1).

[†] The superscript ^{T} identifies a transposed vector.

[‡] Symbolic superscripts are used to mark special vectors.

performances are of particular interest, while for the remaining ones it is enough to meet certain minimum requirements. If for all performances optimization means minimization, then these requirements can be given as upper bound specifications. Mathematically, they have the same form as the sizing constraints. Additionally, both performances and constraints have to be evaluated using simulation. Thus, they can both be combined into an extended sizing constraint vector $\tilde{\mathbf{c}}(\mathbf{p})$. If, for example, the focus is on the first two performances out of n ($n \geq 4$ here), such a two-dimensional optimization problem would be stated as

$$\begin{gathered}\text{“optimize”}_{\mathbf{p}} \; \tilde{\mathbf{f}}(\mathbf{p}) \quad \text{s.t.} \quad \tilde{\mathbf{c}}(\mathbf{p}) \geq \mathbf{0} \\ \text{with} \quad \tilde{\mathbf{f}}(\mathbf{p}) \Leftrightarrow \begin{bmatrix} f_1(\mathbf{p}) \\ f_2(\mathbf{p}) \end{bmatrix} \quad \text{and} \quad \tilde{\mathbf{c}}(\mathbf{p}) \Leftrightarrow \begin{bmatrix} \mathbf{c}(\mathbf{p}) \\ f_{3,\max} - f_3(\mathbf{p}) \\ \vdots \\ f_{n,\max} - f_n(\mathbf{p}) \end{bmatrix}, \end{gathered} \tag{3.4}$$

where $\mathbf{c}(\mathbf{p})$ represents the original sizing constraints. A comparison to (3.1) reveals that the mathematical problem remains unchanged. Therefore, only the original problem (3.1) is considered further.

For practical problems, it is usually not possible to obtain a description of the entire Pareto optimal front in closed form. Instead, this region hast to be discretized by a limited number of points. Practical experience shows that a linear interpolation of the resulting Pareto points yields a good approximation in most cases.

A good performance space exploration method has to meet two requirements [Zha03]: First, it should find advantageously distributed points on the Pareto front for efficiency reasons. Second, it must be able to deal with nonconvex fronts as well.

In the remainder of this section, well-known multi-objective optimization approaches are reviewed with emphasis on deterministic techniques. After that, an advanced deterministic trade-off analysis technique is introduced in Section 3.2.

3.1.2 Stochastic Solution Techniques

The most common stochastic optimization techniques are simulated annealing and genetic algorithms. Both of them mimic natural phenomena: Simulated annealing resembles the cooling process of a melted mass, which leads to a perfect crystal lattice if the temperature is lowered infinitely slowly. Genetic algorithms, in contrast, imitate the evolution of life with its mechanism of promoting superior individuals over the remaining ones. For further details, see Appendix C.

Especially genetic algorithms have been applied to the area of multi-objective optimization [EK96, DG03, WKWC03], although simulated annealing approaches have been reported as well [DHBS90, WGP96, UTO98].

Stochastic techniques are popular because they are less likely to get stuck in local optima than deterministic approaches. Unfortunately, this feature comes at the price of enormous resource requirements [DG02, DG03].

3.1.3 Deterministic Solution Techniques

Deterministic gradient-based techniques transform the multi-objective optimization problem (3.1) into a single-objective optimization formulation, which is then repeatedly solved. This is exemplified in the following subsections. For a more in-depth treatment, see [HM79, Sta88, EKO90, Ehr97, Deb01].

3.1.3.1 Weighted Sum

One way to turn (3.1) into a single-objective optimization problem is to combine all objectives into one by means of a scalar cost function $s : \mathbb{R}^n \mapsto \mathbb{R}$. In the simplest and most common implementation of this idea, s is a weighted sum:

$$s(\mathbf{f}(\mathbf{p})) = \mathbf{w}^T \cdot \mathbf{f}(\mathbf{p}) \,, \quad \mathbf{w} = [w_1 \ldots w_n]^T \,, \quad \underset{i \in \{1 \ldots n\}}{\forall} w_i > 0 \,. \tag{3.5}$$

This leads to the following scalar optimization problem:

$$\min_{\mathbf{p}} \; \mathbf{w}^T \cdot \mathbf{f}(\mathbf{p}) \quad \text{s.t.} \quad \mathbf{c}(\mathbf{p}) \geq \mathbf{0} \,. \tag{3.6}$$

This approach is illustrated in Figure 3.2, where two contour lines of $s(\mathbf{f})$ with constant objective values are shown. The solution $\mathbf{f}^*$ is where such a line is tangential to $\partial\mathcal{F}^{\prec}$. The orientation of the contour lines is determined by the weight vector $\mathbf{w}$. It can be shown that for a convex Pareto optimal front, every efficient point is the solution of a problem according to (3.6) with a proper weight vector $\mathbf{w}$ [DD97].

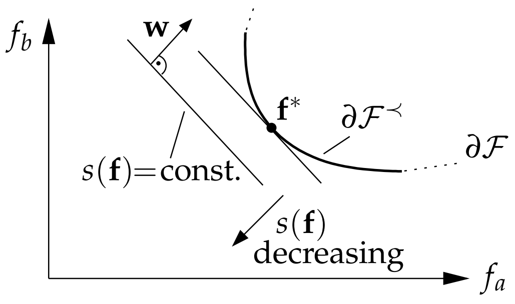


Figure 3.2: Weighted Sum: Efficient Point $\mathbf{f}^*$ According to (3.6)

Yet, the weighted sum method has a major drawback: It is not able to detect all points on nonconvex Pareto fronts. In Figure 3.3, the points on the arc between $\mathbf{f}^{\Diamond}$ and $\mathbf{f}^{\Box}$ would not be obtainable using this method. Other approaches employing a scalar cost function basically suffer from the same problem.

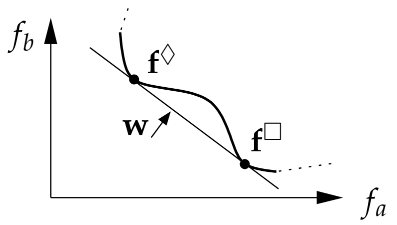


Figure 3.3: Weighted Sum Drawback: Nondetectable Efficient Points in Nonconvex Regions

3.1.3.2 Objectives as Constraints

A different method to obtain a scalar optimization problem is to turn all objectives but one into constraints:

$$\min_{\mathbf{p}} \; f_j(\mathbf{p}) \quad \text{s.t.} \quad \underset{i\in\{1\ldots n\}\backslash\{j\}}{\forall} f_i \;\leq\; f_{i,max} \;\wedge\; \mathbf{c}(\mathbf{p}) \;\geq\; \mathbf{0}\,,\; j \;\in\; \{1\ldots n\}\,. \tag{3.7}$$

Here, the optimization is controlled by varying the maximum values of the n–1 performance constraints $f_{i,max}$. In contrast to the weighted sum method, this approach also yields the efficient points in nonconvex parts of the Pareto optimal front as shown on the left in Figure 3.4.

In fact, the constraint method does not yield any dominated points. On the right in Figure 3.4, an optimization with $f_{i,max3}$ as a bound would yield $\mathbf{f}^{\lozenge}$. Therefore, the arc between $\mathbf{f}^{\lozenge}$ and $\mathbf{f}^{\square}$ would result in a gap in the solution curve. However, this consideration is primarily of academic interest since experience shows that such a curvature is very unlikely for actual analog circuits.

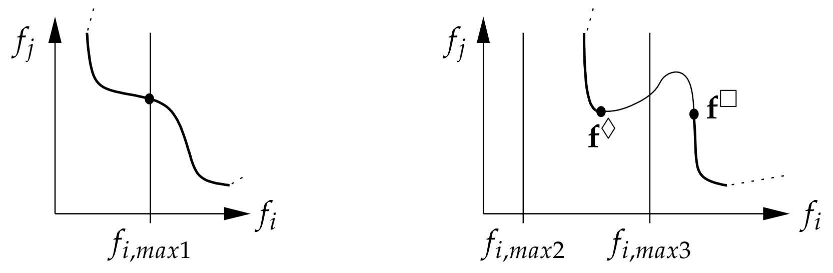


Figure 3.4: Constraint Method Detects Efficient Points in Nonconvex Regions and Leaves Out Non-efficient Parts

In contrast to the weighted sum formulation, which yields solutions for any weight vector with nonnegative components, the constraint method does not lead to solutions if the specified bounds are infeasible as shown with $f_{i,max2}$ on the right in Figure 3.4. On the other hand, different bounds may lead to the same Pareto point. Both effects waste valuable computing resources.

In summary, the constraint method overcomes the limitation regarding nonconvex Pareto fronts. Yet, unfavorable choices of the adjustable performance bounds do not yield useful results.

3.1.3.3 Goal Attainment

The approach from the previous section transforms all performances but one into constraints. The goal attainment method due to [GH75] goes even further in the sense that it transforms *all* objectives into constraints:

$$\min_{\mathbf{p}} z \quad \text{s.t.} \quad \mathbf{f}(\mathbf{p}) \leq \mathbf{f}_{\text{goal}} + z \cdot \mathbf{w} \;\; \wedge \;\; \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,, \qquad \underset{i \in \{1 \ldots n\}}{\forall} w_i > 0\,. \tag{3.8}$$

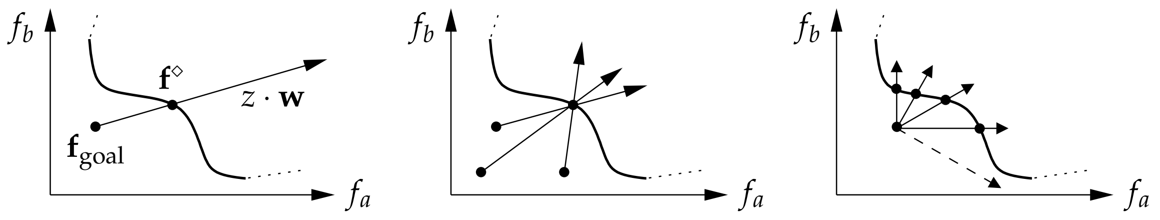


Figure 3.5: Goal Attainment

The geometric idea behind this approach is illustrated in the left part of Figure 3.5. Originally developed for the determination of a single solution point, goal attainment allows to specify a performance target $\mathbf{f}_{\text{goal}}$. Depending on the value of the slackness variable z, the goal can be overachieved ($z < 0$) or underachieved ($z > 0$). The weight vector $\mathbf{w}$ relates the different performances. Geometrically, $(\mathbf{f}_{\text{goal}} + z \cdot \mathbf{w})$ describes a ray in the performance space. At the solution point $\mathbf{f}^\diamond$, the constraint $(\mathbf{f}(\mathbf{p}) \leq \mathbf{f}_{\text{goal}} + z \cdot \mathbf{w})$ is satisfied with equality.

With a proper choice of performance goal and weight vector, the determination of all efficient points is possible, even in nonconvex Pareto fronts. The required values of $\mathbf{f}_{\text{goal}}$ and $\mathbf{w}$ are not obvious, though. First of all, there are some degrees of freedom because a single efficient point can be the result of numerous $\mathbf{f}_{\text{goal}}$ / $\mathbf{w}$ combinations, cf. Figure 3.5, middle. Furthermore, even for a fixed performance goal, a uniform variation of the weight vector does not lead to an even spread of solution points, cf. Figure 3.5, right. Finally, for certain choices of $\mathbf{f}_{\text{goal}}$ and $\mathbf{w}$, there is no solution at all, as indicated by the dashed ray in Figure 3.5, right.

Note that the idea of a search ray in the performance space allows the identification of efficient points even in nonconvex parts of the Pareto front. Yet, the goal attainment method does not yield advantageously distributed solution points and does not provide a methodology for the proper selection of $\mathbf{f}_{\text{goal}}$ and $\mathbf{w}$. The next section introduces an approach which bears resemblance to goal attainment but successfully overcomes its limitations.

3.2 Normal-Boundary Intersection

3.2.1 General Idea

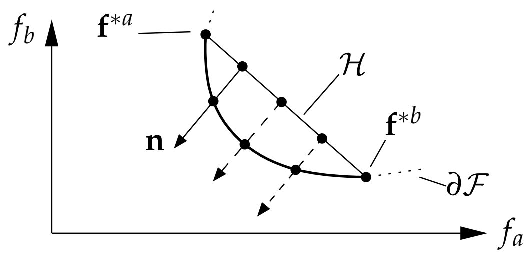


Figure 3.6: Normal-Boundary Intersection

The Normal-Boundary Intersection method (NBI) [DD98] relies on four corner stones as illustrated in Figure 3.6:

1. Individual minima: $\mathbf{f}^{*i}$
   There are special points $\mathbf{f}^{*i}$ in a Pareto front where the individual performances f_i show their global minima. These points are boundary points of the Pareto front and are called *individual minima*. It has been pointed out earlier that caution has to be taken not to initiate searches in regions where no efficient points can be found. Therefore, extreme points of the Pareto front are sought in a first step. If no two performances can be minimized at the same time, there is one unique individual minimum for each of the n performances. Beyond their algorithmic importance, the individual minima are of great interest to a designer.
2. Convex hull of individual minima: $\mathcal{H}$
   With the observation that the individual minima are boundary points of the Pareto front, it is intuitive that there are efficient points "in between" them. The simplest way to describe an area "in between" the individual minima is their convex hull $\mathcal{H}$. Geometrically, $\mathcal{H}$ is a polyhedron with the individual minima as its corner points. It is (n–1)-dimensional if n individual minima exist. The basic idea of NBI is to initiate searches for efficient points starting from points on $\mathcal{H}$.
3. Normal vector to convex hull: $\mathbf{n}$
   A search on a line perpendicular to $\mathcal{H}$ toward the origin yields Pareto optimal points if the respective portion of $\partial\mathcal{F}$ is convex. If the searches start from points evenly distributed on $\mathcal{H}$ and run in the direction of a vector $\mathbf{n}$ normal to $\mathcal{H}$, then the solution points on the efficient front are also well-balanced. Note that it is not even necessary for $\mathbf{n}$ to be exactly orthogonal to $\mathcal{H}$. The only requirement on $\mathbf{n}$ is that it contains significant negative entries for every performance f_i.
4. NBI optimization problem formulation
   A suitable optimization formulation captures the geometric deliberations from above mathematically. The NBI optimization problem can be solved using nonlinear optimization.

The NBI algorithm assumes that the individual minima $\mathbf{f}^{*i}$ limit the Pareto Front. This is based on the requirement that the individual minima comprise of the *global* minima of the distinct performances. If for some reason only local minima were identified in the first step, the NBI method remains operational but yields inferior results. In the graphs shown in Figure 3.7, the individual minimum $\mathbf{f}^{*b}$ was identified correctly, whereas the other one, $\widetilde{\mathbf{f}^{*a}}$, is erroneous. In the left graph, only the fraction of the Pareto front between $\widetilde{\mathbf{f}^{*a}}$ and $\mathbf{f}^{*b}$ is identified, and it cannot be recognized that the dashed part has not been found. In the right graph, the NBI algorithm yields superfluous non-efficient points on the boundary curve between $\widetilde{\mathbf{f}^{*a}}$ and $\mathbf{f}^{*a}$. In this case, however, the error can be noticed easily and the algorithm can be restarted with the correct individual minimum $\mathbf{f}^{*a}$. Fortunately, these problems are primarily of academic interest because experience shows that real circuit behavior is much more well-behaved as soon as the sizing constraints are satisfied.

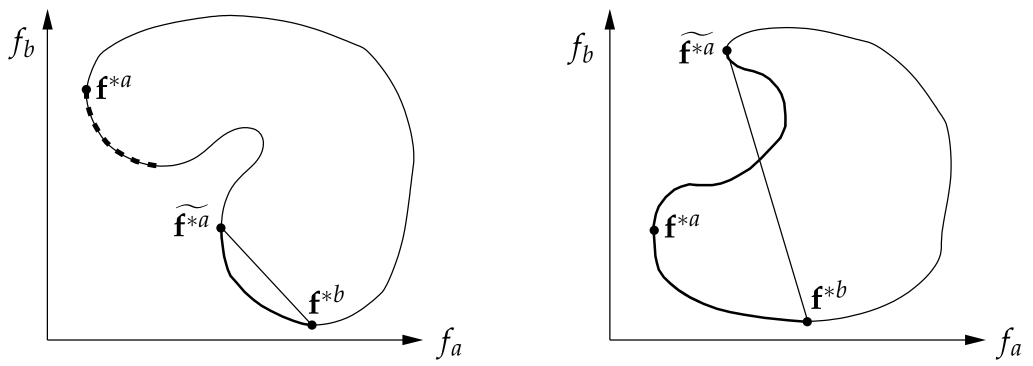


Figure 3.7: Effect of Erroneous Individual Minima: Omission of Efficient (left) and Inclusion of Non-Efficient Points (right)

3.2.2 Algorithmic Details

In this section, the four main ideas from above are elaborated mathematically.

1. Individual minima: $\mathbf{f}^{*i}$
   The individual minima $\mathbf{f}^{*i}$ can be determined using a conventional single-objective optimization formulation in f_i:

$$\mathbf{p}^{*i} = \underset{\mathbf{p}}{\operatorname{argmin}}\, f_i(\mathbf{p}) \quad \text{s.t.} \quad \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,, \quad i \in \{1 \dots n\}\,, \quad \mathbf{f} = [f_1 \dots f_n]^T\,,$$
$$\mathbf{f}^{*i} = \mathbf{f}(\mathbf{p}^{*i})\,. \quad (3.9)$$

   The argmin operator yields the argument leading to the minimum objective value.

Let

$$f_{i,min} = f_i^{*i} \tag{3.10}$$

denote the optimal value of performance f_i as obtained from (3.9). Analogously, $f_{i,max}$ could be defined as the upper bound of performance f_i within $\partial\mathcal{F}^{\prec}$. Since the exact knowledge of the upper bounds is of minor interest, it is sufficient to estimate them by the largest objective values occurring in the set of individual minima $\{\mathbf{f}^{*1}, \mathbf{f}^{*2}, \ldots, \mathbf{f}^{*n}\}$:

$$f_{i,\widetilde{max}} = \max(f_i^{*1}, \ldots, f_i^{*n}), \quad i \in \{1 \ldots n\}. \tag{3.11}$$

Adequate normalization of the objective values is crucial to any numeric optimization. A normalization of the performance values according to (3.10), (3.11) and

$$\hat{f}_i = \frac{(f_i - f_{i,min})}{(f_{i,\widetilde{max}} - f_{i,min})}, \quad i \in \{1 \ldots n\}, \tag{3.12}$$

yields a good improvement of the numerical condition of the resulting optimization problems[§]. Provided that $f_{i,min}$ is the global minimum of performance i, this normalization results in $\hat{f}_i \geq 0$. Negative values indicate an erroneous individual minimum according to Figure 3.7, right. Since the largest value of f_i within the Pareto front was only estimated by $f_{i,\widetilde{max}}$, even larger performance values might occur during the optimization process. Therefore, the upper bound of the normalized performance value range is usually greater than 1. Consequently, the suggested normalization maps the performance values to a range $[0, u]$ with $u \geq 1$.

2. Convex hull of individual minima: $\mathcal{H}$
If the normalized individual minima are combined in a matrix

$$\mathbf{F} = \left[\hat{\mathbf{f}}^{*1} \ \ldots \ \hat{\mathbf{f}}^{*n}\right], \tag{3.13}$$

then their convex hull can be written as

$$\mathcal{H} = \mathbf{F} \cdot \mathbf{w}, \quad \mathbf{w} = [w_1 \ldots w_n]^T, \quad w_i \geq 0, \sum_i w_i = 1, \quad i \in \{1 \ldots n\}. \tag{3.14}$$

Note that the diagonal elements of $\mathbf{F}$ are all zero due to the normalization according to (3.12). Geometrically, $\mathbf{F}$ maps a given weight vector $\mathbf{w}$ to a particular point on $\mathcal{H}$. This linear map establishes a straightforward relation between weights and points as illustrated in Figure 3.8: An even spread of weights results in an even distribution of points on $\mathcal{H}$, see also Figure 3.11.

3. Normal vector to convex hull: $\mathbf{n}$
The search along a family of normal vectors $\mathbf{n}$ offers the benefit of well-balanced

[§] Obviously, this normalization cannot be used for the determination of the individual minima. Here, other strategies should be applied such as the one outlined in Appendix A.2.1.

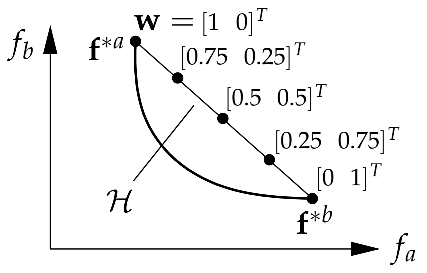


Figure 3.8: Geometric Interpretation of Weights

solution points. This trait is only impaired marginally if the search direction is not exactly normal to $\mathcal{H}$. Hence, the following quasi-normal vector $\tilde{\mathbf{n}}$, which is very easy to calculate, is sufficient:

$$\tilde{\mathbf{n}} = -\mathbf{F} \cdot \mathbf{1}\,, \quad \mathbf{1} = [1 \ldots 1]^T\,. \tag{3.15}$$

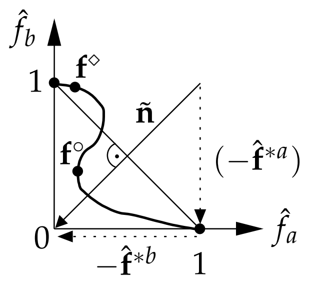


Figure 3.9: Normalized NBI in 2 Dimensions

In the two-dimensional case, $\tilde{\mathbf{n}}$ turns out exactly normal to $\mathcal{H}$ owing to (3.12): $\tilde{\mathbf{n}}$=[-1 -1]^T. This is illustrated in Figure 3.9: $\tilde{\mathbf{n}}$ results from the negative sum of the normalized individual minima $\hat{\mathbf{f}}^{*a}$ and $\hat{\mathbf{f}}^{*b}$, as indicated by the dotted arrows.
This figure also shows that NBI can trace nonconvex surfaces as well. In case of extreme curvature, NBI also identifies points on dominated parts of $\partial\mathcal{F}$, such as the arc between $\mathbf{f}^{\diamond}$ and $\mathbf{f}^{\circ}$. While this is a drawback in the strict mathematical sense, a designer might prefer a contiguous surface to a Pareto front with a gap. Fortunately, such extreme curvatures are very unlikely for practical circuits.
Figure 3.10 shows that if the normalization does not map all the individual minima to unit vectors, then $\tilde{\mathbf{n}}$ is no longer normal to $\mathcal{H}$. This happens if more than two performances are examined. Yet, a search along this quasi-normal direction also produces good results.
Note how for moderate curvature, evenly spread points on $\mathcal{H}$ yield well-balanced efficient points on the Pareto front. In the case of extreme curvature, a higher sampling density in certain parts of the Pareto front might be preferable. However, some more research is required for a reliable adaptive weight selection strategy, cf. Section 3.2.4.6.

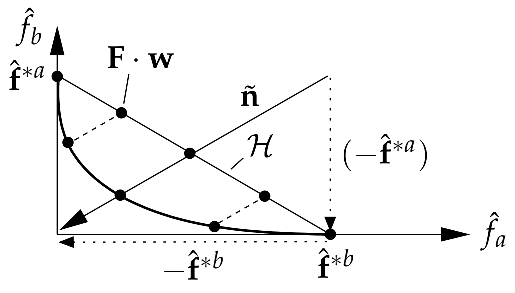


Figure 3.10: NBI with quasi-normal vector $\tilde{\mathbf{n}}$

4. NBI optimization problem formulation
   Finally, the following NBI optimization formulation combines all components described above:

$$\begin{bmatrix}\mathbf{p}^*\\ t^*\end{bmatrix} = \underset{\left[\begin{smallmatrix}\mathbf{p}\\ t\end{smallmatrix}\right]}{\operatorname{argmax}}\ t \quad \text{s.t.} \quad \mathbf{F}\cdot\mathbf{w} + t\cdot\tilde{\mathbf{n}} = \hat{\mathbf{f}}(\mathbf{p}) \ \wedge\ \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,;$$
$$w_i \geq 0\,, \sum_{i\in\{1\ldots n\}} w_i = 1\,; \qquad \mathbf{f}^* = \mathbf{f}(\mathbf{p}^*)\,. \tag{3.16}$$

   For a given vector $\mathbf{w}$, the solution to this problem is a unique Pareto optimal point $\mathbf{f}^*$. Note that the objective function just consists of a newly introduced parameter t. The geometric idea behind NBI is coded in a vector equality constraint: The term $\mathbf{F}\cdot\mathbf{w}\in\mathcal{H}$ defines the base point of a search ray, which is oriented in the direction of the quasi-normal vector $\tilde{\mathbf{n}}$. Therefore, the left-hand side of the equality constraint describes all points on this ray. The parameter t is a measure of the distance from $\mathcal{H}$. Since $\mathbf{c}(\mathbf{p})\geq\mathbf{0}$ ensures feasibility of the parameter values, $\hat{\mathbf{f}}(\mathbf{p})\in\mathcal{F}$ is a feasible performance vector according to (2.15). This optimization formulation can be seen as a *line search in the performance space*: The goal is to find a feasible performance vector that lies on the search ray and has the maximum distance from $\mathcal{H}$.
   For an even spread of efficient points, the components of $\mathbf{w}$ can be chosen according to

$$w_i = \frac{j_i}{k}, \quad k \in \mathbb{N}^+, j_i \in \begin{cases} \{0,1,\ldots,k\cdot(1-\sum_{l=1}^{i-1} w_l)\}\,, & 1\leq i<n \\ \{k\cdot(1-\sum_{l=1}^{i-1} w_l)\}\,, & i=n \end{cases}. \tag{3.17}$$

   This results in a total number of

$$N_n = \underbrace{\sum_{\alpha=0}^{k}\sum_{\beta=0}^{k-\alpha}\sum_{\gamma=0}^{k-\beta}\cdots\sum_{\omega=0}^{k-\psi}}_{(n-1)\text{ sums}} 1 \tag{3.18}$$

   points on $\mathcal{H}$. For the cases $n=2$ and $n=3$, (3.18) simplifies to

$$N_2 = k+1 \tag{3.19}$$
$$N_3 = \frac{(k+1)\cdot(k+2)}{2}\,. \tag{3.20}$$

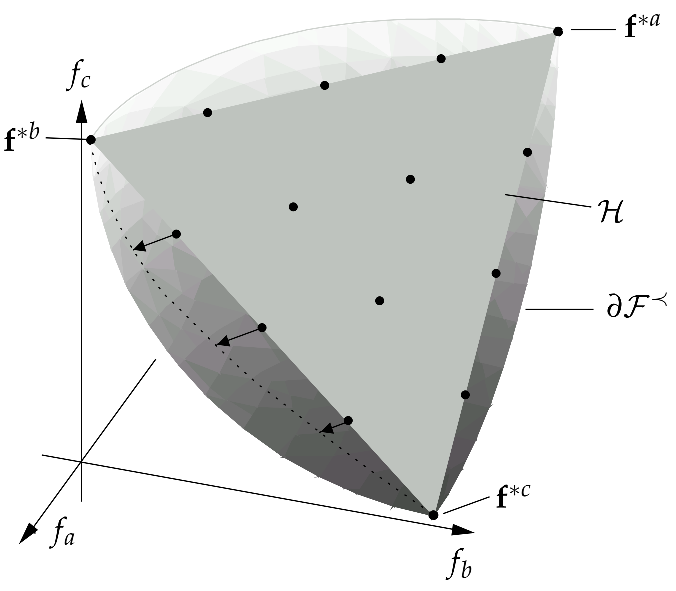


Figure 3.11: Coverage Problems in 3-dimensional Performance Space

Figure 3.11 shows the arrangement of base points on $\mathcal{H}$ for $n = 3$ and $k = 4$. These points are well-balanced and ensure a good coverage of the major part of the Pareto front. The result is shown in the left part of Figure 3.12. Near the boundary of $\partial\mathcal{F}^{\prec}$, however, some efficient points are left out if no extra care is taken. The dotted line in Figure 3.11 indicates that the boundary of $\mathcal{H}$ does not exactly correspond to the boundary of $\partial\mathcal{F}^{\prec}$. The reason is that the latter represents the trade-off of performance *pairs*, whereas the quasi-normal vector $\tilde{\mathbf{n}}$ governs *all three* performances. Consequently, an additional calculation of all two-performance trade-offs in the three-dimensional space guarantees a complete coverage of the entire Pareto front as shown in the right part of Figure 3.12. An analogous reasoning applies to the exploration of more than three performances.

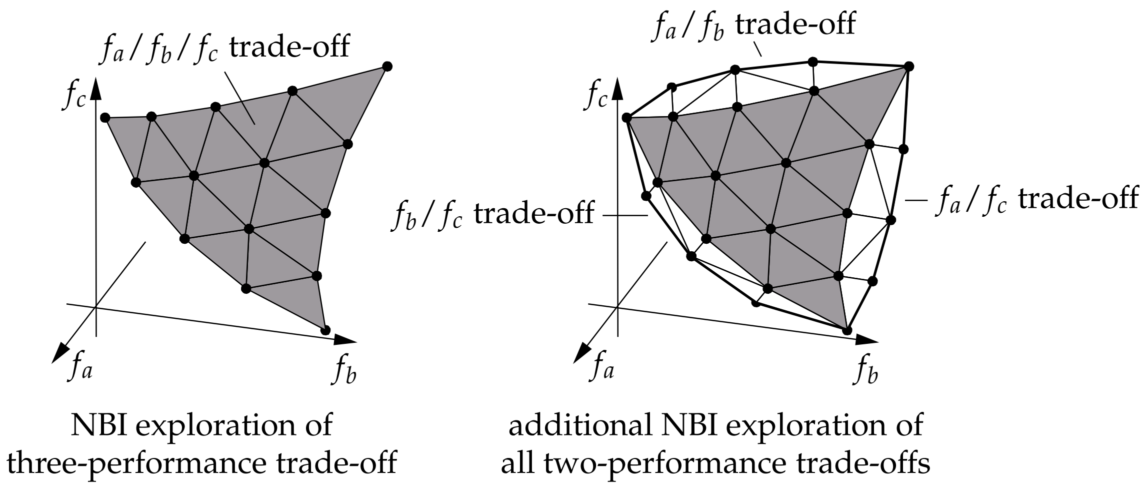


Figure 3.12: Complete Coverage of 3D Pareto Fronts

3.2.3 Practical Solution via SQP

To operationally solve the NBI optimization problems, the work at hand uses an adapted state-of-the-art SQP algorithm [Mat] in combination with a SPICE-like circuit simulator. Admittedly, like any gradient-based technique, SQP can get trapped in local optima. Yet, it was observed that circuit performances are usually well-natured as soon as all sizing constraints are fulfilled [GZEA01,DG03].

Genetic optimization approaches use the genetic difference of subsequent populations as a stop criterion. Therefore, they only yield solution points more or less close to the Pareto front. In contrast, a deterministic search advances to the actual boundary of $\mathcal{F}$ because it identifies the solution based on an analysis of the active constraints and of the slope of the objective function. The advantage is two-fold: First of all, the solution points are really located *on* $\partial\mathcal{F}$. Additionally, the SQP algorithm identifies the entire set of active sizing constraints, which effectively limit the performances. The stringency of the active constraints can even be quantified as explained below.

3.2.3.1 Analysis of Active Constraints

With the abbreviation

$$\mathbf{g}(\mathbf{p},t) \quad := \quad \hat{\mathbf{f}}(\mathbf{p}) - \mathbf{F}\cdot\mathbf{w} - t\cdot\tilde{\mathbf{n}}\,, \tag{3.21}$$

the Lagrangian function corresponding to (3.16) is given by

$$L(\mathbf{p},t,\boldsymbol{\lambda}) = t - \sum_{j\in\{1\ldots q\}} \lambda_j\cdot c_j(\mathbf{p}) - \sum_{j\in\{1\ldots n\}} \lambda_{q+j}\cdot g_j(\mathbf{p},t)\,. \tag{3.22}$$

Let $\mathbf{p}^*$ and t^* represent a solution to (3.16). With an appropriate vector of Lagrange multipliers $\boldsymbol{\lambda}^*$, the Karush-Kuhn-Tucker conditions [Fle87,NW99] can be satisfied:

$$\nabla_{\mathbf{p}} L(\mathbf{p}^*,t^*,\boldsymbol{\lambda}^*) \quad = \quad \mathbf{0} \tag{3.23}$$
$$\nabla_t L(\mathbf{p}^*,t^*,\boldsymbol{\lambda}^*) \quad = \quad \mathbf{0} \tag{3.24}$$
$$\mathbf{c}(\mathbf{p}^*) \quad \geq \quad \mathbf{0} \tag{3.25}$$
$$\mathbf{g}(\mathbf{p}^*,t^*) \quad = \quad \mathbf{0} \tag{3.26}$$
$$\underset{j\in\{1\ldots q\}}{\forall} \lambda_j \quad \geq \quad 0 \tag{3.27}$$
$$\underset{j\in\{1\ldots q\}}{\forall} \lambda_j^*\cdot c_j(\mathbf{p}^*) \quad = \quad 0 \tag{3.28}$$
$$\underset{j\in\{1\ldots n\}}{\forall} \lambda_{q+j}^*\cdot g_j(\mathbf{p}^*,t^*) \quad = \quad 0\,. \tag{3.29}$$

The SQP algorithm iteratively finds a solution to this nonlinear problem, which makes the gradient of the Lagrangian function disappear ((3.23), (3.24)), and which satisfies both the inequality (3.25) and the equality (3.26) constraints.

The j^{th} inequality constraint is called *active* in $\mathbf{p}^*$ if it is satisfied with equality, i.e. $c_j(\mathbf{p}^*) = 0$. Only in this case, the associated Lagrange multiplier λ_j may be positive due to (3.27) and (3.28). Conversely, the multipliers are zero for all inactive inequality constraints. For a designer, the active inequality constraints in $\mathbf{p}^*$ are of particular interest because they prevent a further performance improvement. After all, the circuit performances cannot further be improved without violating the corresponding sizing constraints. Note that the vector equality constraint of the NBI problem (3.16) is not of interest in this context because it does not correspond to any sizing constraint.

Beyond a mere identification of the performance-limiting constraints, their influence can be quantified using the Lagrange multipliers. Mathematically, the question is how the optimum value t^* changes if the constraints are tightened slightly:

$$c_j(\mathbf{p}) - \epsilon_j \geq 0\,, \quad \epsilon_j > 0\,, \quad j \in \{1 \dots q\}\,. \tag{3.30}$$

Then, (3.22) turns into

$$L'(\mathbf{p}, t, \boldsymbol{\lambda}, \boldsymbol{\epsilon}) = t - \sum_{j\in\{1\dots q\}} \lambda_j \cdot (c_j(\mathbf{p}) - \epsilon_j) - \sum_{j\in\{1\dots n\}} \lambda_{q+j} \cdot g_j(\mathbf{p}, t)\,. \tag{3.31}$$

Consequently, $\mathbf{p}^*$, t^*, and $\boldsymbol{\lambda}^*$ become functions of $\boldsymbol{\epsilon}$. Due to (3.22), (3.28), and (3.29)

$$t^*(\boldsymbol{\epsilon}) = L'(\mathbf{p}^*, t^*, \boldsymbol{\lambda}^*, \boldsymbol{\epsilon})\,. \tag{3.32}$$

Hence,

$$\frac{\mathrm{d}t^*(\boldsymbol{\epsilon})}{\mathrm{d}\boldsymbol{\epsilon}} = \frac{\mathrm{d}L'(\mathbf{p}^*, t^*, \boldsymbol{\lambda}^*, \boldsymbol{\epsilon})}{\mathrm{d}\boldsymbol{\epsilon}} \tag{3.33}$$

$$= \frac{\partial L'}{\partial \mathbf{p}^*} \cdot \frac{\partial \mathbf{p}^*}{\partial \boldsymbol{\epsilon}} + \frac{\partial L'}{\partial t^*} \cdot \frac{\partial t^*}{\partial \boldsymbol{\epsilon}} + \frac{\partial L'}{\partial \boldsymbol{\lambda}^*} \cdot \frac{\partial \boldsymbol{\lambda}^*}{\partial \boldsymbol{\epsilon}} + \frac{\partial L'}{\partial \boldsymbol{\epsilon}}\,. \tag{3.34}$$

Here, the operators $\frac{\mathrm{d}}{\mathrm{d}\mathbf{x}} = [\frac{\mathrm{d}}{\mathrm{d}x_1}\ \frac{\mathrm{d}}{\mathrm{d}x_2}\ \dots]$ and $\frac{\partial}{\partial \mathbf{x}} = [\frac{\partial}{\partial x_1}\ \frac{\partial}{\partial x_2}\ \dots]$ yield row vectors, whereas the nabla operator yields a column vector by convention. A Jacobian matrix $\frac{\partial \mathbf{y}}{\partial \mathbf{x}}$ has the derivative $\frac{\partial y_i}{\partial x_k}$ at row i and column k. Recall from (3.23) and (3.24) that

$$\left(\frac{\partial L'}{\partial \mathbf{p}^*}\right)^T = \nabla_{\mathbf{p}} L'(\mathbf{p}^*, t^*, \boldsymbol{\lambda}^*) = \mathbf{0} \tag{3.35}$$

$$\text{and} \quad \left(\frac{\partial L'}{\partial t^*}\right)^T = \nabla_t L'(\mathbf{p}^*, t^*, \boldsymbol{\lambda}^*) = \mathbf{0}\,. \tag{3.36}$$

The partial derivative of the Lagrangian function with respect to the Lagrange multipliers is

$$\frac{\partial L'}{\partial \boldsymbol{\lambda}^*} = -\sum_{j\in\{1\dots q\}} (c_j(\mathbf{p}^*) - \epsilon_j) - \sum_{j\in\{1\dots n\}} g_j(\mathbf{p}^*, t^*)\,. \tag{3.37}$$

Let the set A contain the indices of the active constraints, and let I comprise of the indices of the inactive ones. In (3.37),

$$c_j(\mathbf{p}^*) - \epsilon_j = 0\,, \quad j \in A\,, \tag{3.38}$$

for active inequality constraints and

$$g_j(\mathbf{p}^*, t^*) = 0 \tag{3.39}$$

due to (3.26). For inactive inequality constraints with $c_j(\mathbf{p}^*) - \epsilon_j > 0$, (3.28) yields

$$\lambda_j^* = 0\,, \quad j \in I\,. \tag{3.40}$$

Assume that the components of $\boldsymbol{\epsilon}$ are sufficiently small not to change the set of active constraints. Then the lambda values of the inactive inequality constraints remain zero and thus show a zero sensitivity with respect to a variation of $\boldsymbol{\epsilon}$:

$$\frac{\partial \lambda_j^*}{\partial \boldsymbol{\epsilon}} = \mathbf{0}\,, \quad j \in I\,. \tag{3.41}$$

A combination of (3.37), (3.38), (3.39) and (3.41) yields

$$\frac{\partial L'}{\partial \boldsymbol{\lambda}^*} \cdot \frac{\partial \boldsymbol{\lambda}^*}{\partial \boldsymbol{\epsilon}} = \mathbf{0}\,. \tag{3.42}$$

Furthermore, with $\boldsymbol{\lambda}_{(1\ldots q)} = [\lambda_1 \ldots \lambda_q]$ and (3.31),

$$\frac{\partial L'}{\partial \boldsymbol{\epsilon}} = \boldsymbol{\lambda}_{(1\ldots q)}\,. \tag{3.43}$$

Finally, (3.34), (3.35), (3.36), (3.42), and (3.43) yield

$$\frac{\mathrm{d}t^*(\boldsymbol{\epsilon})}{\mathrm{d}\boldsymbol{\epsilon}} = \boldsymbol{\lambda}_{(1\ldots q)}\,. \tag{3.44}$$

Therefore, the Lagrange multipliers can be interpreted as the sensitivities of the optimum objective values with respect to a tightening of the active constraints.

From (3.16) it can be derived that

$$\mathrm{d}\hat{\mathbf{f}}^* = \tilde{\mathbf{n}}\,\mathrm{d}t\,. \tag{3.45}$$

Substitution and denormalization (cf. (3.12)) yield the sensitivities of the circuit performances

$$\frac{\mathrm{d}f_i^*}{\mathrm{d}\epsilon_j} = (f_{i,\widetilde{max}} - f_{i,min}) \cdot \tilde{n}_i \cdot \lambda_j\,. \tag{3.46}$$

These sensitivities are of great importance to a designer because they are a measure of how stringent the active constraints are in a certain Pareto point.

3.2.4 Efficiency Considerations

3.2.4.1 Choice of Beneficial Starting Point

It is well-known that especially deterministic optimization methods such as the SQP algorithm used here require a suitable starting point in order to find a good optimization result efficiently. If no problem-specific a priori knowledge is available, then such a starting point should safely satisfy all sizing constraints and leave room for the optimizer to move in any direction. The determination of such a starting point is described in Appendix A.

3.2.4.2 Elimination of Conceptual Parameter

A comparison of the generic optimization formulation (3.1) to the NBI problem (3.16) reveals that in the latter the set of designable parameters has been extended by the conceptual parameter t. Effectively, one dimension was added to the design space, which should make the optimization problem harder to solve.

Note that t only appears in the objective function and in the vector equality constraint. The latter comprises of n scalar equalities according to the number of performances. Each of them relates t to one performance $\hat{f}_i(\mathbf{p})$ analytically. Hence, one equality could be removed from the vector of equality constraints in order to replace t by a performance $\hat{f}_i(\mathbf{p})$. For the two-dimensional case, (3.16) results in

$$\min_{\mathbf{p}} \hat{f}_1 \quad \text{s.t.} \quad \hat{f}_1 - \hat{f}_2 + 2\,w - 1 \;=\; 0 \quad \wedge \quad \mathbf{c}(\mathbf{p}) \;\geq\; \mathbf{0}\,, \quad 0 \;<\; w \;<\; 1\;. \tag{3.47}$$

Practical experience shows that this simplification hardly affects the numerical efficiency of the NBI algorithm.

3.2.4.3 Jump Start

The NBI method yields discrete points on the Pareto front. For each of them, a numerical optimization is carried out using an SQP algorithm. Often, a new solution is located in the neighborhood of the previous one. Hence, an improved efficiency can be expected if an optimization run is initialized with the results from the preceding run.

For the identification of the individual minima, the associated performances f_i serve as objective functions, cf. (3.9). Hence, the respective optimization problems are substantially different and have to be solved afresh.

In contrast, the intermediate points on the Pareto front are identified by solving very similar optimization problems, which only differ in the choice of the weight vector $\mathbf{w}$, cf. (3.16). To efficiently solve such an optimization problem, the SQP algorithm builds a second-order model of the Lagrangian function (3.22). Using finite differences or the adjoint method, the Jacobian matrix of the Lagrangian can be determined via simulation. In contrast, second-order information, as represented by the Hessian matrix, is usually not accessible as easily. To obtain an approximation to the Hessian of the Lagrangian function, the utilized SQP algorithm iteratively combines the first-order information represented by the Jacobian with the help of the BFGS update formula [Fle87,NW99]. The final approximation to the Hessian matrix captures valuable second-order information on the Lagrangian at the solution point. Fortunately, a variation of the weight vector $\mathbf{w}$ in (3.21) does not affect the derivatives of the Lagrangian (3.22) and hence its Hessian. Therefore, the Hessian matrix and the solution vector found in one optimization run can be used to initialize the SQP algorithm for the subsequent run.

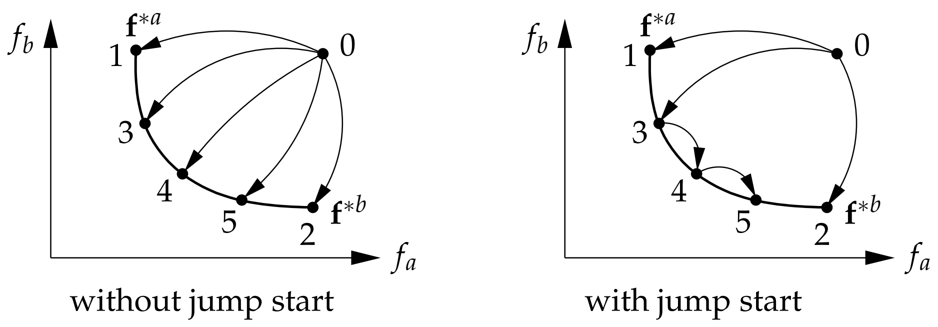


Figure 3.13: Efficiency Gain through Jump Start of Subsequent Optimization Runs

The effect of this jump start on the optimization process is visualized in Figure 3.13. The numbers indicate the order in which the Pareto points are determined, where 0 is the starting point. Without jump start, every optimization starts afresh from point 0. Even with jump start enabled, substantially different objective functions apply to the two individual minima (points 1 and 2) and to the first intermediate Pareto point 3. Hence, three optimization runs have to be performed which start with a newly initialized approximation to the Hessian matrix. Of course, the optimization resulting in point 3 could be started from point 1. The Hessian could not be reused, though. Experiments showed that in this case the optimization algorithm can easily terminate prematurely because during the short trajectory to point 3 not enough information can be collected for a good Hessian approximation. Hence, a start from point 0 turned out to be preferable in this case.

The benefit of the jump-start strategy is two-fold: First, the start from the previous result usually shortens the optimization trajectory, and second, the reuse of the Hessian matrix provides for an accurate local model of the Lagrangian right from the beginning of the optimization run. In practice, this strategy is rewarded by a substantial efficiency gain: In the situation depicted in Figure 3.13 with 5 Pareto points and two jump-start optimization runs, a 5%–30% reduction of the simulation count can be expected.

3.2.4.4 Equality vs. Inequality Constraint

In the NBI problem formulation (3.16), the equality constraint

$$\mathbf{F} \cdot \mathbf{w} + t \cdot \tilde{\mathbf{n}} = \hat{\mathbf{f}}(\mathbf{p}) \tag{3.48}$$

defines a line in the performance space according to Figure 3.14.

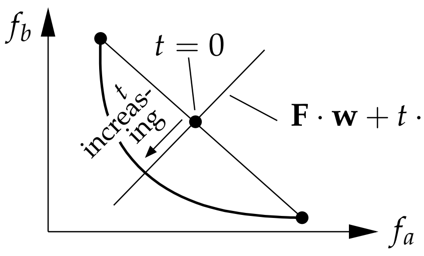


Figure 3.14: Definition of Search Line by Equality Constraint

Accordingly, the iterative SQP algorithm is forced to move on this line during the optimization process. This is illustrated conceptually in Figure 3.15: Starting from an initial point 0 in the performance space, the optimization engine first steps on the search line (point 1) and then takes a few more steps to find the Pareto optimal point.

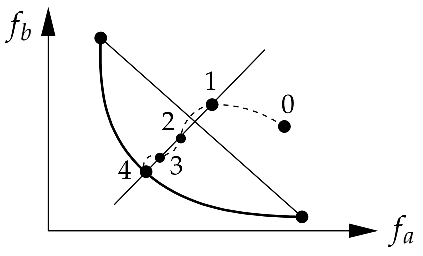


Figure 3.15: Optimization Run Subject to Equality Constraint

While the definition of a search line according to (3.48) is elegant conceptually, it is not advantageous numerically. After all, it heavily restricts the movements of the optimizer. The replacement of the vector equality constraint by an inequality constraint

$$\mathbf{F} \cdot \mathbf{w} + t \cdot \tilde{\mathbf{n}} \geq \hat{\mathbf{f}}(\mathbf{p}) \tag{3.49}$$

yields the same optimization results at an increased efficiency. Note that this change results in one more similarity between NBI and the Goal Attainment method from Section 3.1.3.3.

In (3.48), a given value of t designates a single point on the search line. A comparison to (3.49) reveals that the inequality constraint includes the equality constraint as a special case. For a fixed value of t, (3.49) designates a point on the line and additionally all the points, which it is dominated by, cf. (3.2). The shaded areas in Figure 3.16 represent all these points for three increasing values of t. In each case, the upper right corner point of the shaded area corresponds to the special case (3.48).

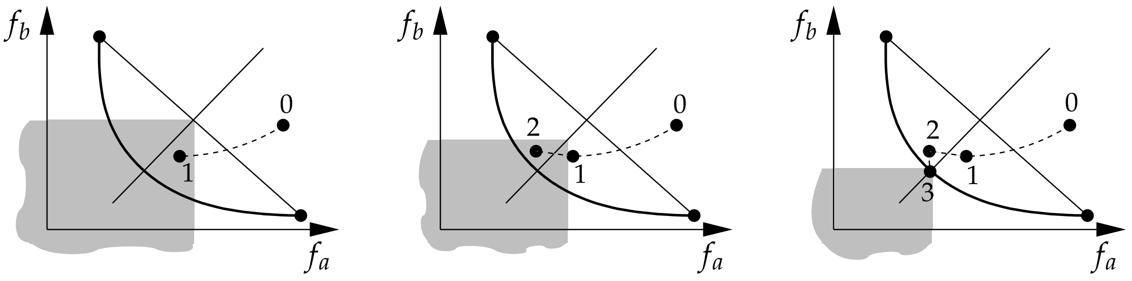


Figure 3.16: Optimization Run Subject to Inequality Constraint

The subfigures in Figure 3.16 illustrate three iteration steps of the modified optimization formulation. With an inequality constraint instead of an equality one, the optimization engine is free to surpass the performance values associated with a certain value of t. Practical experience shows that this increased flexibility results in a significant gain in numerical efficiency, especially in combination with the jump start feature described above. In comparison to the equality-based optimization with jump start, an additional reduction of the simulation count between 5% and 20% is typical.

3.2.4.5 Parallelization

For an even shorter execution time, the NBI approach can readily be parallelized: In a first stage, the individual minima can be calculated in parallel according to (3.9). In a second stage, the remaining efficient points are calculated using (3.16), which can be done in parallel again. In order to retain the jump start feature, the examination of the Pareto front should proceed from the individual minima toward the center of $\mathcal{H}$.

3.2.4.6 Adaptive Sample Density

In Section 3.2.2, an even spread of Pareto points was suggested for a good discretization of the Pareto front, cf. (3.17), Page 35. For the convex case, a computationally more efficient arrangement of sampling points can be obtained if curvature information of the Pareto front is taken into account.

In [Zha03] it was shown how the gradient projection method [GMW81, NW99] can be used to identify planes which are tangential to the Pareto front. The main idea is to project the gradients of the performances onto the feasible search space as defined by the nullspace of the active constraints. This enables a description of the tangent planes *entirely in the performance space*. The result can be interpreted as mutual sensitivities S_{ba} and S_{ca} of the conflicting performances $f_a^* = f_a(\mathbf{p}^*)$, $f_b^* = f_b(\mathbf{p}^*)$, and $f_c^* = f_c(\mathbf{p}^*)$ in a Pareto point $\mathbf{f}^* = [f_a^* \quad f_b^* \quad f_c^*]^T$:

$$S_{ba} = \left.\frac{\partial f_b}{\partial f_a}\right|_{\mathbf{p}^*}, \quad S_{ca} = \left.\frac{\partial f_c}{\partial f_a}\right|_{\mathbf{p}^*}. \tag{3.50}$$

For the two-dimensional case, this is illustrated in Figure 3.17.

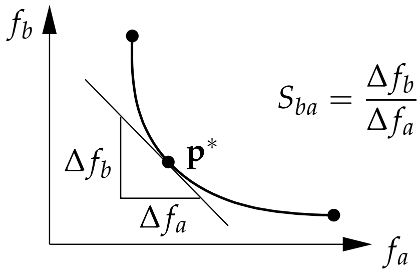


Figure 3.17: Tangent to Pareto Front

For convex Pareto fronts, these tangents can be used to realize an adaptive sampling strategy as illustrated in the left part of Figure 3.18. The intersection point of two tangent lines or three tangent planes, respectively, is projected onto the convex hull of individual minima $\mathcal{H}$. This yields a new weight vector $\mathbf{w}$ which identifies a search ray in a region where the Pareto front is likely to have a strong curvature. Unfortunately, this approach may exert an unpredictable behavior for nonconvex fronts. This is exemplified in the right part of Figure 3.18.

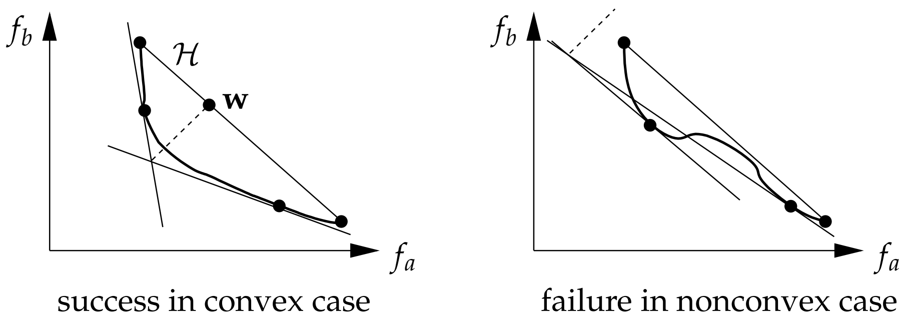


Figure 3.18: Location of Additional Refinement Point by Intersection of Tangents

3.3 Summary

This chapter explained why nonlinear performance space exploration amounts to multi-objective optimization. A systematic introduction to this optimization problem was given, whereupon a number of nonlinear solution strategies, both stochastic and deterministic, were reviewed along with their shortcomings. The Normal-Boundary Intersection (NBI) algorithm was introduced as a particularly beneficial optimization strategy and algorithmic details were discussed. The main advantages of the NBI method are two-fold: First, it identifies the Pareto front requiring only moderate resources. Second, it is able to identify the obstacles for a further performance improvement and even to quantify their stringency.

Chapter 4

Linear Performance Space Exploration

The previous chapter introduced a new approach to accurately identify the performance limits of a given circuit topology based on nonlinear optimization. Unfortunately, the nonlinear exploration of a high-dimensional performance space consumes a prohibitive amount of computational resources. Therefore, the corresponding techniques are forced to focus on a small number of performances, typically two or three. However, analog building blocks are often characterized by far more performances. That is the reason why an efficient high-dimensional performance space exploration technique can play a key role in topology selection and hierarchical sizing.

In this chapter, a novel performance space exploration technique is suggested, which calculates a *linearized approximation* to the range of feasible performance values. The algorithm is based on the idea that, by its operation, a circuit maps its feasible parameter space to its feasible performance space. Accordingly, a linearized description of the feasible parameter space is determined first. Then, the map into the performance space is approximated by a linear relation of circuit parameters and circuit performances. A new algorithm based on the *Fourier-Motzkin elimination* algorithm finally yields a linear approximation to the feasible performance space. Owing to the efficiency of this approach, it can deal with high-dimensional performance spaces which are too complex for nonlinear exploration methods.

In Section 2.3, the feasible performance space was mathematically characterized as

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{f} = \mathbf{f}(\mathbf{p}) \wedge \mathbf{p} \in \mathcal{P}\}\,, \quad \mathbf{p} \in \mathcal{P} \Leftrightarrow \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,. \tag{2.15}$$

In this formulation, $\mathcal{F}$ is still coupled to the parameter space. Yet, the goal of performance space exploration is a description of $\mathcal{F}$ *entirely in the performance space* in analogy to (2.14):

$$\mathcal{F} = \{\mathbf{f} \mid \mathbf{k}(\mathbf{f}) \geq \mathbf{0}\}\,. \tag{4.1}$$

Since the nonlinear maps $\mathbf{f}(\mathbf{p})$ and $\mathbf{c}(\mathbf{p})$ can only be evaluated pointwise, there is no straight way to map $\mathcal{P}$ onto $\mathcal{F}$ in its entirety. Combining (2.15) and (4.1), the task at hand can formally be described as follows: Find a vector inequality $\mathbf{k}(\mathbf{f})$ such that

$$\underbrace{\mathbf{c}(\mathbf{p}) \geq \mathbf{0}}_{\substack{\text{(I)} \\ \text{feasible} \\ \text{parameter space} \\ \mathcal{P}}} \quad \wedge \quad \underbrace{\mathbf{f} = \mathbf{f}(\mathbf{p})}_{\substack{\text{(II)} \\ \text{map:} \\ \text{parameter to} \\ \text{performance space} \\ \mathbf{p} \mapsto \mathbf{f}}} \quad \Leftrightarrow \quad \underbrace{\mathbf{k}(\mathbf{f}) \geq \mathbf{0}}_{\substack{\text{(III)} \\ \text{feasible} \\ \text{performance space} \\ \mathcal{F}}} \quad . \tag{4.2}$$

In most practical cases, the accurate calculation of a nonlinear description of $\mathcal{F}$ according to (4.2)/(III) is not feasible. Using a novel Fourier-Motzkin-based technique, however, it is possible to calculate a linear approximation to $\mathcal{F}$ efficiently.

4.1 Linearized Performance Space Description

The new linear performance space exploration technique comprises of three main steps: Based on a linearized representation of the feasible parameter space (4.2)/(I) and of the map from the parameter to the performance space (4.2)/(II), a linearized description of the feasible performance space (4.2)/(III) can be calculated. These three steps will be discussed elaborately in the following sections.

With these computationally inexpensive linearized approximations, good results were achieved in hierarchical sizing, cf. Section 5.4. If more accurate performance space descriptions are required, they can be computed nonlinearly according to Chapter 3 if the computational resources allow it.

4.1.1 Linearized Feasible Parameter Space

In the neighborhood of a point $\mathbf{p}^{(0)}$ in the parameter space, the constraint function $\mathbf{c}(\mathbf{p})$ can be approximated by a linear Taylor expansion:

$$\mathbf{c}(\mathbf{p}) \approx \mathbf{c}(\mathbf{p}^{(0)}) + \left.\frac{\partial \mathbf{c}(\mathbf{p})}{\partial \mathbf{p}}\right|_{\mathbf{p}^{(0)}} \cdot \Delta\mathbf{p}\,, \quad \Delta\mathbf{p} = \mathbf{p} - \mathbf{p}^{(0)}\,. \tag{4.3}$$

This linear approach is justified because experience shows that $\mathcal{P}$ is usually comparatively small, cf. Appendix A.1.6.1 and [SPS+03, DSV04].

Of course, the choice of $\mathbf{p}^{(0)}$ in (4.3) is critical. If an approximation to the entire feasible performance space is sought, then the sizing constraints should be satisfied with maximum safety margins in the linearization point. Geometrically, this means that $\mathbf{p}^{(0)}$ is located near the center of $\mathcal{P}$, where both the sizing constraints and the circuit performances are usually only weakly nonlinear [GZEA01]. The algorithm presented in Appendix A yields a linearization point which meets these demands. This results in a good linearization accuracy as exemplified in Section A.2.3.

The combination of (4.2)/(I) and (4.3) yields

$$\underbrace{\left.\frac{\partial \mathbf{c}(\mathbf{p})}{\partial \mathbf{p}}\right|_{\mathbf{p}^{(0)}}}_{\mathbf{C}} \cdot \Delta \mathbf{p} \geq \underbrace{-\,\mathbf{c}(\mathbf{p}^{(0)})}_{\mathbf{c}} \,. \tag{4.4}$$

The Jacobian matrix **C** describes the linearized behavior of the sizing constraints with respect to the circuit parameters. Depending on the available circuit simulator, this can be simulated directly or by finite differences.

Based on (4.4), the linearized feasible parameter space $\overline{\mathcal{P}}$ is defined by

$$\overline{\mathcal{P}} = \{\mathbf{p}^{(0)} + \Delta\mathbf{p} \mid \mathbf{C} \cdot \Delta\mathbf{p} \geq \mathbf{c}\} \,. \tag{4.5}$$

Geometrically, $\overline{\mathcal{P}}$ represents a polytope in the parameter space [Zie95].

4.1.2 Linearized Map from Parameter to Performance Space

The performance function $\mathbf{f}(\mathbf{p})$ can be treated similarly to the constraints in the previous section:

$$\mathbf{f}(\mathbf{p}) \approx \mathbf{f}(\mathbf{p}^{(0)}) + \left.\frac{\partial \mathbf{f}(\mathbf{p})}{\partial \mathbf{p}}\right|_{\mathbf{p}^{(0)}} \cdot \Delta\mathbf{p} \,. \tag{4.6}$$

The combination of (4.2)/(II) and (4.6) results in the wanted linearization

$$\underbrace{\left.\frac{\partial \mathbf{f}_{\mathbf{P}}(\mathbf{p})}{\partial \mathbf{p}}\right|_{\mathbf{p}^{(0)}}}_{\mathbf{F}} \cdot \Delta \mathbf{p} = \underbrace{\mathbf{f} - \mathbf{f}(\mathbf{p}^{(0)})}_{\Delta\mathbf{f}} \,. \tag{4.7}$$

In (4.7), the vector difference $\Delta\mathbf{f}$ can be seen as variations of the circuit performances, and the Jacobian matrix **F** describes the linearized behavior of the performances **f** with respect to the parameters **p**.

4.1.3 Linearized Feasible Performance Space

Based on the results from the previous sections, namely the linearized feasible parameter space (4.4) and the linearized map from parameter to performance space (4.7), a linear approximation to the feasible performance space is sought, cf. (4.2):

$$\underbrace{\mathbf{C} \cdot \Delta\mathbf{p} \geq \mathbf{c}}_{\substack{(4.4)\\ \text{linearized feasible}\\ \text{parameter space}\\ \overline{\mathcal{P}}}} \quad \wedge \quad \underbrace{\mathbf{F} \cdot \Delta\mathbf{p} = \Delta\mathbf{f}}_{\substack{(4.7)\\ \text{linearized map:}\\ \text{parameter variations}\\ \text{to performance variations}\\ \Delta\mathbf{p} \mapsto \Delta\mathbf{f}}} \quad \Leftrightarrow \quad \underbrace{\mathbf{K} \cdot \Delta\mathbf{f} \geq \mathbf{k}}_{\substack{(4.8)\\ \text{linearized feasible}\\ \text{performance space}\\ \overline{\mathcal{F}}}} \quad .$$

If $\mathbf{F}$ is nonsingular, then the parameter vector $\Delta\mathbf{p}$ in (4.7) can formally be obtained by matrix inversion:

$$\Delta\mathbf{p} = \mathbf{F}^{-1} \cdot \Delta\mathbf{f}\,. \tag{4.9}$$

Using this result in (4.4) yields

$$\underbrace{\mathbf{C} \cdot \mathbf{F}^{-1}}_{\mathbf{K}} \cdot \Delta\mathbf{f} \geq \underbrace{\mathbf{c}}_{\mathbf{k}} \quad . \tag{4.10}$$

Accordingly, the linearized feasible performance space $\overline{\mathcal{F}}$ is given by

$$\overline{\mathcal{F}} = \{\mathbf{f}^{(0)} + \Delta\mathbf{f} \mid \mathbf{K} \cdot \Delta\mathbf{f} \geq \mathbf{k}\}\,. \tag{4.11}$$

In the general case, $\mathbf{F}$ is not invertible. For this situation, a two-step method was developed to calculate (4.8), as described in the following two subsections.

4.1.3.1 Equation-Based Parameter Substitution

For the following derivations it is assumed that the parameters outnumber the performances, i.e. $m > n$, and that $\mathbf{F}$ has full rank. These assumptions do not constitute significant restrictions because for almost all practical circuit design problems there are several degrees of freedom, which means that $m > n$. Moreover, if $\mathbf{F}$ does not have full rank, then the algorithm described here can be applied to its range space.

Given these assumptions, there are permutations of the columns of $\mathbf{F}$ which allow a partitioning of this matrix into a regular quadratic part $\mathbf{F}_\Box$ and a remainder $\mathbf{F}_\Diamond$,

$$\mathbf{F}^{\leftrightarrows} = [\mathbf{F}_\Box \quad \mathbf{F}_\Diamond]\,, \quad \mathbf{F}_\Box \in \mathbb{R}^n \times \mathbb{R}^n\,, \quad \mathbf{F}_\Diamond \in \mathbb{R}^n \times \mathbb{R}^{m-n}\,, \tag{4.12}$$

Applying the same permutation to the parameter vector, (4.7) can be rewritten as

$$[\mathbf{F}_\Box \quad \mathbf{F}_\Diamond] \cdot \begin{bmatrix} \Delta\mathbf{p}_\Box \\ \Delta\mathbf{p}_\Diamond \end{bmatrix} = \Delta\mathbf{f}\,. \tag{4.13}$$

Then, $\Delta\mathbf{p}_\Box$ can formally be obtained from

$$\Delta\mathbf{p}_\Box = \mathbf{F}_\Box^{-1} \cdot \Delta\mathbf{f} \; - \; \mathbf{F}_\Box^{-1}\mathbf{F}_\Diamond \cdot \Delta\mathbf{p}_\Diamond \, . \tag{4.14}$$

In analogy to (4.13), (4.4) corresponds to

$$[\mathbf{C}_\Box \quad \mathbf{C}_\Diamond] \cdot \begin{bmatrix} \Delta\mathbf{p}_\Box \\ \Delta\mathbf{p}_\Diamond \end{bmatrix} \geq \mathbf{c} \, . \tag{4.15}$$

The substitution of $\Delta\mathbf{p}_\Box$ in (4.15) by the expression found in (4.14) yields

$$[\mathbf{C}_\Box \quad \mathbf{C}_\Diamond] \begin{bmatrix} \mathbf{F}_\Box^{-1} & -\mathbf{F}_\Box^{-1}\mathbf{F}_\Diamond \\ \mathbf{0} & \mathbf{I} \end{bmatrix} \begin{bmatrix} \Delta\mathbf{f} \\ \Delta\mathbf{p}_\Diamond \end{bmatrix} \geq \mathbf{c} \tag{4.16}$$

$$\Leftrightarrow \quad [\underbrace{\mathbf{C}_\Box \, \mathbf{F}_\Box^{-1}}_{\mathbf{X}} \quad -\underbrace{\mathbf{C}_\Box \, \mathbf{F}_\Box^{-1}}_{\mathbf{X}}\mathbf{F}_\Diamond + \mathbf{C}_\Diamond] \begin{bmatrix} \Delta\mathbf{f} \\ \Delta\mathbf{p}_\Diamond \end{bmatrix} \geq \mathbf{c} \tag{4.17}$$

In (4.16), $\mathbf{I}$ is an identity matrix of dimension $(m-n) \times (m-n)$.

Instead of determining $\mathbf{F}_\Box^{-1}$ explicitly, the expression $\mathbf{X} = \mathbf{C}_\Box \; \mathbf{F}_\Box^{-1}$ in (4.17) can be obtained from solving

$$\mathbf{X} \, \mathbf{F}_\Box = \mathbf{C}_\Box \tag{4.18}$$

for $\mathbf{X}$ by Gaussian elimination or any other method to solve a system of linear equations. It is obvious that for this operation a good partitioning of $\mathbf{F}$ is essential. In this work, those n columns from $\mathbf{F}$ are combined in $\mathbf{F}_\Box$, which yield the best numerical condition. While for the solution of (4.14) this is the optimum strategy, this approach does not consider the numerical impact of the chosen partitioning on the subsequent manipulation steps, cf. Section 4.1.3.2. The development of a globally advantageous partitioning requires additional research.

Note that in (4.16), the performances $\Delta\mathbf{f}$ replaced an equal number of parameters $\Delta\mathbf{p}_\Box$. Geometrically, the feasible parameter polytope was mapped from the $\Delta\mathbf{p}$ space into the $\Delta\mathbf{f}/\Delta\mathbf{p}_\Diamond$ space. To obtain the feasible performance polytope in the $\Delta\mathbf{f}$ space according to (4.8), the remaining parameters $\Delta\mathbf{p}_\Diamond$ have to be eliminated as well. Geometrically, this corresponds to an orthogonal projection along the coordinate axes of the remaining parameters $\Delta\mathbf{p}_\Diamond$.

4.1.3.2 Inequality-Based Parameter Elimination

In a second step, the remaining $(m-n)$ parameters $\Delta\mathbf{p}_\Diamond$ have to be eliminated from (4.16). This can be done by *Fourier-Motzkin elimination* [Dan63, DE73], which is described in detail in Section 4.2. This manipulation transforms (4.17) into

$$\mathbf{K} \cdot \Delta\mathbf{f} \geq \mathbf{k} \, , \tag{4.8}$$

which is the sought linearized description of the feasible performance space.

Although Fourier-Motzkin elimination has been known for a long time, it has only recently gained increased attention in the area of combinatorial optimization and compiler construction [AI91, CR96, JLF98, CL]. Since the resulting algorithms are geared toward discrete problems, they implement integer or rational arithmetic. Therefore, they are not suitable for multi-dimensional performance space exploration based on floating-point simulation data. For this reason, a dedicated Fourier-Motzkin elimination algorithm was developed, which meets the demands of a high-dimensional performance space exploration based on floating-point simulation data.

4.2 Fourier-Motzkin Elimination

As described above, equation-based parameter substitution and parameter elimination according to Fourier and Motzkin are used to map the linearized approximation to the feasible parameter space into the performance space. Since Fourier-Motzkin elimination constitutes the core of the new performance space exploration algorithm, it is discussed in detail in this section. After a general overview, an example illustrates the method, whereupon practical implementation aspects follow.

4.2.1 Basic Algorithm

The elimination of a variable x_r, $1 \leq r \leq N$, from a system of linear inequalities

$$\mathbf{A}\mathbf{x} \geq \mathbf{b}\,, \quad \mathbf{A} = \begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ \mathbf{a}_M^T \end{bmatrix} = [a_{ij}]\,, \quad \mathbf{a}_i^T = [a_{i1} \ldots a_{iN}]\,,$$
$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_N \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ \vdots \\ b_M \end{bmatrix}, \qquad M, N \geq 2\,, \tag{4.19}$$

corresponds to the calculation of a new vector inequality with $a_{ir} = 0$ for $1 \leq i \leq M$. To this end, Fourier-Motzkin elimination exploits two properties of inequalities:

$$s \geq t \ \wedge \ y \geq z \ \Rightarrow s + y \geq t + z\,, \tag{4.20}$$
$$s \geq t \ \wedge \ y \geq 0 \ \Rightarrow \ s \cdot y \geq t \cdot y\,. \tag{4.21}$$

The elimination of a variable x_r from (4.19) comprises of two steps:

1. Sorting of inequalities
   The individual inequalities $\mathbf{a}_i^T\mathbf{x} \geq b_i$ from (4.19) are partitioned into three sets:

$$\begin{array}{cl} (I) & I_{>} = \{\mathbf{a}_g^T\mathbf{x} \geq b_g \mid a_{gr} > 0 \wedge g \in G\} \\ (II) & I_{<} = \{\mathbf{a}_l^T\mathbf{x} \geq b_l \mid a_{lr} < 0 \wedge \ l \in L\} \\ (III) & I_{=} = \{\mathbf{a}_e^T\mathbf{x} \geq b_e \mid a_{er} = 0 \wedge \ e \in E\} \end{array} \tag{4.22}$$
$$\text{with} \qquad G \cap L = G \cap E = L \cap E = \emptyset \quad \wedge \quad G \cup L \cup E = \{1 \ldots M\} \quad .$$

2. Elimination of x_r by linear combination
   Note that the inequalities in $I_=$ are already independent of x_r since $a_{er}=0$. They can be included in the final set of inequalities without any further manipulation. Geometrically, the elimination of a variable x_r can be interpreted as a projection of the original polytope along the x_r axis. An inequality with $a_{er}=0$ describes a halfspace, the boundary hyperplane of which is parallel to the x_r axis. The (N-1)-dimensional image of this hyperplane is equivalent to its intersection with the subspace supported by the remaining parameters. This is depicted in Figure 4.1 for the three-dimensional case, where the shaded plane is projected onto the x_a/x_b plane.

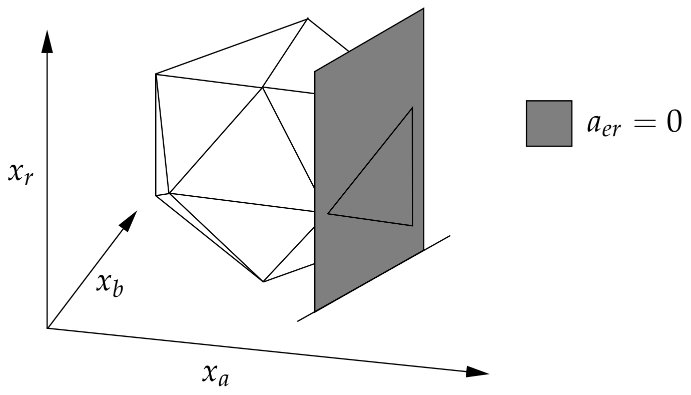


Figure 4.1: Projection of a Parallel Boundary Plane

The set $I_=$ is then augmented by all pairwise linear combinations of inequalities from $I_>$ and $I_<$ such that the r^{th} coefficient is zero:

$$I_= \cup \{(a_{gr}\cdot \mathbf{a}_l^T - a_{lr}\cdot \mathbf{a}_g^T)\cdot \mathbf{x} \geq a_{gr}b_l - a_{lr}b_g \mid g \in G \wedge l \in L\}\,. \tag{4.23}$$

Figure 4.2 shows how such a linear combination results in a zero coefficient of x_r and hence yields a plane which is parallel to the x_r axis.

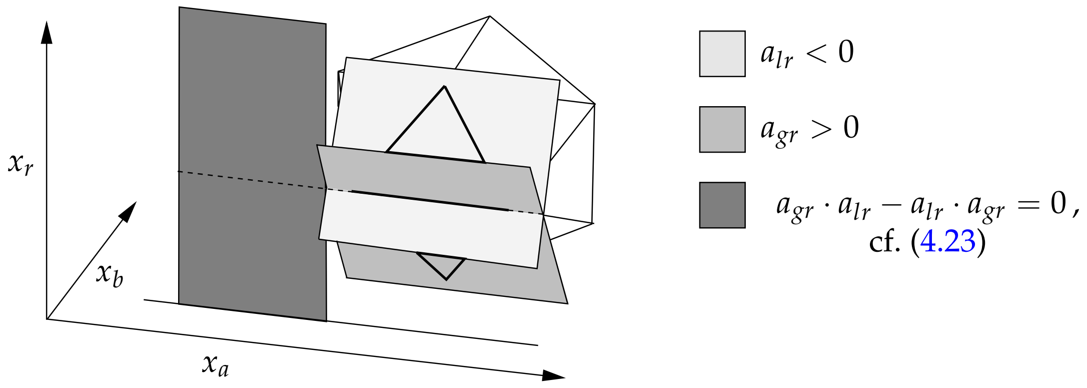


Figure 4.2: Projection of a Linear Combination of Planes

The final set of inequalities can be written in vector/matrix notation as

$$\mathbf{A}^{\backslash r}\cdot \mathbf{x} \geq \mathbf{b}^{\backslash r}\,. \tag{4.24}$$

The entries in the r^{th} column of $\mathbf{A}^{\backslash r}$ are all zeros. Hence, the variable x_r has effectively been removed from the system of inequalities.

4.2.2 Example

To demonstrate the Fourier-Motzkin algorithm, a hypothetic example with three circuit parameters, p_1 to p_3, and one circuit performance, $f_1 = f_1(p_1, p_2, p_3)$ is considered. Let the feasible parameter space be linearized at a suitable operating point according to (4.4) and the circuit performance according to (4.7). Assume that Δp_1 has been replaced by Δf_1 with equation-based parameter substitution:

$$\begin{bmatrix} 5 & 1 & 7 \\ 1 & 1 & -1 \\ 5 & -1 & -10 \\ 0 & -3 & -2 \\ -7 & 0 & -2 \end{bmatrix} \cdot \begin{bmatrix} \Delta f_1 \\ \Delta p_2 \\ \Delta p_3 \end{bmatrix} \geq \begin{bmatrix} 63 \\ 1 \\ -75 \\ -74 \\ -43 \end{bmatrix} . \tag{4.25}$$

The value range of Δf_1 is sought, with the remaining circuit parameters varying within the allowed bounds. Consequently, Δp_2 and Δp_3 have to be eliminated. Let Δp_2 be removed first. To this end, the inequalities are sorted according to (4.22):

$$\begin{array}{c} (I) \\ \\ (II) \\ \\ (III) \end{array} \left[\begin{array}{rrr||r} 5 & 1 & 7 & 63 \\ 1 & 1 & -1 & 1 \\ \hline 5 & -1 & -10 & -75 \\ 0 & -3 & -2 & -74 \\ \hline -7 & 0 & -2 & -43 \end{array}\right] \begin{array}{l} (a) \\ (b) \\ (c) \\ (d) \\ (e) \, . \end{array} \tag{4.26}$$

For ease of notation, the resulting system of inequalities is written as an extended matrix with a double line separating the left-hand from the right-hand side. The horizontal lines separate the three sets of inequalities. The individual original inequalities are marked by lowercase letters for further reference.

Four new inequalities according to (4.23) can be created. Consequently, inequalities (a) through (d) are replaced by their linear combinations (a, d), (a, c), (b, c), and (b, d):

$$\begin{array}{c} (I) \\ (II) \\ \\ \\ \\ \end{array} \left[\begin{array}{rrr||r} 15 & 0 & 19 & 115 \\ \hline 10 & 0 & -3 & -12 \\ 6 & 0 & -11 & -74 \\ 3 & 0 & -5 & -71 \\ -7 & 0 & -2 & -43 \end{array}\right] \begin{array}{l} (a,d) \\ (a,c) \\ (b,c) \\ (b,d) \\ (e) \quad . \end{array} \tag{4.27}$$

In this new matrix, there are one positive, four negative and no zero coefficients of Δp_3. Accordingly, the final elimination step yields a system of four inequalities:

$$\begin{array}{c} (I) \\ \\ (II) \\ \\ \end{array} \left[\begin{array}{rrr||r} -103 & 0 & 0 & -587 \\ \hline 235 & 0 & 0 & 117 \\ 279 & 0 & 0 & -141 \\ 132 & 0 & 0 & -774 \end{array}\right] \begin{array}{l} (a,d,e) \\ (a,c,d) \\ (a,b,c,d) \\ (a,b,d) \, . \end{array} \tag{4.28}$$

The inequality in the second line, for example, was derived from (4.27),(a, d) and (4.27),(a, c). Hence, it comprises of the original inequalities (a), (c), and (d). With (4.28), the elimination is finished and the resulting inequalities in Δf_1 are

$$\begin{cases} \Delta f_1 & \leq & 587/103 \approx & 5.699 & (a, d, e) \\ \Delta f_1 & \geq & 117/235 \approx & 0.498 & (a, c, d) \\ \Delta f_1 & \geq & -141/279 \approx & -0.505 & (a, b, c, d) \\ \Delta f_1 & \geq & -774/132 \approx & -5.864 & (a, b, d)\,. \end{cases} \tag{4.29}$$

Obviously, the feasible circuit performance range is only limited by inequalities (a, d, e) and (a, c, d):

$$0.498 \leq \Delta f_1 \leq 5.699\,. \tag{4.30}$$

Fourier-Motzkin elimination can be interpreted geometrically as an orthogonal projection along the coordinate axes as depicted in Figure 4.3: The original vector inequality (4.26) describes a polytope in the 3-dimensional space. The elimination of Δp_2 yields a polygon in the $\Delta p_3/\Delta f_1$ space. The final removal of Δp_3 leads to a line indicating the value range of Δf_1.

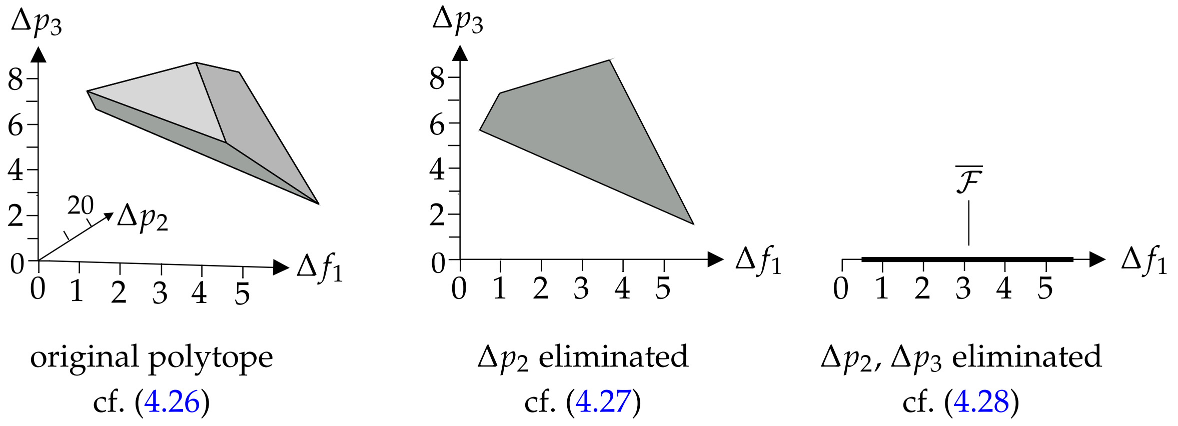


Figure 4.3: Fourier-Motzkin Elimination as Orthogonal Projection

4.2.3 Redundancy Detection

The example above was deliberately simple. Therefore, it does not immediately reveal complexity as the severest challenge of the Fourier-Motzkin elimination. It turned out in (4.29) that two inequalities were redundant. In fact, especially in high-dimensional spaces, and after several elimination steps, a huge number of inequalities can exhaust computing resources easily if no special care is taken. In Figure 4.4, the two-dimensional projection of a dodecahedron is shown, which was originally described by twelve inequalities. Note how many redundant inequalities occur.

The elimination order severely influences the generation of redundant inequalities [ABS97]. Obviously, the most efficient way to cope with redundant inequalities is

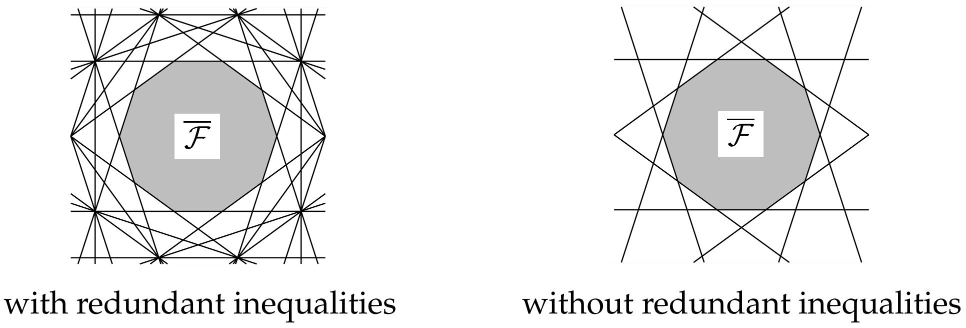


Figure 4.4: Generation of Redundant Inequalities

their avoidance in the first place. The number M of inequalities after an elimination step can be calculated from the powers of the three inequality sets before the elimination:

$$M = |I_>| \cdot |I_<| + |I_=| \,. \tag{4.31}$$

A locally optimal choice of the next parameter to be eliminated can be made by determining the inequality sets from (4.22) and calculating the associated number of inequalities according to (4.31). Finding a globally optimal elimination order, however, is still an unsolved problem [ABS97].

Let M_i be the number of inequalities before elimination step i. In the worst case, an even number of inequalities splits up equally between $I_>$ and $I_<$ resulting in

$$M_{i+1} = \left(\frac{M_i}{2}\right)^2 . \tag{4.32}$$

Practical experience shows that typical application data comes close to this worst-case scenario. Therefore, it is mandatory to exhaustively detect redundancies after each elimination step. For this purpose, an efficient two-layered redundancy filter was developed.

4.2.3.1 Fast Structural Redundancy Detection

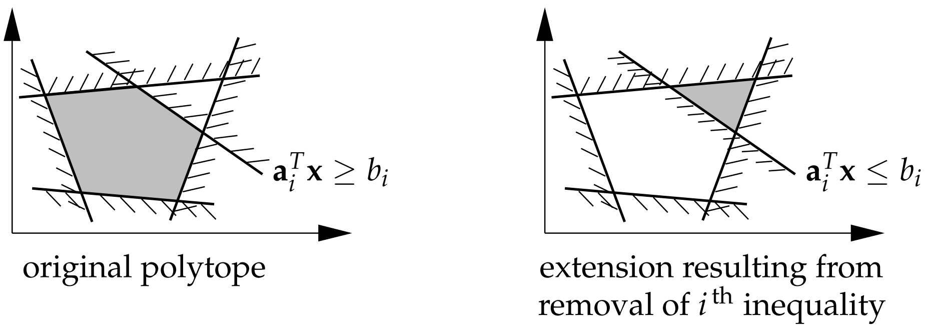
An inequality is redundant if and only if it can be written as a positive combination of other inequalities in the system [Zie95]. From this rule, Chernikov derived a criterion, which allows a fast and easy identification of redundant inequalities [Che71, Zie95]:

An inequality is redundant after step i if it is based on more than $i+1$ original inequalities.

In (4.28), this criterion would have identified the third inequality, (a, b, c, d), as redundant: After the second elimination step, it is based on more than three original inequalities. The redundancy of the fourth inequality in (4.28), (a, b, d), would not have been detected, though. The reason is that this criterion only identifies *structural* redundancy and does not consider the numerical properties. In spite of this drawback, the Chernikov criterion has its value as a computationally cheap method to expunge obvious redundancies.

4.2.3.2 Exhaustive Numerical Redundancy Detection

Practical experience shows that an additional test identifying *all* redundancies is required to keep the number of inequalities within practical limits. After the first filtering step using Chernikov's criterion, each of the remaining inequalities is examined *numerically* using linear programming [GLP].



$\mathbf{a}_i^T\mathbf{x} \geq b_i$
$\mathbf{a}_i^T\mathbf{x} \leq b_i$
original polytope
extension resulting from removal of i th inequality

Figure 4.5: Numerical Detection of Redundant Inequalities

In a system of inequalities, the i th inequality $\mathbf{a}_i^T\mathbf{x} \geq b_i$ is irredundant if and only if its removal extends the feasible space. As illustrated in Figure 4.5, this extension is described by the original system of inequalities with the i th inequality replaced by its inverse, $\mathbf{a}_i^T\mathbf{x} \leq b_i$ [BW94]. Hence, inequality i is irredundant if and only if

$$\begin{bmatrix} \mathbf{a}_1^T \\ \vdots \\ -\mathbf{a}_i^T \\ \vdots \\ \mathbf{a}_N^T \end{bmatrix} \mathbf{x} \geq \begin{bmatrix} b_1 \\ \vdots \\ -b_i \\ \vdots \\ b_N \end{bmatrix} \tag{4.33}$$

has feasible solutions. This can be checked efficiently using the first phase of the simplex algorithm. It turns out that often more than 90% of the CPU time of the entire performance space exploration algorithm is required by this second filtering step. Yet, it is this very step that keeps the overall resource requirements in practical limits, cf. Section 5.3.2.2. In this way, the Fourier-Motzkin elimination method can be used for real-world circuits in spite of its non-polynomial complexity [Sch98].

4.3 Comparison to Alternative Approaches

Geometrically, the main task of the exploration algorithm presented in this chapter is to determine the image of a polytope under a projective linear map. While this approach is new within the area of performance space exploration, there are publications dealing with similar geometric problems, albeit from different points of view. Two of them are outlined in the following sections. They are discussed in more detail in Appendix B.

4.3.1 Geometric Approach: Feasible Performance Space as an Image of a Hypercube under a Linear Map

In the fabrication process of analog circuits, the circuit parameters are subject to random variations. If the parameters are statistically independent and upper and lower deviation limits are known, then the parameter range can be described by means of a hyperbox. If a linear relation of parameters and performances is assumed, then the resulting range of performance values is characterized by a polytope. Geared toward analog test, [MV89] describes an algorithm to determine such a performance polytope. Even though analog test is quite different from performance space exploration, with a mere change of terminology, the method could be applied to performance space exploration as well.

When parameters vary independently within known bounds, then their values can be described by a hyperbox. With a suitable normalization, this hyperbox can be transformed into a hypercube which is centered within the coordinate system:

$$\Delta\overline{\mathcal{P}} = \{\Delta\mathbf{p} \mid \underset{i\in\{1\ldots m\}}{\forall} -1 \leq \Delta p_i \leq 1\}\,. \tag{4.34}$$

Applied to $\Delta\overline{\mathcal{P}}$, a linear map

$$\Delta\mathbf{f} = \mathbf{F}\cdot\Delta\mathbf{p}\,. \tag{4.35}$$

yields a centro-symmetric polytope

$$\Delta\overline{\mathcal{F}} = \{\Delta\mathbf{f} \mid \mathbf{K}\cdot\Delta\mathbf{f} \geq \mathbf{k}\}\,. \tag{4.36}$$

The key idea of the algorithm is to interpret the columns of $\mathbf{F}$ as a basis of the performance space $\Delta\overline{\mathcal{F}}$. The method exploits the fact that according to (4.34) the parameter vectors are mutually independent, and the coordinates of the vertices consist of the numbers 1 and -1 only. Based on geometric reasoning, the boundary hyperplanes of $\Delta\overline{\mathcal{F}}$ can be derived from sums and differences of the columns of $\mathbf{F}$. More details are provided in Appendix B.1.

Comparing (4.34), (4.35) and (4.36) to (4.4), (4.7) and (4.8), it turns out that this approach is less general than the new Fourier-Motzkin-based technique: It does not allow linearized sizing constraints in their general form, where boundary hyperplanes are not parallel to the coordinate axes. Unfortunately, the algorithm cannot be adapted for this case since it relies on the special properties of $\Delta\overline{\mathcal{P}}$ according to (4.34).

4.3.2 Algebraic Approach: Feasible Performance Space as Solution Space of a System of Linear Equations

Chernikov [Che71] developed an algorithm to determine all nonnegative solutions of an underdetermined homogeneous system of linear equations:

$$\mathbf{A}\cdot\mathbf{x} = \mathbf{0}\,,\quad \mathbf{x} \geq \mathbf{0}\,,\quad \mathbf{A}\in\mathbb{R}^{(n\times m)}\,,\quad n < m\,. \tag{4.37}$$

In [Lee90], linear symbolic equations are used to describe the behavior of small circuits. By elementary algebraic transformations, variable substitutions and the introduction of slack variables (cf. Section B.2), the circuit equations can be written according to (4.37). Depending on the actual circuit, the vector $\mathbf{x}$ can comprise of branch currents, node voltages, power dissipations, impedance values, and so on.

In a simulation-based environment, a linearized circuit description according to (4.7) can also be written in the form of (4.37):

$$\mathbf{F} \cdot \Delta\mathbf{p} = \Delta\mathbf{f} \Leftrightarrow [\mathbf{F} \;\; -\mathbf{I}] \cdot \left[\begin{array}{c} \Delta\mathbf{p} \\ \Delta\mathbf{f} \end{array} \right] = \mathbf{0} \,. \tag{4.38}$$

Here, $\mathbf{I}$ is an $(n \times n)$ identity matrix. With (4.3), (4.4) and

$$\mathbf{c}(\mathbf{p}) - \mathbf{c}(\mathbf{p}^{(0)}) = \Delta\mathbf{c} \,, \tag{4.39}$$

sizing constraint variations can be related to parameter variations:

$$\mathbf{C} \cdot \Delta\mathbf{p} = \Delta\mathbf{c} \,. \tag{4.40}$$

Then, (4.38) can be extended by the linearized constraint functions:

$$\left[\begin{array}{ccc} \mathbf{F} & -\mathbf{I} & \mathbf{O} \\ \mathbf{C} & \mathbf{O}' & -\mathbf{I}' \end{array} \right] \cdot \left[\begin{array}{c} \Delta\mathbf{p} \\ \Delta\mathbf{f} \\ \Delta\mathbf{c} \end{array} \right] = \mathbf{0} \,. \tag{4.41}$$

In (4.41), $\mathbf{I}$ and $\mathbf{I}'$ are identity matrices of appropriate dimensions, whereas $\mathbf{O}$ and $\mathbf{O}'$ are null matrices of correct size.

With variable substitutions and the introduction of slack variables according to [Lee90], lower and upper bounds can be imposed on the unknowns:

$$\Delta\mathbf{p}_{\text{min}} \leq \Delta\mathbf{p} \leq \Delta\mathbf{p}_{\text{max}} \tag{4.42}$$

$$\Delta\mathbf{f}_{\text{min}} \leq \Delta\mathbf{f} \leq \Delta\mathbf{f}_{\text{max}} \tag{4.43}$$

$$\Delta\mathbf{c}_{\text{min}} \leq \Delta\mathbf{c} \leq \Delta\mathbf{c}_{\text{max}} \,. \tag{4.44}$$

Owing to the available degrees of freedom in the circuit sizing process, (4.41) is satisfied by an entire solution space. In the case of electronic circuits, the parameter, performance, and constraint values are bounded for physical reasons. Therefore, and due to the linearity of (4.41)–(4.44), the entirety of all solutions constitutes a polytope. It can be expressed as the convex hull of its vertices $\mathbf{v}_i$, which may be written as the columns of a matrix $\mathbf{V}$:

$$\left[\begin{array}{c} \Delta\mathbf{p} \\ \Delta\mathbf{f} \\ \Delta\mathbf{c} \end{array} \right] = \mathbf{V} \cdot \mathbf{u} \,, \quad \mathbf{V} \in \mathbb{R}^{(m+n+q) \times s} \,, \quad \mathbf{u} \geq \mathbf{0} \,, \quad \sum_{i \in \{1 \ldots s\}} u_i = 1 \,. \tag{4.45}$$

Here, s is the number of vertices of the solution polytope. Chernikov's algorithm [Che71] can be used to determine (4.45) [Lee90].

This approach can be used for linearized performance space exploration. Usually, bounds are given for all parameters according to (4.42). Due to (4.2)/(I), the sizing constraints only have a lower bound. With (4.4) and (4.40), the linearized sizing constraints can be written as

$$\mathbf{c} \leq \Delta\mathbf{c}\,. \tag{4.46}$$

This is in line with (4.44) since the specification of bounds is optional and no upper bounds need to be given. The resulting linearized feasible performance space can be obtained from those n equations of (4.45) which refer to $\Delta\mathbf{f}$:

$$\overline{\mathcal{F}} = \{\mathbf{f}_0 + \Delta\mathbf{f} \mid \Delta\mathbf{f} = \widetilde{\mathbf{V}} \cdot \mathbf{u}\}\,,$$

$$\widetilde{\mathbf{V}} = \begin{bmatrix} v_{(m+1),1} & \cdots & v_{(m+1),s} \\ \vdots & \vdots & \vdots \\ v_{(m+n),1} & \cdots & v_{(m+n),s} \end{bmatrix}, \quad \mathbf{u} \geq \mathbf{0}\,, \quad \sum_{i \in \{1 \ldots s\}} u_i = 1\,. \tag{4.47}$$

Note that even if an irredundant polytope description according to (4.45) is available, the selection of a subsystem of equations introduces redundancy. This is plausible even without a rigorous derivation: If, for example, only one performance f_i were of interest, its feasible range would be described by the convex hull of s points:

$$\Delta f_i = [v_{i,1} \ldots v_{i,s}] \cdot \mathbf{u}\,, \quad \mathbf{u} \geq \mathbf{0}\,, \quad \sum_{i \in \{1 \ldots s\}} u_i = 1\,. \tag{4.48}$$

Yet, for a definition of a scalar interval, a maximum number of two points is required. Hence, at least $(s-2)$ points in (4.48) must be redundant. It is shown in Appendix B.2 that Chernikov's algorithm is the dual of the Fourier-Motzkin elimination method. Consequently, it also tends to produce a large amount of redundancy even if no selection according to (4.47) is done: In (4.45), the matrix $\mathbf{V}$ does not only contain actual vertices of the solution polytope, but also numerous redundant points which are located inside the polytope. In [Lee90], the problem of redundancy was not discussed since only small circuits were considered.

So far, the discussion focused on how the algorithm from [Lee90] can be used for linear performance space exploration. Yet, this algorithm has no notion of bottom-up constraint propagation and top-down specification propagation. Instead, it *simultaneously* calculates the feasible spaces of parameters, performances, and constraints, cf. (4.45). For performance space exploration, only those n rows of $\mathbf{V}$ are exploited which refer to performances. Analogously, a top-down propagation of performance specifications can be done if only the first m equations of (4.45) are considered*. The feasible

* A linear top-down propagation of specifications is trivial if the resulting parameter space has to be represented by a system of inequalities. Let performance specifications be written as $\mathbf{f} \geq \mathbf{f}_{\text{spec}}$. Upper bounds can easily be accommodated by variable substitution. With (4.4) and (4.7), the feasible parameter space is

$$\begin{bmatrix} \mathbf{C} \\ \mathbf{F} \end{bmatrix} \cdot \Delta\mathbf{p} \geq \begin{bmatrix} \mathbf{c} \\ \mathbf{f}_{\text{spec}} - \mathbf{f}(\mathbf{p}^{(0)}) \end{bmatrix}. \tag{4.49}$$

space of the sizing constraints are usually of minor concern. Since Chernikov's algorithm raises similar complexity challenges as Fourier-Motzkin elimination, it does not appear favorable to solve the complex problem (4.41) and then only use part of the solution. This was of minor importance in [Lee90], where only small circuits were examined, but for linear performance space exploration with a large number of sizing constraints this should be avoided. In addition, Chernikov's algorithm requires non-negative variables. To accommodate general bounds according to (4.42)–(4.44), slack variables have to be defined (cf. Appendix B.2), which further enlarge the problem.

In (4.45), the feasible performance space is described by a convex hull of vertices. For a given performance vector $\mathbf{f}^*$, the check for containedness is not cheap in this case. In contrast, this check is trivial if a representation of $\overline{\mathcal{F}}$ as an intersection of halfspaces according to (4.8) is available: It amounts to a numeric evaluation of the corresponding system of inequalities and a subsequent check for negative components of the resulting vector. This is particularly beneficial, for example, in hierarchical sizing where the optimization engine has to check the feasibility of parameter vectors frequently.

4.4 Summary

This chapter described how a linear approximation to the feasible performance space of a circuit can be derived: A linear Taylor expansion of the sizing constraints yields an approximation to the feasible parameter space. By its operation, a circuit maps its parameters to its performances. When this map is linearized as well, the resulting linear description of the feasible performance space can be calculated. It was shown how equation-based parameter replacement and inequality-based parameter elimination yield the desired result. The corresponding elimination algorithm according to Fourier and Motzkin was discussed in detail. It was interpreted graphically and explained with the help of an elaborate numeric example. The generation of redundant inequalities was identified as the greatest challenge of this algorithm, whereupon an effective two-step redundancy removal strategy was presented. Finally, two algorithms from the literature were briefly reviewed, which could be adapted for linear performance space exploration. However, it should be noted that none of them was originally designed for this purpose.

Chapter 5

Experimental Results

After a thorough discussion of the theory behind the two new performance space exploration techniques, this chapter provides experimental results. They illustrate typical applications and provide additional information concerning the algorithmic behavior of the algorithms. Section 5.2 deals with nonlinear performance space exploration using Normal-Boundary Intersection. Linear performance space exploration based on Fourier-Motzkin elimination is demonstrated in Section 5.3, whereupon Section 5.4 suggests hierarchical optimization approach which incorporates the new linear exploration technique.

5.1 Sample Circuits

Most of the experimental results in Sections 5.2 and 5.3 were obtained from a folded cascode and a Miller compensated operational amplifier as depicted in Figure 5.1. The corresponding numbers of transistors, inequality sizing constraints and designable circuit parameters are given in Table 5.1.

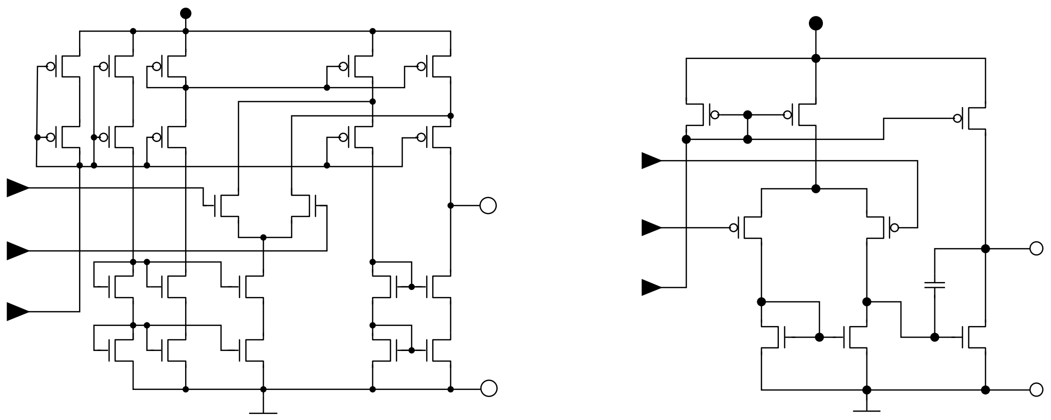


Figure 5.1: Folded Cascode (l) and Miller (r) Operational Amplifiers

Circuit	# Transistors	# Inequality Sizing Constraints	# Parameters
Folded Cascode	22	189	11
Miller	8	79	8

Table 5.1: Characteristics of Sample Circuits

5.2 Normal-Boundary Intersection

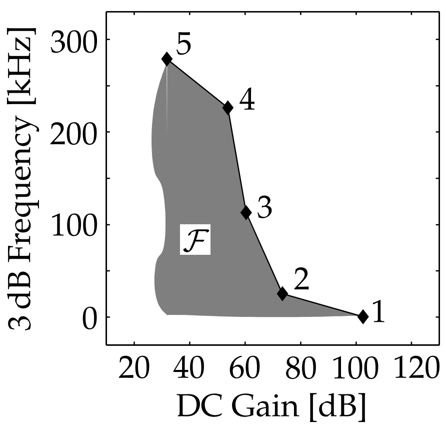
In this section, the major features of the NBI-based performance space exploration technique are highlighted: Section 5.2.1 shows how it can accurately identify the trade-offs of competing performances even in the nonconvex case. Section 5.2.2 shows how the sizing constraints which are active in the Pareto points can be used to identify those components of the circuit, which inhibit a further performance improvement. In addition, the stringency of the active constraints can also be specified quantitatively. Since Pareto fronts represent the ultimate performance capabilities of the underlying circuit topologies, these fronts can be used to compare different implementation alternatives as illustrated in Section 5.2.3. In a design process, this allows the selection of the topology which matches the given specifications best. In spite of the high accuracy of this nonlinear exploration method, its resource requirements are moderate as shown in Section 5.2.4.

5.2.1 Trade-off Analysis

For the folded cascode architecture from Figure 5.1, a few two-performance trade-offs are given below. While for the NBI-based exploration method only two or three performances can be selected for examination, constraints can be given for the remaining "invisible" ones. Here, a minimum 3 dB frequency of 500 Hz was specified in order to avoid pathologic circuit behavior. The other performances were left unconstrained.

In Figure 5.2, all the performances have to be maximized. So the NBI method identifies the upper right part of the boundary of the feasible performance space[*]. Therefore, the performance values have to be multiplied by –1 in order to obtain a minimization problem according to (3.9)–(3.16). In this sense, the points labeled 1 are the individual minima of the DC gain and of the gain margin procedurally (cf. (3.9)), while they actually indicate their maxima. The same applies to the 3 dB frequency and to the transit frequency in the points labeled 5. Three line searches in the performance space are performed according to (3.16) in order to obtain points 2 to 4. Linear interpolation yields an approximation to the sought Pareto curve. These two examples show that Pareto fronts by no means have to be convex. In such cases, traditional exploration methods would have run into trouble.

[*] The shaded areas merely symbolize the feasible performance space. Of course, the extension of $\mathcal{F}$ in the unexamined directions remains unknown.



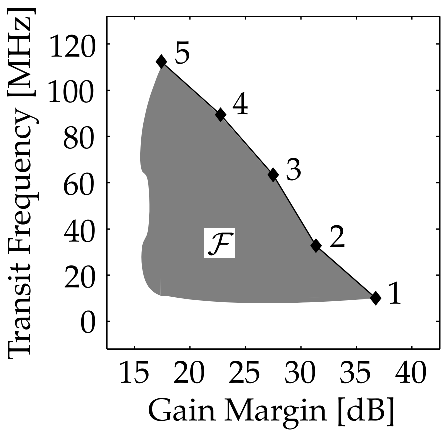
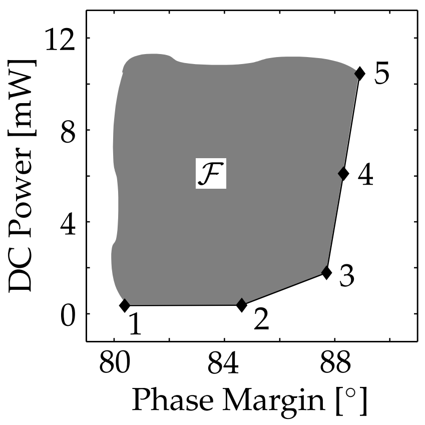


Figure 5.2: Nonconvex Pareto Curves

The left trade-off curve in Figure 5.2 features two parts with strong curvatures, sometimes referred to as *knees* [Das98]. In the mathematical sense, all efficient points are "equally optimal". Yet, if a high 3 dB frequency is a concern, then the DC gain should not exceed 55 dB. On the other hand, gain values beyond 75 dB can only be realized for very low 3 dB frequencies.
In contrast, the trade-off between gain margin and transit frequency in the right part of Figure 5.2 is almost linear, without any distinct region.



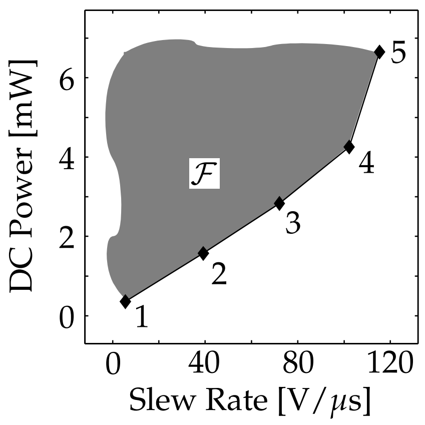


Figure 5.3: Convex Pareto Curves

Two trade-off curves involving the static DC power consumption are shown in Figure 5.3. Here, power has to be minimized, while phase margin and slew rate are subject to maximization. Both Pareto curves are convex with one distinct knee each.

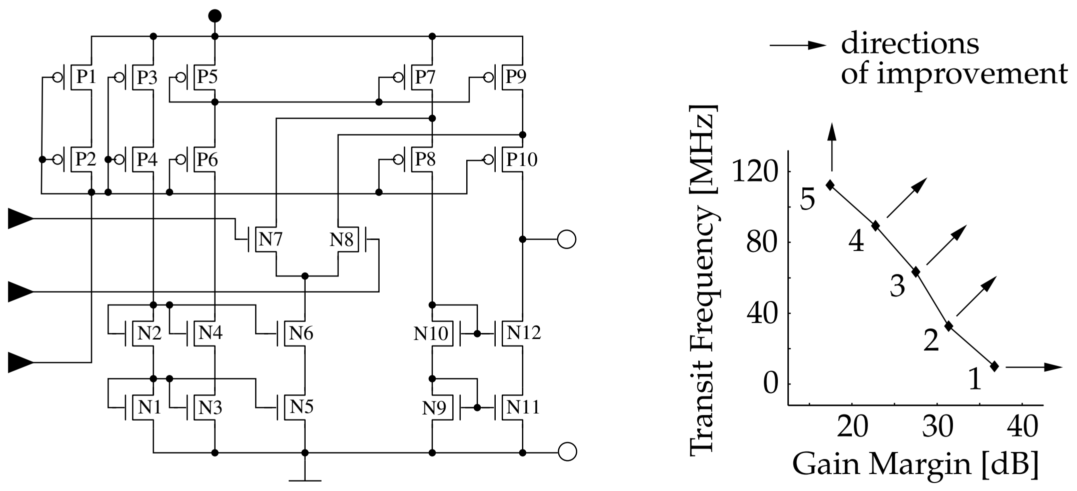
It is evident from Figure 5.3, left, that the folded cascode amplifier can achieve a maximum phase margin of roughly 88.5°. The trade-off curve shows that a phase margin up to about 85° can be achieved at a fairly low power consumption, while the "last few degrees" come at the price of prohibitively raising the energy dissipation. On the other hand, lowering the phase margin below 80° does not further reduce the power consumption.

The right Pareto curve in Figure 5.3 shows that for slew rates up to about 100 V/μs, one mW of DC power consumption buys about 25 V/μs of slew rate. For extreme slew rates, this rate deteriorates slightly.

These are only a few examples of how trade-off curves convey an insight to the behavior of a given circuit. In the actual design process, the designer can use this type of analysis to examine the capabilities of a newly designed circuit or to evaluate the suitability of an existing topology for a given set of specifications.

5.2.2 Examination of Limiting Sizing Constraints

In addition to a mere identification of the trade-off relationships as demonstrated above, the new NBI-based exploration technique can also identify the limiting sizing constraints and quantify their stringency. To demonstrate this feature, the trade-off between gain margin and transit frequency from Figure 5.2 was selected for an in-depth analysis. The corresponding Pareto curve and a labeled version of the amplifier schematic are combined in Figure 5.4 for further reference. Table 5.2 gives an overview of the active constraints and will now be discussed in detail.



Folded Cascode Amplifier

Gain Margin / Transit Frequency Trade-off

Figure 5.4: Amplifier Schematic and Corresponding Pareto Curve

Discussion of Table Structure

In the example presented here, a number of 12 sizing constraints turn out to have a limiting effect on the performances. They are labeled by roman numbers in the leftmost column. The second column lists the name of the constraints along with the

			Pareto Points				
			1	2	3	4	5
I	minimum length N2, 4, 6, 10, 12, P2, 4, 6, 8, 10	sens. GM $\left[\frac{\text{dB}}{\%}\right]$	0.0385	0.0586	0.0716	0.0763	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$	0.0000	0.3192	0.3896	0.4155	0.4853
II	minimum width N7, 8	sens. GM $\left[\frac{\text{dB}}{\%}\right]$	0.1102	0.0305			
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$	0.0000	0.1658			
III	minimum length N1, 3, 5, 7, 8, 9, 11, P1, 3, 5, 7, 9	sens. GM $\left[\frac{\text{dB}}{\%}\right]$		0.0174	0.0391	0.0600	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$		0.0949	0.2129	0.3269	0.8493
IV	maximum width N9, 10, 11, 12	sens. GM $\left[\frac{\text{dB}}{\%}\right]$	0.0269				
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$	0.0000				
V	maximum width P7, 8, 9, 10	sens. GM $\left[\frac{\text{dB}}{\%}\right]$		0.0098			
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$		0.0535			
VI	maximum width N5, 6	sens. GM $\left[\frac{\text{dB}}{\%}\right]$			0.0125	0.0262	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$			0.0683	0.1426	0.5600
VII	maximum width N7, 8	sens. GM $\left[\frac{\text{dB}}{\%}\right]$					0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$					0.4136
VIII	minimum drain-source overdrive P5	sens. GM $\left[\frac{\text{dB}}{\%}\right]$	0.0006	0.0033	0.0032	0.0034	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$	0.0000	0.0179	0.0219	0.0185	0.0160
IX	transistor in strong inversion N8	sens. GM $\left[\frac{\text{dB}}{\text{mV}}\right]$	0.1023				
		sens. Ft $\left[\frac{\text{MHz}}{\text{mV}}\right]$	0.0000				
X	minimum gate-source overdrive N12	sens. GM $\left[\frac{\text{dB}}{\%}\right]$	0.0077				
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$	0.0000				
XI	minimum drain-source overdrive P6	sens. GM $\left[\frac{\text{dB}}{\%}\right]$		0.0027	0.0030	0.0033	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$		0.0146	0.0164	0.0177	0.0171
XII	minimum drain-source overdrive N6	sens. GM $\left[\frac{\text{dB}}{\%}\right]$			0.0020	0.0034	0.0000
		sens. Ft $\left[\frac{\text{MHz}}{\%}\right]$			0.0109	0.0186	0.0684

Table 5.2: Active Sizing Constraints

transistors they refer to. The different types of active constraints are separated by a double line: I to III are lower parameter bounds, IV to VII constitute upper parameter bounds, and VIII to XII are electric constraints.

For each constraint, there are two rows which list the sensitivities of the two performances gain margin (GM) and transit frequency (Ft) with respect to constraint variations. The numeric values indicate how much the performances improve when the respective sizing constraints are relaxed. For lower and upper bounds, relaxation means a decrease and an increase of the limit values, respectively. Most electric constraints feature a safety margin, which can be lowered. Hence, the sensitivities could be given in dB/μm or MHz/mV, for example. However, this would make it impossible to compare the stringency of different types of constraints. For this reason, the performance variations in dB and MHz, respectively, are related to a *relative* relaxation of the constraints. For upper and lower bounds, these relaxations are given in percent of the limit values, and for electric constraints in percent of the safety margins. For example, for gain margin and a lower bound, a sensitivity value of 1 dB/% would mean that if the lower bound were decreased by 1%, then the gain margin would increase by 1 dB in the linear model. Of course, the actual improvement in the real circuit might differ due to nonlinear effects. Additionally, a relaxation of one constraint can activate another one which used to be inactive.

The last five columns list the actual sensitivity values for the five Pareto points according to Figure 5.4. Empty fields mean that a constraint is not active. For a proper interpretation of the sensitivity values, some specifics of the NBI method have to be taken into account. Recall that for the determination of the individual minima, only one performance is considered regardless of the other one. Graphically, this corresponds to a further move in parallel to the respective coordinate axis, as indicated by the arrows at point 1 and 5 in Figure 5.4. That is the reason why in Pareto point 1 the sensitivity of the transit frequency is always zero and in point 5 the same is true for the gain margin. For the intermediate points 2 to 4, a line search in the performance space is carried out in the direction of the quasi-normal to the convex hull of individual minima, cf. Figure 3.10. Consequently, there are always nonzero sensitivity values for both performances in these points.

Analysis of Active Constraints

A look at constraints I and III reveals that the entire biasing circuitry is driven toward minimum lengths. Constraint III shows that for all Pareto points above the minimum transit frequency, the lengths of both the PMOS (P1, 3, 5, 7, 9) and the NMOS (N1, 3, 5, 9, 11) current mirrors hit their minimum values. Together with corresponding level shifters (PMOS: P2, 4, 6, 8, 10; NMOS: N2, 4, 6, 10, 12), these current mirrors form a cascode current mirror according to Table 2.1. The level shifters even have minimum lengths for all Pareto points, cf. constraint I.
The only minimum width can be found in the differential input pair (N7, 8) for ambitious gain margins (constraint II, points 1, 2).

The examination of the active upper parameter bounds shows that they only refer to device widths, but not to lengths. The focus moves from the NMOS part of the output stage (N9, 10, 11, 12) in point 1 (constraint IV) to the corresponding PMOS part (P7, 8, 9, 10) in point 2 (constraint V) and to the active load (N5, 6) of the differential input pair in points 3 to 5 (constraint IV).
While for high gain margins the input pair (N7, 8) requires minimum widths (constraint II), it has the maximum width for the maximum transit frequency (constraint VII, point 5).

As can be seen from constraint VIII, keeping transistor P5 in saturation is a problem throughout the entire Pareto front. The associated sizing constraint specifies the minimum drain-source overdrive voltage:

$$v_{ds}(Ni) - (v_{gs}(Ni) - V_{th,n}) \geq V_{sat,min,n} \quad \text{for NMOS devices,} \quad 1 \leq i \leq 12 \tag{5.1}$$
$$-(v_{ds}(Pj) - (v_{gs}(Pj) - V_{th,p})) \geq V_{sat,min,p} \quad \text{for PMOS devices,} \quad 1 \leq j \leq 10 \tag{5.2}$$

In these formulae, $v_{ds}(\cdot)$ denotes the drain-source voltage of a transistor and $v_{gs}(\cdot)$ its gate-source voltage. The threshold voltages for NMOS and PMOS are given by $V_{th,n}$ and $V_{th,p}$, respectively. The safety margins $V_{sat,min,n}$ and $V_{sat,min,p}$ are chosen by the designer.
Constraint XI shows that not only the saturation of transistor P5 causes a performance limitation, but also the saturation of P6 which forms a cascode with P5.
For increasing transit frequencies (points 3 to 5), the saturation of N6 which is located in the active load of the differential pair becomes an obstacle for further performance improvement (constraint XII).
In Pareto point 1, where the highest gain margin is achieved, the output load seems to cause an asymmetry in the two branches of the output stage which are formed by P7, P8, N9, N10 and P9, P10, N11, N12, respectively. This asymmetry is reflected by two active constraints which ensure sufficient gain-source voltages for the transistors N12 (X) and N8 (IX).
The minimum gate-source overdrive voltage of transistor N12 is specified by

$$v_{gs}(N12) - V_{th,n} \geq V_{gs,min,n}\,. \tag{5.3}$$

Strong inversion of N8 is ensured by

$$v_{gs}(N12) - V_{th,n} \geq 0\,. \tag{5.4}$$

Since in (5.4) there is no safety margin which could be used as a reference, the corresponding sensitivity has to be given in dB/mV and MHz/mV, respectively. That is no drawback because this is a "hard" constraint which cannot be relaxed without sacrificing functionality.

So far, the actual numeric values of the performance sensitivities have not been considered. As discussed above, these values quantify how a relaxation of a certain constraint can be traded off for a performance improvement. If a constraint relaxation of about 10% is considered acceptable, then the achievable gain improvements are about 1 dB at most (constraint II, point 1). For the transit frequency, additional 8.5 MHz are achievable in the best case (constraint III, point 5). At this point, it should be noted that the sensitivities referring to electric constraints are even about one order of magnitude smaller than those relating to parameter bound constraints[†]. Considering these numbers, it becomes obvious that a constraint relaxation for the sake of performance improvement should only be done carefully. First of all, the improvements will usually only be moderate. Furthermore, lowering safety margins and minimum device sizes is likely to promote nonlinear effects, which can severely impact the robustness of the circuit. An enlargement of devices is less critical if circuit size is no major concern. Yet, a topological modification might be preferable to mere constraint relaxations in most cases.

In summary, valuable design information can be gained by examining the set of active sizing constraints and the associated sensitivities: problem areas can be spotted and the corresponding sensitivity information can be interpreted as a measure of stringency.

5.2.3 Topology Selection

As mentioned before, the Pareto fronts of a circuit topology describe its ultimate performance capabilities for a given technology. Therefore, these fronts can be used to compare different topologies in order to select the best one for a certain design task.

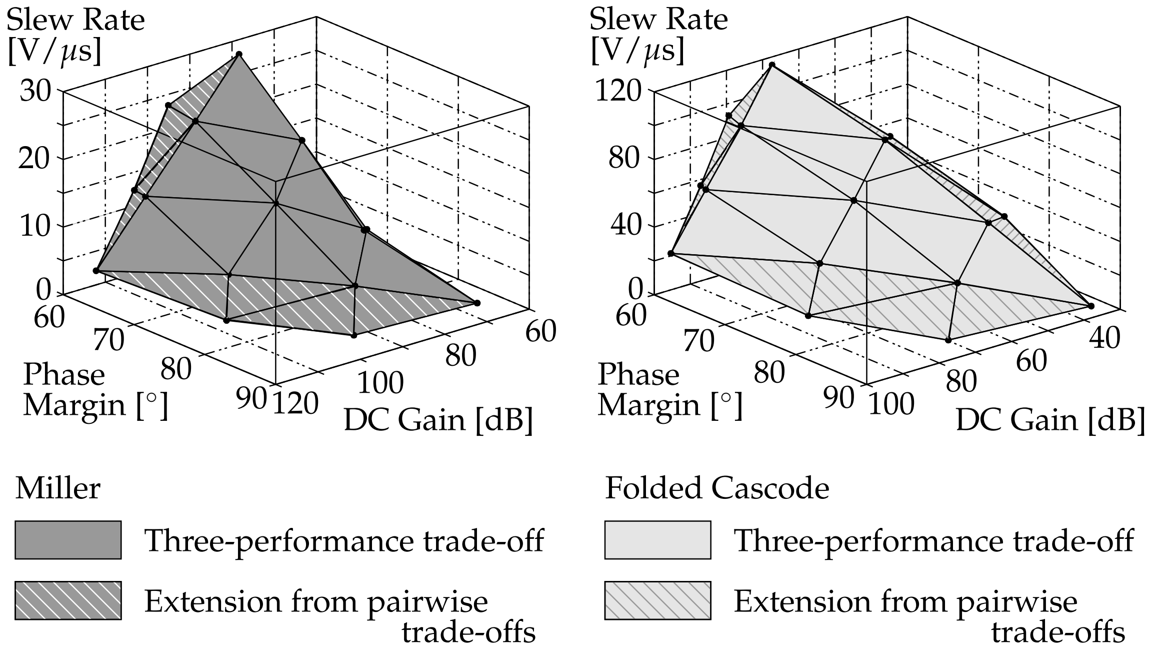


Figure 5.5: 3D Pareto Fronts

[†] The only exception is constraint IX, which, however, is an inflexible constraint as discussed earlier.

Figure 5.5 shows two 3D Pareto fronts relating DC gain, phase margin, and slew rate for the Miller (left) and the folded cascode architecture (right). In both cases, a phase margin exceeding 60° was specified due to stability considerations. As in the previous section, the folded cascode amplifier additionally had to exhibit 3 dB frequencies beyond 500 Hz to avoid pathologic behavior.

The homogeneously filled areas represent the three-performance trade-offs, whereas the hatched parts indicate the extensions of the Pareto fronts due to the additional pairwise trade-offs, cf. Figures 3.11 and 3.12 in Section 3.2.2. Note, for example, that in the case of the Miller amplifier, the trade-off between phase margin and DC gain significantly extends the overall Pareto front, whereas the trade-off between phase margin and slew rate only contributes an insignificant part. Furthermore, the Miller architecture yields a slightly nonconvex Pareto front, whereas the folded cascode Pareto front is convex.

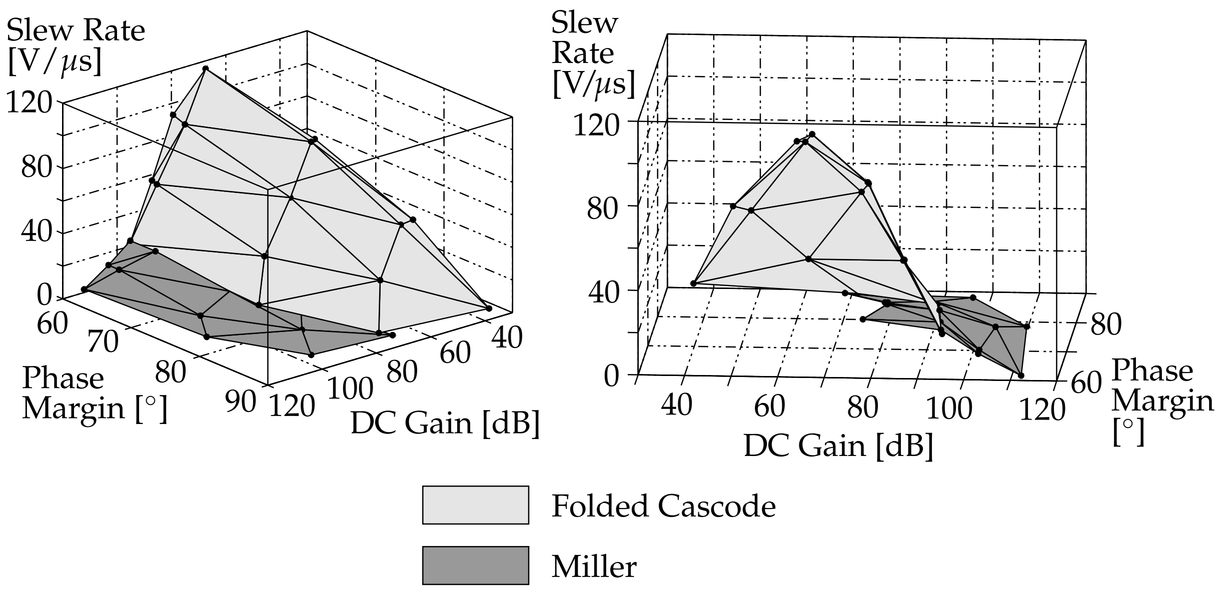


Figure 5.6: Topology Comparison Using Three-dimensional NBI

To allow an easy comparison, both Pareto fronts are shown in the same graph in Figure 5.6. The left subfigure adopts the viewpoint from Figure 5.5. While both circuits are comparable in terms of phase margin, the Miller amplifier achieves slightly higher DC gains. The most striking difference is the range of achievable slew rates. The folded cascode amplifier is far superior in this respect. The right graph in Figure 5.6 depicts the same surfaces from behind and impressively shows how the folded cascode Pareto front arches over the Miller counterpart.

Three-dimensional graphs unfold their full potential in an interactive environment, where by turning and moving the designer can get a vivid impression of the surface. In static contexts, 2D representations of the essential trade-offs can be sufficient. As an example, Figure 5.7 focuses on the trade-off between DC gain and slew rate.

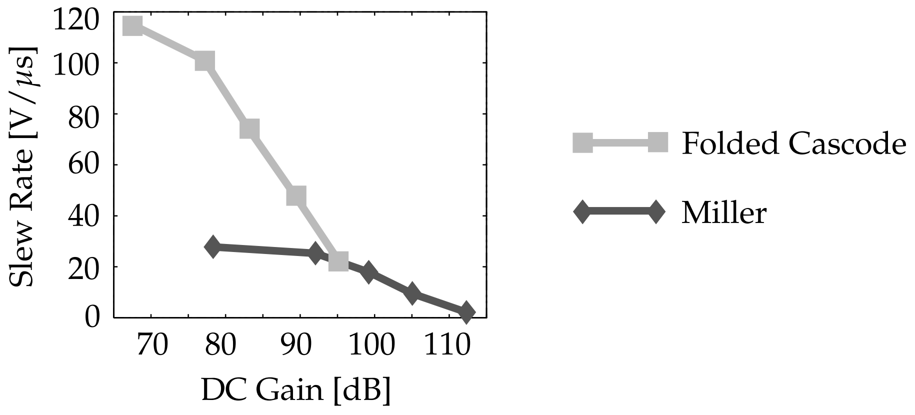


Figure 5.7: Topology Comparison Using Two-Dimensional NBI

From the above Pareto fronts it is evident that for low-speed applications the Miller amplifier can be a good architecture considering its small size. For more ambitious speed requirements, the folded cascode amplifier is the architecture of choice. That way, Pareto fronts can be a valuable tool for interactive topology selection.

5.2.4 Computational Effort

The new NBI-based nonlinear performance space exploration method relies on transistor-level circuit simulations for accuracy. Hence, the resource requirements are a legitimate concern. All the experiments in this section were done on a single 3 GHz Pentium 4 machine with 1 GB of RAM. Table 5.3 summarizes the characteristics of the experiments along with the respective computational effort.

To correctly interpret the number of simulations, it should be noted that the sizing constraints and the performances are combined into *simulation groups*. This means, for example, that for a certain parameter vector, all corresponding electric constraint values including the static power consumption can be extracted from a single DC simulation. Analogously, one AC simulation yields all small-signal performances. Finally, one more simulation is required for the transient performances.
In the available simulation environment, the determination of constraint and performance sensitivities had to be done using finite differences. To calculate the sensitivity of a single performance group with respect to parameter variations, m+1 simulations were required with m being the number of parameters. Consequently, a single sensitivity calculation required 12 simulations for the folded cascode amplifier and 9 for the Miller architecture. This is one of the reasons for the significant differences in the simulation counts. Obviously, the simulation time can be reduced significantly if a simulator is used which yields the sensitivities directly.

For the stochastic simulation-based exploration method presented in [DG02, DG03], computation times of several hours were reported. In contrast, the suggested approach yields 2D Pareto fronts within less than fifteen minutes and 3D fronts within

Amplifier	Performance Trade-Off	# Pareto Points	Figure Reference	# Simu-lations	Wallclock Time [min:sec]
folded cascode	DC gain 3 dB frequency	5	5.2	1553	15:13
folded cascode	gain margin transit frequency	5	5.2	1368	13:18
folded cascode	phase margin DC power	5	5.3	1348	12:38
folded cascode	slew rate DC power	5	5.3	1537	14:15
Miller	DC gain phase margin slew rate	16	5.5, 5.6	3264	31:18
folded cascode	DC gain phase margin slew rate	16	5.5, 5.6	5434	49:30
Miller	DC gain slew rate	5	5.7	1133	10:35
folded cascode	DC gain slew rate	5	5.7	1681	14:34

Table 5.3: Characteristics of All Experiments from Section 5.2

tens of minutes. Since these runtimes were obtained on a single machine, a parallelization would even shorten the execution times. Both a parallel simulation and a concurrent determination of different Pareto points are feasible. The moderate runtimes make the new exploration method applicable to practical circuit design and to topology selection in analog synthesis. Of course, this technique is well-suited not only for cell-level design using circuit simulation but also for system-level design using adequate models for the circuit performances.

5.3 Fourier-Motzkin Elimination

In the following sections, a few features of the Fourier-Motzkin-based exploration technique are presented. Section 5.3.1 illustrates its use for the visualization of three-dimensional performance spaces, which enables a fast interactive topology selection. To evaluate the approximation accuracy, linearized results are compared to accurate nonlinear ones. The runtime behavior of the Fourier-Motzkin elimination is discussed in detail in Section 5.3.2.

5.3.1 Topology Selection

The primary goal of the suggested linear performance space exploration technique is to derive a linear description of the entire high-dimensional circuit performance space. After an equation-based parameter replacement, the remaining parameters are removed using Fourier-Motzkin elimination. Of course, it is also possible to eliminate all but three performances. The resulting three-dimensional feasible performance space can then readily be visualized.

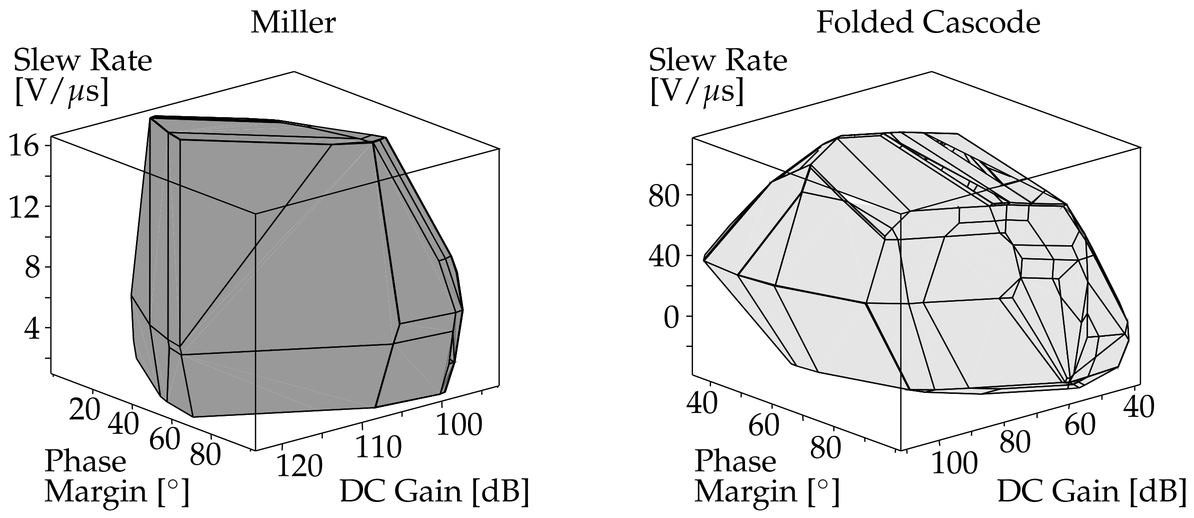


Figure 5.8: 3D Performance Spaces

Figure 5.8 shows the estimated feasible performance spaces of DC gain, phase margin, and slew rate for the Miller and the folded cascode amplifiers from Figure 5.1. This experiment was similar to the nonlinear one that led to the results in Figure 5.5. The only difference was that no minimum phase or minimum 3 dB frequency were specified. In spite of obvious linearization errors such as negative slew rates in the case of the folded cascode amplifier, these linearized feasible performance spaces immediately reveal the strengths and weaknesses of the different amplifiers: The Miller architecture achieves higher DC gains, but it is far inferior to the folded cascode amplifier in terms of slew rate. The combination of both feasible performance spaces into a single graph allows an even easier topology comparison as shown in Figure 5.9, left.

Since only phase margins exceeding 60° are displayed in the combined graph, the results are directly comparable to the 3D Pareto fronts from Figure 5.6. Those fronts are shown again in the right part of Figure 5.9 for convenience. It is evident that the linear approximations are reasonably accurate. Yet, a closer look reveals some inaccuracies: While the achievable slew rates were estimated very well for the folded cascode amplifier, they have been underestimated slightly for the Miller topology. For both amplifiers, the maximum DC gains were met accurately, whereas the minimum gain was predicted too optimistically for the Miller amplifier. Finally, the phase margins were overestimated by roughly 10° in both cases. In spite of these inaccura-

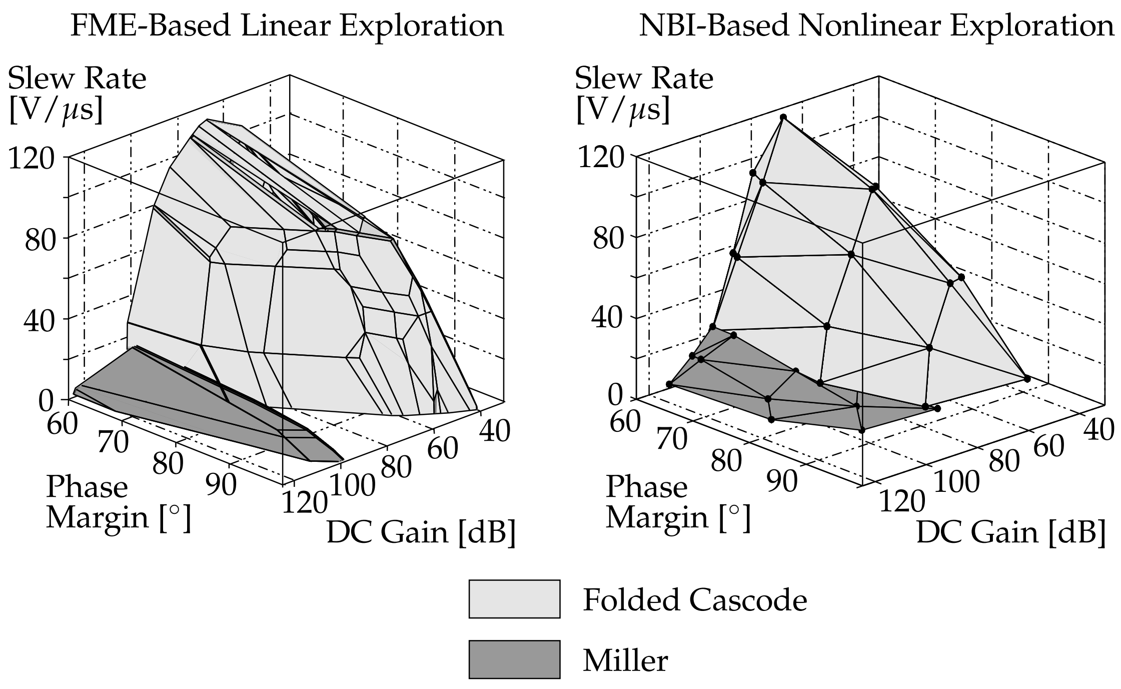


Figure 5.9: Performance Comparison of Two Operational Amplifiers

cies the linear estimates correctly reflect the characteristics of the two amplifiers. This is an excellent trade-off between accuracy and speed. The total computation time of these estimates, including simulation, performance space exploration and visualization, was about 1.5 minutes on a 3 GHz Pentium 4 machine, cf. Section 5.3.2.3. In contrast, the nonlinear exploration required 81 minutes in total.

5.3.2 Runtime Behavior

This section provides a detailed insight into procedural details of the Fourier-Motzkin elimination through the discussion of intermediate data: Section 5.3.2.1 shows how the number of inequalities evolves during the determination of an eight-dimensional performance space, whereupon the effectiveness of the overall redundancy detection strategy is demonstrated in Section 5.3.2.2. Effective CPU times are presented in Section 5.3.2.3.

5.3.2.1 Determination of an Eight-Dimensional Performance Space

Low-dimensional performance spaces can be handled by nonlinear techniques, albeit with long runtimes. The true strength of the linearized approach, in contrast, lies in the examination of high-dimensional spaces which can no longer be handled by nonlinear methods.

For the following example, the eight-dimensional performance space of the folded cascode amplifier from Figure 5.1 was examined. According to Table 5.1, this circuit consists of 22 transistors and has 11 designable parameters. An automatic topological analysis resulted in a total number of 189 inequality sizing constraints. For performance space exploration, the following steps were carried out:

- Simulation-based determination of linearized circuit descriptions according to (4.4) and (4.7).
- Initial numerical redundancy check according to Section 4.2.3.2: 38 out of 189 inequalities were identified as irredundant.
- Equation-based replacement of designable parameters 1, 5, 6, 7, 8, 9, 10, and 11.
- Fourier-Motzkin elimination of remaining parameters 2, 3, and 4:

	Parameter 2	Parameter 3	Parameter 4	
$\vert I_<\vert$	19	142	357	(I)
$\vert I_>\vert$	· 16	· 74	360	(II)
$\vert I_=\vert$	+ 3	+ 2	+ 0	(III)
# Inequalities	= 307	= 10510	= 128520	(IV)
# Void	– 5	– 8	– 6	(V)
# Redundant Inequalities Detected by Chernikov	– 0	– 9674	– 126136	(VI)
# Redundant Inequalities Detected by Linear Programming	– 84	– 111	– 562	(VII)
# Irredundant Inequalities	= 218	= 717	= 1816	(VIII)

Recall that for the elimination of a certain parameter, the set of all inequalities is partitioned into three subsets $I_<$, $I_>$, and $I_=$. They comprise of those inequalities which have a negative, a positive or a zero coefficient at the position of the respective parameter. The total number of inequalities right after the parameter elimination and before any redundancy detection can be calculated from the powers of these sets according to lines (I) to (III) and (4.31). The respective results are given in line (IV).

The following lines give details on the redundancy detection process: A few inequalities had only zero coefficients and could be discarded (line (V)). The Chernikov criterion (line (VI)) and the linear-programming-based redundancy detection (line (VII)) left a final number of irredundant inequalities as given in the last line. Note how the efficiency of the Chernikov criterion improved in the course of the successive elimination steps. Although the relative importance of the second redundancy removal step seems to vanish at the same rate, the opposite is true as will be shown in the next section.

5.3.2.2 Effectiveness of Redundancy Detection

To illustrate the effectiveness of the suggested two-step redundancy detection, an operational transconductance amplifier was examined. It featured 11 parameters and 3 AC performances. Consequently, eight Fourier-Motzkin elimination steps were required. An initial redundancy check using linear programming reduced the number of original inequalities from 120 to 47. The impact of different detection strategies is illustrated in Figure 5.10. Without any redundancy detection, the number of inequalities exploded already after the second elimination step and the calculations had to be stopped due to resource limitations. The exclusive use of the Chernikov criterion was only able to postpone this explosion by one step. With the help of redundancy detection via linear programming, the resource consumption could be kept within practical limits.

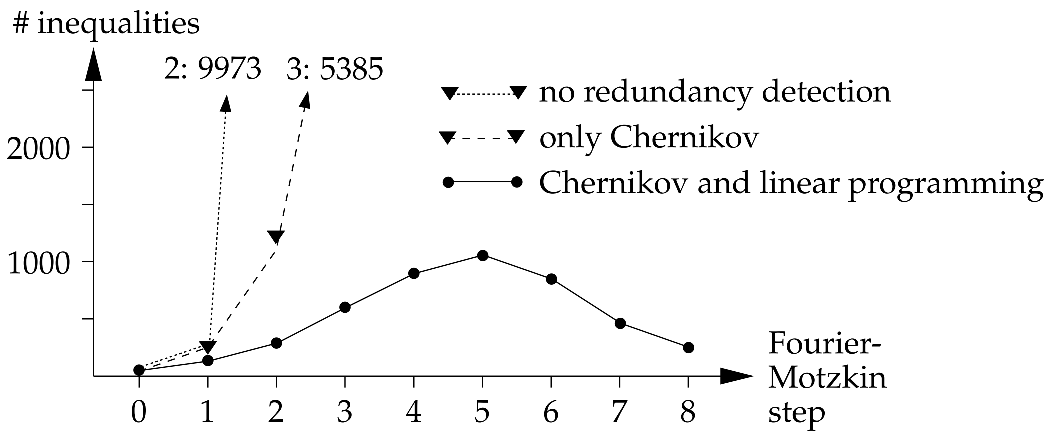


Figure 5.10: Effect of Different Redundancy Detection Strategies

5.3.2.3 Computational Effort

An outstanding efficiency in high-dimensional spaces is the greatest strength of the new linear performance space exploration technique. This claim is well supported by the runtimes of the experiments presented above, which have all been carried out on a 3 GHz Pentium 4 machine with 1 GB of RAM.

Topology Selection The left part of Figure 5.9 shows three-dimensional linearized performance spaces of the Miller and the folded cascode amplifiers from Figure 5.1. The entire performance space exploration process which was required to create that graph consisted of the following steps:

1. Circuit simulation to obtain a linearized circuit description according to (4.4) and (4.7).
2. Equation-based parameter replacement and inequality-based parameter elimination via Fourier-Motzkin.
3. Visualization of the resulting geometric object [Fra04].

To put the different contributions to the overall computation time into perspective, they are listed separately for both amplifiers in Table 5.4. The contribution of the second step was not subdivided further because the effort of parameter replacement is negligible compared to the cost of parameter elimination as will be shown later.

Amplifier	Simulation	Parameter Replacement/ Elimination	Visualization
Miller	6 sec	2 sec	4 sec
folded cascode	15 sec	56 sec	14 sec

Table 5.4: Contributions to Overall Computation Time for Linearized Performance Spaces in Figure 5.9

It is striking that the parameter replacement and elimination, respectively, only plays a minor role for the Miller amplifier. The reason is that this circuit has only eight parameters, which means that just five elimination steps had to be done. For the folded cascode amplifier with eleven parameters, eight elimination steps were needed. Additionally, the folded cascode amplifier is much larger and is subject to far more inequality sizing constraints (189 vs. 79, cf. Table 5.1).
Nonetheless, the complete topology comparison required just 97 seconds in total. This is especially remarkable in comparison to nonlinear exploration times of far more than an hour.

The table above highlighted two exploration runs for a certain combination of three performances. To get a better idea of the resource consumption of the Fourier-Motzkin elimination method, it is instructive to perform the projection for all performance triples and to compare the computation times. Since the available amplifier testbench provided a total number of eight performances, there were 56 different triples. Figure 5.11 summarizes the resulting projection times for both amplifiers in the form of histograms. The x-axis, which represents the calculation time, was divided into 50 equal intervals or *bins*. The y-axis corresponds to the number of projection runs, which had calculation times from within the respective bins.

First of all, it is evident that the projections for the Miller amplifier are more than two orders of magnitude faster than those for the folded cascode architecture. This observation is in line with the result presented in Table 5.4, and the reasons were given above.
Yet, the most important effect to note is the distribution of the computation times. In both cases, most of the samples cumulate in the lower half of the entire time range. The individual computation times differ significantly, though. It is obvious that the

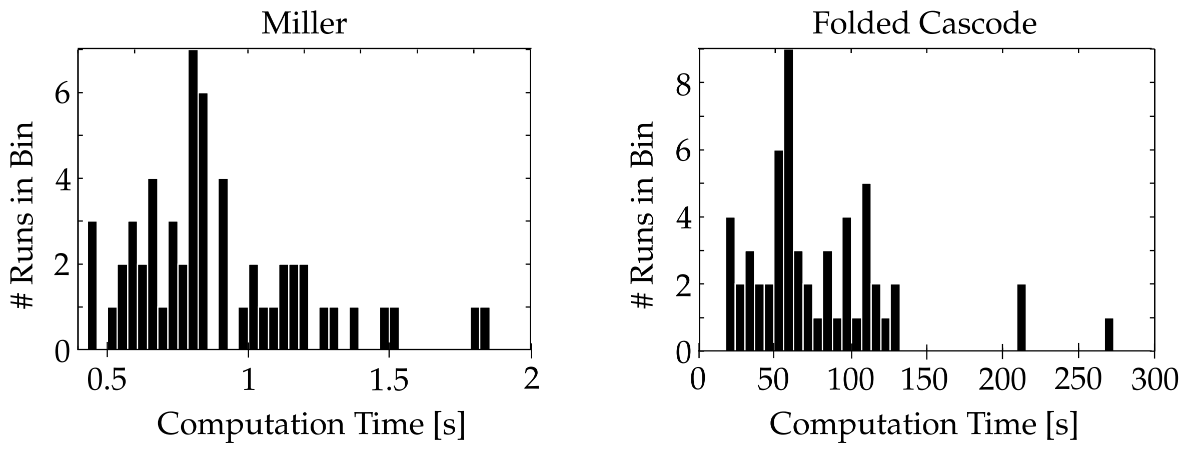


Figure 5.11: Histogram Showing Projection Times for All Performance Triples

execution time of the Fourier-Motzkin elimination algorithm is not only determined by the circuit size and the number of elimination steps. Instead, the computational effort clearly depends on the numerical properties of the actual problem. Unfortunately, the derivation of reliable numerical criteria which allow an evaluation of input data and a prediction of the computation times to be expected is still in an early stage, cf. footnote on Page 102 in Section A.1.4.1.

Eight-Dimensional Performance Space Section 5.3.2.1 presented a detailed analysis of how the number of inequalities evolved during the exploration of an eight-dimensional performance space. A breakdown of the corresponding calculation times can be found in the left part of Table 5.5.

This table reflects three procedural stages:

- *Initialization*: The simulation data is read and the equation-based parameter replacement is carried out. This yields the original set of inequalities. At this stage, the Chernikov criterion is not applicable since it can only detect structural redundancy, which is introduced in the course of the elimination process. Therefore, the inequalities are directly passed on to an initial numerical redundancy detection. Note that the computation time for these steps, and for the equation-based parameter replacement in particular, is negligible.
- *Parameter Elimination*: To obtain the eight-dimensional feasible performance space, three Fourier-Motzkin elimination steps were required. It is evident that the computational effort increases with each step due to a rising number of inequalities. Note that the numerical redundancy detection using linear programming consumes the largest fraction of the CPU time.
- *Output of Final Inequalities*: The internal representation of the final inequalities is converted into a text format which can easily be read by other tools. The resource consumption is marginal.

	Experiment from Section 5.3.2.1		Experiment from Section 5.3.2.2	
	Creation of Inequalities; Chernikov	Numerical Redundancy Detection	Creation of Inequalities; Chernikov	Numerical Redundancy Detection
Initialization	< 1 sec	< 1 sec	< 1 sec	< 1 sec
Elimination Step 1	< 1 sec	1 sec	< 1 sec	< 1 sec
Elimination Step 2	< 1 sec	8 sec	< 1 sec	1 sec
Elimination Step 3	3 sec	169 sec	< 1 sec	2 sec
Elimination Step 4			1 sec	29 sec
Elimination Step 5			3 sec	74 sec
Elimination Step 6			2 sec	14 sec
Elimination Step 7			2 sec	18 sec
Elimination Step 8			1 sec	7 sec
Data Output	2 sec		< 1 sec	
Total	183 sec		154 sec	

Table 5.5: Numerical Redundancy Detection as Major Contributor to Overall Computation Time

Effectiveness of Redundancy Detection To justify the effort of an expensive numerical redundancy detection, Section 5.3.2.2 showed that only with this second detection step the overall resource requirements could be kept under control. The right part of Table 5.5 provides details on the calculation times. In this example, the effort for both initialization and the final data output was negligible. As could be expected from Figure 5.10, the resource requirements for redundancy detection reached their maximum in the fifth elimination step in accordance to the number of inequalities. The determination of the final 327 inequalities took 154 seconds in total and required just 7.3 MB of main memory. The numerical redundancy detection consumed about 94% of the entire CPU time.

5.4 Hierarchical Sizing

While the previous section focused on the Fourier-Motzkin-based exploration technique itself, a more comprehensive view follows, which puts this method into a larger context.

When design decisions can be based on the trade-off of only a few performances, then nonlinear optimization-based methods are the tools of choice. Yet, for example in hierarchical sizing as outlined in Section 1.1.3, high-dimensional performance spaces have to be explored. This cannot be done using nonlinear techniques due to excessive resource requirements. This is the area where the suggested linear exploration technique can unfold its full potential.

The problem of hierarchical sizing is described in Section 5.4.1. A hierarchical sizing method which embeds the suggested linear exploration technique is presented in Section 5.4.2. As an illustration, a biquad bandpass filter design is demonstrated.

5.4.1 Problem Description

Simulation-based circuit sizing requires a large number of circuit simulations. Small to medium-sized analog circuits with short simulation times and a moderate number of designable parameters can be sized based on transistor models and simulators like SPICE. Yet, for complex electronic circuits this approach results in impractical resource requirements due to long simulation times and a large number of designable parameters. There is common agreement about the necessity of a hierarchical approach for the simulation-based sizing of larger analog circuits, e. g. [CCC+97, DNAV99, VDL+01, HS96, DGV+04]. Yet, there is little material in the literature when it comes to concrete solutions. A problem of paramount importance is the suitable formulation of physically motivated, possibly high-dimensional sizing constraints at higher levels of abstraction. This problem is a main target of the suggested linear performance space exploration method. As an illustration, a hierarchical sizing approach is presented below, which features two levels of abstraction.

In the preceding parts of this thesis, circuits were considered from a non-hierarchical point of view as a netlists of transistors. This level of abstraction is referred to as *block level* henceforth, and the designable parameters **p** are called *block parameters*. Accordingly, the performances **f** are named *block performances*. At this level, the sizing process can be written in the form of the following optimization problem[‡]:

$$\hat{\mathbf{p}} = \underset{\mathbf{p}}{\operatorname{argmin}} \left\| \left(\mathbf{f}(\mathbf{p}) - \hat{\mathbf{f}}\right) / \hat{\mathbf{f}} \right\| \quad \text{s.t.} \quad \mathbf{c}(\mathbf{p}) \geq \mathbf{0}\,. \tag{5.5}$$

Here, the operator / denotes a component-wise vector division. The specifications and the corresponding sizing result are marked by a hat. The goal of the above optimization formulation is to meet the specifications as closely as possible. The sizing constraints make sure that only “technically meaningful” sizings are considered.

Larger analog circuits, such as the PLL in Section 1.1.3, are composed of functional blocks of small to medium size. Each of these blocks can be modeled behaviorally, e.g. using a hardware description language like VHDL-AMS [CB99, APT02]. If the entire circuit is described as a netlist of such behavioral models, then a simulation speedup of two to three orders of magnitude can be expected [Ziz01, SGA03b]. Hence, simulation-based sizing becomes viable again. This level of abstraction is called *system level* henceforth, and the large circuit is referred to as *system* accordingly. In the case of a PLL, the *system performances* **s** include the lock time and the

[‡] This is a simple optimization formulation for conceptual purposes. For a more in-depth discussion, cf. [Sch02].

natural frequency, among others. The actual behavior of the models can directly be adjusted by appropriate parameters. Since the models and their underlying circuit realizations are supposed to behave identically, the performances of the functional blocks also serve as model parameters, which are called *system parameters* here. Examples are delay, gain, or dynamic properties. The system-level netlist might inherit some elementary components from the block-level netlist, which do not need behavioral modeling. Examples are capacitors or resistors in feedback networks. Since these components are not subject to abstraction, their parameters are shared between both levels of abstraction. Therefore, they are called *shared parameters* here.

Assume that a given electronic system is partitioned into N functional blocks. Let the performances of the i^{th} block be referred to as $\mathbf{f}_i$. Then the parameters at system level comprise of the block performances and the shared parameters, which are written as $\mathbf{f}_{N+1}$ in this context:

$$\mathbf{f} = \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_N \\ \mathbf{f}_{N+1} \end{bmatrix} . \tag{5.6}$$

The system performances $\mathbf{s}$ result from a quick behavioral simulation:

$$\mathbf{s} = \mathbf{s}(\mathbf{f}) . \tag{5.7}$$

In analogy to the block level, the range of feasible system parameters should be restricted by *sizing constraints at system level*. Obviously, the feasible range of the system parameters $\mathbf{f}_i$, $1 \leq i \leq N$, is limited by the feasible block performance spaces. Of course, there may be constraints on the shared parameters also: $\mathbf{k}_{N+1}(\mathbf{f}_{N+1}) \geq \mathbf{0}$. Additionally, all system parameters $\mathbf{f}$ may be subject to constraints, which refer to the system as a whole: $\mathbf{k}_{\text{sys}}(\mathbf{f}) \geq \mathbf{0}$. These constraints control system behavior, which would usually not be specified explicitly. They might, for example, ensure stability. Since they have to be evaluated using system simulation, they correspond to electric sizing constraints at block level.

Let $\mathbf{k}_i(\mathbf{f}_i) \geq \mathbf{0}$ with $i \in \{1 \ldots N\}$ denote the feasible performance space of block i according to (4.2). Then the sizing constraints at system level can be written as

$$\mathbf{k}(\mathbf{f}) \geq \mathbf{0} \Leftrightarrow \begin{bmatrix} \mathbf{k}_1(\mathbf{f}_1) \\ \vdots \\ \mathbf{k}_N(\mathbf{f}_N) \\ \mathbf{k}_{N+1}(\mathbf{f}_{N+1}) \\ \mathbf{k}_{\text{sys}}(\mathbf{f}) \end{bmatrix} \geq \mathbf{0} . \tag{5.8}$$

With such a system description, a two-step hierarchical sizing can be carried out as depicted in Figure 5.12.

In a first step, a single optimization is carried out *entirely at system level*:

$$\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmin}} \left\| \left(\mathbf{s}(\mathbf{f}) - \hat{\mathbf{s}}\right) / \hat{\mathbf{s}} \right\| \quad \text{s.t.} \quad \mathbf{k}(\mathbf{f}) \geq \mathbf{0} . \tag{5.9}$$

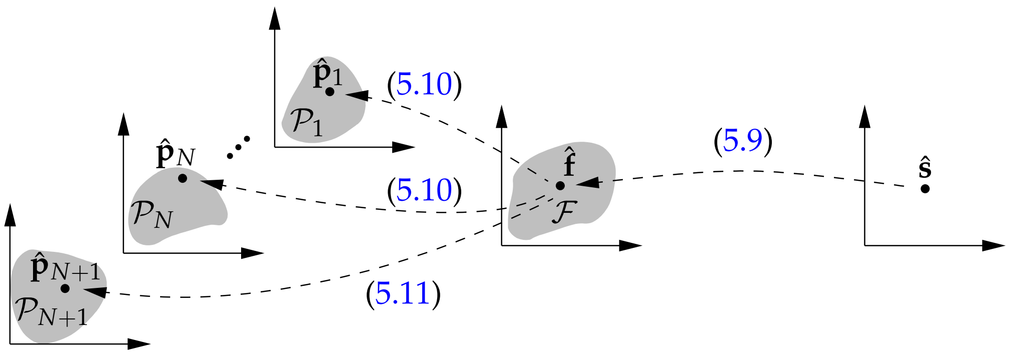


Figure 5.12: Hierarchical Sizing

In this way, the system specifications $\hat{\mathbf{s}}$ are broken down to individual block specifications $\hat{\mathbf{f}}_i$. The system sizing constraints $\mathbf{k}(\mathbf{f})$ make sure that these specifications can actually be met by the available circuit topologies. One more optimization per functional block yields the actual component values at block level:

$$\underset{1\le i\le N}{\forall}\left(\hat{\mathbf{p}}_i = \underset{\mathbf{p}_i}{\operatorname{argmin}} \left\|\left(\mathbf{f}_i(\mathbf{p}_i) - \hat{\mathbf{f}}_i\right)/\hat{\mathbf{f}}_i\right\| \quad \text{s.t.} \quad \mathbf{c}_i(\mathbf{p}_i) \ge \mathbf{0}\right) \tag{5.10}$$

The shared parameters of the elementary components keep their values regardless of the level of abstraction. The alias

$$\hat{\mathbf{p}}_{N+1} \equiv \hat{\mathbf{f}}_{N+1} \tag{5.11}$$

emphasizes their validity at the lower level of abstraction also. The final sizing at block level is

$$\hat{\mathbf{p}} = \begin{bmatrix} \hat{\mathbf{p}}_1 \\ \vdots \\ \hat{\mathbf{p}}_N \\ \hat{\mathbf{p}}_{N+1} \end{bmatrix}. \tag{5.12}$$

Figure 5.13 summarizes this hierarchical sizing approach.

5.4.2 Iterative Sizing Approach

If an accurate nonlinear description of the high-dimensional system sizing constraints $\mathbf{k}(\mathbf{f})$ were available, then a straight top-down optimization according to Figure 5.13 would solve the sizing problem.

The Normal-Boundary Intersection technique presented in Section 3.2 can be used to accurately identify the boundaries of feasible block performance spaces. Like all nonlinear performance space exploration methods, however, this method consumes a prohibitive amount of resources if high-dimensional performance spaces have to be examined.

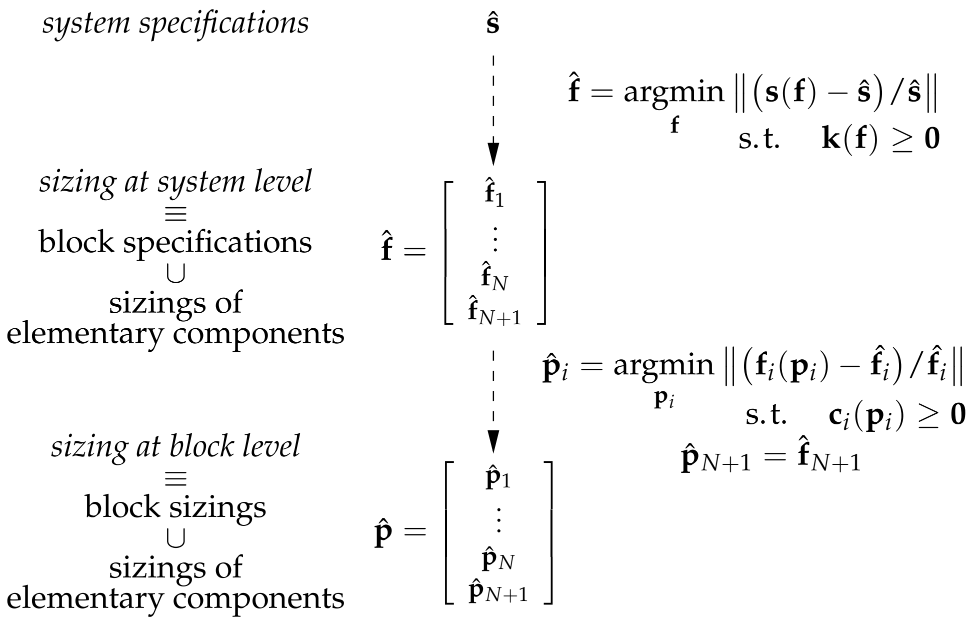

system specifications
$\hat{\mathbf{s}}$
$\hat{\mathbf{f}} = \operatorname{argmin}_{\mathbf{f}} \left\| \left(\mathbf{s}(\mathbf{f}) - \hat{\mathbf{s}}\right) / \hat{\mathbf{s}} \right\|$ s.t. $\mathbf{k}(\mathbf{f}) \geq \mathbf{0}$
sizing at system level ≡ block specifications ∪ sizings of elementary components
$\hat{\mathbf{f}} = \begin{bmatrix} \hat{\mathbf{f}}_1 \\ \vdots \\ \hat{\mathbf{f}}_N \\ \hat{\mathbf{f}}_{N+1} \end{bmatrix}$
$\hat{\mathbf{p}}_i = \operatorname{argmin}_{\mathbf{p}_i} \left\| \left(\mathbf{f}_i(\mathbf{p}_i) - \hat{\mathbf{f}}_i\right) / \hat{\mathbf{f}}_i \right\|$ s.t. $\mathbf{c}_i(\mathbf{p}_i) \geq \mathbf{0}$
$\hat{\mathbf{p}}_{N+1} = \hat{\mathbf{f}}_{N+1}$
sizing at block level ≡ block sizings ∪ sizings of elementary components
$\hat{\mathbf{p}} = \begin{bmatrix} \hat{\mathbf{p}}_1 \\ \vdots \\ \hat{\mathbf{p}}_N \\ \hat{\mathbf{p}}_{N+1} \end{bmatrix}$

Figure 5.13: Overview of Hierarchical Sizing Procedure

In this situation, an approximation to the feasible system parameter space can be obtained using the Fourier-Motzkin-based exploration approach. Due to linearization inaccuracies, however, unachievable block specifications might result from (5.9). Using a biquad filter as an example, the following section demonstrates how an iterative technique compensates for linearization inaccuracies and thus enables a successful hierarchical sizing.

5.4.2.1 Example Circuit

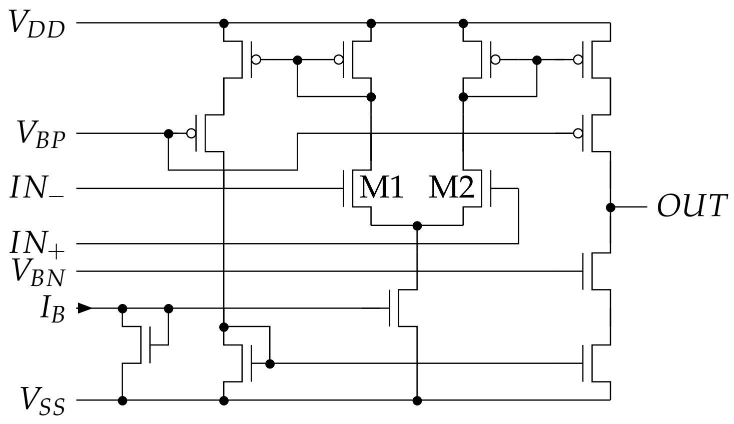


Figure 5.14: Operational Transconductance Amplifier

In Figure 5.14, an operational transconductance amplifier (OTA) is given in the form of a transistor netlist [GESSL95]. Based on symmetry requirements and additional heuristics, the number of essential block parameters $\mathbf{p}_i$ was reduced to nine. They comprise of the widths and lengths of the transistors and the bias current and voltages. To adequately describe the electrical AC behavior of the OTA, the block performances $\mathbf{f}_i$ do not only include its transconductance g, but also its input and output impedance, among others. In total, five important AC performances were identified. This OTA can be used as a building block for the biquad bandpass filter given in Figure 5.15 [GESSL95].

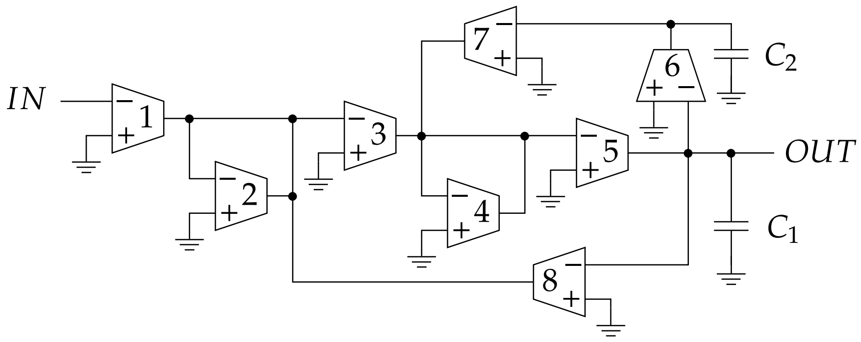


Figure 5.15: OTA-C Biquad Bandpass Filter Schematic

For an efficient simulation of the entire filter, the OTAs were modeled behaviorally in VHDL-AMS based on [GESSL95]. The models feature electrical pins in spite of the elevated level of abstraction. Consequently, the filter is represented by a netlist of eight behavioral OTA models and two capacitors. In AC domain, the system performances $\mathbf{s}$ comprise of the center frequency f_0, the pass-band gain G_0, and the quality factor Q.

Due to symmetries in the filter, two pairs of OTAs, namely $\text{OTA}_4/\text{OTA}_7$ and $\text{OTA}_5/\text{OTA}_6$ can be sized identically. The values of the two capacitors are shared parameters, which are not transformed hierarchically. Consequently, the biquad filter has $6 \cdot 5 + 2 = 32$ system parameters.

With $i \in I = \{1, 2, 3, (4/7), (5/6), 8\}$ and $\mathbf{f}_C = \mathbf{p}_C = [C_1\ C_2]$, Table 5.6 summarizes the characteristics of the two abstraction levels.

	Block Level	System Level
Netlist Elements	transistors, elementary components	VHDL-AMS models, elementary components
Parameters	$\mathbf{p}_i = [W_{M1}\ L_{M1}\ \dots\ I_B]$, $\mathbf{p}_C$	$\mathbf{f} = [\mathbf{f}_1\ \dots\ \mathbf{f}_8\ \mathbf{f}_C]$
Performances	$\mathbf{f}_i = [g\ R_{\text{out}}\ C_{\text{out}}\ \dots]$	$\mathbf{s} = [f_0\ G_0\ Q]$

Table 5.6: Abstraction Levels of Hierarchical Filter Description

5.4.2.2 Iterative Sizing Algorithm

Given high-dimensional block performance spaces, the sizing constraints at system level can only be approximated. Hence, the capabilities of the functional blocks can be over- or underestimated. In the first case, unachievable block specifications could result from a system sizing according to (5.9). In the second case, the capabilities of the entire system would not fully be exploited. In engineering and science, a wide range of nonlinear problems is routinely solved by algorithms which combine linearization and iteration. For hierarchical optimization, too, this is a viable approach.

Linear approximations are accurate in the neighborhood of the linearization points. Ideally, the performance spaces of the functional blocks should be linearized in the neighborhood of the final sizing. Of course, there is no a priori knowledge to do this. Yet, iteration enables a simultaneous improvement of both the linearization and the sizing: Based on initial approximations to the feasible block performance spaces, the system specifications are translated to a block-level sizing according to Figure 5.13. If at system and/or block level some specifications were missed, then another iteration can be started and the linearizations are updated with the current sizings as new linearization points. The main idea is that even if a sizing result does not meet the specifications, at least it identifies promising regions in the block parameter spaces. The circuit behavior in these regions can then be approximated more accurately by new linearizations. After that another top-down sizing run is carried out.

```
$\forall_{i\in I}$ determine $\hat{\mathbf{p}}_i$ with $\mathbf{c}(\hat{\mathbf{p}}_i) \geq \mathbf{0}$ by initial sizing, cf. Appendix A    (I)
    $\forall_{i\in I}$ determine $\mathbf{C}_i, \mathbf{c}_i, \mathbf{F}_i, \mathbf{f}_i(\hat{\mathbf{p}}_i)$ via simulation at $\hat{\mathbf{p}}_i$, cf. (4.4) and (4.7)    (II)
    $\forall_{i\in I}$ calculate $\mathbf{K}_i, \mathbf{k}_i$, cf. (4.8)    (III)
    $\hat{\mathbf{f}} = \underset{\mathbf{f}}{\operatorname{argmin}} \left\| (\mathbf{s}(\mathbf{f}) - \hat{\mathbf{s}})/\hat{\mathbf{s}} \right\|$ s.t. $\forall_{i\in I} \mathbf{K}_i \cdot (\mathbf{f}_i - \mathbf{f}_i(\hat{\mathbf{p}}_i)) \geq \mathbf{k}_i$
        $\wedge\ \mathbf{f}_C \geq \mathbf{f}_{C,\min} \wedge -\mathbf{f}_C \geq -\mathbf{f}_{C,\max}$    (IV)
    $[\hat{C}_1\ \hat{C}_2] = \hat{\mathbf{p}}_C = \hat{\mathbf{f}}_C$    (V)
    $\forall_{i\in I} \left( \hat{\mathbf{p}}_i = \underset{\mathbf{p}_i}{\operatorname{argmin}} \left\| (\mathbf{f}_i(\mathbf{p}_i) - \hat{\mathbf{f}}_i)/\hat{\mathbf{f}}_i \right\| \text{ s.t. } \mathbf{c}_i(\mathbf{p}_i) \geq \mathbf{0} \right)$    (VI)
$\left\| (\mathbf{s}(\hat{\mathbf{f}}) - \hat{\mathbf{s}})/\hat{\mathbf{s}} \right\|_\infty < \hat{\epsilon}_\mathbf{s} \quad \wedge \quad \left\| (\mathbf{f}_i(\hat{\mathbf{p}}_i) - \hat{\mathbf{f}}_i)/\hat{\mathbf{f}}_i \right\|_\infty < \hat{\epsilon}_\mathbf{f}$
    $\wedge \left\| (\mathbf{s}(\hat{\mathbf{p}}_1, \ldots, \hat{\mathbf{p}}_8, \hat{\mathbf{p}}_C) - \hat{\mathbf{s}})/\hat{\mathbf{s}} \right\|_\infty < \hat{\epsilon}_\mathbf{v}$    (VII)
```

$I = \{1, 2, 3, (4/7), (5/6), 8\}$; $/$ is element-wise vector division; $\|\cdot\|_\infty$ denotes L-Infinity-Norm

Figure 5.16: Hierarchical Sizing Algorithm

The outline of a hierarchical sizing algorithm for the above filter is given in Figure 5.16. There are seven procedural steps. The linear performance space exploration described in Chapter 4 is done in steps II) and III).

I) *Determination of initial linearization point by means of initial sizing*:
For a technically meaningful sizing, the OTAs have to satisfy all sizing constraints at block level. For symmetry reasons, for example, the drain-source voltages of the two transistors M1 and M2 must not differ too much. Therefore, it is advisable to choose an initial linearization point which safely satisfies all block-level constraints $\mathbf{c}(\hat{\mathbf{p}}_i) \geq \mathbf{0}$. This is true for the center point of the feasible parameter space $\mathcal{P}_i$, which is the result of the initial sizing process according to Appendix A. In addition, this point yields good linearization results since the circuit behavior is usually only weakly nonlinear in the center of the feasible parameter space.

II) *Determination of linearized circuit description via simulation*:
A linear approximation to the feasible parameter space $\mathcal{P}_i$ of OTA i is given by $\mathbf{C}_i$ and $\mathbf{c}_i$, cf. (4.4). The relation of block parameters $\mathbf{p}_i$ and block performances $\mathbf{f}_i$ is approximated by $\mathbf{F}_i$ and $\mathbf{f}_i(\hat{\mathbf{p}}_i)$ according to (4.7).

III) *Calculation of system sizing constraints*:
For each OTA, a linearized representation of its feasible performance space $\mathcal{F}_i$ according to (4.8) is determined using the Fourier-Motzkin-based exploration method.

IV) *System sizing*:
This sizing step is carried out entirely at system level. The associated sizing task can be interpreted mathematically as an optimization problem. The argmin operator yields the system parameter vector $\hat{\mathbf{f}}$ leading to the minimum relative deviation of the achieved performances from the specified values $\hat{\mathbf{s}}$. The system sizing constraints limit the optimization engine to the feasible OTA performance spaces and to the upper and lower capacitance bounds.

V) *Assignment of shared parameters*:
The determination of the shared parameters is trivial.

VI) *Simultaneous block sizing*:
In the block sizing step, the remaining system parameters $\hat{\mathbf{f}}_1 \ldots \hat{\mathbf{f}}_8$ turn into the block specifications of the individual OTAs. One block-level sizing run per OTA yields the block parameter values.

VII) *Check of exit conditions*:
If both the system sizing and the block sizing were successful, then the filter performances can be simulated at block level (if the circuit size allows it) in order to verify the sizing result[§].

[§] Of course, further criteria are required to ensure a reliable termination.

5.4.2.3 Exemplary Hierarchical Sizing Run

To illustrate the hierarchical sizing algorithm, it is applied to the above filter for the specifications

$$\hat{\mathbf{s}} = [\hat{f}_0 \;\; \hat{G}_0 \;\; \hat{Q}] = [1\,\text{MHz} \;\; 1.00 \;\; 50]\,. \tag{5.13}$$

The procedural steps are as follows:

- Initial sizing
- First iteration
  - *Linear performance space exploration* with initial sizing result as linearization point.
  - The *system sizing* yields $\mathbf{s}(\hat{\mathbf{f}}) = [1.050\,\text{MHz} \;\; 1.020 \;\; 45.00]$.
    The maximum relative difference between the achieved system performances $\mathbf{s}(\hat{\mathbf{f}})$ and the system specifications $\hat{\mathbf{s}}$ is $\epsilon_{\mathbf{s}} = \left\| \frac{45-50}{50} \right\| = 10\%$.
  - The detailed results of the *block sizing* are omitted here for brevity. Instead, the table below gives the maximum relative difference between the achieved block performances and the block specifications for each OTA.

	OTA$_1$	OTA$_2$	OTA$_3$	OTA$_{4,7}$	OTA$_{5,6}$	OTA$_8$
$\epsilon_{\mathbf{f}}$ [%]	*28.6*	2.0	1.9	2.0	2.1	2.1

  - The *exit conditions* are not met: For $\hat{\epsilon}_{\mathbf{s}} = \hat{\epsilon}_{\mathbf{f}} = 2\%$, both the system sizing and the block sizing terminate with unacceptable errors. Consequently, another iteration is required.
- Second iteration
  - *Linear performance space exploration*:
    Based on sizing result from previous iteration.
  - *System sizing*:
    $\mathbf{s}(\hat{\mathbf{f}}) = [0.9974\,\text{MHz} \;\; 1.006 \;\; 49.87] \quad \Rightarrow \quad \epsilon_{\mathbf{s}} = 0.6\%$.
  - *Block sizing:*

	OTA$_1$	OTA$_2$	OTA$_3$	OTA$_{4,7}$	OTA$_{5,6}$	OTA$_8$
$\epsilon_{\mathbf{f}}$ [%]	0.23	0.14	0.12	0.13	0.13	0.11

  - *Exit conditions*:
    The sizing errors both at system and at block level meet the tolerances of $\hat{\epsilon}_{\mathbf{s}} = \hat{\epsilon}_{\mathbf{f}} = 1\%$. A final non-hierarchical block-level simulation of the entire filter is easily possible in this case due to a small circuit size. It yields the system performances

$$\mathbf{s}(\hat{\mathbf{p}}_1, \ldots, \hat{\mathbf{p}}_8, \hat{\mathbf{p}}_C) = [0.9985\,\text{MHz} \;\; 1.002 \;\; 49.50] \quad \Rightarrow \quad \epsilon_{\mathbf{v}} = 1.0\% = \hat{\epsilon}_{\mathbf{v}}\,.$$

The final verification confirms the success of the hierarchical sizing process.

In this example, just two iterations were required in order to find a suitable sizing. The initial approximation to the feasible system parameter space did not allow the optimization engine to choose system parameter values which could make the filter meet the specifications. Besides, the specifications for OTA_1 turned out as overambitious. Nevertheless, the first iteration identified parameter vectors in promising areas of the feasible OTA parameter spaces. New linearizations in the neighborhood of the block-level sizing results $\hat{\mathbf{p}}_i$ led to an improved accuracy in the areas of interest. In the second iteration, it turned out that the capabilities of the filter had been underestimated at system level previously. By the same token, more realistic approximations to the feasible OTA performance spaces were obtained, which prevented unachievable block specifications for all OTAs.

5.5 Summary

This chapter demonstrated how the suggested performance space exploration methods can be applied in practice. It was shown how the nonlinear Normal-Boundary Intersection technique enables a trade-off analysis between competing performances and allows an analysis of the performance-limiting sizing constraints both in terms of identification and quantification. The resulting Pareto fronts represent the ultimate performance capabilities of a given circuit topology. Therefore, they facilitate the comparison of competing circuit realizations and hence enable a topology selection according to a given set of specifications. For all examples, the simulation times were given, which show that the resource requirements are moderate, especially in comparison to stochastic techniques.

In contrast to Normal-Boundary Intersection, the Fourier-Motzkin-based linear exploration technique primarily aims at high-dimensional performance spaces. Of course, this method can be applied to three dimensions as well in order to perform a topology selection by inspection. This was demonstrated using the same circuits and performances as for Normal-Boundary Intersection. A comparison of the results revealed a reasonable accuracy of the linear technique. It was shown that the demands on the computational resources were remarkably low even in high-dimensional performance spaces.

To illustrate an important application of linear performance space exploration, this technique was embedded in a hierarchical sizing method which features two levels of abstraction. In this context, the descriptions of the feasible block performance spaces provide the link between the levels of abstraction. Using a biquad bandpass filter as an example, it was shown that this sizing approach works even with approximate performance space descriptions because iteration compensates for linearization inaccuracies.

Chapter 6

Conclusion

In recent years, there has been an ongoing trend toward more and more functionality even in medium or low cost consumer electronics. The mass market of handheld computing and communication devices has seen a tremendous growth. Within these gadgets, digital processing cores offer mobile computing power that was found in desktop computers a short time ago. RF circuitry allows communication at any location and networking on the fly. Yet, complexity is only *one* challenge. In a global competition, a short product development cycle is essential since only the first few suppliers in the market can benefit from attractive profit margins, whereas the followers might suffer financial loss in spite of good products. In this situation, it seems paradox that modern process technologies offer more performance potential than is actually used in the designs. The reason is that the design tools improve at a much slower pace than the process technology. This so-called design productivity gap widens although there are decent tools for digital synthesis readily available. Unfortunately, the situation is even worse in the analog domain because of the many degrees of freedom inherent to analog design. This discipline often becomes the bottleneck in the entire design process due to the lack of automatic tools. This problem is even going to intensify for "smart" systems, which feature analog and digital electronics in order to combine sensing, computing and acting.

Within analog design, the exploration and evaluation of different implementation alternatives in the conceptual stage is a particularly time-consuming task. Therefore, new design tools which focus on this particularly rewarding part of the entire design process have been presented. The suggested methods unfold their full potential in an environment where two methodological premises hold:

- *Simulation* is employed to analyze the circuit behavior. As an alternative, symbolic equations could be used. They can be evaluated very fast, but only yield approximate results. Moreover, symbolic equations partially require a laborious setup process and usually refer to DC and AC performances only. In contrast, simulation can be applied to any type of performance and yields accurate results. Finally, the use of a simulator reflects the industrial status quo. This facilitates the transition of the tools from the academic to the industrial environment.

- To keep the resource requirements in moderate bounds in spite of expensive circuit simulations, *deterministic* optimization algorithms are used, as opposed to stochastic ones. The latter are less likely to get stuck in local optima, but their resource requirements are much higher.

The need for performance space exploration arises naturally in the design process: For a given circuit topology and production technology, the question occurs what the ultimate performance capabilities of the circuit are. More precisely, the feasible range of performance values is sought. While most other publications on performance space exploration remain vague about what actually restricts the feasible performance space, this thesis shows that a systematic and comprehensive setup of sizing constraints is the basis of any successful performance space exploration. This setup relies on two cornerstones:

- A given circuit topology is analyzed hierarchically in a bottom-up fashion starting from the individual transistors. It was shown how the circuit topology can be represented by means of a topology tree. With the entire circuit as the root and the individual transistors as the leaves, each node of the tree represents an analog building block at a particular hierarchical level.
- A generic rule set captures fundamental circuit design knowledge. For each path from root to leaf the appropriate rules are instantiated, which results in a number of sizing constraints for the individual transistors. These constraints specify the DC operating point and ensure saturation, matching or symmetry conditions, among others.

Any particular circuit sizing result which satisfies the sizing constraints is in line with good design practice. Using simulation it can be mapped to a point in the feasible performance space. Consequently, the sizing constraints represent the basis of the entire performance space exploration process.

There is a trade-off between the accuracy of the exploration technique and the number of performances which can be analyzed at the same time. In this thesis, two exploration methods were presented which feature a combination of accuracy and efficiency well beyond the state of the art. The two approaches focus on different application scenarios:

- For the *accurate* exploration of *low-dimensional* performance spaces with typically two or three performances, a nonlinear method was derived. It incorporates the Normal-Boundary Intersection (NBI) algorithm and offers full simulator accuracy.
- For the *approximate* exploration of *high-dimensional* performance spaces with possibly ten or more performances, a linear method was developed. It relies on the Fourier-Motzkin elimination (FME) algorithm and operates on a linear description of the circuit behavior as obtained from a simulation-based sensitivity analysis.

The analog sizing process comprises of three main steps: topology selection or generation, circuit sizing, and layout generation. Out of these three steps, the two new performance space exploration methods support the first two as was demonstrated by a number application examples.

Accurate Low-Dimensional Performance Space Exploration via Normal-Boundary Intersection (NBI)

Methodology For a given unsized topology it is impossible to accurately describe the boundaries of its feasible performance space without actually having performed the sizing. That is the reason why NBI-based performance space exploration automatically performs multiple sizing runs with well-defined optimization criteria. In fact, the advantageous translation of the exploration problem into a sequence of optimization problems is the main contribution which differentiates the new method from the current state of the art.
To keep the exploration times within moderate bounds, only the two or three most important performances are selected, while fixed bounds can be imposed on the remaining performances. Furthermore, out of the entire boundary of the feasible performance space, only the trade-off region is identified. There, one performance can only be improved at the cost of the remaining performances. More precisely, this region is referred to as the Pareto front.
The suggested exploration procedure can be partitioned into two stages, which constitute the actual Normal-Boundary Intersection algorithm: First, those points in the performance space are identified where the different performances show their individual optima regardless of the remaining performances. The convex hull of these points represents a first approximation to the Pareto front. In the second stage, a successive refinement is carried out by line searches in the performance space. They emanate from evenly distributed points on the convex hull and proceed in a normal direction. Each such optimization yields one additional point on the Pareto front.
In contrast to other exploration techniques found in the literature, the NBI-based method yields well-discretized Pareto fronts and is applicable even to the nonconvex case.

Topology Selection Since the Pareto fronts represent the ultimate performance capabilities of a circuit, they can be used to compare different topologies and to select the most appropriate one for a given set of specifications. In addition, a visualization of the Pareto fronts can help the designers to obtain a more intuitive grasp of the circuit characteristics. It was exemplified that two-dimensional explorations take between 10 and 15 minutes on a single computer. For three-dimensional Pareto fronts the simulation times are considerably less than an hour.

Circuit Sizing The contribution of NBI-based performance space exploration to the circuit sizing process is two-fold:
First, the designer can determine the ultimate performance limits of a certain topology. This conveys a sense of how ambitious actual sizing targets are.
Second, the obstacles to a further performance improvement are identified. Thanks to the systematic setup of sizing constraints and the deterministic optimization algorithm, the designer gets comprehensive information on what constraints are active in the Pareto fronts and how stringent they are. From the design point of view, the active constraints highlight those parts of the design which need modification if extra performance has to be achieved.
Based on an elaborate example it was demonstrated how the presented nonlinear exploration method allows an in-depth design analysis which can be of great value in the circuit sizing process.

In essence, the novel NBI-based performance space exploration technique is a valuable enhancement of current interactive design environments because it is the first tool to combine the following features:

- Full simulator accuracy
- Moderate execution times
- Well-balanced discretization of Pareto front
- Applicability to nonconvex Pareto fronts
- Identification of performance-limiting sizing constraints and quantification of their stringency

Approximate High-Dimensional Performance Space Exploration via Fourier-Motzkin Elimination (FME)

Methodology The exploration of high-dimensional performance spaces based on time-consuming circuit sizing runs is clearly infeasible due to excessive resource requirements. In this thesis, it was demonstrated that for an approximate exploration actual sizing runs are not needed. Instead, a linearization of the circuit behavior allows the estimation of the feasible performance space.
Owing to the resulting efficiency, no restriction to a subset of performances is required. In addition, no focus on the Pareto front is needed. The feasible performance space is described as a closed region instead.
Conceptually, the approximate exploration method comprises of three steps: First, the sizing constraints are linearized based on circuit simulation. This yields a linear description of the feasible parameter space. Mathematically, a circuit realizes a map from parameters to performances. In the second procedural step, this map is linearized in the same way as the sizing constraints. In the third step, the image of the linearized feasible parameter space under this linear map is calculated. The result is a linear description of the feasible performance space. This last step is the algorithmic

core of the linear exploration algorithm and it usually requires the largest fraction of the entire computational effort. The calculation of the feasible performance space is done in two stages: An initial coordinate transformation maps the feasible parameter space into an intermediate space, which features the same number of dimensions as the parameter space, but comprises of all performances and an appropriate number of remaining parameters. A subsequent elimination of the last parameters using the Fourier-Motzkin elimination method yields the feasible performance space.
In contrast to similar algorithms in the literature, the new Fourier-Motzkin-based approach was specifically designed for linear performance space exploration. Hence, it meets all its requirements while at the same time offering a remarkable efficiency.

Topology Selection It was shown that in spite of inevitable linearization inaccuracies, the FME-based exploration method gives a good idea of the feasible performance spaces of different circuit topologies. This allows topology comparisons within runtimes of a few minutes. Of course, this approach can be used to obtain three-dimensional performance spaces for visual inspection in an interactive environment. However, it unfolds its true strength in combination with automatic tools, which perform topology selection or even hierarchical sizing in high-dimensional spaces.

Circuit Sizing With the help of a circuit example it was shown that the FME-based linear performance exploration method is a key enabler for hierarchical sizing. In this design style, which is mandatory for larger analog designs, the specifications of the entire circuit are successively broken down to specifications of the functional blocks until at the lowest level transistor widths and lengths are obtained.
Sizing constraints play a key role in flat circuit sizing because they restrict the optimization engine to technically meaningful parameter values. In a hierarchical circuit description, the entire circuit is partitioned into functional blocks which are described phenomenologically by behavioral models. The performances of the models can directly be adjusted by means of behavioral parameters. At this level of abstraction, a parameter value is technically meaningful if the corresponding block performance value can be achieved by the actual topology of the functional block. Hence, the feasible parameter space of a behavioral model is identical to the feasible performance space of the corresponding transistor implementation. That is the reason why performance space exploration yields sizing constraints at higher levels of abstraction. In this way, the top-down propagation of circuit specifications is complemented by a bottom-up propagation of physically motivated constraints. These constraints ensure technically meaningful circuit realizations even at higher levels of abstraction.
Since behavioral models often feature far more than just two or three parameters, a high-dimensional performance space exploration technique is of paramount importance to a successful hierarchical sizing. In exemplary fashion it was shown that iteration can successfully compensate for linearization inaccuracies.

In summary, the suggested FME-based performance space exploration method is unique within the literature because it offers the following features:

- Exploration of high-dimensional performance spaces at an outstanding efficiency
- Reasonable accuracy
- Compatibility with simulation-based design environments
- Key enabler for hierarchical sizing

Appendix A

Initial Sizing

Circuit sizing is a computationally expensive problem, especially when variations in the manufacturing process and in the operating conditions are also taken into account. For an increased efficiency, the sizing task has traditionally been divided into two steps as outlined in Section 1.1.2: Nominal design identifies a circuit sizing which meets the specifications under nominal (or fixed worst-case) conditions. Design centering then improves the result with regard to robustness. In this way, a sizing step with limited resource requirements provides a promising starting point for a subsequent improvement with tighter requirements and higher computational costs.

The fundamental importance of sizing constraints was discussed in Section 2.1.3: Any technically meaningful sizing has to satisfy these constraints. In fact, the traditional two-step sizing method can be extended by one more step called *initial sizing*. Disregarding any specifications, this step only focuses on the fulfillment of the sizing constraints. Table A.1[*] shows how the scope of the resulting three-step sizing procedure is gradually widened at the cost of increasing computational costs.

	Initial Sizing	Nominal Sizing	Design Centering
Computational Costs	low	moderate	high
Sizing Constraints Met	yes	yes	yes
Specifications Met at Nominal Conditions	no	yes	yes
Specifications Met Even With Variations	no	no	yes

Table A.1: Three-step Sizing Procedure

The result of the initial sizing step does not only provide a good starting point for a subsequent nominal sizing, but it also plays important roles in performance space exploration as will be shown later.

[*] A *"no"* entry means *"only by chance"*.

Especially for modern low-voltage designs, the feasible parameter space $\mathcal{P}$ can be extremely small. Thus, the mere identification of a sizing which meets all sizing constraints can be a challenging problem. This chapter discusses an initial sizing algorithm which efficiently and reliably solves this task.

A.1 Initial Sizing Algorithm

A.1.1 General Idea

The fulfillment of the sizing constraints is a prerequisite to any further performance optimization. In the case of violated functional sizing constraints, a circuit might not even exhibit the desired fundamental functionality (e.g. constant signal from an "oscillator"). Therefore, and for the sake of efficiency, the initial sizing step should be separated from the actual performance optimization. This section outlines a method to reliably calculate an initial sizing, denoted as $\mathbf{p}_s$.

Recall that the feasible parameter space $\mathcal{P}$ was defined by

$$\mathcal{P} = \{\mathbf{p}|\mathbf{c}(\mathbf{p}) \geq \mathbf{0}\}\,, \quad \mathbf{c}(\mathbf{p}) \in \mathbb{R}^q\,, \quad \mathbf{p} \in \mathbb{R}^m\,. \tag{2.14}$$

Obviously, $\mathbf{p}_s \in \mathcal{P}$ is a necessary condition. Furthermore, it is beneficial to choose a point "in the center" of $\mathcal{P}$ for two reasons: First, in this point all sizing constraints are satisfied with maximum safety margins. This means that all the parameter values in the neighborhood of $\mathbf{p}_s$ will be feasible. Second, the performances are only weakly nonlinear when all sizing constraints are safely satisfied, as shown in [GZEA01].

Within the area of performance space exploration, the initial sizing result is useful in various contexts:

- *Normalization*:
  The initial sizing $\mathbf{p}_s$, the corresponding constraint values $\mathbf{c}(\mathbf{p}_s)$, and the performance values $\mathbf{f}(\mathbf{p}_s)$ can be used as reference values for numeric normalization, cf. Section A.2.1.
- *Optimization*:
  The parameter vector $\mathbf{p}_s$ is a beneficial optimization starting point, especially for deterministic techniques as used in the Normal-Boundary Intersection algorithm, cf. Section A.2.2.
- *Linearization*:
  A linearization of the circuit behavior in $\mathbf{p}_s$ yields a good approximation accuracy, cf. Section A.2.3. This is especially important to high-dimensional linear performance space exploration.

A.1.2 Geometric Illustration

Finding the center of the feasible parameter space $\mathcal{P}$ is a nonlinear problem, which cannot be solved directly. After all, the sizing constraints only provide an implicit description of $\mathcal{P}$ and can only be evaluated pointwise using simulation. Yet, this problem can be solved by an iterative numerical optimization process, which relies on approximations to $\mathcal{P}$. In the neighborhood of a particular sizing vector $\mathbf{p}^{(0)}$, the function $\mathbf{c}(\mathbf{p})$ can be approximated by a linear Taylor expansion:

$$\mathbf{c}(\mathbf{p}) \approx \frac{\partial \mathbf{c}(\mathbf{p})}{\partial \mathbf{p}}\bigg|_{\mathbf{p}^{(0)}} \cdot (\mathbf{p} - \mathbf{p}^{(0)}) + \mathbf{c}(\mathbf{p}^{(0)}) =: \mathbf{C}^{(0)} \cdot \Delta\mathbf{p}^{(0)} + \mathbf{c}^{(0)} . \tag{A.1}$$

This yields the following linear approximation to (2.14):

$$\overline{\mathcal{P}}^{(0)} = \{\mathbf{p}^{(0)} + \Delta\mathbf{p}^{(0)} \mid \mathbf{C}^{(0)} \cdot \Delta\mathbf{p}^{(0)} + \mathbf{c}^{(0)} \geq \mathbf{0}\} . \tag{A.2}$$

The Jacobian matrix $\mathbf{C}^{(0)} \in \mathbb{R}^q \times \mathbb{R}^m$ contains the sensitivities of the sizing constraints with respect to the transistor parameters. It can be approximated by finite differences from a number of quick DC circuit simulations or based on the adjoint method. Geometrically, (A.2) describes a polytope in the parameter space [Zie95].

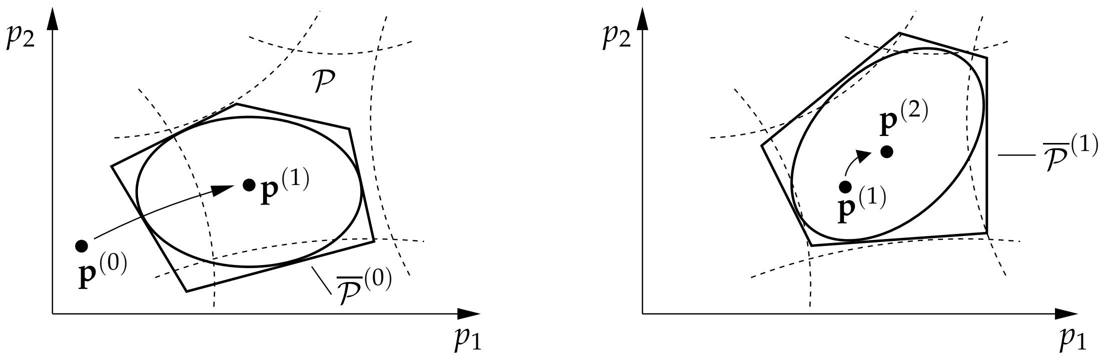


Figure A.1: Iteratively Finding Center Point Using Ellipsoids

Figure A.1 illustrates the basic idea behind the suggested algorithm. The dotted lines indicate the bounding hypersurfaces of $\mathcal{P}$. The linearization at $\mathbf{p}^{(0)}$ according to (A.2) yields the polytope $\overline{\mathcal{P}}^{(0)}$. The further the point $\mathbf{p}^{(0)}$ is outside $\mathcal{P}$, the worse $\overline{\mathcal{P}}^{(0)}$ usually approximates $\mathcal{P}$.

Computational geometry provides algorithms to determine an ellipsoid with maximum volume which is contained in a given polytope [BGT81, ZG03]. It can be seen from Figure A.1, left, that the center $\mathbf{p}^{(1)}$ of such an ellipsoid roughly lies in the middle of the polytope $\overline{\mathcal{P}}^{(0)}$. In addition, $\mathbf{p}^{(1)}$ moves closer to the actual center of $\mathcal{P}$.

As shown in Figure A.1, right, a new linearization at $\mathbf{p}^{(1)}$ yields a better estimate $\overline{\mathcal{P}}^{(1)}$ of $\mathcal{P}$. Newly inscribing a maximum volume ellipsoid yields a further improved center point estimate $\mathbf{p}^{(2)}$. This procedure is repeated until point $\mathbf{p}^{(j+1)}$ lies close to $\mathbf{p}^{(j)}$. Then, the sought initial sizing is $\mathbf{p}_s = \mathbf{p}^{(j+1)}$.

A.1.3 Overview of Algorithm

```
choose $\mathbf{p}^{(0)}$ with $\mathbf{p}_{\min} \le \mathbf{p}^{(0)} \le \mathbf{p}_{\max}$, and let $j = -1$    (I)
$\mathbf{c}^{(0)} = \mathbf{c}(\mathbf{p}^{(0)})$ from simulation    (II)
    increase $j$ by 1    (III)
    $\mathbf{C}^{(j)} = \mathbf{C}(\mathbf{p}^{(j)})$ from finite differences, DC simulation    (IV)
    $\mathbf{p}^{(j+1)} = \mathbf{p}^{(j+1)}(\mathbf{C}^{(j)}, \mathbf{c}^{(j)})$ via Ellipsoidal Update (cf. Sec. A.1.4)    (V)
    $\mathbf{c}^{(j+1)} = \mathbf{c}(\mathbf{p}^{(j+1)})$ from simulation    (VI)
until $\|\mathbf{p}^{(j+1)} - \mathbf{p}^{(j)}\| \le \epsilon \wedge \mathbf{c}^{(j+1)} \ge \mathbf{0}$    (VII)
$\mathbf{p}_s = \mathbf{p}^{(j+1)}$    (VIII)
```

Figure A.2: Overview of Initial Sizing Algorithm

Figure A.2 gives an overview of the algorithm. It requires hardly any a priori knowledge but lower and upper parameter bounds, which, however, should be known for any sizing problem. As an initial approximation to the center point, an arbitrary sizing $\mathbf{p}^{(0)}$ can be chosen which lies within the parameter bounds $\mathbf{p}_{\min}$ and $\mathbf{p}_{\max}$ and which has DC convergence.

$\mathbf{C}^{(j)}$ and $\mathbf{c}^{(j)}$ are obtained in a loop from simulations at the current point $\mathbf{p}^{(j)}$, and a new center $\mathbf{p}^{(j+1)}$ is calculated until convergence is detected. For termination, all sizing constraints have to be satisfied and the distance between the center point approximations $\|\mathbf{p}^{(j+1)} - \mathbf{p}^{(j)}\|$ has to be sufficiently small. This distance is measured using the Euclidean norm and an appropriate normalization of the parameter vectors. In most practical cases, the value of $\|\mathbf{p}^{(j+1)} - \mathbf{p}^{(j)}\|$ decreases monotonically. While no strict proof of convergence can be given, extremely reliable convergence behavior was observed for practical circuits, cf. Section A.1.6.2.

A.1.4 Determination of New Linearization Point via Ellipsoidal Update

The determination of the new linearization point $\mathbf{p}^{(j+1)}$ in line (V) of Figure A.2 is the critical step in the entire algorithm. For this purpose, a robust ellipsoidal update procedure was developed. The MVE algorithm which is used to calculate the maximum inscribed ellipsoid is introduced in the following subsection. The entire update procedure which embeds the MVE algorithm is discussed afterwards.

A.1.4.1 MVE Algorithm

Originally developed for linear programming, the ellipsoid algorithm according to Khachiyan [Kha79] can be used to find a maximum volume ellipsoid inscribed in a polytope [AMH91, BGT81]. For this particular application, however, the algorithm is not very efficient. Recently, a much superior maximum volume ellipsoid (MVE) algorithm was published that aims at practical performance rather than theoretical properties [ZG03]. It owes its remarkable performance to two corner stones: First, it is based on an advantageous mathematical formulation of the underlying geometric idea. Second, the resulting problem was partially solved symbolically. Hence, at runtime, only a simplified problem has to be addressed numerically. The key ideas of this algorithm are summarized in the following paragraphs.

Given a center point $\mathbf{p}_e \in \mathbb{R}^m$ and a symmetric, positive definite matrix $\mathbf{E} \in \mathbb{R}^m \times \mathbb{R}^m$, an ellipsoid $\mathcal{E}$ is uniquely defined by

$$\mathcal{E}(\mathbf{p}_e, \mathbf{E}) = \{\mathbf{p} \mid \ \mathbf{p} = \mathbf{p}_e + \mathbf{E} \cdot \mathbf{v} \ \ \wedge \ \ \|\mathbf{v}\| \le 1\}\,. \tag{A.3}$$

Geometrically, the ellipsoid is the image of a unit ball under the linear map $\mathbf{E}$ with its center point shifted to $\mathbf{p}_e$.

With $\Delta\mathbf{p}_e^{(j)} = \mathbf{p}_e^{(j)} - \mathbf{p}^{(j)}$, the ellipsoid $\mathcal{E}^{(j)}$ is inscribed in $\overline{\mathcal{P}}^{(j)}$, i.e. $\mathcal{E}^{(j)} \subset \overline{\mathcal{P}}^{(j)}$, if and only if

$$\forall_{i \in \{1\ldots q\}} \ \inf_{\|\mathbf{v}\|=1} \left(\mathbf{c}_i^{(j)\,T} \cdot (\Delta\mathbf{p}_e^{(j)} + \mathbf{E}^{(j)} \cdot \mathbf{v}) + c_i^{(j)}\right) \ge 0 \tag{A.4}$$

$$\Leftrightarrow \quad \forall_{i \in \{1\ldots q\}} \ \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e^{(j)} - \|\mathbf{c}_i^{(j)\,T} \cdot \mathbf{E}^{(j)}\| + c_i^{(j)} \ge 0\,. \tag{A.5}$$

Here, $\mathbf{c}_i^{(j)\,T}$ is the i^{th} row of $\mathbf{C}^{(j)}$ and $c_i^{(j)}$ is the i^{th} entry of $\mathbf{c}^{(j)}$. The geometric idea behind (A.4) is that for a containment check it is sufficient to examine the boundary points which are characterized by $\|\mathbf{v}\| = 1$. The infimum operator identifies the greatest lower bound of the bracketed expression:

$$\begin{aligned}
& \inf_{\|\mathbf{v}\|=1} \left(\mathbf{c}_i^{(j)\,T} \cdot (\Delta\mathbf{p}_e^{(j)} + \mathbf{E}^{(j)} \cdot \mathbf{v}) + c_i^{(j)}\right) && \text{(A.6)}\\
= \ & \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e^{(j)} + c_i^{(j)} + \inf_{\|\mathbf{v}\|=1} \left(\mathbf{c}_i^{(j)\,T} \cdot \mathbf{E}^{(j)} \cdot \mathbf{v}\right) && \text{(A.7)}\\
= \ & \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e^{(j)} + c_i^{(j)} - \mathbf{c}_i^{(j)\,T} \cdot \mathbf{E}^{(j)\,T} \cdot \frac{\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)}}{\|\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)}\|}\,; \quad \text{note: } \mathbf{E}^{(j)} = \mathbf{E}^{(j)\,T} && \text{(A.8)}\\
= \ & \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e^{(j)} + c_i^{(j)} - \frac{(\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)})^T \cdot (\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)})}{\|\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)}\|} && \text{(A.9)}\\
= \ & \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e^{(j)} + c_i^{(j)} - \|\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)}\| && \text{(A.10)}
\end{aligned}$$

The main idea in (A.8) is that the infimum occurs when $\mathbf{v}$ and $\mathbf{E}^{(j)} \cdot \mathbf{c}_i^{(j)}$ point in the opposite direction.

If V_b is the volume of the n-dimensional unit ball, then the Ellipsoid $\mathcal{E}$ from (A.3) has the volume [BV04]

$$V_e = \det(\mathbf{E}) \cdot V_b \, . \tag{A.11}$$

Therefore, $\det(\mathbf{E})$ could be used as an objective function for the maximization of the ellipsoid volume. Yet, using the logarithm of the determinant, (A.5) and (A.11) yield the following *convex* optimization problem [BV04, Hin04]:

$$\begin{aligned} [\Delta\mathbf{p}_e^{(j)} \;\; \mathbf{E}^{(j)}] &= \underset{[\Delta\mathbf{p}_e \;\; \mathbf{E}]}{\operatorname{argmax}} \;\; \log\, \det(\mathbf{E}) \\ &\text{s. t.} \quad \underset{i \in \{1 \dots q\}}{\forall} \; \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}_e - \|\mathbf{c}_i^{(j)\,T} \cdot \mathbf{E}\| + c_i^{(j)} \geq 0 \, . \end{aligned} \tag{A.12}$$

Here, the notation $[\Delta\mathbf{p}_e^{(j)} \;\; \mathbf{E}^{(j)}]$ denotes a matrix consisting of $\Delta\mathbf{p}_e^{(j)}$ and $\mathbf{E}^{(j)}$. The argmax operator yields the argument leading to the maximum objective value.

The ellipsoid center

$$\mathbf{p}_e^{(j)} = \mathbf{p}^{(j)} + \Delta\mathbf{p}_e^{(j)} \tag{A.13}$$

is used as an improved linearization point in the next iteration of the initial sizing algorithm:

$$\mathbf{p}^{(j+1)} = \mathbf{p}_e^{(j)} \, . \tag{A.14}$$

The matrix $\mathbf{E}^{(j)}$ is not needed for initial sizing. Yet, it could be used to estimate the shape and the orientation of $\mathcal{P}^{(j)}$ [AMH91][†].

It is possible to directly solve problem (A.12) numerically. Yet, in [ZG03] it was shown how efficiency can be improved by a far-reaching symbolic simplification: Setting up the Karush-Kuhn-Tucker (KKT) conditions for (A.12) yields $m^2 + m + 2q$ single equations. With clever transformations, m^2 equations can be eliminated symbolically. In the MVE algorithm, the thus simplified KKT conditions are solved numerically using a primal-dual algorithm. This approach involves iteratively finding the roots of the perturbed KKT conditions [NW99]. A symbolic transformation of the resulting system of equations to triangular shape further improves the performance because, at runtime, the roots can be found by simple back substitution.

Note that these symbolic simplification steps were done manually only once. The resulting algorithm is purely numeric in nature.

[†] It was stated in Section 5.3.2.3 that so far no reliable criteria have been available to estimate the computational effort of a given Fourier-Motzkin elimination problem. The MVE algorithm might be useful in that context: Recall that the equation-based parameter replacement results in a polytope according to (4.17). Using Fourier-Motzkin elimination, the remaining parameters are eliminated from the system of inequalities. The MVE algorithm could be applied to the polytope from (4.17). The ellipsoid principal axes and the condition of $\mathbf{E}$ might play an important role in estimating the difficulty of the given elimination problem.

A.1.4.2 Robust Ellipsoidal Update Algorithm

The MVE algorithm requires a starting point $\Delta\mathbf{p}_a^{(j)}$ which is located inside the polytope, i.e. $\mathbf{C}^{(j)} \cdot \Delta\mathbf{p}_a^{(j)} + \mathbf{c}^{(j)} > \mathbf{0}$. This point can be obtained by solving the following auxiliary linear programming (LP) problem:

$$[z_a^{(j)} \quad \Delta\mathbf{p}_a^{(j)}] = \underset{[z \;\; \Delta\mathbf{p}]}{\text{argmax}} \; z \quad \text{s.t.} \quad \mathbf{C}^{(j)} \cdot \Delta\mathbf{p} + \mathbf{c}^{(j)} \geq \mathbf{1} \cdot z\,. \tag{A.15}$$

In (A.15), the vector $\mathbf{1} \in \mathbb{R}^q$ consists of all ones. It is obvious that the constraints can always be satisfied by choosing z sufficiently small, i.e. negative and with a large absolute value. The parameter $z = 0$ yields the original constraints. Thus, increasing z as required by the argmax operator successively tightens the constraints. Depending on the resulting value of $z_a^{(j)}$, two cases can be distinguished:

Case 1, $z_a^{(j)} > 0$: An interior point exists if and only if $z_a^{(j)} > 0$. Then, $\Delta\mathbf{p}_a^{(j)}$ is the required starting point for the MVE algorithm which yields $\mathbf{p}^{(j+1)}$ according to (A.12)–(A.14).

Case 2, $z_a^{(j)} \leq 0$: In this case, $\overline{\mathcal{P}}^{(j)}$ represents an empty polytope. However, any meaningful circuit topology under reasonable operating conditions has a non-empty feasible parameter space $\mathcal{P}$. The reason for this discrepancy is that for remote linearization points, $\overline{\mathcal{P}}^{(j)}$ as given by $\mathbf{C}^{(j)}$ and $\mathbf{c}^{(j)}$ often does not approximate the feasible parameter space $\mathcal{P}$ well. Especially in the early stages of the initial sizing process, empty polytopes $\overline{\mathcal{P}}^{(j)}$ occur frequently. That is the reason why an update strategy was developed that reliably deals with this problem.

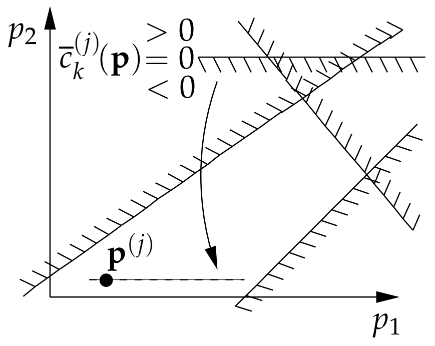
If (A.15) cannot find an interior point, then the linearization point $\mathbf{p}^{(j)}$ is most likely too far outside $\mathcal{P}$, resulting in a poor approximation $\overline{\mathcal{P}}^{(j)}$. In this case, a point $\mathbf{p}^{(j+1)}$ closer to $\mathcal{P}$ is sought: First, a suitable search direction emanating from $\mathbf{p}^{(j)}$ is identified and second, an appropriate step length is estimated. A subsequent simulation-based line search yields $\mathbf{p}^{(j+1)}$.

Search direction: As before, the rows of the Jacobian Matrix $\mathbf{C}^{(j)}$ are denoted as $\mathbf{c}_i^{(j)\,T}$. They contain the gradients of the sizing constraints with respect to $\mathbf{p}^{(j)}$. For the i th sizing constraint, the gradient at $\mathbf{p}^{(j)}$ is

$$\nabla c_i(\mathbf{p})|_{\mathbf{p}^{(j)}} = \mathbf{c}_i^{(j)\,T}\,. \tag{A.16}$$

Since $\mathbf{p} = \mathbf{p}^{(j)}$ means $\Delta\mathbf{p}^{(j)} = 0$, the sizing constraint i is violated at this point if $c_i^{(j)} = c_i(\mathbf{p}^{(j)}) < 0$. Let $V^{(j)} \subseteq \{1 \ldots q\}$ comprise of the indices of the violated sizing constraints in $\mathbf{p}^{(j)}$. Then, a step $\Delta\mathbf{p}^{(j)}$ reduces the violations if

$$\underset{i \in V^{(j)}}{\forall} \; \mathbf{c}_i^{(j)\,T} \cdot \Delta\mathbf{p}^{(j)} > 0\,. \tag{A.17}$$



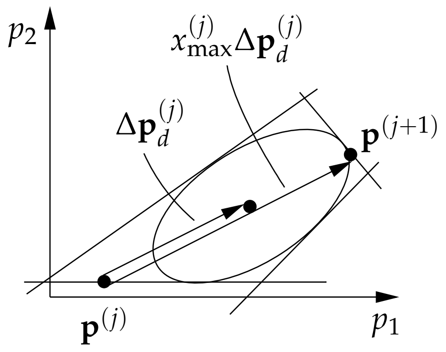

p_2
$x_{\max}^{(j)} \Delta \mathbf{p}_d^{(j)}$
$\Delta \mathbf{p}_d^{(j)}$
$\mathbf{p}^{(j+1)}$
$\mathbf{p}^{(j)}$
p_1

Figure A.3: Polytope Relaxation

Of course, no remaining sizing constraint may be violated:

$$\forall_{i \in \{1 \dots q\} \setminus V^{(j)}} \quad \mathbf{c}_i^{(j)\,T} \cdot \Delta \mathbf{p}^{(j)} + c_i^{(j)} > 0\,. \tag{A.18}$$

Here, the operator \ denotes the set difference. A combination of (A.17) and (A.18) yields

$$\mathbf{C}^{(j)} \cdot \Delta \mathbf{p}^{(j)} + \tilde{\mathbf{c}}^{(j)} > \mathbf{0}\,, \quad \tilde{c}_i^{(j)} = \begin{cases} 0\,, & i \in V^{(j)} \\ c_i^{(j)}\,, & i \in \{1 \dots q\} \setminus V^{(j)}\,. \end{cases} \tag{A.19}$$

From a different point of view, and with $\geq$ instead of $>$, (A.19) can be interpreted as a relaxed version of the original polytope $\overline{\mathcal{P}}^{(j)}$, where the point $\mathbf{p}^{(j)}$ has been made feasible by moving the boundaries of the violated sizing constraints appropriately.

Figure A.3, left, illustrates this interpretation. Four linearized sizing constraints are shown. The hatchings mark the side of the boundaries, where

$$\bar{c}_i^{(j)}(\mathbf{p}) := \mathbf{c}_i^{(j)\,T} \cdot \Delta \mathbf{p}^{(j)} + c_i^{(j)} < 0\,. \tag{A.20}$$

In this example, no point exists that satisfies all sizing constraints. In $\mathbf{p}^{(j)}$, constraint k is violated. Geometrically, modifying the constant $c_k^{(j)}$ means a parallel shift of the associated boundary. Setting $c_k^{(j)} = 0$ forces the boundary to go through the linearization point $\mathbf{p}^{(j)}$, which results in a non-empty relaxed polytope.

As can be seen from Figure A.3, right, the center of this polytope indicates a suitable search direction $\Delta \mathbf{p}_d^{(j)}$ because it provides an improvement in the violated sizing constraint while staying away from the remaining boundaries. Therefore, $\Delta \mathbf{p}_d^{(j)}$ is determined based on (A.19) using the MVE algorithm.

Step length: The point $\mathbf{p}^{(j)} + \Delta \mathbf{p}_d^{(j)}$ could be used as a new linearization point $\mathbf{p}^{(j+1)}$. Yet, the efficiency of the algorithm can further be improved if the search direction $\Delta \mathbf{p}_d^{(j)}$ is maintained and a maximum step length $x_{\max}^{(j)}$ is calculated according to

$$x_{\max}^{(j)} = \max_x x \quad \text{s.t.} \quad \mathbf{C}^{(j)} \cdot \Delta \mathbf{p}_d^{(j)} \cdot x + \tilde{\mathbf{c}}^{(j)} \geq \mathbf{0}\,. \tag{A.21}$$

Note that (A.21) refers to the relaxed polytope, which is reflected by the modified constraint vector $\tilde{\mathbf{c}}^{(j)}$ from (A.19). In the linear approximation, the step $x_{\max}^{(j)} \cdot \Delta\mathbf{p}_d^{(j)}$ yields large reductions of the sizing constraint violations without newly infringing any remaining sizing constraints (cf. Figure A.3, right).

In awareness of the limitations of the linear model, $x_{\max}^{(j)}$ can merely be used as an estimate. The actual step length is determined by a simulation-based line search in the direction of $\Delta\mathbf{p}_d^{(j)}$: To find a point which reduces the actual constraint violations, all constraints are simulated at

$$\mathbf{p} = \mathbf{p}^{(j)} + x^{(j)} \cdot \Delta\mathbf{p}_d^{(j)}, \quad x_{\max}^{(j)} \geq x^{(j)} > 0 \tag{A.22}$$

for different values of $x^{(j)}$. Initially, $x^{(j)} = x_{\max}^{(j)}$. The step length $x^{(j)}$ is gradually decreased until a new linearization point $\mathbf{p}^{(j+1)} = \mathbf{p}^{(j)} + x^{(j)} \cdot \Delta\mathbf{p}_d^{(j)}$ has been found which satisfies

$$\underbrace{\left(|V^{(j+1)}| < |V^{(j)}|\right)}_{(I)} \vee \underbrace{\left(|V^{(j+1)}| = |V^{(j)}| \quad \wedge \sum_{i \in V^{(j+1)}} (c_i^{(j+1)})^2 < \sum_{i \in V^{(j)}} (c_i^j)^2\right)}_{(II)}. \tag{A.23}$$

Recall that the set $V^{(j+1)}$ contains the indices of the violated constraints in $\mathbf{p}^{(j+1)}$ with $c_i^{(j+1)} < 0$. The operator $|\cdot|$ returns the cardinality of a set.

This line search strategy ensures that the new linearization point $\mathbf{p}^{(j+1)}$ combines a large step length with reduced constraint violations as specified by (A.23): Criterion (I) means that the number of violated constraints is reduced. Criterion (II) describes the case where the number of violated constraints remains unchanged, but the actual values of the constraint violations are diminished. The performance of this strategy is demonstrated in Section A.1.6.2.

A.1.5 Comparison to Ellipsoid-Based Design Centering

The presented initial sizing algorithm bears some resemblance to geometric design centering approaches. These techniques seek the center of the feasible parameter space as given by *explicit* performance specifications. Deterministic [AMH91, AMHH99, DH77, SVP98, WV93] and stochastic [Kje91, KT81] methods were suggested. Both rely on ellipsoidal approximation.

The new technique can be interpreted as a new solution method for ellipsoid-based geometric design centering. It has the following characteristics:

- *Arbitrary starting point* $\mathbf{p}^{(0)}$:
  Deterministic design centering procedures usually require some a priori knowledge of the location of the feasible space. Even stochastic techniques [Kje91,KT81] can have difficulties in finding feasible designs due to the small size of the feasible parameter space (c.f. Section A.1.6.1). No such information is required for the new initial sizing method. In fact, an efficient and robust convergence to the center of the feasible parameter space from a remote starting point with numerous heavily violated sizing constraints is an emphasis of the algorithm.
- *Efficient approximation of the feasible parameter space* $\mathcal{P}$:
  To identify the boundaries of the feasible parameter space, most deterministic design centering techniques identify a number of boundary points either by nonlinear optimization [SVP98, WV93] or a multitude of line searches [AMH91,AMHH99,DH77]. These strategies are not favorable considering a high-dimensional parameter space and a large number of sizing constraints. An increased efficiency results from the determination of a linear approximation to the feasible parameter space by *simultaneous linearization* of all sizing constraints *in one point*. That way, the time-consuming identification of boundary points can be avoided. While most design centering techniques aim at accuracy, initial sizing aims at efficiency.
- *Advanced solution to the maximum volume ellipsoid problem*:
  The suggested algorithm takes advantage of the new MVE algorithm, as opposed to [AMH91, AMHH99], where the traditional ellipsoid method is used. Experimental results show that this results in a speedup of almost two orders of magnitude.

A limitation of this approach is that for nonconvex feasible spaces there might not be a unique center point. In fact, this limitation applies to all the ellipsoid-based design centering techniques. Practical experience, however, suggests that this concern is primarily of academic interest.

A.1.6 Experiments

The experimental results in the following Sections A.1.6.1–A.1.6.3 were obtained from the folded cascode and the Miller compensated operational amplifier as depicted in Figure 5.1, Page 63.

A.1.6.1 Smallness of Feasible Parameter Space

As indicated earlier, only a small fraction of the entire parameter space yields legal sizings. For each amplifier, a number of 10,000 samples $\mathbf{p}$ with $\mathbf{p}_{\min} \leq \mathbf{p} \leq \mathbf{p}_{\max}$ was generated and simulated. For the folded cascode amplifier *none* of the samples satisfied all sizing constraints, while for the Miller amplifier only 182 of them were feasible.

A.1.6.2 Performance of Initial Sizing Algorithm

The initial sizing procedure has to be carried out only once for a given circuit topology in a certain technology. Yet, in order to demonstrate the reliability of the initial sizing algorithm, 100 random samples from within the lower and upper parameter limits were chosen and each of them was used as a starting point $\mathbf{p}^{(0)}$ for one initial sizing run.

Circuit	# Failures	Average # Steps	Average # DC simulations
folded cascode	0	6.0	74.5
Miller	0	3.3	30.1

Table A.2: Performance Statistics for 100 Initial Sizing Runs

Table A.2 gives some overall performance statistics. First of all, there was not a single failure, which demonstrates that the algorithm reliably converges from practically any starting point. Moreover, the algorithm used only little computing resources. On an average it took about 6.0 iteration steps for the folded cascode architecture, resulting in 74.5 quick DC simulations which can readily be parallelized. Reflecting the simpler architecture, only 3.3 steps with 30.1 simulations were sufficient for the Miller operational amplifier. These results are remarkable considering how small the feasible parameter spaces are.

Circuit	# Runs With $z_m^{(j)} \leq 0$	Average # Line Searches
folded cascode	97	2.2
Miller	23	0.3

Table A.3: Handling of Poor Linearizations (100 Initial Sizing Runs)

In the ellipsoidal update algorithm, special emphasis was put on the case where the linearizations yield empty polytopes, i.e. $z_a^{(j)} \leq 0$, cf. Section A.1.4.2, Case 2. The first column in Table A.3 shows that, in practice, there is a substantial number of sizing runs which have to deal with poor linear approximations in one or more steps. In this experiment with 100 sizing runs, this situation arose in 97 and 23 cases, respectively. Therefore, the reliable handling of this case is crucial to a dependable initial sizing algorithm.

Recall that for empty polytopes a line search is done to achieve a good reduction of the sizing constraint violations. Experience shows that steps which are based on line searches only occur at the beginning of the initial sizing procedure. Once a good linearization describing a non-empty polytope has been found, a small number of ellipsoidal steps completes the centering. For the folded cascode amplifier the first 2.2 steps out of 6.0 were on an average carried out based on line searches rather than direct ellipsoidal centering. It may be surprising that there were only 0.3 steps with line searches out of 3.3 for the Miller amplifier. This can be put into perspective from a different point of view: If the average number of line searches is related to the fraction of sizings that required line searches at all, it turns out that the folded cascode needed 2.2/(97/100)=2.3 line search steps and the Miller required 0.3/(23/100)=1.3 of them.

Circuit	Average # Steps	Average # DC Simulations
folded cascode	8.3	100.1
Miller	3.4	30.5

Table A.4: Performance Statistics With Line Search Disabled

The discussion above showed that the line search procedure is an integral part of the initial sizing algorithm. In Section A.1.4.2, it was argued that for Case 2 performing an actual line search instead of simply solving the relaxed problem yields significant speedups. To support this claim, the line search feature was disabled temporarily. Then, the algorithm was rerun using the same starting points as for the previous experiment. Table A.4 shows the results. The fact that the Miller amplifier is comparatively well-natured explains the minimal increase in computational cost for this circuit. In contrast, for the folded cascode architecture, the number of steps increased by 38% and the number of simulations by 34%. This indicates that especially for challenging circuits the line search technique yields a significant performance improvement.

A.1.6.3 Performance of MVE Algorithm

Algorithm	Number of Ellipsoidal Iterations in Step 1	Step 2	Step 3	Step 4	Total Duration
MVE	19	25	20	19	13 sec
KE	17510	17584	18152	17463	17 min 27 sec

Table A.5: Performance Comparison

As mentioned in Section A.1.4.1, under the constraints of initial sizing the new MVE algorithm [ZG03] is superior to alternative algorithms such as the original ellipsoid algorithm due to Khachiyan (KE) [AMH91, BGT81]. As a demonstration, an initial sizing run of the folded cascode amplifier with four iteration steps was examined. In each step, the two algorithms mentioned above, which are iterative themselves, calculated the ellipsoid centers. Table A.5 shows the results obtained on a cluster of 800 MHz Pentium III machines. It is striking that the MVE algorithm is almost two orders of magnitude faster than KE. Considering that the numerical circuit simulations only took additional 18 seconds in this example, the necessity of a high-performance ellipsoid algorithm becomes obvious.

A.2 Relevance to Performance Space Exploration

The result of the initial sizing process plays important roles in the context of the performance space exploration techniques presented in this thesis.

A.2.1 Normalization

Normalization is of great importance to numerical algorithms. Obviously, these algorithms only deal with *numbers*, usually in floating-point format. Yet, as in the case of circuit sizing, the parameter and performance values are products of numerical values and physical units. Of course, the physical nature of these values is ignored in the computations. This can be interpreted as normalization mathematically. Additionally, the numerical values of the parameters and performances often differ by many orders of magnitude. Owing to the limited accuracy of floating-point computations, this may lead to severe numerical problems which can easily render a computation result worthless. Hence, normalization has to deal with both the physical units and the numerical values. In the actual implementation of the presented exploration methods, all the input data was normalized as stated below. In the discussion of the algorithms this detail was skipped for simplicity.

Parameters Initial sizing identifies a parameter vector $\mathbf{p}_s$ in the center of the feasible parameter space. This vector was used as a reference value and all parameter vectors were normalized according to

$$\tilde{\mathbf{p}} = \frac{\mathbf{p} - \mathbf{p}_s}{\mathbf{p}_s} \,. \tag{A.24}$$

This maps the initial sizing result to the center of the normalized parameter coordinate system. Additionally, this normalization makes sure that the occurring parameter values are roughly in the same order of magnitude.

Performances A similar reasoning yields the following normalization of the performances:

$$\tilde{\mathbf{f}} = \frac{\mathbf{f} - \mathbf{f}_s}{\mathbf{f}_s} \,, \quad \mathbf{f}_s = \mathbf{f}(\mathbf{p}_s) \,. \tag{A.25}$$

Constraints The initial sizing $\mathbf{p}_s$ satisfies all sizing constraints with maximum safety margins. Therefore,

$$\tilde{\mathbf{c}} = \frac{\mathbf{c}}{\mathbf{c}_s} \,, \quad \mathbf{c}_s = \mathbf{c}(\mathbf{p}_s) \,, \tag{A.26}$$

leads to reasonably scaled constraint values. In contrast to (A.24) and (A.25), there is no shift in order to retain the convention that a sizing constraint is satisfied if and only if its value is nonnegative.

A.2.2 Optimization Starting Point

The initial sizing $\mathbf{p}_s$ is a beneficial starting point for subsequent optimization steps. Starting from this point, a performance optimization algorithm can choose any search direction without violating the sizing constraints early. Thus, $\mathbf{p}_s$ is an unbiased optimization starting point. Besides, the circuit behavior is usually only weakly nonlinear in $\mathbf{p}_s$, which is particularly beneficial for gradient-based deterministic optimization techniques. Therefore, $\mathbf{p}_s$ is used to initialize the Normal-Boundary Intersection algorithm presented in Section 3.2.

To illustrate the impact of the starting point on the optimization process, three different optimization strategies were applied to an industrial bandgap amplifier with 30 transistors, 107 inequality constraints, and 15 designable parameters. Lower bounds were specified for the slew rate and the gain, among others. Upper bounds were given for the power consumption and the overshoot, for example. Each optimization started from the same set of parameter values which were located outside of $\mathcal{P}$. The results, which are in line with experiences gained from other circuits, are summarized in Table A.6.

Optimization Strategy	# Simulations
gradient-based	— (failed)
combined evolutionary / gradient-based	7444
initial sizing / gradient-based	596

Table A.6: Performances of Different Optimization Strategies

A purely gradient-based optimization failed due to convergence problems. This is not surprising because the circuit performances tend to become heavily nonlinear as soon as the sizing constraints are violated. In such a situation one might resort to a stochastic technique because it does not require an explicit starting point. In this experiment, an evolutionary technique was applied to the problem. Upon termination of the this algorithm, a few gradient-based optimization steps quickly identified the final optimization result. This approach "implicitly" found its way into the feasible parameter space and generated a good result after 7444 simulations. Yet, the combination of the initial sizing procedure with the gradient-based optimization algorithm terminated successfully after only 596 simulations with a comparable quality of the result.

This experiment suggests two conclusions: In comparison to stochastic techniques, the dedicated initial sizing algorithm shows a much better efficiency in finding the feasible parameter space. Furthermore, it is evident that a starting point centered in the feasible parameter space is well-suited to initialize (especially gradient-based) optimization procedures. Considering that nonlinear deterministic optimization techniques gather higher-order information by iteratively evaluating first-order data, it is obvious that they benefit greatly from the good linearization quality at the initial sizing.

A.2.3 Linearization Quality

The accuracy of a linear approximation to a nonlinear function is best in the neighborhood of the linearization point. If a linear approximation to the entire feasible performance space $\mathcal{F}$ is sought and no further a priori knowledge is available, then the initial sizing $\mathbf{p}_s$ is a beneficial linearization point.

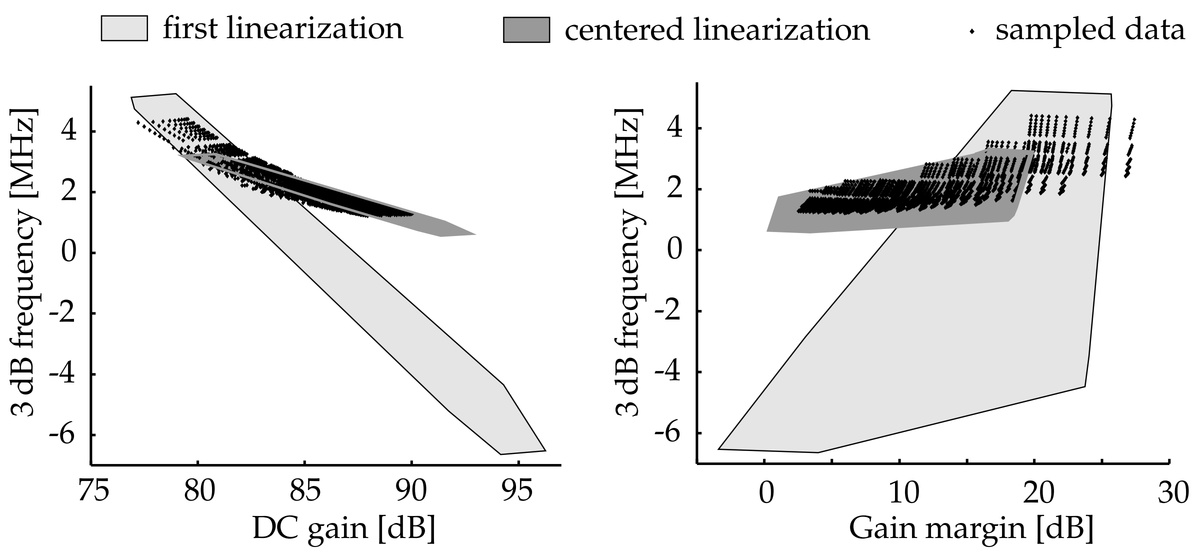


Figure A.4: Approximations to Feasible Performance Space Based on Different Linearization Points

For the folded cascode operational amplifier, the linearization quality for different linearization points is compared in Figure A.4. The projections for two performance combinations, namely DC gain vs. 3 dB frequency and gain margin vs. 3 dB frequency, were first calculated at a parameter vector $\mathbf{p}^{(0)}$ at the boundary of $\mathcal{P}$ and then at the initial sizing. The filled polygons indicate the two-dimensional projections of $\overline{\mathcal{F}}$. All performance values within these polygons are achievable according to the linear approximation. The light gray areas represent the approximation at $\mathbf{p}^{(0)}$, and the dark gray polygons represent the approximation at the initial sizing $\mathbf{p}_s$. For validation purposes, a large number of sizings from within the nonlinear feasible parameter space $\mathcal{P}$ were mapped to the according performances by simulation. A comparison to the linear approximations clearly reveals that the first linearization severely overestimates $\mathcal{F}$, whereas the linear model at the initial sizing point much closer resembles the actual feasible performance space.

Appendix B

Comments on Alternative Algorithms for Linear Performance Space Exploration

B.1 Determination of the Image of a Hypercube under a Linear Map by Geometric Reasoning

The algorithm presented in [MV89] calculates the image of a centered m-dimensional hypercube

$$\overline{\mathcal{P}} = \{\mathbf{p} \mid \underset{i \in \{1 \ldots m\}}{\forall} -1 \le p_i \le 1\} \tag{B.1}$$

under a linear map $\mathbf{F} : \mathbb{R}^m \mapsto \mathbb{R}^n$ with

$$\mathbf{f} = \mathbf{F} \cdot \mathbf{p}\,, \quad \mathbf{F} = [\mathbf{f}_1 \ldots \mathbf{f}_m]\,, \quad \mathbf{f}_i \in \mathbb{R}^n\,, \quad m > n\,. \tag{B.2}$$

The result is a centro-symmetric n-dimensional polytope

$$\overline{\mathcal{F}} = \{\mathbf{f} \mid \mathbf{K} \cdot \mathbf{f} \ge \mathbf{k}\}\,, \quad \mathbf{K} = \begin{bmatrix} \mathbf{k}_1^T \\ \vdots \\ \mathbf{k}_r^T \end{bmatrix}\,, \quad \mathbf{k} = \begin{bmatrix} k_1 \\ \vdots \\ k_r \end{bmatrix}\,. \tag{B.3}$$

Geometrically, $\overline{\mathcal{F}}$ is described by an intersection of r halfspaces:

$$\mathbf{k}_i^T \cdot \mathbf{f} \ge k_i\,, \quad i \in \{1 \ldots r\}\,. \tag{B.4}$$

The vector $\mathbf{k}_i^T$ can be interpreted as a normal vector of the associated boundary hyperplane of $\overline{\mathcal{F}}$. By the same token, the constant k_i reflects the distance of that hyperplane from the origin of the coordinate system.

Determination of K: The matrix $\mathbf{F}$ realizes a linear map $\mathbb{R}^m \mapsto \mathbb{R}^n$. Each subset of $(n-1)$ linearly independent columns of $\mathbf{F}$ from (B.2) constitutes the basis of a hyperplane H in $\mathbb{R}^n$ passing through the origin. Let the index set

$$I \subseteq \{1 \dots m\}\,, \quad |I| = n-1 \tag{B.5}$$

comprise of the indices of the selected columns, where $|I|$ defines the power of I. Then, the associated hyperplane H can be described by

$$H = \{\mathbf{f} \mid \mathbf{f} = \sum_{i \in I} \mathbf{f}_i \cdot p_i\}\,, \quad p_i \in \mathbb{R} \tag{B.6}$$

$$\Leftrightarrow \quad H = \{\mathbf{f} \mid \mathbf{f} = \mathbf{F} \cdot \mathbf{p}\}\,, \qquad p_i \in \begin{cases} \mathbb{R}\,, & i \in I \\ \{0\}\,, & i \in \bar{I}\,, \end{cases} \tag{B.7}$$

with $\bar{I} = \{1 \dots m\} \setminus I$. The generalized cross product[*] of all $\mathbf{f}_i$, $i \in I$, yields a vector $\mathbf{u}$ which is normal to H. Then, an alternative way to describe this hyperplane is

$$H = \{\mathbf{f} \mid \mathbf{u}^T \cdot \mathbf{f} = 0\}\,. \tag{B.11}$$

The assignment of *constant* values other than 0 to the parameters p_i, $i \in \bar{I}$, in (B.7) results in a parallel shift of H without affecting the normal vector $\mathbf{u}$. Consequently, (B.11) becomes

$$H = \{\mathbf{f} \mid \mathbf{u}^T \cdot \mathbf{f} = v\}\,, \tag{B.12}$$

where $v \neq 0$ indicates that H no longer passes through the origin.

[*] The cross product of two three-dimensional vectors, say $\mathbf{a}$ and $\mathbf{b}$, can be written as a formal determinant:

$$\mathbf{a} \times \mathbf{b} = \begin{vmatrix} a_1 & b_1 & \mathbf{e}_1 \\ a_2 & b_2 & \mathbf{e}_2 \\ a_3 & b_3 & \mathbf{e}_3 \end{vmatrix}\,. \tag{B.8}$$

The entries $\mathbf{e}_i$, $i \in \{1,2,3\}$, symbolize orthonormal basis vectors with 1 at their i^{th} component and 0 elsewhere. This concept can informally be generalized by a "cross product" of $(n-1)$ vectors in an n-dimensional space:

$$\mathbf{z} = \times(\underbrace{\mathbf{a}, \mathbf{b}, \dots, \mathbf{x}}_{\substack{\text{(n-1)} \\ \text{vectors}}}) = \begin{vmatrix} a_1 & b_1 & \dots & x_1 & \mathbf{e}_1 \\ a_2 & b_2 & \dots & x_2 & \mathbf{e}_2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ a_n & b_n & \dots & x_n & \mathbf{e}_n \end{vmatrix}\,. \tag{B.9}$$

As in the three-dimensional case, the resulting vector is orthogonal to all $(n-1)$ vectors $\mathbf{a}, \dots, \mathbf{x}$. More precisely, $\mathbf{z}$ is the Hodge dual of the exterior product of the vectors $\mathbf{a}, \dots, \mathbf{x}$ [HS84, Dar94]:

$$\mathbf{z} = *(\mathbf{a} \wedge \mathbf{b} \wedge \dots \wedge \mathbf{x})\,. \tag{B.10}$$

Here, $*$ denotes the Hodge star operator. Due to the shape of the operator $\wedge$, the exterior product is often referred to as *wedge product*.

Now assume that H is shifted in the direction of $\mathbf{u}$, which means $v > 0$. Since for $i \in \bar{I}$ the parameters are restricted to $-1 \leq p_i \leq 1$, there are two distinct parallel hyperplanes that both have a maximum distance $|\hat{v}|$ from the origin:

$$\begin{aligned} H_1 &= \{\mathbf{f} \mid \mathbf{u}^T \cdot \mathbf{f} = \hat{v}\} & \text{(B.13)} \\ H_2 &= \{\mathbf{f} \mid \mathbf{u}^T \cdot \mathbf{f} = -\hat{v}\}\,. & \text{(B.14)} \end{aligned}$$

In fact, H_1 and H_2 represent boundary hyperplanes of $\overline{\mathcal{F}}$. The region between H_1 and H_2 is given by

$$\begin{aligned} A &= \{\mathbf{f} \mid \mathbf{f} = \mathbf{F} \cdot \mathbf{p}\}\,, \quad p_i \in \begin{cases} \mathbb{R}\,, & i \in I \\ [-1, 1]\,, & i \in \bar{I}\,. \end{cases} \\ &= \{\mathbf{f} \mid -\hat{v} \leq \mathbf{u}^T \cdot \mathbf{f} \leq \hat{v}\} \qquad \text{(B.15)} \\ &= \left\{ \mathbf{f} \,\middle|\, \begin{bmatrix} \mathbf{u}^T \\ -\mathbf{u}^T \end{bmatrix} \cdot \mathbf{f} \geq \begin{bmatrix} -\hat{v} \\ -\hat{v} \end{bmatrix} \right\}. \end{aligned}$$

For each index set I, there is one such region. Their intersection yields $\overline{\mathcal{F}}$.

A comparison of (B.3) to (B.15) reveals that for each set of basis vectors $\mathbf{f}_i$, $i \in I$, the matrix $\mathbf{K}$ contains one pair of normal vectors, $\mathbf{u}^T$ and $-\mathbf{u}^T$. The calculation of the associated constants $k_i = -\hat{v}$ is described in the next section.

Determination of $\mathbf{k}$: While for each boundary hyperplane, its normal vector $\mathbf{u}$ is derived from $\mathbf{f}_i$ with $i \in I$, its distance from the origin can be calculated from the remaining column vectors of $\mathbf{F}$. Note that $\mathbf{f}$ can be written as

$$\mathbf{f} = \mathbf{F} \cdot \mathbf{p} = \sum_{i \in I} \mathbf{f}_i \cdot p_i + \sum_{j \in \bar{I}} \mathbf{f}_j \cdot p_j \qquad \text{(B.16)}$$

and recall that

$$\mathbf{u}^T \cdot \mathbf{f}_i = 0\,, \quad i \in I\,. \qquad \text{(B.17)}$$

For a hyperplane H according to (B.12), the value of v then results from

$$v = \mathbf{u}^T \cdot \sum_{i \in \bar{I}} \mathbf{f}_i \cdot p_i\,, \quad -1 \leq p_i \leq 1\,. \qquad \text{(B.18)}$$

If a *boundary* hyperplane has to be determined, then v must be maximized or minimized, respectively. Since (B.18) is linear in p_i, it is obvious that $p_i \in \{-1; 1\}$ in this case. If, again, $v > 0$ is assumed, then maximization results from

$$\hat{v} = \mathbf{u}^T \cdot \sum_{i \in \bar{I}} \mathbf{f}_i \cdot \mathrm{sgn}(\mathbf{u}^T \cdot \mathbf{f}_i)\,, \qquad \text{(B.19)}$$

where $\mathrm{sgn}(\cdot)$ is the signum function.

Conclusion This algorithm draws its efficiency from two algorithmic benefits:

- Owing to the restriction of the parameter space to a unit cube, cf. (B.1), both **K** and **k** can be derived from the columns of **F** directly. Note that any hyperbox can be transformed into a hypercube by proper normalization.
- Redundancy is avoided by construction: In [MV89] it is shown that for every vector **u** which is normal to a subset of $(n-1)$ linearly independent columns of **F**, there are exactly two boundary hyperplanes. The identification of their distance from the origin according to (B.19) implicitly avoids the introduction of redundant hyperplanes.

Unfortunately, the assumption that the parameters are uncorrelated is crucial to this algorithm. Therefore, it cannot be applied to to linearized sizing constraints in their general form.

B.2 Duality of Fourier-Motzkin Elimination

For an underdetermined homogeneous system of linear equations,

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{0}\,, \quad \mathbf{A} \in \mathbb{R}^{(n \times m)}\,, \quad n < m\,, \tag{B.20}$$

the space $\mathcal{X}$ of all nonnegative solutions[†]

$$\mathcal{X} = \{\mathbf{x} \mid \mathbf{A} \cdot \mathbf{x} = \mathbf{0}\,,\ \mathbf{x} \geq \mathbf{0}\}\,, \tag{B.21}$$

can be determined using an algorithm due to Chernikov [Che71]. This space can be described by the nonnegative linear combinations of its basis vectors $\mathbf{b}_i$:

$$\mathcal{X} = \{\mathbf{x} \mid \mathbf{x} = \mathbf{B} \cdot \mathbf{u}\,,\ \mathbf{u} \geq \mathbf{0}\}\,, \quad \mathbf{B} = [\mathbf{b}_1 \dots \mathbf{b}_s]\ . \tag{B.22}$$

Geometrically, $\mathcal{X}$ is a *polyhedral cone* [Che71,Zie95], which implies the following properties:

$$\text{pointedness:} \quad \mathbf{0} \in \mathcal{X} \tag{B.23}$$

$$\text{unboundedness:} \quad \mathbf{x} \in \mathcal{X} \Rightarrow c \cdot \mathbf{x} \in \mathcal{X}\,, \quad c \geq 0\,. \tag{B.24}$$

In [Lee90], this algorithm is used to calculate the solution space of an inhomogeneous system of linear equations, where upper and lower bounds may be given for each variable[‡]:

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{d}\,, \quad \mathbf{x}_{\min} \leq \mathbf{x} \leq \mathbf{x}_{\max}\,, \quad \mathbf{A} \in \mathbb{R}^{(n \times m)}\,, \quad n < m\,. \tag{B.25}$$

Since Chernikov's algorithm requires a *homogeneous* system of equations and *nonnegative* variables, two transformation steps have to be applied to (B.25):

[†] Note that $\mathcal{X}$ is not the *kernel* of (B.20) because it is restricted to the nonnegative solutions.

[‡] Since in [Lee90] both circuit parameters and circuit performances are combined in **x**, the components x_i are referred to as *variables* here for generality.

- Variable substitutions and the introduction of slack variables transform (B.25) into a problem with only nonnegative variables.
  - An *unbounded* variable x can be written as a difference of two nonnegative variables:
  $$x = \hat{x}_1 - \hat{x}_2\,, \quad \hat{x}_1,\, \hat{x}_2 \geq 0\,. \tag{B.26}$$
  - For a *non-positive* variable x, use the substitution
  $$x = -\hat{x}\,, \quad \hat{x} \geq 0\,. \tag{B.27}$$
  - For the general case of finite *lower and upper bounds*,
  $$d_1 \leq x \leq d_2\,, \tag{B.28}$$
  the substitution
  $$\hat{x}_1 = x - d_1 \tag{B.29}$$
  yields a nonnegative variable with an upper bound:
  $$0 \leq \hat{x}_1 \leq d_2 - d_1\,. \tag{B.30}$$
  The upper bound can be transformed into an equality by the introduction of a nonnegative slack variable $\hat{x}_2$. Therefore, (B.28) results in
  $$\hat{x}_1 \geq 0 \quad \wedge \quad \hat{x}_1 + \hat{x}_2 = d_2 - d_1 \quad \wedge \quad \hat{x}_2 \geq 0\,. \tag{B.31}$$

  If all variables including substitution and slack variables are combined in a vector $\hat{\mathbf{x}}$, and all the constants in $\hat{\mathbf{d}}$, then (B.25) turns into
  $$\hat{\mathbf{A}} \cdot \hat{\mathbf{x}} = \hat{\mathbf{d}}\,, \quad \hat{\mathbf{x}} \geq \mathbf{0}\,. \tag{B.32}$$
- With the help of one additional variable $\hat{y}$, (B.32) is transformed into a homogeneous system of equalities:
$$\begin{bmatrix} \hat{\mathbf{A}} & -\hat{\mathbf{d}} \end{bmatrix} \cdot \begin{bmatrix} \hat{\mathbf{x}} \\ \hat{y} \end{bmatrix} = \mathbf{0}\,. \tag{B.33}$$

Chernikov's algorithm can then be applied to (B.33). It yields a solution space in $\hat{\mathbf{x}}$ and $\hat{y}$ which is unbounded due to (B.24). Finally, the substitutions are reversed and the variable $\hat{y}$ is replaced by the constant value 1 again. This yields the solution space of (B.25) which does not have to be bounded for *mathematical* reasons.

However, when the above method is applied to circuit design according to [Lee90], then the solution space comprises of the feasible circuit parameter and performance values. Consequently, this space is definitely bounded for *physical* reasons. This means that if (B.25) describes an actual circuit, (B.33) necessarily contains a condition like
$$\sum_i \hat{x}_i + \hat{y} = const\,. \tag{B.34}$$

In the process of back substitution, this condition ensures boundedness of the original solution space. Geometrically, the latter represents a bounded polyhedron, or polytope, for short [Zie95].

In fact, the above algorithm is the *dual* of the Fourier-Motzkin elimination method. This interrelation is outlined briefly here; for more details see [Che71, DE73]. To convey a rough idea, it is instructive to examine how the solutions of

$$\mathbf{A} \cdot \mathbf{x} = \mathbf{0}\,, \quad \mathbf{x} \geq \mathbf{0}\,, \quad \mathbf{A} \in \mathbb{R}^{(n \times m)}\,, \quad n < m\,, \tag{B.35}$$

can be found using Fourier-Motzkin elimination: Consider the system of linear inequalities

$$\mathbf{A}^T \cdot \mathbf{r} + \mathbf{t} \geq \mathbf{0}\,. \tag{B.36}$$

The elimination of all variables r_i yields

$$\mathbf{C} \cdot \mathbf{t} \geq \mathbf{0}\,. \tag{B.37}$$

Then, (B.35) is solved by [Che71, DE73]

$$\mathcal{X} = \{\mathbf{x} \mid \mathbf{x} = \mathbf{C}^T \cdot \mathbf{u}\,,\ \mathbf{u} \geq \mathbf{0}\}\,. \tag{B.38}$$

Note that in (B.35), the solution vectors $\mathbf{x}$ describe the nonnegative weights which make the weighted sum of the *columns* of $\mathbf{A}$ disappear. Similarly, Fourier-Motzkin elimination finds the nonnegative linear combinations of the individual inequalities of (B.36) which make up the *rows* of $\mathbf{A}^T$ (= columns of $\mathbf{A}$) disappear. This explains the transposition of $\mathbf{A}$ in (B.36).

In (B.36), the vector $\mathbf{t}$ is introduced in order to create a "log" of the elimination process. Every inequality is uniquely identified by one distinctive variable t_i . Therefore, the coefficients of the linear combinations resulting from Fourier-Motzkin elimination are preserved in the *rows* of $\mathbf{C}$. These linear combinations make the rows of $\mathbf{A}^T$ and hence the columns of $\mathbf{A}$ disappear. Consequently, the respective coefficients are the *columns* of $\mathbf{C}^T$ and form the basis of the solution space $\mathcal{X}$ according to (B.38).

As a side note, recall that the Fourier-Motzkin elimination can only build *nonnegative* linear combinations of inequalities since the orientation of the inequality sign has to be preserved. This corresponds to the nonnegativity requirement in (B.35).

The duality of Chernikov's algorithm and Fourier-Motzkin elimination is also revealed by the way the resulting solution spaces are described. There are two dual, mathematically equivalent ways to describe a polyhedron [Zie95, ABS97]: An *H-Polyhedron* is given as an intersection of halfspaces, whereas a *V-Polyhedron* is a positive hull of vertices. The Fourier-Motzkin elimination describes its solution space in terms of an H-Polyhedron, while Chernikov's algorithm yields a V-Polyhedron. While a row of $\mathbf{C}$ in (B.37) describes a bounding halfspace of the associated H-Polyhedron, a column of $\mathbf{C}^T$ in (B.38) represents a vertex of the respective V-Polyhedron. Recall that Fourier-Motzkin elimination tends to produce redundant inequalities which correspond to redundant halfspaces. Accordingly, Chernikov's algorithm yields redundant points which are located inside the polyhedron. This implies the introduction of redundant variables u_i.

B.3 A Note on Range Arithmetic

Most numerical algorithms in engineering and science rely on floating point calculations which produce results in the form of numeric values, yet without quantifying their accuracy. This is a serious concern because a limited machine precision can result in severe rounding errors if no care is taken. Also, the input data might be uncertain due to measurement inaccuracies. As a solution to this problem, *range arithmetic* does not produce a single numeric value, but provides lower and upper bounds which provably contain the exact result. Two realizations of range arithmetic are briefly outlined in the following sections and then contrasted to Fourier-Motzkin elimination and to the two algorithms presented in Sections B.1 and B.2.

Interval Arithmetic

The idea of *interval arithmetic*, also known as *interval analysis* [Moo66], is to represent a real quantity, say x, by an interval $\hat{x}$ which definitely contains the exact value of x:

$$\hat{x} = [x_l, x_u] \quad \text{with} \quad x_l \leq x \leq x_u \,. \tag{B.39}$$

In a computer algorithm, x_u and x_l are floating-point numbers which are conservatively rounded according to the respective IEEE standard [IA85].

In line with this idea, conventional arithmetic operations are extended to entire intervals in such a way that the resulting interval is guaranteed to contain the exact value of the represented quantity. Subtraction, for example, is defined by

$$\hat{x} - \hat{y} \quad := \quad [x_l - y_u, x_u - y_l] \,. \tag{B.40}$$

More complex operations such as multiplication, division, or trigonometric functions can be taken into account by minimum and maximum calculations in combination with detailed case analyses. Thus, conventional algorithms can be replaced by their interval equivalents entirely. For applications, refer to [Kea96].

The greatest weakness of interval arithmetic is that the results are often far too pessimistic. With $\hat{x} = [7, 11]$, for example,

$$\hat{x} - \hat{x} = [7, 11] - [7, 11] = [-4, 4] \,, \tag{B.41}$$

whereas $[0, 0]$ should be the actual range of the expression. Such overestimations accumulate throughout the steps of an algorithm and can render a computation result worthless in extreme cases.

For a classification of the algorithms from Sections 4.2, B.1, and B.2 consider that in interval arithmetic, ranges are assigned to the variables which are regarded as mutually independent. Geometrically, this corresponds to *hypercubes* in the case of orthogonal coordinate axes. In fact, both algorithms presented earlier in this chapter assume independent input variables. Yet, their results are *polytopes* geometrically. Hence, their capabilities go beyond what traditional interval arithmetic can offer. The same is true for the Fourier-Motzkin algorithm which deals with polytopes throughout.

Affine Arithmetic

In *affine arithmetic* [dS97], a real quantity x is represented by an *affine form* which is a linear polynomial of the following form:

$$\tilde{x} = x_0 + x_1\epsilon_1 + x_2\epsilon_2 + \cdots + x_n\epsilon_n\,, \quad \underset{1\le i\le n}{\forall} -1 \le \epsilon_i \le 1\,. \tag{B.42}$$

The coefficients x_i, $0 \le i \le n$, are floating point numbers. The *noise symbols* ϵ_i, $0 \le i \le n$, are symbolic real variables for which only bounds are known. Each of them captures one independent component of uncertainty of x. The quantitative contribution of a particular noise symbol to the overall uncertainty is given by the corresponding coefficient. The summand x_0 is called *central value* of $\tilde{x}$. If the actual value of x is a floating-point number, then x can be represented by $\tilde{x} = x_0$ precisely.

Based on (B.42), the standard arithmetic operations can be extended to ranges. For example, the sum and the difference of two affine forms can be written as

$$\begin{aligned} \tilde{x} + \tilde{y} &:= (x_0 + y_0) + (x_1 + y_1)\epsilon_1 + (x_2 + y_2)\epsilon_2 + \cdots + (x_n + y_n)\epsilon_n & \text{(B.43)} \\ \tilde{x} - \tilde{y} &:= (x_0 - y_0) + (x_1 - y_1)\epsilon_1 + (x_2 - y_2)\epsilon_2 + \cdots + (x_n - y_n)\epsilon_n\,. & \text{(B.44)} \end{aligned}$$

If a certain noise symbol is not common to $\tilde{x}$ and $\tilde{y}$, then the respective coefficient is zero in one of the affine forms. In linear operations such as sum and difference, the set of noise symbols remains unchanged. Nonlinear operations, however, have to be *approximated* by affine forms. The unavoidable uncertainty is reflected by the introduction of new noise symbols.

The greatest advantage of affine arithmetic is that different affine forms can share noise symbols, which allows the formulation of dependencies. Let two independent quantities be described by

$$\begin{aligned} \tilde{x} &= x_0 + x_1\epsilon_1 & \text{(B.45)} \\ \text{and} \quad \tilde{y} &= y_0 + y_2\epsilon_2\,. & \text{(B.46)} \end{aligned}$$

Their sum is

$$\tilde{z} = (x_0 + y_0) + x_1\epsilon_1 + y_2\epsilon_2\,. \tag{B.47}$$

In contrast to interval arithmetic which would just yield a lower and an upper bound for z, affine arithmetic preserves the information that $\tilde{z}$ depends on $\tilde{x}$ and $\tilde{y}$. This is especially important in multi-step computations where it helps to avoid an excessive blow-up of the guaranteed bounds. Hence,

$$\tilde{z} - \tilde{y} = x_0 + x_1\epsilon_1 = \tilde{x} \tag{B.48}$$

and in particular

$$\tilde{x} - \tilde{x} = 0\,. \tag{B.49}$$

Such a cancellation of terms would not have been possible in interval arithmetic.

Since the variables are regarded as independent in interval arithmetic, their joint ranges are hyperboxes geometrically. In [dS97] it is pointed out that in affine arithmetic, the joint ranges of two partially dependent variables are centro-symmetric polytopes. Recall that the algorithm from Section B.1 [MV89] deals with exactly the same geometric structures. In fact, a comparison of (B.42) to (B.1) and (B.2) reveals that the problem description of [MV89] is given as a system of affine forms whose central values are all zero. Yet, the algorithm does not feature affine arithmetic: The focus of affine arithmetic is the efficient algorithmic *manipulation* of affine forms. During this process, new affine forms are derived from the original data until the final result is obtained. In contrast, the aim of the algorithm from [MV89] is the *interpretation* of a fixed set of constant affine forms and its representation as a system of linear inequalities. The values which are described by the affine forms do not change, just their representation. That is the reason why this algorithm does not fall into the category of affine arithmetic. Instead, it could be used as a final procedural step to transform the result of an affine arithmetic algorithm into a hyperplane representation.

Interval arithmetic and affine arithmetic in particular generalize real arithmetic in the sense that guaranteed lower and upper bounds are provided for the exact but unknown real-valued results. For example, this allows to calculate ranges which contain the roots of nonlinear functions [Wil81,dS97]. In [LHB02] it is shown how guaranteed bounds on an optimization result can be calculated when the designable parameters are subject to uncertainty. The motivation behind Chernikov's algorithm from Section B.2 is different: It calculates all nonnegative solutions of an underdetermined homogeneous system of linear equations, cf. (B.21). The result is a set of vertices whose positive combinations describe the desired solution space without safeguarding overestimation. No uncertainty is considered and consequently no methods of interval or affine arithmetic are used. For this reason, both Chernikov's algorithm and Fourier-Motzkin elimination as its dual are no representatives of interval or affine arithmetic.

Conclusion

In essence, neither of the algorithms presented in Sections 4.2, B.1, and B.2 relies on interval or affine arithmetic. Yet, all of them operate on intervals. For this reason, it is justified to refer to them as interval methods in a broader sense.

Appendix C

Basic Concepts of Popular Stochastic Optimization Techniques

While numerous stochastic optimization techniques were proposed [RH01], two of them moved to mainstream: genetic algorithms and simulated annealing. The following sections give an overview of the basic concepts behind these approaches.

C.1 Genetic Algorithms

Genetic algorithms are nonlinear optimization strategies, which can be applied to a wide variety of engineering problems. Circuit optimization is one of them: For a given circuit topology, i.e. a certain map $\mathbf{f}(\cdot)$, a particular circuit instance is fully characterized by its parameter values $\hat{\mathbf{p}}$. In the area of genetic optimization, such a unique instance is referred to as an *individual*. In a biological analogy, its parameter values are called *genes* because they determine the traits of that particular individual. The quality of these traits is quantified by a scalar value which is referred to as *fitness* of the individual. This value is calculated from the performances $\mathbf{f}(\hat{\mathbf{p}})$ and the sizing constraints $\mathbf{c}(\hat{\mathbf{p}})$. In contrast to most other optimization techniques which successively optimize one distinct individual, genetic algorithms operate on sets, or *populations*, of individuals. Therefore, they seem particularly suited for multi-objective optimization where not only a single optimum solution exists, but a large number of Pareto optimal solutions. Genetic optimization is an iterative process and the consecutive populations are interpreted as *generations*. The optimization starts from an initial population, the genes of which are randomly distributed in the entire parameter space. Imitating the mechanisms of *survival of the fittest*, *mating* and random *mutations*, a new *offspring generation* is derived from the current *parent generation* in each iteration. That way, the average fitness of the generations tends to increase in the course of the evolution process.

The following operations procedurally describe the creation of a new generation:

I) Selection
From the current population, two individuals at a time are selected for mating. Their offspring will be placed into the new generation which is usually equal in size to the parent generation. There are numerous selection schemes which share a fundamental common trait: The chance for an individual of being selected depends on its scalar fitness value. In multi-objective optimization, a Pareto ranking has to be implemented which ensures that the non-dominated individuals have a higher fitness than all others and that infeasible individuals have low fitness values. Furthermore, genetic diversity has to be encouraged to avoid clustering in certain regions of the feasible performance space. Since the selection process is usually stochastic, a certain individual might be selected multiple times, and also dominated individuals can be selected, albeit at a lower probability. In this way, inferior traits are likely to die out.

II) Crossover
In the mating process, a new individual is created that inherits genetic information from both parents. The details of the crossover algorithm depend on how the genes encode the parameter values. Binary genetic algorithms use bit strings of finite length as genes. In this case, a new gene is composed by combining bit sequences from one gene with sequences from the other one. In spite of their algorithmic convenience, binary genes are of limited suitability for continuous parameter values. For real-valued genes the crossover step is more intricate, which has lead to a large number of different algorithms.

III) Mutation
In the crossover step, genetic information is exchanged between individuals. Mutation, in contrast, causes a random perturbation within the genes of a single individual. In binary genetic algorithms, for example, single bits may be altered at a low probability. For real-valued genetic algorithms, on the other hand, the parameter values may, for example, be perturbed according to a zero-mean Gaussian distribution.

In summary, the crossover and mutation steps promote diversity and hence drive the exploration of the performance space. The selection step is responsible for increasing the average fitness of successive populations. An in-depth treatment of multi-objective optimization using genetic algorithms can be found in [Deb01].

While the manipulation of entire sets of individuals is an appealing feature in the area of multi-objective optimization, the main drawback of genetic approaches is their huge computational effort. Recall that each individual corresponds to a circuit instance, the fitness of which has to be evaluated based on simulation. Furthermore, convergence can only be detected by examining the genetic dissimilarity of consecutive populations. Therefore, in contrast to deterministic techniques, there is no precise optimality criterion.

C.2 Simulated Annealing

Simulated annealing imitates the natural process of cooling a melted mass, which leads to a perfect crystal lattice if the cooling process proceeds slowly enough. In contrast to genetic algorithms, single-objective simulated annealing algorithms work with a single *search agent*, i.e. they successively improve one particular solution candidate. For multi-objective optimization, a number of independent search agents can be used in parallel [NP00]. Starting from initial parameter vectors and a certain virtual *temperature*, the optimization relies on the iterative execution of the following operations:

I) Perturbation
To explore the performance space, the momentary parameter vectors are perturbed by a certain amount. This operation is similar to the mutation operation of genetic algorithms. Yet, mutation only results in minor variations, whereas the main source of diversity in genetic algorithms is crossover. In simulated annealing, however, perturbation yields major variations because it is the only mechanism for exploration. The implementation details of the perturbation operator depend on how the parameter values are encoded – either binary or as real values.

II) Selection
For each search agent, the performance values resulting from the perturbed parameter values are compared to the old performances. If the new performance values dominate the old ones, then the perturbed parameter values are accepted immediately. Otherwise, the new parameters are accepted at a certain probability, which nonlinearly depends on the temperature and on the performance values. At high temperatures, a performance degradation is readily accepted, while for falling temperatures the importance of performance improvement increases steadily. This strategy combines versatile exploration in the first phase and fine-tuning in the final phase.

III) Cooling
After each parameter update by perturbation and selection, the temperature is lowered. The actual nonlinear cooling scheme has a major impact on the convergence speed and the quality of the final result.

While simulated annealing can be applied to multi-objective optimization, it has been reported that it is inferior to genetic algorithms both in terms of computational resources and solution quality [NP00].

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.

[ABS97] D. Avis, D. Bremner, R. Seidel: *How good are convex hull algorithms?*, Computational Geometry: Theory and Applications, volume 7(5–6), pages 265–301, April 1997.

[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.

[AI91] C. Ancourt, F. Irigoin: *Scanning polyhedra with DO loops*, in: *ACM SIGPLAN Symposium on Principles and Practice of Parallel Programming*, pages 39–50, 1991.

[AMH91] H. Abdel-Malek, A. Hassan: *The ellipsoidal technique for design centering and region approximation*, IEEE Transactions on Computer-Aided Design of Circuits and Systems, volume 10, pages 1006–1013, 1991.

[AMHH99] H. L. Abdel-Malek, A.-K. S. O. Hassan, M. H. Heaba: *A boundary gradient search techniques and its application in design centering*, IEEE Transactions on Computer-Aided Design of Circuits and Systems, volume 18(11), pages 1654–1661, November 1999.

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

[BGH04] O. Bajdechi, G. E. Gielen, J. H. Huijsing: *Systematic design exploration of delta-sigma ADCs*, IEEE Transactions on Circuits and Systems I: Fundamental Theory and Applications, volume 51(1), pages 86–95, January 2004.

[BGT81] R. Bland, D. Goldfarb, M. Todd: *The ellipsoid method: a survey*, Operations Research, volume 29, pages 1039–1091, 1981.

[Bri03] R. Bridgelall: *Enabling mobile commerce through pervasive communications with ubiquitous RF tags*, in: *IEEE Wireless Communications and Networking Conference*, volume 3, pages 2041–2046, March 2003.

[BV04] S. Boyd, L. Vandenberghe: *Convex Optimization*, Cambridge University Press, 2004.

[BW94] A. J. C. Bik, H. A. G. Wijshoff: *Implementation of Fourier-Motzkin elimination*, Technical report, High Performance Computing Division, Department of Computer Science, Leiden University, 1994.

[Cad] *Cadence Design Systems (www.cadence.com).*

[CB99] E. Christen, K. Bakalar: *VHDL-AMS – A hardware description language for analog and mixed-signal applications*, IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing, volume 46(10), pages 1263–1272, October 1999.

[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.

[CCH$^+$99] H. Chang, L. Cooke, M. Hunt, G. Martin, A. McNelly, L. Todd: *Surviving the SOC Revolution*, Kluwer Academic Publishers, 1999.

[CDSS02] L. P. Carloni, F. De Bernardinis, A. L. Sangiovanni-Vincentelli, M. Sgroi: *The art and science of integrated systems design*, in: *European Solid-State Circuits Conference (ESSCIRC)*, pages 25–36, 2002.

[CG78] H. S. Coxeter, S. L. Greitzer: *Geometry revisited*, Math. Assoc. of America, 1978.

[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.

[Che71] S. N. Chernikov: *Lineare Ungleichungen*, Deutscher Verlag der Wissenschaften, 1971.

[CL] T. Christof, A. Loebel:
Porta - polyhedron representation transformation algorithm., http://www.zib.de/Optimization/Software/Porta/index.html.

[CR96] T. Christof, G. Reinelt: *Combinatorial optimization and small polytopes*, Top (Spanish Statistical and Operations Research Society), volume 4, pages 1–64, 1996.

[Dan63] G. Dantzig: *Linear Programming and Extensions*, Princeton University Press, 1963.

[Dar94] R. W. R. Darling: *Differential Forms and Connections*, Cambridge University Press, 1994.

[Das98] I. Das: *On characterizing the 'knee' of the Pareto curve based on Normal-Boundary Intersection*, Structural Optimization, volume 18(2/3), pages 107–115, August 1998.

[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.

[DE73] G. Dantzig, B. Eaves: *Fourier-Motzkin elimination and its dual*, Journal of Combinatorial Theory (A), volume 14, pages 288–297, 1973.

[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.

[Deb01] K. Deb: *Multi-objective optimization using evolutionary algorithms*, Wiley-Interscience Series in Systems and Optimization, Wiley, 2001.

[DG02] B. De Smedt, G. Gielen: *WATSON: a multi-objective design space exploration tool for analog and RF IC design*, in: *IEEE Custom Integrated Circuits Conference (CICC)*, pages 31–34, 2002.

[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 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 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.

[DH77] S. Director, G. Hachtel: *The simplicial approximation approach to design centering*, IEEE Transactions on Circuits and Systems CAS, volume 24, pages 363–372, 1977.

[DHBS90] F. Durbin, J. Haussy, G. Berthiau, P. Siarry: *A novel approach to integrated circuit CAD: optimization by coupling simulated annealing method to open circuit simulator SPICE-PAC*, L'Onde Electrique, volume 70(6), pages 59–65, 1990.

[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 Circuits and Systems, volume 20(1), pages 1–21, January 2001.

[dMHHM$^+$99] M. del Mar Hershenson, A. Hajimiri, S. S. Mohan, S. P. Boyd, T. H. Lee: *Design and optimization of LC oscillators*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 1999.

[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.

[dS97] L. H. de Figueiredo, J. Stolfi: *Self-Validated Numerical Methods and Applications*, Brazilian Mathematics Colloquium monographs, IMPA/CNPq, Rio de Janeiro, Brazil, 1997.

[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.

[Ehr97] M. Ehrgott: *Multiple Criteria Optimization*, Shaker Verlag, 1997.

[EK96] H. Esbensen, E. S. Kuh: *Design space exploration using the genetic algorithm*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 500–503, 1996.

[EKO90] H. Eschenauer, J. Koski, A. Osyczka: *Multicriteria design optimization: procedures and applications*, Springer-Verlag, 1990.

[ETP89] F. El-Turky, E. Perry: *BLADES: An artificial intelligence approach to analog circuit design*, IEEE Transactions on Computer-Aided Design of 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.

[Fra04] M. Franz: *Convex - a Maple package for convex geometry*, http://www-fourier.ujf-grenoble.fr/~franz/convex/, 2004.

[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. Gómez, S. H. K. Embabi, E. Sánchez-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.

[GH75] F. W. Gembicki, Y. Y. Haimes: *Approach to performance and sensitivity multiobjective optimization: The goal attainment method*, IEEE Transactions on Automatic Control, volume 20(6), pages 769–771, December 1975.

[Gil01] B. Gilbert: *Analog at milepost 2000: a personal perspective*, Proceedings of the IEEE, volume 89(3), pages 289–304, March 2001.

[GK03] K. A. Grajski, E. Kirk: *Towards a mobile multimedia age - location-based services: a case study*, Wireless Personal Communications, pages 105–116, 2003.

[GLP] *GLPK (GNU Linear Programming Kit), http://www.gnu.org/software/glpk/glpk.html*.

[GMW81] P. E. Gill, W. Murray, M. H. Wright: *Practical Optimization*, Academic Press. Inc., London, 1981.

[GR00] G. G. E. Gielen, R. A. Rutenbar: *Computer-aided design of analog and mixed-signal integrated circuits*, Proceedings of the IEEE, volume 88(12), pages 1825–1852, December 2000.

[GRFRV02] O. Guerra, E. Roca, F. V. Fernandez, A. Rodriguez-Vazquez: *Approximate symbolic analysis of hierarchically decomposed analog circuits*, Analog Integrated Circuits and Signal Processing, volume 31(2), pages 131–145, 2002.

[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.

[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 Circuits and Systems, volume 11, pages 1402–1417, 1992.

[Hin04] H. Hindi: *A tutorial on convex optimization*, in: *American Control Conference*, 2004.

[HM79] C.-L. Hwang, A. S. M. Masud: *Multiple Objective Decision Making*, Springer, 1979.

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

[HS84] D. Hestenes, G. Sobczyk: *Clifford Algebra to Geometric Calculus*, D. Reidel Publishing Company, 1984.

[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.

[IA85] IEEE Computer Society Standards Committee. Working group of the Microprocessor Standards Subcommittee, American National Standards Institute: *IEEE standard for binary floating-point arithmetic*, ANSI/IEEE Std 754-1985., IEEE Computer Society Press, 1985.

[ITRS03] International SEMATECH, *International Technology Roadmap for Semiconductors: 2003 Edition*, 2003, http://public.itrs.net/Files/2003ITRS/Home2003.htm.

[JLF98] M. Jimenez, J. Llaberia, A. Fernandez: *Loop bounds computation for multilevel tiling*, in: *Sixth Euromicro Workshop on Parallel and Distributed Processing*, 1998.

[JLSW03] S. Jung, C. Lauterbach, M. Strasser, W. Weber: *Enabling technologies for disappearing electronics in smart textiles*, in: *IEEE International Solid State Circuits Conference*, pages 386–387, 2003.

[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 Circuits and Systems, volume 19(12), pages 1561–1571, December 2000.

[Kea96] R. B. Kearfott: *Interval computations: Introduction, uses, and resources*, Euromath Bulletin, volume 2(1), pages 95–112, 1996.

[Kha79] L. Khachiyan: *A polynomial algorithm in linear programming*, Soviet Mathematics Doklady, volume 20, pages 191–194, 1979.

[Kje91] G. Kjellström: *On the efficiency of Gaussian adaptation*, Journal of Optimization Theory and Applications, volume 3, pages 589–597, 1991.

[Kon04] J.-T. Kong: *CAD for nanometer silicon design challenges and success*, IEEE Transactions on VLSI Systems, volume 12(11), pages 1132–1147, November 2004.

[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 Circuits and Systems, volume 9, pages 113–125, 1990.

[KT81] G. Kjellström, L. Taxen: *Stochastic optimization in system design*, IEEE Transactions on Circuits and Systems CAS, volume 28, pages 702–715, 1981.

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

[Lee90] D. Leenaerts: *Application of interval analysis for circuit design*, IEEE Transactions on Circuits and Systems CAS, volume 37, pages 803–807, 1990.

[LH91] D. M. W. Leenaerts, J. A. Hegt: *Finding all solutions of piecewise linear functions and application to circuit design*, International Journal of Circuit Theory and Applications, volume 19, pages 107–123, 1991.

[LHB02] A. Lemke, L. Hedrich, E. Barke: *Analog circuit sizing based on formal methods using affine arithmetic*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, pages 486–489, 2002.

[LTZ87] M. Lightner, T. Trick, R. Zug: *Circuit optimization and design*, Circuit Analysis, Simulation and Design, Part 2 (A. Ruehli). Advances in CAD for VLSI 3, pages 333–391, 1987.

[Mat] *MATLAB®, The Mathworks (www.mathworks.com).*

[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.

[MCR95] P. Maulik, L. R. Carley, R. Rutenbar: *Integer programming based topology selection of cell-level analog circuits*, IEEE Transactions on Computer-Aided Design of Circuits and Systems, volume 14(4), pages 401ff, April 1995.

[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.

[Moo66] R. E. Moore: *Interval Analysis*, Prentice-Hall, 1966.

[MSG03] T. Massier, G. Stehr, H. Graeb: *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.

[Mun] *MunEDA GmbH (www.muneda.com).*

[MV89] L. Milor, V. Visvanathan: *Detection of catastrophic faults in analog integrated circuits*, IEEE Transactions on Computer-Aided Design of Circuits and Systems, volume 8, pages 114–130, 1989.

[MV01] P. Mandal, V. Visvanathan: *CMOS Op-Amp sizing using a geometric programming formulation*, IEEE Transactions on Computer-Aided Design of 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.

[Neo] *Neolinear Inc. (acquired by Cadence in 2004)(www.cadence.com/neolinear).*

[Ana92] *ELDO: Analog and System Simulation*, ANACAD GmbH, Ulm, 1992.

[NP00] D. Nam, C. H. Park: *Multiobjective simulated annealing: A comparative study to evolutionary algorithms*, International Journal of Fuzzy Systems, pages 87–97, June 2000.

[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 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 Circuits and Systems, volume 15(3), pages 273–294, March 1996.

[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 Circuits and Systems, volume 19(6), pages 703–717, June 2000.

[RH01] C. C. Ribeiro, P. Hansen: *Essays and Surveys in Metaheuristics*, volume 15 of *Operations Research/Computer Science Interfaces*, Kluwer Academic Publishers, 2001.

[Sch98] A. Schrijver: *Theory of Linear and Integer Programming*, John Wiley & Sons, 1998.

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

[SEGA99] R. Schwencker, J. Eckmueller, H. Graeb, K. Antreich: *Automatische Nominalpunktdimensionierung analoger CMOS-Schaltungen mit Parameterabständen als Zielgrößen*, in: *GME/ITG-Diskussionssitzung Entwicklung von Analogschaltungen mit CAE-Methoden*, Munich, February 1999.

[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.

[SGA01] G. Stehr, H. Graeb, K. Antreich: *A hierarchical optimization approach for analog and mixed-signal systems*, in: *Forum on Design Languages (FDL)*, September 2001.

[SGA02] G. Stehr, H. Graeb, K. Antreich: *Dimensionierungsnebenbedingungen bei der hierarchischen Optimierung von Mixed-Signal-Systemen*, in: *6. GMM/ITG Diskussionssitzung Entwurf von Analogschaltungen (ANALOG '02)*, 2002.

[SGA03a] G. Stehr, H. Graeb, K. Antreich: *Feasibility regions and their significance to the hierarchical optimization of analog and mixed-signal systems*, International Series of Numerical Mathematics, volume 146, pages 167–184, 2003.

[SGA03b] G. Stehr, H. Graeb, K. Antreich: *Hierarchische Simulation von Mixed-Signal-Schaltungen*, in: *Informationstagung Mikroelektronik 2003 (ME '03)*, 2003.

[SGA03c] 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.

[SGA03d] G. Stehr, H. Graeb, K. Antreich: *Untersuchung der Leistungsfähigkeit analoger Schaltungen mit Hilfe von Dimensionierungsregeln und nichtlinearer Mehrziel-Optimierung*, in: *7. ITG/GMM-Diskussionssitzung Entwurf von Analogschaltungen (ANALOG '03)*, 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.

[SGA05] G. Stehr, H. Graeb, K. Antreich: *Ein linearer Ansatz zur effizienten Abschätzung der Leistungsfähigkeit analoger Schaltungen*, in: *GME/ITG-Diskussionssitzung Entwicklung von Analogschaltungen mit CAE-Methoden (ANALOG '05)*, 2005.

[SPS$^+$03] G. Stehr, M. Pronath, F. Schenkel, H. Graeb, K. Antreich: *Initial sizing of analog integrated circuits by centering within topology-given implicit specifications*, in: *IEEE/ACM International Conference on Computer-Aided Design (ICCAD)*, 2003.

[Sta88] W. Stadler: *Multicriteria optimization in engineering and in the sciences*, Plenum Press, 1988.

[Ste01] G. Stehr: *Hierarchical optimization of analog and mixed-signal systems*, Technical Report TUM-LEA-01-7, TUM, December 2001.

[SVP98] A. Seifi, J. Vlach, K. Ponnambalam: *Statistical design of integrated circuits using maximum likelihood estimation of the covariance matrix*, in: *IEEE International Symposium on Circuits and Systems (ISCAS)*, page VI/338, 1998.

[Syn] *Synopsys Inc. (www.synopsys.com).*

[UTO98] E. L. Ulungu, J. Teghem, C. Ost: *Efficiency of interactive multi-objective simulated annealing through a case study*, Journal of the Operational Research Society, volume 49(10), October 1998.

[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 Circuits and Systems, volume 20(9), pages 1037–1058, September 2001.

[VG03] M. Vogels, G. Gielen: *Architectural selection of A/D converters*, in: *ACM/IEEE Design Automation Conference (DAC)*, pages 974–977, 2003.

[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.

[VVS87] M. Vlach, J. Vlach, K. Singhal: *SABER: A design tool for analog systems*, Analogy Inc., Beaverton, U.S.A., 1987.

[WGP96] J. F. Whidborne, D. W. Gu, I. Postlethwaite: *Simulated annealing for multi-objective control system design*, in: *UKACC International Conference on Control*, pages 376–381, September 1996.

[Wil81] P. Wildenauer: *Construction of domains with all solutions, and the existence of extreme solutions*, SIAM Journal on Numerical Analysis, volume 18(5), pages 801–807, October 1981.

[WKWC03] R. A. Wildman, J. I. Kramer, D. S. Weile, P. Christie: *Multi-objective optimization of interconnect geometry*, IEEE Transactions on VLSI Systems, volume 11(1), pages 15–23, February 2003.

[WV93] J. Wojciechowski, J. Vlach: *Ellipsoidal method for design centering and yield estimation*, IEEE Transactions on Computer-Aided Design of Circuits and Systems, volume 12, pages 1570–1579, 1993.

[ZG03] Y. Zhang, L. Gao: *On numerical solution of the maximum volume ellipsoid problem*, SIAM Journal on Optimization, volume 14(1), pages 53–76, 2003.

[Zha03] W. H. Zhang: *Pareto optimum sensitivity analysis in multicriteria optimization*, Finite Elements in Analysis and Design, volume 39(5–6), pages 505–520, March 2003.

[Zie95] G. M. Ziegler: *Lectures on Polytopes*, Springer Verlag, New York, 1995.

[Ziz01] S. Zizala: *Numerische Verhaltensmodellierung analoger CMOS-Schaltungen unter Berücksichtigung von Dimensionierungsregeln*, Ph.D. thesis, Technical University of Munich, 2001.

Nomenclature

	General Conventions	
$\mathbf{A}$	Matrix $\mathbf{A}$	16
a	Scalar a	16
$\mathbf{a}$	Vector $\mathbf{a}$	16
a_i	i^{th} component of vector $\mathbf{a}$	16
$\mathbf{f}$	Vector of circuit performances, n-dimensional	16
$\mathbf{p}$	Vector of designable parameters, m-dimensional	16
$\mathbf{c}(\cdot)$	Sizing constraint vector function, q-dimensional	22
$\mathbf{f}(\cdot)$	Performance vector function	16
$\mathcal{C}$	Feasible constraint space	22
$\mathcal{F}$	Feasible performance space	23
$\partial\mathcal{F}$	Boundary of feasible performance space	23
$\mathcal{P}$	Feasible parameter space	22
	Normal-Boundary Intersection	
$\partial\mathcal{F}^{\prec}$	Pareto optimal front	26
$\mathbf{f}^{*i}$	Individual minimum. In $\mathbf{f}^{*i}$, performance f_i has its minimum value within $\mathcal{F}$.	31
$\mathbf{F}$	Matrix comprising of all individual minima	33
$\mathcal{H}$	Convex hull of individual minima	31
$\mathbf{n}$	Normal vector to $\mathcal{H}$	31
$\tilde{\mathbf{n}}$	Quasinormal vector to $\mathcal{H}$	33
t	NBI objective function, can be interpreted as distance to $\mathcal{H}$	35
$\mathbf{w}$	Weight vector. In NBI, $\mathbf{F} \cdot \mathbf{w} \in \mathcal{H}$	33

	Fourier-Motzkin Elimination	
$\overline{\mathcal{F}}$	Linearized feasible performance space	50
$\overline{\mathcal{P}}$	Linearized feasible parameter space	49
$\Delta\mathbf{f}$	Performance variation	49
$\Delta\mathbf{p}$	Parameter variation	48
$\mathbf{C}$	Jacobian matrix of sizing constraints in $\mathbf{p}^{(0)}$	49
$\mathbf{C}_\Box$	Subset of n columns of $\mathbf{C}$	51
$\mathbf{C}_\Diamond$	Subset of remaining $m-n$ columns of $\mathbf{C}$	51
$\mathbf{c}$	Right-hand side of linearized sizing constraints	49
$\mathbf{F}$	Jacobian matrix of performances in $\mathbf{p}^{(0)}$	49
$\mathbf{F}_\Box$	Regular quadratic subset of n columns of $\mathbf{F}$	50
$\mathbf{F}_\Diamond$	Subset of $m-n$ remaining columns of $\mathbf{F}$	50
$\Delta\mathbf{p}_\Box$	Subset of n entries of $\mathbf{p}$ corresponding to $\mathbf{F}_\Box$	50
$\Delta\mathbf{p}_\Diamond$	Subset of $m-n$ remaining entries of $\mathbf{p}$ corresponding to $\mathbf{F}_\Diamond$	50
$\mathbf{K}$	Jacobian matrix of linearized description of feasible performance space	50
$\mathbf{k}$	Right-hand side of linearized description of feasible performance space	50

	Initial Sizing	
$\mathbf{p}_s$	Initial sizing result	98
$\mathbf{p}^{(j)}$	Approximation to $\mathbf{p}_s$ in the j th iteration of the initial sizing algorithm	99
$\mathbf{c}^{(j)}$	Values of sizing constraints at $\mathbf{p}^{(j)}$	99
$\mathbf{C}^{(j)}$	Jacobian of the sizing constraints in the j th iteration.	99
$\mathbf{c}_i^{(j)\,T}$	i th row of $\mathbf{C}^{(j)}$	101
$V^{(j)}$	Set containing the indices of the violated constraints in point $\mathbf{p}^{(j)}$	103
$\mathbf{p}_e$	Center of inscribed maximum volume ellipsoid	101
$\mathbf{E}$	Matrix describing size and orientation of inscribed maximum volume ellipsoid	101
$\Delta\mathbf{p}_d^{(j)}$	Search direction in ellipsoidal update, relaxed polytope case	104
$x^{(j)}$	Step length in ellipsoidal update, relaxed polytope case	104

List of Figures

List of Tables

Abstract in German

Sowohl beim flachen als auch beim hierarchischen Entwurf analoger Schaltungen spielt die Frage nach den realisierbaren Eigenschaftswerten einer gegebenen Schaltungsstruktur eine wichtige Rolle. Im Rahmen dieser Arbeit wurden zwei simulationsbasierte Verfahren zur Eigenschaftsraum-Exploration vorgestellt, die für unterschiedliche Anwendungsgebiete optimiert sind. Ein nichtlineares Verfahren ermöglicht die genaue und effiziente Identifikation der Tradeoffs zweier oder dreier Eigenschaften. Dabei werden auch jene Schaltungskomponenten identifiziert, die eine weitere Eigenschaftsverbesserung verhindern. Muss wie beim hierarchischen Entwurf eine Vielzahl von Eigenschaften gleichzeitig untersucht werden, sind nichtlineare Verfahren wegen ihres hohen Simulationsaufwandes nicht mehr geeignet. Zu diesem Zweck wurde eine Methodik entwickelt, die das Schaltungsverhalten linearisiert und daraus eine lineare Approximation des realisierbaren Eigenschaftsbereiches berechnet.