2025 Design, Automation & Test in Europe Conference (DATE 2025)

Circuits in a Box: Computing High-Dimensional Performance Spaces for Analog Integrated Circuits

1st Benedikt Ohse
Ernst-Abbe-Hochschule Jena,
University of Applied Sciences
07743 Jena, Germany
benedikt.ohse@eah-jena.de

2nd Jürgen Kampe
Ernst-Abbe-Hochschule Jena,
University of Applied Sciences
07743 Jena, Germany
juergen.kampe@eah-jena.de

3rd Christopher Schneider
Ernst-Abbe-Hochschule Jena,
University of Applied Sciences
07743 Jena, Germany
christopher.schneider@eah-jena.de

***Abstract*— Performance spaces contain information about all combinations of attainable performance parameters of analog integrated circuits. Their exploration allows designers to evaluate given circuits without considering implementation details, making them a valuable tool to support the design process. The computation of performance spaces—even for a small number of considered parameters—is time-consuming because it requires solving multi-objective, non-convex optimization problems that involve costly circuit simulations. We present a numerical method for efficiently approximating high-dimensional performance spaces, which is based on the box-coverage method known from Pareto optimization. The resulting implementation not only outperforms state-of-the-art solvers based on the well-known Normal-Boundary Intersection method in terms of computational complexity, but also offers several advantages, such as a practical stopping criterion and the possibility of warm starting. Furthermore, we present an interactive visualization technique to explore performance spaces of any dimension, which can help system designers to make reliable topology decisions even without detailed technical knowledge of the underlying circuits. Numerical experiments that confirm the efficiency of our approach are performed by computing seven-dimensional performance spaces for an analog low-dropout regulator as used in the radio-frequency identification domain.**

***Index Terms*—analog circuits, performance space, optimization**

I. Introduction

Performance spaces are an essential tool for the design of analog integrated circuits because they provide information about all attainable combinations of *performance parameters* (gain, bandwidth, etc.). Thus, they illustrate the performance capabilities of a given circuit, and their exploration assists designers, e.g., in topology selection [1]. Approximations of performance spaces can be obtained using *multi-objective optimization* methods. However, their computation is very time-consuming. In recent years, performance spaces and their *Pareto fronts* (set of all non-dominated performances, see Def. 1) have been computed mainly using the well-known *Normal-Boundary Intersection* (NBI) method [2], [3]. Thereby, Pareto fronts [3]–[5] and complete performance spaces could be computed for up to three performance criteria [6]–[8], while real-world scenarios often require higher dimensions.

Recently, a novel approach for solving multi-objective optimization problems has been published [9]–[11]. This method aims at a *box-based coverage* of the Pareto front and offers several advantages over the NBI method, such as adjustable stopping criteria based on the actual quality of the approximation, a warm start option, and a high degree of parallelizability. Boxes in any dimension can be defined by only two points (vertices), resulting in a simpler implementation compared to the sophisticated NBI method. Our goal in this paper is to provide an efficient implementation of the so-called *box approximation method* for the case of computing high-dimensional performance spaces. After formulating the problem of computing performance spaces mathematically in Sec. II, our modified and parallelized version of the box approximation method is described in Sec. III.

Naturally, another issue arises in high dimensions (greater than three), namely *visualization*. Approximations of performance spaces consist of point clouds, which can be visualized by their (non-convex) hulls in dimensions two or three, e.g. using α-shapes [12], [13]. As a visualization concept for higher dimensions, *Parallel Coordinates* plots [14]–[16], like in Fig. 1, are used. They allow the designer to intuitively and interactively explore the performance spaces and are a tremendous help in the system design task. Parallel Coordinates plots are introduced in Sec. IV and applied to a real-world example in Sec. V, where the design of an analog low-dropout regulator (ALDO) is supported by seven-dimensional performance spaces.

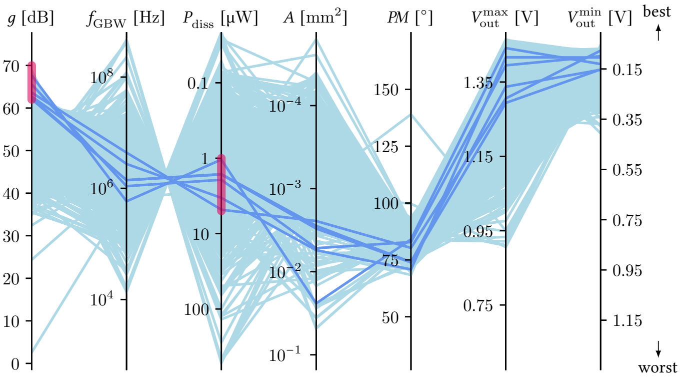


Fig. 1. Parallel Coordinates plot of a performance space $\mathcal{P}$.
■ point on approximation of $\mathcal{P}$ ■ specification marker
■ point on approximation of $\mathcal{P}$ that meets specification

B.O. is supported by the graduate fellowship of Thuringia in Germany.

978-3-9826741-0-0/DATE25/© 2025 EDAA

II. Multi-Objective Optimization

A. Mathematical Problem Definition

Throughout this paper, scalars t and vectors v are written in lower-case, matrices M in upper-case, and sets $\mathcal{S}$ in calligraphic notation. All relations ($\leq, \geq$) for vectors are to be understood element-wise.

For integrated circuits, the choice of *design parameters* (e.g., channel lengths and widths, resistances, and capacitances) determines the resulting *performance parameters* (e.g., bandwidth, gain, and power dissipation). We denote the set of n design parameters, gathered in a vector $x \in \mathbb{R}^n$, by

$$\mathcal{D} = \{x \in \mathbb{R}^n \mid c\,(x) \geq 0\,,\ x_\ell \leq x \leq x_u\}\,,$$

where the vector-valued function $c\colon \mathbb{R}^n \to \mathbb{R}^p$ defines p possible constraints and $[x_\ell, x_u] \subseteq \mathbb{R}^n$ denote the lower and upper bounds. The m performance criteria are defined by a vector-valued function $f\colon \mathbb{R}^n \to \mathbb{R}^m$. For a given combination of design parameters $x \in \mathbb{R}^n$, this results in an ensemble of performance parameters $f(x) = [f_1(x), f_2(x), \ldots, f_m(x)] \in \mathbb{R}^m$. The set of all possible combinations of performance parameters for a circuit is called *performance space*:

$$\mathcal{P} = \{y \in \mathbb{R}^m \mid y = f\,(x)\,,\ x \in \mathcal{D}\}$$

Note that for the evaluation of both functions, f and c, SPICE[1] simulations are necessary [17]. In addition, there is a set of system parameters that remain constant during the design process (e.g., temperature, operating voltages), see [18, p. 15]. For the sake of simplicity, they are omitted in the notation, and their values are given in the text whenever necessary.

The optimization of performance parameters can now be formulated as *multi-objective optimization problem*:

$$\min_{x \in \mathcal{D}} \quad f(x) = [f_1(x), f_2(x), \ldots, f_m(x)] \tag{MOP}$$

Performance criteria are conflicting, as it is usually the case in most multi-objective problems. Therefore, a single combination of design parameters will not optimize all objectives simultaneously (*utopia point*). A solution concept for (MOP) is given by the following:

Definition 1. A point $\bar{x} \in \mathcal{D}$ is called *Pareto solution* for Problem (MOP), if there exists no $x \in \mathcal{D}$ with $f_i(x) \leq f_i(\bar{x})$ for all $i \in \{1, \ldots, m\}$ and with $f_j(x) < f_j(\bar{x})$ for at least one $j \in \{1, \ldots, m\}$.

Images $f(\bar{x})$ of Pareto solutions are said to be *non-dominated* by all other images $f(x)$, $x \in \mathcal{D}$. The set of all non-dominated (image) points of Problem (MOP) is called *Pareto front*. Fig. 2 shows the Pareto front of a two-dimensional image space (🟦) highlighted in red (🟥).

For some applications, not only the Pareto front but the complete performance space is relevant, see [8], [19]. Then, the task is:

$$\text{approximate} \quad \mathcal{P}$$

In this paper, we focus on computing such an approximation in an efficient way by using a *box-based coverage* in the criterion space $\mathcal{P}$. In [9], the concept of an enclosure is introduced, which realizes such a box-based coverage.

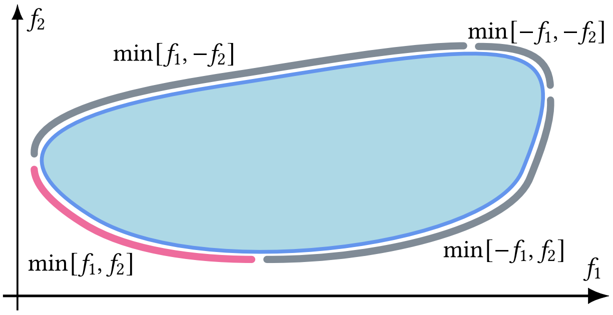


Fig. 2. Two-dimensional performance space (🟦) with Pareto front (🟥) and other fronts (◼).

Since maximizing a function f_i is equivalent to minimizing $-f_i$, we are able to describe all fronts of $\mathcal{P}$ as Pareto fronts of (MOP) by changing the signs of the components f_i of the objective function f, see Fig. 2. This results in 2^m different combinations. We will therefore apply box coverages to all 2^m Pareto fronts in order to obtain an approximation of the complete performance space $\mathcal{P}$. Since the computation of the different Pareto fronts is independent of each other, this can be done in parallel.

B. Box-Based Coverage and Scalarization

The concept of *box-based coverage* [9] is used, e.g., in [10], [11]. Our implementation is based on the algorithm AdEnA in [11, Algo. 3], but contains several modifications. In order to compute not only the Pareto front but the complete performance space $\mathcal{P}$, an efficient parallelization scheme is applied. These improvements will be described later in Sec. III. In the following, the basic procedure for the box coverage of a Pareto front is explained.

The idea behind the approach is to cover the Pareto front by several connected boxes. In any dimension m, a box $\mathcal{B}$ is defined by a lower bound $\ell \in \mathbb{R}^m$ and an upper bound $u \in \mathbb{R}^m$:

$$\mathcal{B} = [\ell, u] = \{y \in \mathbb{R}^m \mid \ell \leq y \leq u\}$$

All lower and upper bounds are stored in the sets $\mathcal{L}$ and $\mathcal{U}$, resp. Then $\left(\mathcal{L} + \mathbb{R}^m_+\right) \cap \left(\mathcal{U} - \mathbb{R}^m_+\right)$ is an *enclosure* of the Pareto front, see Fig. 3 and Fig. 5 (blue (🟦) areas).

Iterative Process: Step (A) – First, all individual minima $\hat{f}_i$ of the functions f_i are computed. These values serve as components for the lower and upper bounds of the initial box (Fig. 3-A).

Step (B) – Using the bounds ℓ and u, the following scalarized version of (MOP) is solved, cf. Fig. 3-B:

$$\begin{aligned} \min_{x \in \mathcal{D}, t \in \mathbb{R}} \quad & t \\ \text{s.t.} \quad & f(x) \leq t\,(u - \ell) + \ell \end{aligned} \tag{SUP(ℓ, u)}$$

Although the solution $\bar{x}$ of SUP(ℓ, u) is only *weakly* Pareto optimal, so that $\bar{y} = f(\bar{x})$ is only *weakly* non-dominated (cf. [11, Def. 2.1 and 2.2]), the correctness of the algorithm is ensured by [11, Lemma 4.7]. To obtain non-weak Pareto solutions, a second scalarized problem would have to be solved,

[1] Simulation Program with Integrated Circuit Emphasis

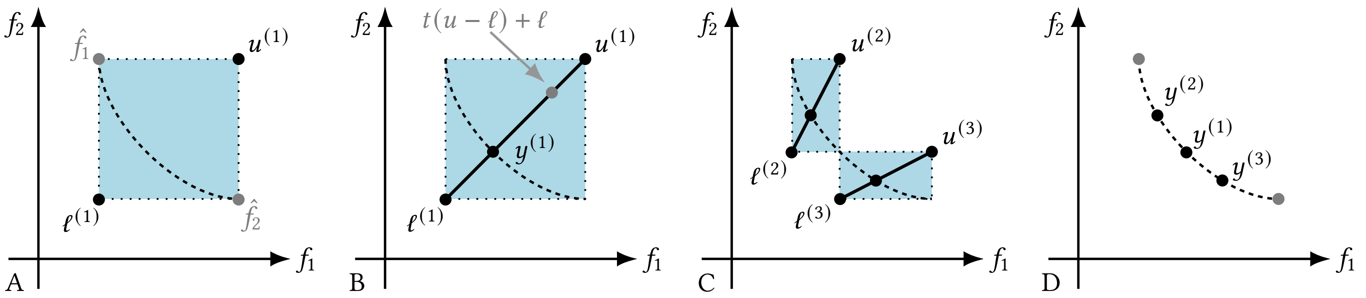


Fig. 3. Box approximation: A – initialization; B – update point computation; C – update of lower and upper bounds; D – resulting points on the Pareto-front.

see [10, Sec. 4.2]. Skipping this second optimization reduces the computational complexity of the approach.

Step (C) – With the solution $(\bar{x}, \bar{t})$ of SUP(ℓ, u), the set $\mathcal{L}$ is updated with $\hat{y} = \bar{t}\,(u - \ell) + \ell$ and the set $\mathcal{U}$ is updated with $\bar{y} = f(\bar{x})$ using [11, Algs. 1 and 2]. Fig. 3-C illustrates this update. Iteratively, all boxes are reduced in size until their smallest edges are smaller than a given tolerance ε. The accuracy of the approximation can be controlled by choosing of $\varepsilon > 0$.

III. Parallelized Box Approximation

We are now able to describe the implementation of our modified box approximation method for computing the complete performance space $\mathcal{P}$. The method consists of several algorithms, which are divided into several parts in order to process the boxes in parallel. Within the `main` algorithm (Fig. 4-left), the `call` function (Fig. 4-right) is executed in a pool of processes. This pool is used to perform the actual subdivision of the boxes and the numerical solution of the scalar optimization problems. The communication between the main process and the pool is done via two queues.

A. Main Algorithm

The `main` algorithm (Fig. 4-left) starts with an initialization, where the lower and upper bounds are computed by individually min- and maximizing all f_i. Then, the `buildBoxes` function creates a list of initial boxes—one for each "Pareto" front (see also Fig. 2). The `sendQ` and `getQ` queues are responsible for the process communication. They prioritize the largest box so that the approximation is systematically refined. In the `pool`, the `call` function is executed (see Fig. 4-right). This fills `sendQ` with all initial boxes, clears the `ToDo` list, and creates an empty `readyBoxes` list. A while loop to refine the box approximation is executed, as long as the stopping criterion (Sec. III-C) is not satisfied. When a single box $\mathcal{B}$ satisfies the stopping criterion, it is transferred into the `readyBoxes` list.

B. Lower and Upper Bounds Update

The `call` function (Fig. 4-right) is responsible for the refinement of boxes by updating the sets of lower and upper bounds. This is done by solving a modified version of the scalarization SUP(ℓ, u). In order to get an approximation of the complete performance space $\mathcal{P}$, and not just its Pareto front, we introduce a sign vector $\rho \in \mathbb{R}^m$, with $\rho_i \in \{-1, +1\}$. Using the Hadamard product for element-wise multiplication, $\rho \odot f(x) \in \mathbb{R}^m$ represents one of the 2^m different combinations for the objective function of (MOP). Each ρ thus characterizes one of the fronts of $\mathcal{P}$. E.g., for Fig. 2, the sign vectors $\rho \in \{[+1,+1], [+1,-1], [-1,-1], [-1,+1]\}$ are used to describe all fronts of $\mathcal{P}$, where $\rho = [+1,+1]$ belongs to the conventional Pareto front. Applying this change to the scalarization gives:

$$\begin{aligned} \min_{x\in\mathcal{D},t\in\mathbb{R}} \quad & t \qquad\qquad\qquad\qquad (\text{FSUP}([s,\gamma]_\rho)) \\ \text{s.t.} \quad & \rho \odot f(x) \le t\,(\gamma - s) + s \end{aligned}$$

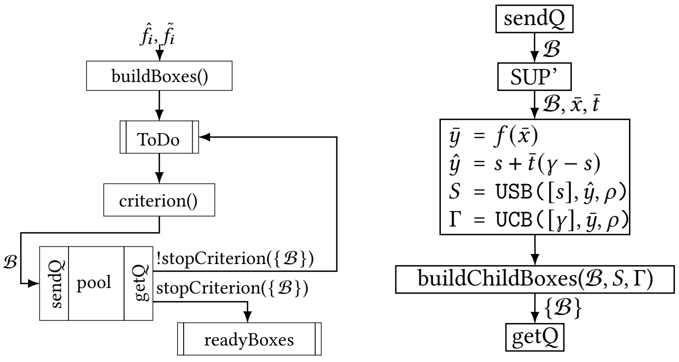

$\hat{f}_i, \tilde{f}_i$
buildBoxes()
ToDo
criterion()
$\mathcal{B}$
sendQ
pool
getQ
!stopCriterion($\{\mathcal{B}\}$)
stopCriterion($\{\mathcal{B}\}$)
readyBoxes
sendQ
$\mathcal{B}$
SUP'
$\mathcal{B}, \bar{x}, \bar{t}$
$\bar{y} = f(\bar{x})$
$\hat{y} = s + \bar{t}(\gamma - s)$
S = USB([s], $\hat{y}$, ρ)
Γ = UCB([γ], $\bar{y}$, ρ)
buildChildBoxes($\mathcal{B}, S, \Gamma$)
$\{\mathcal{B}\}$
getQ

Fig. 4. Flowcharts of the `main` algorithm (left) and the `call` function of the pool (right, based on [11]).

Since the terms "lower" and "upper" bounds could be misleading when computing box coverages for all fronts of $\mathcal{P}$, they are now called (outer) satellite points s (instead of ℓ) and (inner) center points γ (instead of u). Given a sign vector ρ, a box is thus defined by

$$\mathcal{B} = [s, \gamma]_\rho = \{y \in \mathbb{R}^m \mid \rho \odot s \le \rho \odot y \le \rho \odot \gamma\}\,.$$

With the solution $(\bar{x}, \bar{t})$ of (FSUP$([s,\gamma]_\rho)$), the set of satellite bounds is then updated with $\bar{y} = f(\bar{x})$, while the set of center bounds is updated with $\hat{y} = \bar{t}\,(\gamma - s) + s$.

C. Stopping Criterion

A major advantage of the box approximation method over the commonly used NBI method is the availability of useful *stopping criteria*. The simplest criterion would be to set a *maximum number of boxes* to be computed. Since the queues in the `main` algorithm prioritize large boxes, the resulting approximation will be evenly distributed. It has to be defined

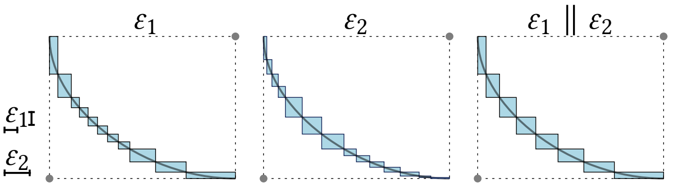


Fig. 5. Comparison of stopping criteria: ε_1, ε_2, and both or-combined.

what *large* means in this context. In [10], the *length of the smallest edge* is used as an indicator. Furthermore, it is not recommended to subdivide a box whose smallest edge length is smaller than a given tolerance ε, which leads to a second stopping criterion. Note that, in practice, this criterion prevents the development of long, thin boxes and can easily be applied in any dimension.

Other possible stopping criteria are related to the *volume* and *diagonal* $\|\gamma - s\|_2$ of the boxes. In practice, both lead to evenly distributed boxes. But a good choice for their tolerances—especially in high dimensions—is not obvious and has to be done by trial and error. This leads to another big advantage of the box approximation scheme: there is the possibility of a *warm start*. If the resulting approximation of $\mathcal{P}$ is not fine enough, it can directly be used as a starting point for another run of the algorithm with smaller tolerances ε or/and a larger maximum number of boxes.

Fig. 5 compares different stopping criteria and their combination for a simple, two-dimensional example (Pareto front only). Here, ε_1 is the tolerance on the length of the smallest edge and ε_2 is the tolerance on the volume and diagonal. With $\varepsilon_1 \parallel \varepsilon_2$, a box is no longer subdivided as soon as one of the criteria is met.

D. Parallelization

The box approximation method is very well-suited for parallelization. The `call` function (Fig. 4-right) is executed completely in parallel by all processes in the pool. As soon as a new box is added to the `sendQ` queue, it is accepted by one of the processes, always giving priority to larger boxes. Therefore, several scalarized problems of type (FSUP($[s, \gamma]_\rho$)) are solved in parallel, allowing further box updates. All resulting child boxes are then passed to `getQ`.

IV. Performance Space Visualization

In previous publications [5], [7], [8], [18], performance spaces (or their Pareto fronts) of dimension $m \in \{2, 3\}$ have been computed. For their visualization, methods such as α-shape [12] are used. Based on a given m-dimensional point cloud, they compute a (non-convex) hull, which can then be visualized, see Fig. 9. Unfortunately, α-shape only works up to dimension $m = 3$, see [13]. Furthermore, exploring these hulls can be difficult for designers.

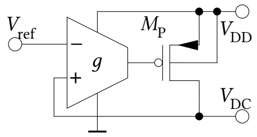
The box approximation method can compute approximations of performance spaces of higher dimensions, such as $m = 7$ in the example of Sec. V. Therefore, new visualization techniques are needed. Since visualizations should enable users to make their circuit design decisions, we provide an approach that allows easy exploration of the performance space. Therefore, the concept of *Parallel Coordinates*, introduced by Inselberg [14]–[16], is used. Parallel Coordinates are a common method for visualizing high-dimensional datasets. Here, all m dimensions are represented as parallel axes, and each m-dimensional (performance) point is represented as a single *polygonal chain* with m vertices. These vertices lie on the parallel axes and mark the coordinates of their associated point. An advantage of this plot over, e.g., radar charts (spider charts) is that all axes can be displayed without distortion and with individual minima and maxima. Since it can be difficult to keep an overview with many points (polygonal chains), it is advisable to use, e.g., colored sections [20]. This is done in our implementation to mark the boundaries of the specification (🟥). In addition, all points matching the specification are highlighted with a darker color (🟦), see Fig. 1.

V. Numerical Experiments

An Intel® Xeon® E5-2667 v3 CPU with 2×16 cores and 128 GB RAM is used for our experiments. The box approximation method is implemented in Python 3.8. Cadence® Spectre® 18.1.0.077 is used for SPICE simulations. All scalar optimization problems are solved by Knitro® 13.2.1 [21] with an AMPL interface [22]. The computation is performed on CPUs, since the simulator does not support GPU acceleration.

A. Analog Low-Dropout Regulator

To demonstrate the efficiency of the proposed (parallelized) box approximation method, we will compute seven-dimensional performance spaces for the development of an *analog low-dropout regulator* (ALDO) as used in the *radio-frequency identification domain* (RFID). With some modifications, we employ an example from [23], where the authors developed a *digital* low-dropout regulator. In this paper, it is designed as *analog* LDO. The corresponding structure and specifications are illustrated in Fig. 6.



Param.	Value	Param.	Value
V_{DD}	1.5 V	P_{diss}	10 µW
V_{DC}	0.8 V	f_{GBW}	8 MHz
V_{ref}	0.8 V	*PM*	80°
$V_{\mathrm{out}}^{\min}$	0.1 V	$V_{\mathrm{out}}^{\max}$	1.4 V

Fig. 6. Schematic and specification of an ALDO consisting of an OTA with gain g and the PMOS M_{P}.

The ALDO consists of a *operational transconductance amplifier* (OTA) and a PMOS M_{P}. A range of 1.0 V to 1.8 V is specified for the voltage V_{DD}. Circuit-1 (see Fig. 7) for the OTA is the OTA-B-N-1 from [8, Tab. 2], and with a folded cascode we use it as more complex Circuit-2 (see Fig. 8). For f_{GBW} the highest system clock from [23] is used. The gain g has only a minimal influence on the remaining control deviation, since the gain of M_{P} is already high enough. $V_{\mathrm{out}}^{\min}$ and $V_{\mathrm{out}}^{\max}$ are set so that M_{P} can be relatively small and still deliver a current from 100 µA to 300 µA.

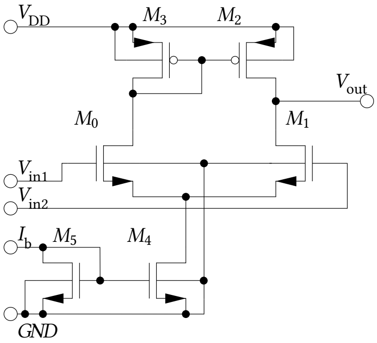


Fig. 7. Circuit-1 based on OTA-B-N-1 from [8, Tab. 2].

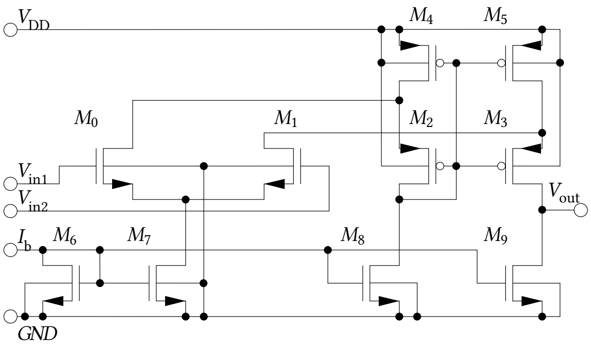


Fig. 8. Circuit-2 based on Circuit-1 with folded cascode.

B. Performance Space Approximation

To illustrate, first the approximations of two-dimensional performance spaces for Circuits-1 and -2 are computed, where gain g competes with bandwidth f_{GBW}. The resulting α-shapes are shown in Fig. 9. It can be seen that Circuit-1—although less complex than Circuit-2—can achieve a higher bandwidth in some cases. Their computations using the box approximation method took just over an hour each, see Tab. I.

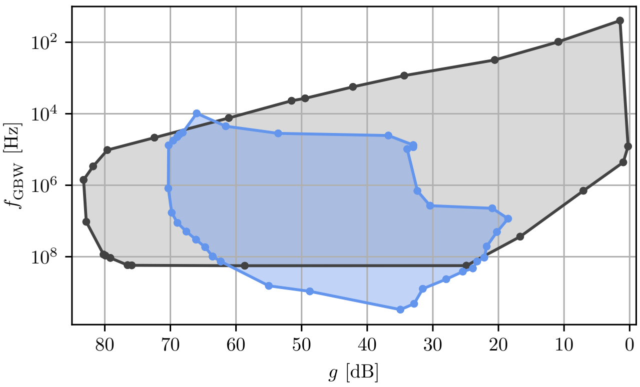


Fig. 9. Two-dimensional performance spaces of ■ Circuit-1 and ■ Circuit-2.

We will now approximate the performance spaces w.r.t. the seven performance parameters phase margin PM, power dissipation P_{diss}, bandwidth f_{GBW}, output voltage limits $V_{\text{out}}^{\text{min}}$, $V_{\text{out}}^{\text{max}}$, gain g, and area A. Fig. 10-left shows the Parallel Coordinates plot of Circuit-1 with nominal settings. The required specification is colored red (■) for each axis. Out of all 3604 computed points (light blue ■) of the performance space, only one point (dark blue ■) meets the requirements. It can be seen that the specification for f_{GBW} has a limiting effect. In [24, p. 270], it is described how f_{GBW} can be increased by increasing the size of the differential input transistors. This is realized by doubling the number of transistor fingers from 1 to 2, deviating from the nominal settings. The results are display in Fig. 10-right. As required, f_{GBW} is increased (by $5.11\,\%$) to reach a range that exceeds the specification. Circuit-2 did not meet the required specification at the nominal settings. We therefore, similar to Circuit-1, doubled the number of transistor fingers. Fig. 11 shows the resulting Parallel Coordinates plot, with the f_{GBW} requirements disabled. As can be seen, there is no feasible combination that meets the specification. The missing gap for f_{GBW} is well illustrated. Circuit-2 therefore does not meet the specification, even though it is more complex than Circuit-1 (with doubled finger count).

C. Computational Effort

The seven-dimensional performance spaces in Sec. V-B were computed in about 15 hours. This means that in practice they can be generated by the begining of the next working day. The computational times given in Tab. I show that the box approximation method is far superior to previous implementations based on the NBI method. We compared our approach to [8, Tab. IV], which used a parallelized implementation of the NBI method to compute performance spaces. Similar hardware was used, and to ensure a fair comparison, the tolerances of the box approximation method (Sec. III-C) are adjusted to obtain a similar number N of computed points in the performance space. To be consistent with the parallelization scheme used in [8], p_b limits the maximum number of boxes that can be run in parallel. For $m = 3$, significant savings in computation time can be seen, especially for fine approximations, where a large numbers of points N are computed. The NBI method would not be able to compute seven-dimensional performance spaces in a reasonable time. In order to validate the efficiency of our parallelized implementation, it is necessary to consider the lower bound given by Amdahl's law [25]. Thus, under ideal circumstances, the execution time could be reduced to the sequential part of the algorithm. Therefore, we compare the ratio of the full computation time to the lower bound t_{min} given by the total computation time of the SPICE simulator:

$$t_{\text{min}} = \frac{t_{\text{init}}}{2m} + \frac{t_{\text{app}}}{p_b}$$

Here, t_{init} is the computation time for simulations during the initialization phase, and t_{app} is the computation time for simulations during the iterative approximation process. While for low dimensions, especially $m = 2$, the overhead due to parallelization is significant, it can be seen that our implementation is very efficient in high dimensions ($m = 7$). Here, the parallelization overhead is only about $6 - 11\,\%$.

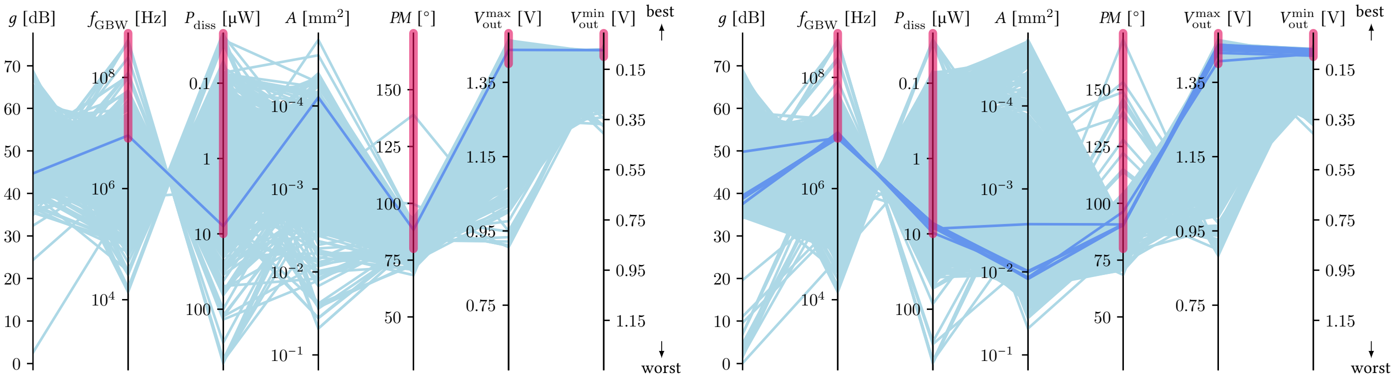


Fig. 10. Performance spaces $\mathcal{P}$ of Circuit-1 with nominal settings (left) and doubled finger count (right).
■ point on approximation of $\mathcal{P}$ ■ specification marker ■ point on approximation of $\mathcal{P}$ that meets specification

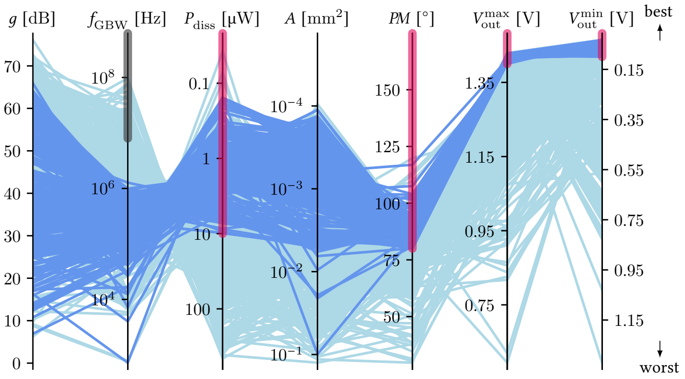


Fig. 11. Performance space $\mathcal{P}$ of Circuit-2 with doubled finger count.
■ point on approximation of $\mathcal{P}$ ■ specification marker ■ deactivated specification marker ■ point on approximation of $\mathcal{P}$ that meets specification

TABLE I
COMPUTATION TIMES IN [HH:MM]

m	N	p_b	NBI [8]	Box Method	$t_{\min}$	$\frac{\text{Box}}{t_{\min}}$ [%]
3	53	1	15:31	10:01	9:27	94.30 %
3	70	8	13:41	2:30	1:49	72.83 %
3	69	64	7:11	1:28	0:37	41.83 %
3	348	8	50:40	8:46	7:56	90.62 %
3	464	64	25:03	3:05	2:14	72.51 %
$7^{(1)}$	3604	64	—	15:18	14:19	93.56 %
$7^{(2)}$	3581	64	—	15:35	14:04	90.23 %
$7^{(3)}$	3488	64	—	15:35	13:57	89.47 %
$2^{(4)}$	47	64	—	1:13	0:22	30.07 %
$2^{(5)}$	46	64	—	1:17	0:20	26.47 %

$^{(1,2,3,4,5)}$ related to Fig. 10 (left), 10 (right), 11, 9 (blue ■), 9 (gray ■)

VI. Conclusion

With the box approximation method, we presented a powerful algorithm for approximating high-dimensional, complete performance spaces for integrated analog circuits. Due to the non-convexity of the underlying multi-objective optimization problem (MOP) and the need for SPICE simulations in each iteration while solving the scalarized problems FSUP($[s,\gamma]_\rho$), this task is computationally challenging. Using an efficient parallelization scheme, it was possible to compute performance spaces for a real-world example up to dimension seven in reasonable time. This is a significant step forward from the three-dimensional performance spaces that have been possible with the (parallelized) NBI method. Comparing our method with the NBI method for three-dimensional examples, the computational time of the box approximation method was significantly lower while achieving comparable results. In addition, the new method offers adjustable stopping criteria, a warm start option, and is easy to implement and parallelize. The prioritized handling of large boxes ensures a uniform approximation of the performance space whenever the method is stopped (early). If the resulting approximation is not fine enough, the algorithm can be continued.

TABLE II
COMPARISON OF BOX APPROXIMATION WITH NBI METHOD.

	NBI Method	Box Method
Pareto front approximation	✓	✓
performance space approx.	✓	✓
dimension independent	✘	✓
easy to implement	✘	✓
parallelizable	✓	✓
stopping criterion	✘	✓
warm start / refinement	✘	✓

Furthermore, we presented a visualization method called *Parallel Coordinates*. The resulting plots allow not only the visualization of high-dimensional performance spaces, but also their exploration in a user-friendly manner. For example, the use of colored markers to highlight desired specifications adds a high degree of interactivity when working with these plots. In the development of an analog low-dropout regulator, we have shown how performance spaces and their visualization with Parallel Coordinates plots can assist the designer in topology selection and circuit design.

References

[1] G. Stehr, *On the Performance Space Exploration of Analog Integrated Circuits*. PhD thesis, TU München, 2005.

[2] I. Das and J. E. Dennis, “Normal-Boundary Intersection: A New Method for Generating the Pareto Surface in Nonlinear Multicriteria Optimization Problems,” *SIAM Journal on Optimization*, vol. 8, pp. 631–657, Aug. 1998.

[3] D. Mueller-Gritschneder, H. Graeb, and U. Schlichtmann, “A Successive Approach to Compute the Bounded Pareto Front of Practical Multiobjective Optimization Problems,” *SIAM Journal on Optimization*, vol. 20, pp. 915–934, Jan. 2009.

[4] D. Schreiber and J. Kampe, “On Applying Pareto Optimization for Complete Performance Space Modeling of Analog ICs,” in *ANALOG 2018; 16th GMM/ITG-Symposium*, pp. 1–6, 2018.

[5] G. Stehr, H. E. Graeb, and K. J. Antreich, “Analog Performance Space Exploration by Normal-Boundary Intersection and by Fourier–Motzkin Elimination,” *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, pp. 1733–1748, Oct. 2007.

[6] D. Schreiber and J. Kampe, “OTA Performance Space Modeling for System-Level Optimization of SC-circuits,” in *2020 27th IEEE International Conference on Electronics, Circuits and Systems (ICECS)*, (Glasgow, UK), pp. 1–4, IEEE, Nov. 2020.

[7] D. Schreiber, B. Ohse, J. Kampe, and C. Schneider, “Performance Space Supported Design of Analog Electronic Circuits,” *IFAC-PapersOnLine*, vol. 55, no. 20, pp. 13–18, 2022.

[8] B. Ohse, D. Schreiber, J. Kampe, and C. Schneider, “Efficient Approximation of Performance Spaces for Analog Circuits via Multi-Objective Optimization,” in *2023 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, (Antwerp, Belgium), pp. 1–6, IEEE, Apr. 2023.

[9] G. Eichfelder, P. Kirst, L. Meng, and O. Stein, “A general branch-and-bound framework for continuous global multiobjective optimization,” *Journal of Global Optimization*, vol. 80, pp. 195–227, May 2021.

[10] G. Eichfelder and L. Warnow, “An approximation algorithm for multi-objective optimization problems using a box-coverage,” *Journal of Global Optimization*, vol. 83, pp. 329–357, June 2022.

[11] G. Eichfelder and L. Warnow, “Advancements in the computation of enclosures for multi-objective optimization problems,” *European Journal of Operational Research*, vol. 310, pp. 315–327, Oct. 2023.

[12] H. Edelsbrunner, D. Kirkpatrick, and R. Seidel, “On the shape of a set of points in the plane,” *IEEE Transactions on Information Theory*, vol. 29, pp. 551–559, July 1983.

[13] H. Edelsbrunner and E. P. Mücke, “Three-dimensional alpha shapes,” *ACM Transactions on Graphics*, vol. 13, pp. 43–72, Jan. 1994.

[14] A. Inselberg and B. Dimsdale, “Parallel coordinates: a tool for visualizing multi-dimensional geometry,” in *Proceedings of the First IEEE Conference on Visualization: Visualization ‘90*, (San Francisco, CA, USA), pp. 361–378, IEEE Comput. Soc. Press, 1990.

[15] A. Inselberg, *Parallel Coordinates: Visual Multidimensional Geometry and Its Applications*. Springer, 2009.

[16] A. Inselberg, “Multidimensional detective,” in *Proceedings of VIZ ’97: Visualization Conference, Information Visualization Symposium and Parallel Rendering Symposium*, pp. 100–107, 1997.

[17] L. W. Nagel, *SPICE2: A Computer Program to Simulate Semiconductor Circuits*. PhD Thesis, EECS Department, University of California, Berkeley, May 1975. Issue: UCB/ERL M520.

[18] D. Müller-Gritschneder, *Deterministic performance space exploration of analog integrated circuits considering process variations and operating conditions*. Elektronik, München: Verl. Dr. Hut, 1. aufl ed., 2009.

[19] H. E. Graeb, *Analog Design Centering and Sizing*. Springer Dordrecht, 2007.

[20] J. Johansson and C. Forsell, “Evaluation of Parallel Coordinates: Overview, Categorization and Guidelines for Future Research,” *IEEE Transactions on Visualization and Computer Graphics*, vol. 22, pp. 579–588, Jan. 2016.

[21] R. H. Byrd, J. Nocedal, and R. A. Waltz, *Knitro: An Integrated Package for Nonlinear Optimization*, pp. 35–59. Boston, MA: Springer US, 2006.

[22] R. Fourer, D. M. Gay, and B. W. Kernighan, “A Modeling Language for Mathematical Programming,” *Management Science*, vol. 36, pp. 519–554, May 1990.

[23] B. Ohse and J. Tan, “Design and Optimization of a Control Algorithm for a Digital Low-Dropout Regulator in System-on-Chip Applications,” in *SMACD / PRIME 2021; International Conference on SMACD and 16th Conference on PRIME*, pp. 1–4, 2021.

[24] T. C. Carusone, D. Johns, K. W. Martin, and D. Johns, *Analog integrated circuit design*. Hoboken, NJ: John Wiley & Sons, 2nd ed ed., 2012. OCLC: ocn751717304.

[25] G. M. Amdahl, “Validity of the single processor approach to achieving large scale computing capabilities,” in *Proceedings of the April 18-20, 1967, spring joint computer conference on - AFIPS ’67 (Spring)*, (Atlantic City, New Jersey), p. 483, ACM Press, 1967.