Inverse Analog IC Sizing and Exploration through Diffusion Models and Structural Knowledge

Filipe Azevedo*[1], Markus Leibl*[2], Ricardo Martins[1] and Helmut Graeb[2]
[1]Instituto de Telecomunicações / Instituto Superior Técnico - Universidade de Lisboa
[2]Technical University of Munich, Chair of Electronic Design Automation
{filipe.p.azevedo@tecnico.ulisboa.pt, markus.leibl@tum.de, ricmartins@lx.it.pt, helmut.graeb@tum.de}

***Abstract*—In the field of analog integrated circuit sizing, the ability to rapidly and efficiently explore design spaces is crucial due to the demands of fast development cycles, evolving specifications, and increasingly complex circuits. It is also well established that leveraging slack in specifications through adjustments to transistor sizes can enhance yield. To address these challenges, we propose a novel approach that combines two state-of-the-art machine learning techniques to accurately generate optimal performance points, while also enabling the exploration of nearby sizing configurations that may result in more robust designs. Specifically, we integrate a diffusion model with an algorithm that analyzes the circuit and decomposes the problem into simpler subproblems. The proposed models are evaluated on a set of typical operational amplifiers.**

I. Introduction

Since the introduction of SPICE, the industry-standard method for designing analog circuits has remained largely unchanged. While the SPICE models have become increasingly complex [1], the approach of manual device tuning persists [2]. Key challenges in making analog CAD tools useful in practice are robustness, ease of use and applicability across a high range of use cases. All those aspects are inherently difficult to grasp on a fundamental level. On the other hand, machine learning (ML) approaches excel at modeling nonlinear, black box phenomena. Their combination with already existing conventional techniques holds significant potential to bridge remaining gaps that prevent the adoption of tools in industry.

Analog design is inherently an inverse problem. While the calculation of performances, yield, etc., can be reliably achieved through simulation, the reverse process - determining the optimal circuit parameters - typically relies on experience, trial and error, and complex numerical methods. As shown in previous works [3], [4], machine learning (ML) methods can be used to accurately reproduce sizings on (Pareto) optimal fronts. Furthermore, it has been shown [5] that diffusion models are well-suited for addressing the inverse sizing problem. Other ML-based works attempted to solve this same problem. In [6], artificial neural networks (ANNs) are used to guess circuit sizings, utilizing only one ANN for modeling an entire netlist, leading to a higher learning task complexity and thus large sample sizes are required to learn the inverse problem within a smaller performance range. In [7], the problem is broken down into smaller tasks, and a cascade of ANNs is trained to sequentially size a single device. Each network in the cascade builds on the outputs of the previous ANNs, following a methodology similar to [3]. The drawback is that for more complex circuits of, e.g., 15 devices, optimally ordering the ANNs for sizing leads to a complexity of $15! \approx 1.3e12$.

Other works follow the traditional approach of tackling the direct sizing problem, treating it as an optimization problem [8]–[11]. In [12] it is shown that starting points chosen via a simpler device model greatly reduce the necessary time for optimization. Reinforcement learning was also studied, with [13], [14] replacing the optimizer with an ANN, leading to improvements in prediction speed, accuracy, and scalability. However, to the best of our knowledge, none of them show strengths in all four categories, speed, accuracy, sample efficiency and exploration.

In this work, we aim for a combination of all those virtues by applying denoising diffusion probabilistic models (DDPM) to the sizing of OpAmps, utilizing the approach in [3] of subdividing the circuits into simpler structures, which has shown high accuracy for low sample sizes. DDPMs have proven their potential for generating strongly varied, but realistic results in other fields, OpenAI's DALL-E and Stable Diffusion being among the most prominent examples. As stated, [5] proved the efficacy of these in the inverse sizing problem.

Unlike most ML models, DDPMs can suggest different sizings for the same problem by repeatedly sampling them. As shown in Section IV, this approach to the inverse problem can produce a more varied set of sizings, which is very helpful when exploring sizing options in the vicinity of performance optima. This work contributes a ML approach that:

- can suggest valid sizings with resulting performance in the vicinity of the required metrics, allowing for exploration of sizings in order to pursue further optimal performance regions, while still fulfilling specifications,
- is highly data efficient, requiring a fraction of the dataset size used in [5] and [6],
- has an accurate expectation of the generated sizings,
- generates sizing solutions at push-button speed.

* *Both authors contributed equally to this work.*

Parts of this work have been done within the collaborative project HoLoDEC funded by the Federal Ministry of Education and Research Germany (BMBF) under the funding code 16ME0705. and also funded by Fundação para a Ciência e Tecnologia – Ministério da Ciência, Tecnologia e Ensino Superior (FCT/MCTES) through national funds and, when applicable cofounded by European Union (EU) funds under the projects UIDB/50008/2020 (DOI identifier 10.54499/UIDB/50008/2020), ACTON (DOI identifier 10.54499/2023.11981.PEX), and also, by Sony Semiconductor Solutions (project GENERALISE).

In section II, we provide an overview of the proposed method and its connection to structural analysis, as well as describing the sizing process in detail. In section III, a short background of DDPMs and our implemented models are presented. We conclude with a presentation of results for two OpAmps, in section IV.

II. From Structural Analysis to Circuit Sizing via Machine Learning

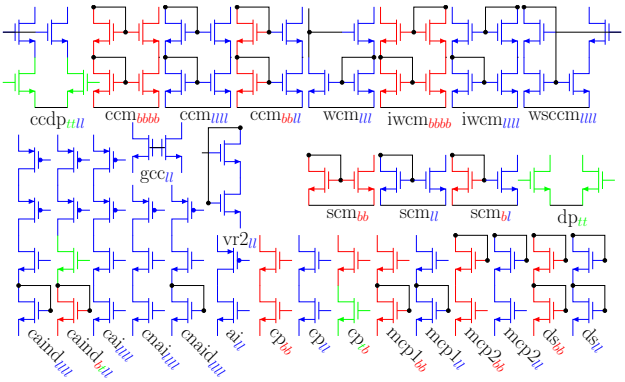


Fig. 1: Splitting various OpAmps into canonical building blocks with different functional characteristics. Indices b, l and t represent *bias*, *load* and *transconductance* functionality of devices. Adapted from [3].

A. Basics of Structural and Functional Blocks

As detailed in [3], the method relies on the decomposition of circuits into subcircuits, which we call structural building blocks. While the structure of the blocks is defined by their devices and interconnections, we identify three functional properties, *bias, load* or *transconductance*, for each individual transistor. This leads to further discrimination among the structural blocks. But because not every combination of functional properties is useful, the total set of possible structures remains small. All possible variants of blocks can be found in Fig. 1.

An example of decomposing a Folded-Cascode OpAmp (FCOA) into blocks can be seen in Fig. 2. Its structural building blocks are shaded, the names annotated close by. Structural and functional block recognition is fully automated. The time required for this is much less than one second.

B. Overview of the Method

In contrast to [3] we use a simplified version of the model. We skip the pretraining step and directly finetune the networks. Furthermore, instead of using gain boosting for sizing of L, we employ the same model type. The sizing process is then straightforward:

- split the opamp into its building blocks via [15].
- train a neural network for each of the buidling blocks, where we use the whole OpAmp performance as guidance for the DDPM model and train to recover only the sizings of the blocks.
- finally, at evaluation time, recombine the predicted sizings. In case of overlap of two blocks, we take the average value.

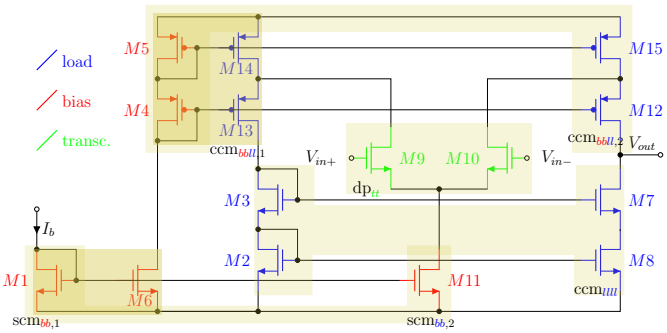


Fig. 2: Folded-Cascode OpAmp (FCOA). The circuit is decomposed into its structural and functional blocks with structural blocks shaded and functionality of the devices represented by colours. Adapted from [3].

Like in the previous work, we enforce symmetry that can be automatically detected by deterministic algorithms. For symmetry within blocks, this is done during training, for symmetry outside of blocks, this is done during recombination.

III. DDPM Background and Implementation

Several DDPMs are trained to learn the sizing and respective performance distribution of the different building blocks that make up an OpAmp. DDPMs were first introduced in [16] as a new diffusion model parametrization, which are a class of ANNs mostly used in an imaging context, like data generation and restoration. These models are composed of 3 different processes, which are briefly highlighted here:

1) Forward Process: First, the training data x_0 is systematically destroyed by the addition of noise over T sequential steps, in the literature called timesteps, until a terminal distribution x_T is reached. This noise is sampled from a Gaussian distribution and at the end of the T timesteps, the resulting training data distribution resembles itself a Gaussian distribution. This process is demonstrated by equation 1, where β_t is the variance schedule, a hyperparameter that controls how much noise is added to the data x at each time timestep t.

$$\begin{aligned} q(x_t|x_{t-1}) &:= \mathcal{N}(x_t; \mu, \Sigma) \\ &:= \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I) \end{aligned} \tag{1}$$

2) Reverse Process: After the data is destroyed until x_t, an ANN is trained to try to reverse the forward process, either by predicting the original data x_0 up front, or by predicting another value that can be used to reconstruct the original data, like the added noise ϵ, etc. This process is represented by equation 2, where $\alpha_t = 1 - \beta_t$ and $\overline{\alpha}_t = \prod_{s=1}^{t} \alpha_s$.

$$\begin{aligned} &\rho_\theta(x_{t-1}|x_t) := \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \\ &:= \mathcal{N}(x_t; \frac{1}{\sqrt{\alpha_t}}(x_t - \frac{1-\alpha_t}{\sqrt{1-\overline{\alpha}_t}}\epsilon_\theta(x_t, t)), \beta_t I) \end{aligned} \tag{2}$$

The correlation between the forward and reverse process to noise and denoise images can be seen in Fig. 3. An overview of the training phase for the FCOA of Fig. 2 can be seen in Fig. 4.

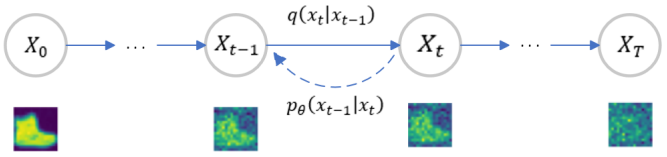


Fig. 3: Correlation between forward and reverse processes [5].

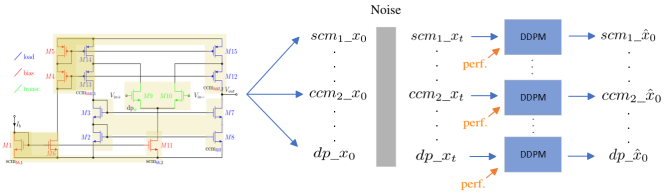


Fig. 4: Training phase for sizing an OpAmp. After the decomposition into sub-blocks, noise is added to the respective sizings. A DDPM is trained for each sub-block to predict the original sizing values. The respective performances of the original sizings are given as guidance.

3) Sampling Process: Finally, after the model is trained, the sampling process is used to generate new data. The model learned to reverse the forward process, and so, when giving it pure Gaussian noise, it tries to reverse the noise addition until arriving at the original data. Except this time, there is no actual data to begin with, marking this a form of generative artificial intelligence. This stage is similar to the training phase shown in Fig. 4, except random noise is given as input to the network instead of the noisy sizings.

It was shown by [5] that DDPMs produce interesting results when applied to the inverse sizing problem. In this work, a similar implementation is considered, with a cosine schedule without clipping [17] for β_t and classifier-free guidance [18], but with 2 key differences:

- instead of predicting the noise ϵ, we implement a model that predicts a velocity equation $v_t = \alpha_t \epsilon - \sigma_t x$ which has been shown to enable a true signal-to-noise ratio of 0 at the last timestep T (meaning that at the last timstep the input of the model is pure Gaussian noise), as opposed to other DDPM parametrizations. This removes an important discrepancy between training (where the model would never see pure noise) and inference [17];
- the backbone of the model implemented is a transformer, with an architecture similar to the "adaLN-Zero" model of [19], as opposed to the simple multi-layer perceptrons implemented by [5]. To predict a more complex value in v, a stronger architecture was required.

IV. Experimental Results

We trained our approach on two different OpAmps, the FCOA previously studied in [3] shown in Fig. 2 and the Miller OpAmp (MOA) in Fig. 5. After training, we sampled the models for all elements in the test sets (containing specification and a ground truth sizing) of both OpAmps, corresponding to 118 for the FCOA and 77 for the Miller OpAmp. After simulating the sampled values, the resulting performances were compared with simulations of the ground truth values.

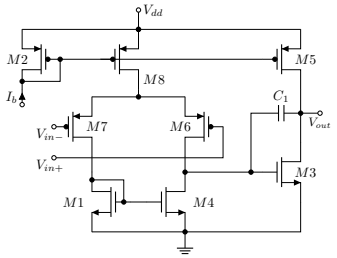
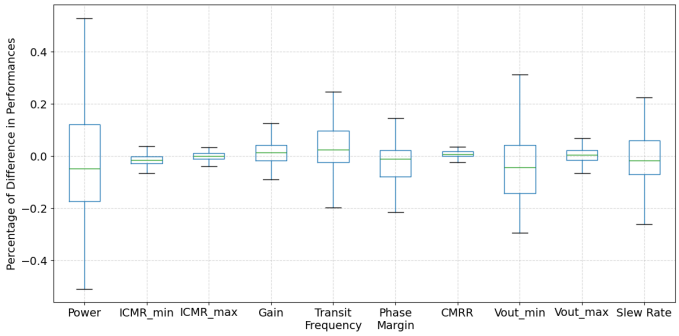
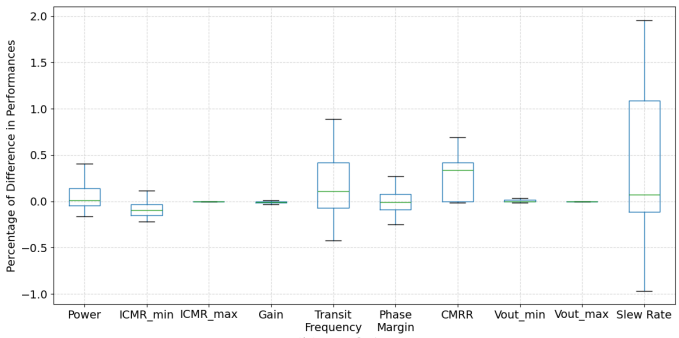


Fig. 5: Miller OpAmp (MOA)



(a) FCOA.



(b) MOA.

Fig. 6: Boxplots of the percentage of differences in performances for the test set of both tested OpAmps, after removal of outliers. The innermost line represents the median, the box 50% of the data.

All models have been trained with a sample size between 700 and 1000 elements. While slightly less efficient as [3], the price for added flexibility, the sample sizes are very low.

Boxplots of the relative performance differences are presented in Fig. 6, bar outliers. For the FCOA in Fig. 6a, most boxes are centered around 0 (i.e. the ground truth), but some performances show a tendency of improvement. For Power, ICMR_{min} and Vout_{min} median and box extend downwards (the predictions' performances are lower than the labels), while Gain and Transit Frequency median and box extend upwards (the predictions' performances are greater than the ground truth sizings). Similar observations can be made for the MOA in Fig. 6b, with the CMRR and Slew Rate having the biggest improvement. Here, the only performance that showed a tendency toward deterioration was Power. This shows that the models are able to accurately learn the distribution of the circuits' sizing and respective performances, sometimes finding interesting improvements that might be attributed to learned trade-offs between sizes of the different sub-blocks.

TABLE I: Estimated mean value and standard deviation of W parameter sizes of the FCOA for 100 samples and same performance specification.

device	mean value $[\mu m]$	standard deviation
0	8.52	2.0
1	43.26	19.0
2	41.74	17.0
3	14.4	1.0
4	90.98	8.0
5	88.47	23.0
6	41.74	17.0
7	43.26	19.0
8	97.88	55.0
9	97.88	55.0
10	114.78	35.0
11	367.21	22.0
12	367.21	22.0
13	244.11	13.0
14	244.11	13.0

In a second experiment we sampled the model for the FCOA 100 times, while keeping the same performance specification. The results can be seen in Fig. 7 and Table I. Most distributions in the performance space in Fig. 7 exhibit a gaussian behaviour, centered around the ground truth performance marked by a red star. Table I indicates that the model learns to estimate the different contributions and sensitivities for the individual parameters. It can be seen that the standard deviation varies heavily depending on the device, indicating that the model learns to scale the effects of parameter deviations.

V. Conclusion

We have shown a machine learning approach to not only reproduce optimal sizings for a given specification, but also to help explore points in the nearby performance space. In the scenario of sampling multiple times for the same specification, the sampling can be interpreted as adding random noise to the ground truth sizing. The model learns the corresponding covariance matrix. As the standard deviation is not equal for all devices, this behaviour cannot be mimicked by simply adding noise to an optimal sizing.

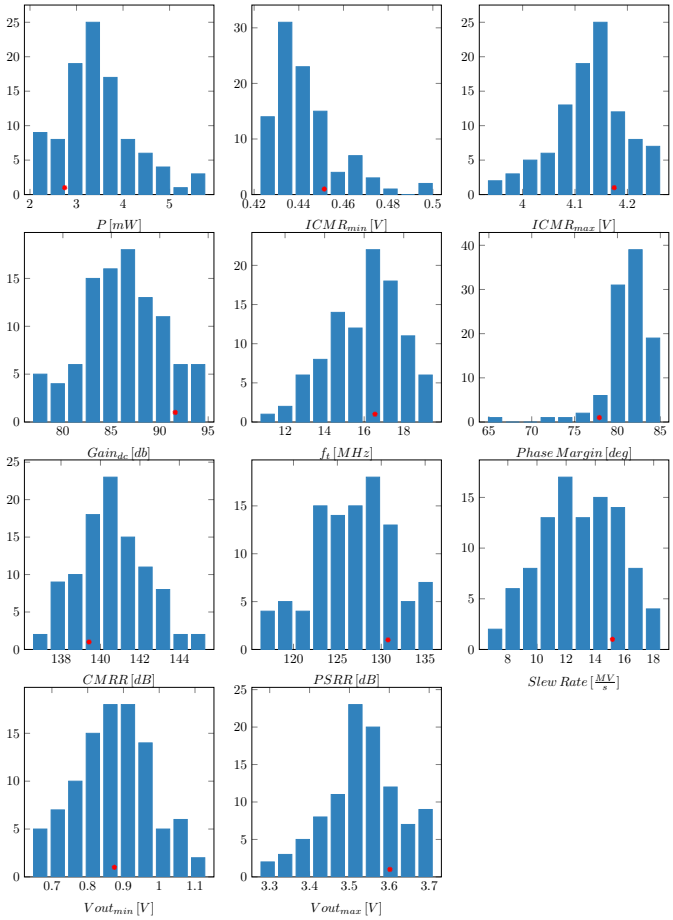


Fig. 7: Histograms for 100 samples of the trained FCOA model with **same** performance specification. Most distributions are well shaped around the ground truth from the test set. A red star marks the performance value of the ground truth sizing.

References

[1] C. Gatermann and R. Sommer, "Teaching the mosfet: A circuit designer's view," in *Int. Conf. on SMACD*, 2022, pp. 1–4.

[2] G. Gielen, "Analog synthesis 3.0: Ai/ml to synthesize and test analog ics: hope or hype ?" in *2023 ACM/IEEE 5th Workshop on Machine Learning for CAD (MLCAD)*, 2023, pp. 1–1.

[3] M. Leibl and H. Graeb, "Optimizer-free sizing of opamps leveraging structural and functional properties," in *Int. Conf. on SMACD*, 2024.

[4] N. Lourenço, E. Afacan *et al.*, "Using polynomial regression and artificial neural networks for reusable analog ic sizing," in *Int. Conf. on SMACD, July 2019*, 2019, pp. 13–16.

[5] P. Eid, F. Azevedo *et al.*, "Solving the inverse problem of analog integrated circuit sizing with diffusion models," in *Int. Conf. on SMACD*, 2024.

[6] N. Lourenço, J. Rosa *et al.*, "On the exploration of promising analog ic designs via artificial neural networks," in *Int. Conf. on SMACD, July 2018*, 2018, pp. 133–136.

[7] P.-O. Beaulieu, E. Dumesnil *et al.*, "Analog rf circuit sizing by a cascade of shallow neural networks," *IEEE TCAD*, vol. 42, no. 12, p. 4391–4401, 2023.

[8] M. Fayazi, M. T. Taba *et al.*, "Angel: Fully-automated analog circuit generator using a neural network assisted semi-supervised learning approach," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 70, no. 11, pp. 4516–4529, 2023.

[9] A. F. Budak, D. Smart *et al.*, "Apostle: Asynchronously parallel optimization for sizing analog transistors using dnn learning," in *Proc. of ASP-DAC, Jan. 2023*, 2023, pp. 70–75.

[10] G. Wolfe and R. Vemuri, "Extraction and use of neural network models in automated synthesis of operational amplifiers," *IEEE TCAD*, vol. 22, no. 2, pp. 198–212, 2003.

[11] K. Hakhamaneshi, M. Nassar *et al.*, "Pretraining graph neural networks for few-shot analog circuit modeling and design," *IEEE TCAD*, vol. 42, no. 7, pp. 2163–2173, 2023.

[12] M. Leibl, A. Lberni *et al.*, "On the importance of initial sizing of analog circuits based on analytical equations," in *31st IEEE ICECS*, 2024.

[13] K. Settaluri, Z. Liu *et al.*, "Automated design of analog circuits using reinforcement learning," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 41, no. 9, pp. 2794–2807, 2022.

[14] J. Zhang, J. Bao *et al.*, "Automated design of complex analog circuits with multiagent based reinforcement learning," in *2023 60th ACM/IEEE Design Automation Conference (DAC)*, 2023, pp. 1–6.

[15] I. Abel and H. Graeb, "Fuboco: Structure synthesis of basic opamps by functional block composition," *ACM Trans. Des. Autom. Electron. Syst.*, vol. 27, no. 6, jun 2022. [Online]. Available: https://doi.org/10.1145/3522738

[16] J. Ho, A. Jain *et al.*, "Denoising diffusion probabilistic models," 2020. [Online]. Available: https://arxiv.org/abs/2006.11239

[17] S. Lin, B. Liu *et al.*, "Common diffusion noise schedules and sample steps are flawed," 2024. [Online]. Available: https://arxiv.org/abs/2305.08891

[18] J. Ho and T. Salimans, "Classifier-free diffusion guidance," 2022. [Online]. Available: https://arxiv.org/abs/2207.12598

[19] W. Peebles and S. Xie, "Scalable diffusion models with transformers," 2023. [Online]. Available: https://arxiv.org/abs/2212.09748