Embedding MATLAB Optimizers in SIMSIDES for the High-Level Design of $\Sigma\Delta$ Modulators

Beatriz Cortés-Delgadillo, Pablo A. Rodríguez-Navas,
Luis I. Guerrero-Linares and José M. de la Rosa, *Senior Member, IEEE*

***Abstract*—This brief shows how to combine SIMSIDES, a SIMULINK-based time-domain behavioral simulator, with different optimization engines available in MATLAB for the automated high-level design of $\Sigma\Delta$ modulators. To this purpose, an updated version of SIMSIDES has been developed, which includes a user-friendly interface that links the simulator with the optimizers, and guides designers through the main steps required to set the design variables, constraints and select the most suitable algorithm to maximize the performance of an arbitrary modulator topology for a given set of specifications. Several examples and results of the optimization procedure are shown to illustrate the benefits of the presented tool for the high-level synthesis of $\Sigma\Delta$ modulators.**

***Index Terms*—Analog-to-digital conversion, sigma-delta modulation, behavioral modeling and simulation, optimization.**

I. Introduction

SIGMA-Delta Modulators ($\Sigma\Delta$Ms) are one of the best techniques to realize Analog-to-Digital Converters (ADCs) for many diverse applications. However, designing efficient $\Sigma\Delta$ ADCs requires optimizing their design parameters at different abstraction levels: from systems to circuits. Although all design steps are required to maximize their performance, one of the most critical synthesis phases takes place at the architectural level, which usually requires a high degree of expertise and know-how about the diverse kinds of $\Sigma\Delta$ loop-filter realizations, impact of circuit error mechanisms, etc [1].

Over the years, a number of approaches have been reported to help designers systematize and maximize the performance of $\Sigma\Delta$Ms [2]–[6]. In the majority of cases, state-of-the-art design methods and CAD tools are based on finding the best set of design variables that optimizes the figures of merit of the modulator. This task is usually carried out by combining a simulator to evaluate the performance and an optimization engine to guide the simulator through the design space in order to obtain the best solution [7]. This process takes sometimes hundreds or even thousands of iterations until the optimum design is found, which requires an accurate–but computationally efficient– way to simulate $\Sigma\Delta$Ms. A well-known simulation method is based on the so-called behavioral-modeling approach, which has been widely used by $\Sigma\Delta$ designers over the years [8]. A good example of behavioral simulation tools is SIMSIDES[1], a time-domain simulator developed in MATLAB/SIMULINK, that includes models for $\Sigma\Delta$M building blocks which have been verified by transistor-level simulations and experimental measurements [3].

In addition to provide a number of signal processing capabilities within a user-friendly interface, MATLAB includes many optimization engines and algorithms that can be combined with behavioral simulation for the high-level synthesis and design of $\Sigma\Delta$Ms. However, many designers–specially those not familiar with optimization procedures–may become confused and lost with the diversity of algorithms available in MATLAB.

This paper contributes to this topic and presents an improved version of SIMSIDES that incorporates an optimization feature, which has been specifically built to assist designers to optimize their $\Sigma\Delta$Ms by combining the benefits of time-domain behavioral simulation with the optimizers available in MATLAB. The Graphical User Interface (GUI) provided by SIMSIDES allows to easily define the design objectives, variables, and constraints, as well as the optimization engine to be combined with the simulator. As a demonstration vehicle, some optimization examples are given to illustrate the use and benefits of the presented toolbox for the high-level synthesis of $\Sigma\Delta$ ADCs.

The paper is organized as follows. Section II gives some background on the optimization-based high-level synthesis methodology of $\Sigma\Delta$Ms. Section III describes the proposed approach based on the combination of SIMSIDES with MATLAB optimization engines. Some optimization examples are shown in Section IV and conclusions are drawn in Section V.

II. Background on Optimization-based High-Level Synthesis of $\Sigma\Delta$Ms

Fig. 1 shows a typical flow diagram of the optimization-based high-level synthesis of $\Sigma\Delta$Ms based on the combination of behavioral simulation (SIMSIDES in this case) as performance evaluation, and an optimizer to explore the design space and find the best (optimum) set of building-block electrical parameters in order to get the maximum modulator performance with minimum cost in terms of energy and silicon area. The starting point in any design consists of selecting the modulator topology, which can be synthesized by using

Manuscript received February 22, 2018; accepted March 26, 2018. This work was partially supported by the Spanish Ministry of Economy and Competitiveness (with support from the European Regional Development Fund) under contracts TEC2013-45638-C3-3-R, TEC2016-75151-C3-3-R, and by "Consejería de Economía, Innovación, Ciencia y Empleo de la Junta de Andalucía", under contract P12-TIC-1481.

Beatriz Cortés-Delgadillo, Pablo A. Rodríguez-Navas, Luis I. Guerrero-Linares and José M. de la Rosa are with the Instituto de Microelectrónica de Sevilla, IMSE-CNM (CSIC/Universidad de Sevilla), C/Américo Vespucio, 41092 Sevilla, SPAIN, e-mail: jrosa@imse-cnm.csic.es.

[1]SIMSIDES can be downloaded for free at the following website: http://www.imse-cnm.csic.es/simsides.

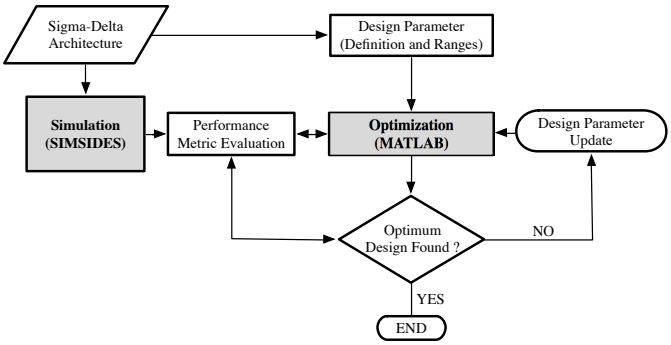

Fig. 1. Flow diagram of the optimization-based high-level synthesis of ΣΔMs using SIMSIDES and MATLAB optimizers.

the well-known Schreier's Delta-Sigma toolbox [9]. Here, the design parameters are the ΣΔM building-block specifications, i.e. the circuit nonideal parameters, which affect the ΣΔM performance and define the electrical specifications of the ΣΔM subcircuits, i.e. integrators, comparators, DAC elements, etc.

Considering arbitrary initial conditions in Fig. 1, a set of perturbations of the design parameters is generated by the optimizer. With the new design parameters, the appropriate simulations are carried out to evaluate the modulator performance and the process is repeated in an iteratively way until the target performance metrics are optimized.

The way in which the optimization-based synthesis procedure of Fig. 1 is implemented strongly depends on the performance evaluator, i.e. the simulator, and the optimization method. Thus, the nature of the algorithms used will affect how the design-variable perturbations are carried out, the number of iterations needed, the dependency on the initial conditions to find either local or global maxima/minima, etc. This work aims to get advantage of the number of optimization methods and algorithms available in MATLAB to be combined with SIMSIDES[2] in a user-friendly way in order to implement the synthesis process of Fig. 1 as described in next section.

III. Using SIMSIDES with MATLAB Optimizers

Fig. 2 shows the main parts of the optimization facility included in SIMSIDES in order to guide designers through the main steps to set and run the high-level synthesis of ΣΔMs. Thus, starting from this optimization menu, designers can choose different ways to automate the high-level synthesis of ΣΔMs, whose behavioral model has been built using SIMSIDES. There are two different alternative ways of running an optimization as discussed below.

A. SIMULINK Optimization Toolbox

The most direct option–depicted in Fig. 2(b)-(c)– consists of directly launching the `SIMULINK Design Optimization` tool. This toolbox provides the necessary functions and tools to assist designers to define the design and optimization problem, involving the definition of design variables, ranges, design objectives, design constraints, etc. as well as to select the solver algorithm, run the optimization and analyse the results. A number of solvers such as `Fmincon` , `Fminsearch` , `Patternsearch`, search methods like `Gradient Descent` ,`Simplex Search` , `Pattern` , etc. as well as diverse optimization algorithms, namely: `Neider-Mead` , `Genetic` , etc. can be used[3].

The `SIMULINK Design Optimization` toolbox can be used with any arbitrary SIMULINK model. However, although a powerful and friendly GUI is provided, the formulation of the optimization problem is not an easy task when applied to the design of ADCs, and particularly to ΣΔMs. The reason is that a suitable performance metric– like for instance the Signal-to-Noise Ratio (SNR)– cannot be directly computed from the optimization toolbox, which is mostly focused on optimizing the frequency response of filters based on fine tuning their Bode diagrams. This limitation aggravates for nonexpert designers, who are not confident when they are starting a new design and they want to do the high-level synthesis of ΣΔMs by combining optimization and behavioral simulation in the MATLAB environment.

B. SIMSIDES Optimization Interface

In order to address the aforementioned problems, a dedicated interface has been developed and embedded in SIMSIDES to help designers combine the benefits of its accurate behavioral models for ΣΔMs with the different optimization methods and algorithms available in MATLAB.

Fig. 3 illustrates an excerpt of the SIMSIDES optimization interface. This optimization menu allows designers to customize their optimization problem by setting all required pieces of information, namely: name of SIMSIDES model, MATLAB script including main simulation parameters, number of design variables, initial values and ranges of variables and the optimization method. To this end, designers need to follow these three steps:

- Build a model of the ΣΔM in SIMSIDES
- Create a MATLAB script with all design variables to be optimized as well as the parameters required to simulate the ΣΔM model
- Enter the information corresponding to the SIMSIDES model and the MATLAB script in the SIMSIDES optimization interface as illustrated in Fig. 3.

It is important to note that the accuracy of the optimization results will strongly depend on how precise the SIMSIDES models are, in terms of the number of circuit non-idealities considered in the models. In addition, depending on the algorithm used, the optimization procedure will be more or less dependent on the initial values of design parameters.

Different performance metrics – such as dynamic range, in-band noise, harmonic distortion, etc – can be used for optimization purposes. Without loss of generality, the SNR has been considered in this work. Therefore, the design objective is to maximize the SNR while optimizing the ΣΔM

[2]SIMSIDES can be also combined with other optimization algorithms not included in MATLAB [3], like multiobjective genetic algorithms such as NSGA-II [6], [10].

[3]The interested reader can find a detailed description of all these optimization methods and algorithms in the MATLAB documentation [11].

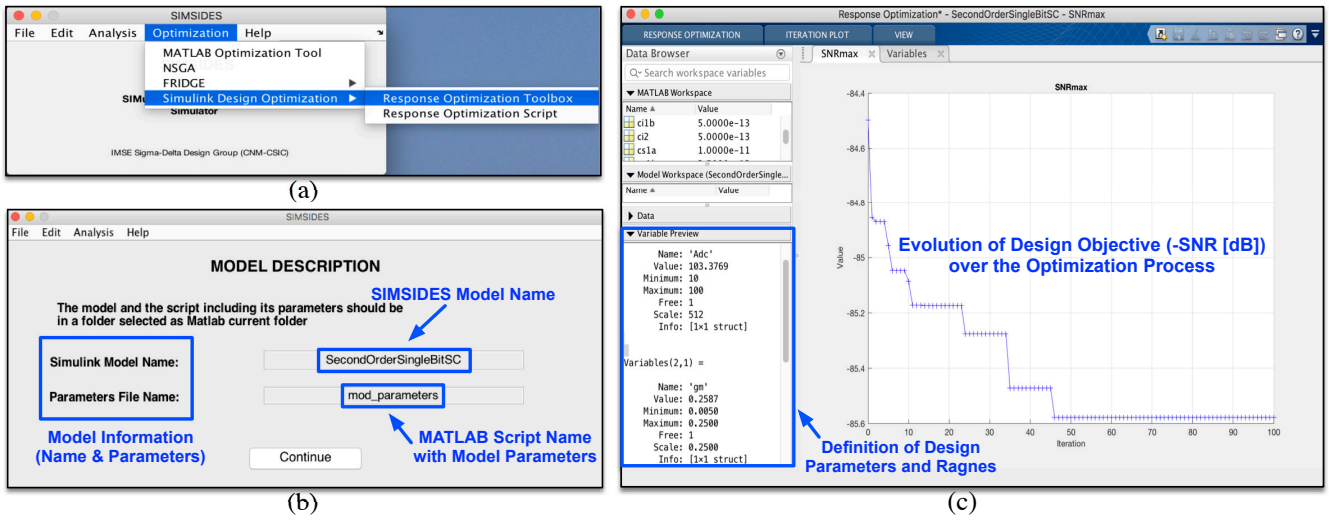

Fig. 2. Combining SIMSIDES and SIMULINK optimization toolbox: (a) SIMSIDES optimization main menu. (b) Launching SIMULINK optimization toolbox from a SIMSIDES model. (c) Optimization toolbox main window, showing the evolution of the design objective over the optimization process.

building-block design variables in order to minimize the power consumption [6]. However, the optimization solvers and algorithms available in MATLAB are intended to minimize a given function, rather than to maximize it – as it is the case here. Therefore, in order to overcome this limitation, an additional block is added to SIMSIDES, which calculates the SNR at the output of the $\Sigma\Delta$M output bitstream and obtains the negative value of the computed SNR, so that the optimization problem can be formulated as:

$$\text{maximize } [f(\bar{x})] = \text{minimize } [-f(\bar{x})] \tag{1}$$

where $f(\bar{x})$ is the performance metric to be optimized – SNR in this case – and $\bar{x}$ denotes the vector of design variables involved in the optimization, i.e. the $\Sigma\Delta$M building-block specifications to be minimized/maximized in order to get the maximum SNR with the minimum power consumption. For the design variables, initial values and ranges need to be entered as depicted in Fig. 3. Also, the algorithm solver and search method used in the optimization are set as well as the maximum number of iterations to be considered in order to limit the CPU time in case the synthesis process does not converge to any solution. During the optimization procedure, the toolbox gives information about both the number of iterations and the number of simulations. The former refers to the number of times the optimization algorithm is run whereas the later stands for the number of times the $\Sigma\Delta$M is simulated.

IV. Case Studies and Optimization Examples

As a demonstration vehicle, let us consider the high-level design of a cascade 2-1 SC-$\Sigma\Delta$M with 1-bit quantization–modeled in SIMSIDES as illustrated in Fig. 4. As stated above, an additional block is added in order to compute the SNR every time the model is simulated. The optimization problem can be formulated into two different ways according to Fig. 3. On the one hand, the design objective may consists of obtaining the maximum SNR while optimizing the design variables. On the other hand, the optimization problem can be focused on optimizing the design variables while achieving a given specification/target for the SNR.

A. Different Ways of Formulating the Optimization Problem

Fig. 5 shows the results of the high-level sizing problem by using a genetic algorithm as optimization engine, by considering the following $\Sigma\Delta$M parameters: sampling frequency, $f_s = 5.12$MHz, input frequency, $f_{\text{in}} = 5$kHz and OverSampling Ratio (OSR)$= 128$. The design parameters taken into account are the finite DC gain, a_o, of the amplifiers and the incomplete settling error, whose main nonideal parameters are the finite transconductance, g_m, and the maximum output current, i_o, of the amplifiers[4]. As the front-end amplifier would be more demanding in terms of electrical specifications, two different set of design parameters are considered, namely: $\{a_{o1}, g_{m1}, i_{o1}\}$, for the front-end integrator, and $\{a_{o2}, g_{m2}, i_{o2}\}$ for the second and third integrators. This way, the design space is made up of six design variables and two different design-target scenarios are considered. The first one–whose results are shown in Fig. 5(a)– aims to achieve the maximum SNR, while the second one–Fig. 5(b)– targets a design objective of SNR $>$ 100dB. In both cases, the optimization-based high-level sizing process aims to minimize the building-block electrical requirements, although in general, less demanding circuit specifications are obtained in Fig. 5(b).

B. Results Obtained with Different Optimizers

In order to illustrate how the presented tool can be used to optimize the design of $\Sigma\Delta$Ms, let us consider the high-level synthesis of the $\Sigma\Delta$M in Fig. 4. Table I compares the results obtained by using different optimization algorithms in

[4]The behavioral model used in this example includes also some additional parameters like the values of sampling and integration capacitors, output swings of integrators, etc. Interested readers can read more details about SIMSIDES behavioral models in [7].

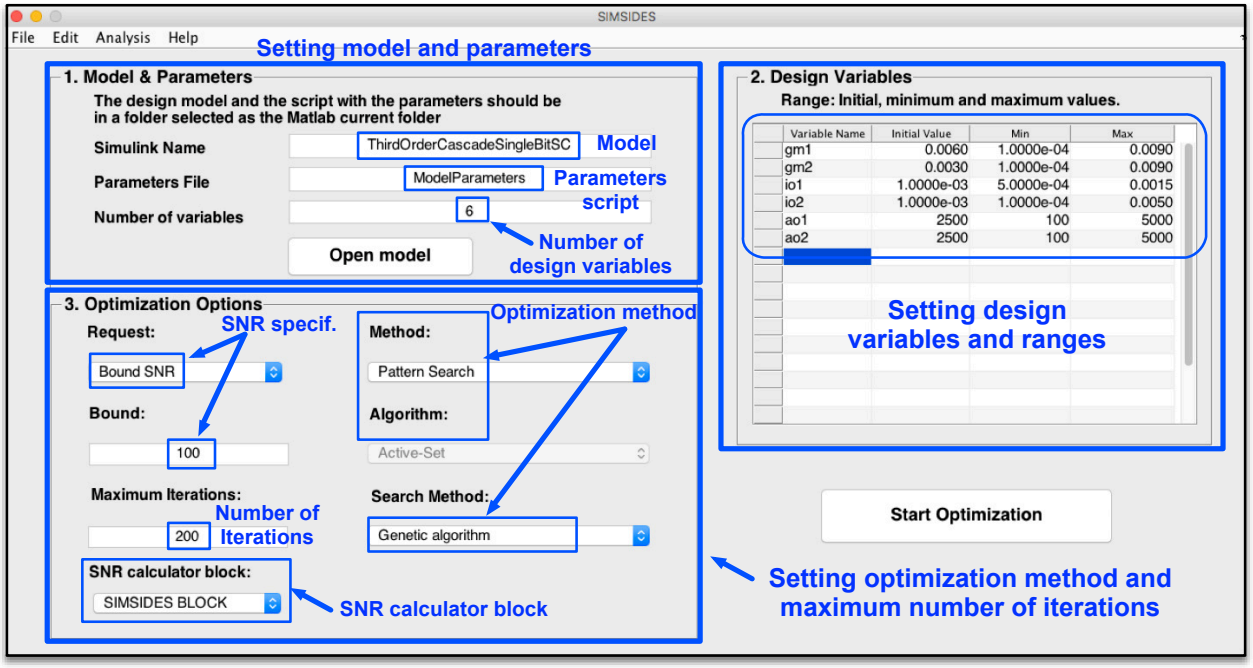

Fig. 3. Illustration of the SIMSIDES optimization interface highlighting its main parts.

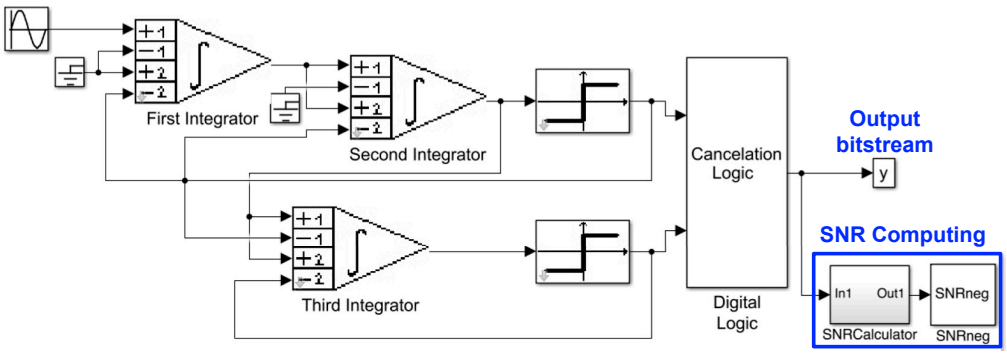

Fig. 4. SIMSIDES model of a cascade 2-1 SC-ΣΔM used as a case study. (It includes main circuit nonidealities such as sampling/integration capacitor nonlinearities, limited output swing, thermal noise, finite DC gain and incomplete settling [7], although only the last two effects will be accounted here).

order to obtain the maximum SNR (denoted as SNRmax) with minimum requirements in terms of finite DC gain and transient response of the amplifiers–which is directly related to the power consumed by the ΣΔM [6]. Note also that the values of SNRmax in Table I will be lower if more circuit non-idealities are considered in the behavioral models [7]. Table I gives also information about the number of simulations required to complete each optimization process, which is in turn directly proportional to the CPU time needed in each simulation[5].

In this case, the best SNR (109.6dB) is achieved by the `Genetic Algorithm` at the price of increasing the number of simulations up to 2039. In contrast, the fastest optimization engines are `Active Set` and `Sequential quadratic programming` [11], obtaining SNR $\geq$ 107.8dB within 61 iterations in both cases. Fig. 6 illustrates the results obtained from this comparative study by

[5]A 2^{16} clock-cycle simulation of the ΣΔM in Fig. 4 takes an average CPU time of 5s in a 4-GHz Intel® Core i7 with 32-GB RAM.

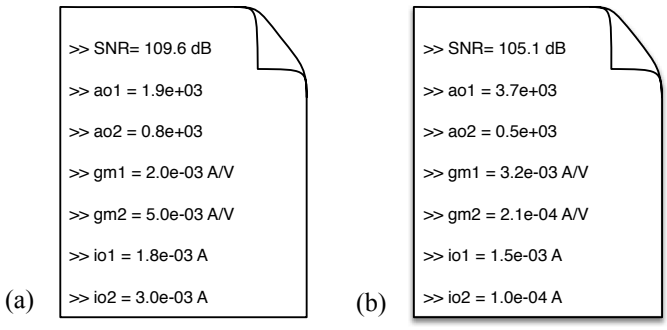

Fig. 5. Results of the high-level synthesis of a cascade 2-1 SC-ΣΔM by combining SIMSIDES with a genetic algorithm. Two objectives are considered: (a) Maximize SNR. (b) Optimize design variables for SNR > 100dB.

showing the final porcentual reduction of the design variables with respect to their initial values (Fig. 6(a)) and the final SNRmax obtained by each optimization method (Fig. 6(b)). It

TABLE I
COMPARISON OF OPTIMIZATION ALGORITHMS FOR THE HIGH-LEVEL SIZING OF THE ΣΔ MODULATOR OF FIG. 4

Algorithm	**Simulations**	**Optimized Design Parameters**						**SNRmax (dB)**
		$a_{o1} \cdot 10^{-3}$	$a_{o2} \cdot 10^{-3}$	g_{m1}(mA/V)	g_{m2}(mA/V)	i_{o1}(mA)	i_{o2}(mA)	
Gradient/Active Set	61	1.55	1.86	6.2	3.9	4.0	4.0	107.8
Gradient/Interior-Point	169	1.32	1.49	2.2	0.7	3.1	2.6	108.2
Gradient/Sequential Quadratic	61	1.56	1.86	6.2	3.9	4.0	4.0	107.9
Pattern/Positive Basis Np1	234	2.0	1.0	2.1	1.2	3.0	4.0	108.9
Pattern/Nelder-Mead	694	2.0	1.2	2.1	2.0	4.0	4.0	109.2
Pattern/Genetic	2039	1.9	0.8	2.0	5.0	1.8	3.0	109.6
Pattern/Latin Hypercube	508	2.0	0.6	3.8	4.8	2.8	3.4	109.3
Simplex Search	277	2.5	2.5	1.0	1.0	5.1	4.9	108.9

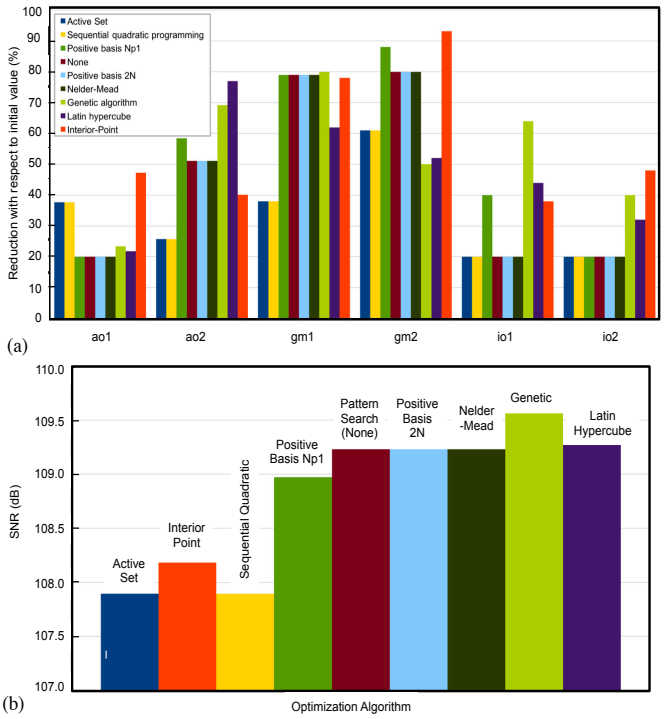

Fig. 6. Illustrating the results obtained by the different optimization algorithms: (a) Percentage reduction of the design parameter with respect to their initial values. (b) Maximum SNR achieved.

can be shown that all optimization methods allows to reduce the values of design parameters with respect to their initial values, becoming more significant for the transconductances.

V. CONCLUSIONS

This paper presented a user-friendly tool to combine optimization and behavioral simulation in the MATLAB/SIMULINK environment for the high-level synthesis of ΣΔMs. To this end, the time-domain simulator SIMSIDES has been improved by adding an interface with the optimization engines and algorithms available in MATLAB. As a case study, the optimized high-level design of a cascade SC-ΣΔM has been carried out by comparing the results obtained by different algorithms. The same methodology can be extended to any arbitrary ΣΔM, by properly building its behavioral model in SIMSIDES and setting the design parameters and performance metrics to be optimized, thus constituting a powerful tool for the design automation of ΣΔ ADCs.

ACKNOWLEDGMENT

The authors would like to thank CETI for their support and anonymous reviewers for their constructive and valuable comments and suggestions to improve the quality of this paper.

REFERENCES

[1] J. M. de la Rosa, "Sigma-Delta Modulators: Tutorial Overview, Design Guide, and State-of-the-Art Survey," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 58, pp. 1–21, January 2011.
[2] K. Francken *et al.*, "DAISY: A simulation-based high-level synthesis tool for ΔΣ modulators," *Proceedings of the 2000 IEEE/ACM Intl. Conference on Computer-Aided Design*, pp. 188–192, 2000.
[3] J. Ruiz-Amaya *et al.*, "High-Level Synthesis of Switched-Capacitor, Switched-Current and Continuous-Time ΣΔ Modulators Using SIMULINK-based Time-Domain Behavioral Models," *IEEE Trans. on Circuits and Systems – I: Regular Papers*, vol. 51, pp. 1795–1810, September 2005.
[4] H. Tang *et al.*, "High-level synthesis of ΔΣ modulator topologies optimized for complexity, sensitivity, and power consumption," *IEEE Trans. Comput.-Aided Des. Integr. Circ. Syst.*, pp. 597–607, March 2006.
[5] T. Bruckner *et al.*, "A GPU-Accelerated Web-based Synthesis Tool for CT Sigma-Delta Modulators," *IEEE Transactions on Circuits and Systems - I: Regular Papers*, vol. 61, pp. 1429–1441, May 2014.
[6] M. Velasco, R. Castro-Lopez, and J. M. de la Rosa, "High-Level Optimization of ΣΔ Modulators Using Multi-Objetive Evolutionary Algorithms," *Proc. of the IEEE Intl. Symp. on Circuits and Systems (ISCAS)*, pp. 1494–1497, May 2016.
[7] J. M. de la Rosa and R. del Río, *CMOS Sigma-Delta Converters: Practical Design Guide.* Wiley-IEEE Press, 2013.
[8] P. Malcovati *et al.*, "Behavioral modeling of switched-capacitor sigma-delta modulators," *IEEE Trans. on Circuits and Systems – I: Regular Papers*, vol. 50, pp. 352–364, March 2003.
[9] R. Schreier, *Delta-Sigma Toolbox.* [Online]. Available: http://www.mathworks.com/matlabcentral/fileexchange/19., 2016.
[10] K. Deb *et al.*, "A Fast and Elitist Multiobjective Genetic Algorithm: NSGA-II," *IEEE Trans. on Evolutionary Computation*, vol. 6, pp. 182–197, April 2002.
[11] Mathworks, "Optimization Toolbox User's Guide," *The Mathworks Inc.*, 2016.