Subgradient Based Multiple-Starting-Point Algorithm for Non-Smooth Optimization of Analog Circuits

Wenlong Lv[1], Fan Yang[1]*, Changhao Yan[1], Dian Zhou[1,2] and Xuan Zeng[1]*
[1]State Key Lab of ASIC & System, School of Microelectronics, Fudan University, Shanghai, P. R. China
[2]Department of Electrical Engineering, University of Texas at Dallas, Richardson, TX, U.S.A.

***Abstract*—Starting from a set of starting points, the multiple-starting-point optimization searches the local optimums by gradient-guided local search. The global optimum is selected from these local optimums. The region-hit property of the multiple-starting-point optimization makes the multiple-starting-point approach more likely to reach the global optimum. However, for non-smooth objective functions, e.g., worst-case optimization, the traditional gradient based local search methods may stuck at non-smooth points, even if the objective function is smooth "almost everywhere". In this paper, we propose a subgradient based multiple-starting-point algorithm for non-smooth optimization of analog circuits. Subgradients instead of traditional gradients are used to guide the local search of the non-smooth optimization. The Shor's R algorithm is used to accelerate the subgradient based local search. A two-stage optimization strategy is proposed to deal with the constraints in analog circuit optimization. Our experiments on 2 circuits show that the proposed method is very efficient for worst-case optimization. The proposed approach can achieve much better solutions with less simulations, compared with the traditional gradient based method, smoothing approximation method, smooth relaxation method and differential evolution algorithms.**

***Index Terms*—Worst-case optimization; Shor's R algorithm; Multiple-starting-point**

I. Introduction

Due to the continuous scaling of the IC technology and the growing demand for higher performance and lower power, manual analog circuit design is hard to meet the performance and power requirements with limited time to market. Automated analog circuit design has attracted great interest in recent years, as it improves circuit performance while reduces the design time. Recently, the optimal device sizing attracts many research interests because it can be well formulated as constrained nonlinear optimization problems and thus possibly be efficiently solved by the well-developed optimization algorithms.

Most analog circuit optimization algorithms fall into two categories: model based and simulation based methods. For model based methods, analytical expressions of the circuits performances are required. The expressions can be obtained by manual analysis or fitting [1]. A well-known model based algorithm is Geometric Programming(GP) [2], where the circuit performances are expressed as posynomials [3]. The posynomial approximation guarantees the optimization problem is convex and the global optimum can be found. Recent advances show that global optimums can be found even for general polynomial optimization problems. The idea is to transform the general polynomial optimization problem to convex programming by Semidefinite-Programming (SDP) relaxations [4]. The advantage of model based algorithms is that given the circuit performance models, the optimal solutions can be found very efficiently even for large-scale problems. But what limits their usage is the generation of accurate performance models. Even with a large number of simulations, the accuracy of the generated performance models still cannot be guaranteed for the whole design space.

The simulation based analog circuit optimization methods require no explicit analytical performance models. SPICE simulations are used to guide the optimization directly. Simulated Annealing (SA) [5], Genetic Algorithms (GA) [6], particle swarm optimization (PSO) [7] and differential evolution (DE) [8] are best-known simulation based approaches. The stochastic characteristics of these algorithms help them to better explore the solution space and avoid falling into the local optimum. Although the above mentioned algorithms can efficiently explore the design space, they suffer from relatively low convergence rate. Note that design specifications like gain, bandwidth, phase margin are smooth and differentiable over the whole design space. As a result, gradients can be used to efficiently guide the local search. It has been reported in [9]-[10] that gradient based local search with multiple starting points (MSP) is very efficient for these smooth performances. If one starting point is located in a valley, it converges rapidly to the local optimum by the local search. The region-hit property of the multiple-starting-point optimization makes the multiple-starting-point optimization approach more likely to reach the global optimum.

In order to improve the robustness of the design under PVT variations, the optimization problem can be reformulated by optimizing the worst performance over several PVT corners. Compared with Monte-Carlo based yield optimization, the worst-case optimization tends to be more pessimistic. However, the worst-case optimization is widely used by designers and accepted by industry. It turns to be a de facto design methodology in industry [10]-[11].

Another example of non-smooth objective function is the rail-to-rail amplifier. The amplifier should satisfy all specifications with input common mode voltage varying from 0 to V_{dd} [12], which means that the transconductance g_m of input stage should keep constant across whole common mode input voltage range. Such an objective can be formulated as optimizing the worst performance with several representative input voltages, e.g., 0, $0.25V_{dd}$, $0.5V_{dd}$, $0.75V_{dd}$, and V_{dd}.

*Corresponding authors: {yangfan, xzeng}@fudan.edu.cn.

The worst-case optimization poses new challenges to gradient-based local search and MSP algorithms. For worst-case optimization which is expressed as the maximum, minimum and absolute value of the smooth performance functions, the objective functions and constrained functions are thus smooth "almost everywhere" but can be non-smooth at some zero-measured subsets of design space.

For the aforementioned cases, the traditional gradient-based methods may stuck at non-smooth points, even if the objective function is smooth "almost everywhere". But we should note that the gradients are still very useful to guide the search and the gradients are available "almost everywhere". The traditional methods for non-smooth optimization try to smooth the non-smooth functions by approximations such as log-sum-exp approximation [13]. However, these smooth approximations would deviate the real non-smooth objective functions and thus degrade the optimization quality. Smooth relaxation transforms the non-smooth optimization problem to smooth optimization problem by introducing new constraints, however, the smooth relaxation approach introduces extra constraints. The objective function and the constraints are tightly coupled in the relaxed formulation, which degrades the convergence rate.

In this paper, we propose a subgradient based multiple-starting-point algorithm for non-smooth optimization of analog circuits. Subgradient instead of traditional gradient is used to guide the local search of the non-smooth optimization. The Shor's R algorithm is used to accelerate the subgradient based local search. A two-stage optimization strategy is proposed to deal with the constraints in automated analog circuit design. In the first stage, we move the solution to the feasible regions. In the second stage, the objective function is minimized while keeping the constraints been satisfied. Our experiments on 2 circuits show that the proposed method is very efficient for worst-case optimization. The proposed approach can achieve much better solutions with less simulations, compared with the traditional gradient based method, smoothing approximation method, smooth relaxation method and differential evolution algorithms.

The rest of paper is organized as follows. In Section II, the background of non-smooth optimization is discussed. The proposed method is presented in Section III. Numerical experiments on are reported in section IV. We conclude the paper in Section V.

II. Background

In this section, we will present the formulation of the worst-case optimization problem firstly. Then, we will review the multiple-starting-point algorithm. Finally, we will review the traditional gradient based local search and its limitations to deal with non-smooth optimization.

A. Problem Formulation

A typical analog circuit optimization problem can be formulated as the following constrained optimization problem.

$$\begin{aligned} \text{minimize} \quad & f(\boldsymbol{x}) \\ \text{s.t.} \quad & c_i(\boldsymbol{x}) < 0 \\ & \forall i \in 1 \ldots N. \end{aligned} \tag{1}$$

In (1), $\boldsymbol{x}$ is the vector formed by circuit design parameters, $f(x)$ is the objective function to be minimized, which corresponds to the FOM (Figure of Merits) of the circuit. $c_1, c_2, \ldots, c_N$ are N constraints which should be met during the optimization. Typically, the objective function $f(x)$ and the constraints $c_1, c_2, \ldots, c_N$ are smooth.

If multiple scenarios are considered, we aim to minimize the worst-case FOM while the constraints are still satisfied. For M scenarios, (1) can be reformulated as

$$\begin{aligned} \text{minimize} \quad & \max(f_1(\boldsymbol{x}), \ \ldots, \ f_M(\boldsymbol{x})) \\ \text{s.t.} \quad & c_{i,j}(\boldsymbol{x}) < 0 \\ & \forall i \in 1 \ldots N \\ & \forall j \in 1 \ldots M. \end{aligned} \tag{2}$$

Compared to (1), the objective function in (2) became non-smooth. The number of constraints increase to $M \times N$. The number of constraints grows linearly with number of scenarios. For analog circuit optimization problem, most constrains are mapped from design specifications, which would be tight and difficult to satisfy during the optimization. The $M \times N$ constraints can also be rewritten as a single non-smooth constraints $g(\boldsymbol{x}) < 0$ as follows.

$$\begin{aligned} \text{minimize} \quad & F(\boldsymbol{x}) \\ \text{s.t.} \quad & g(\boldsymbol{x}) < 0, \end{aligned} \tag{3}$$

where $g(\boldsymbol{x}) = \max(c_1(\boldsymbol{x}), \ \ldots, \ c_{M\times N}(\boldsymbol{x}))$, and $F(\boldsymbol{x}) = \max(f_1(\boldsymbol{x}), \ \ldots, \ f_M(\boldsymbol{x}))$.

The minimax formulation in (3) makes the objective function less correlated with constraints, as in most design space, only one FOM and one design specifications dominate the whole objective function and constraint function. A discussion of the advantages of minimax formulation can be seen in [14].

B. Multiple-Starting-Point Algorithm

Multiple-Starting-Point (MSP) algorithm is an efficient algorithm for general constraint optimization problem. Starting from a set of starting points, the multiple-starting-point optimization searches the local optimums by gradient guided local search. The global optimum is selected from these local optimums.

These starting points are generated uniformly or using quasi-random methods. If one starting point is located in a valley, by using the gradient-guided search, e.g., Sequential Quadratic Programming (SQP) [15], it converges quickly to the corresponding local optimum, which means that this starting point covers this valley. With sufficient number of starting points, they will cover all the valleys. The multiple-starting-point algorithm thus has high probability to find the global optimum. The region-hit property of the multiple-starting-point makes it very efficient for global optimization.

However, gradients are required to guide the optimization for multiple-starting-point algorithm. For the non-smooth optimization problem in (3), there exist non-smooth points, where the gradients are unavailable. But we should note that the gradients are still very useful to guide the search and the gradients are available "almost everywhere".

C. Traditional Gradient Based Methods for Non-Smooth Optimization

If a function is non-smooth only at several zero-measured sets, it is natural to apply the normal gradient-based searching methods directly. Since the gradients are obtained by numerical difference in optimization, the gradients can still be obtained even in the non-smooth points. However, from our experiments, with such a strategy, the optimization may stuck at non-smooth points. We use a simple example in (4) to demonstrate it.

$$f(x, y) = 3|x + y| + |x - y|. \tag{4}$$

Equation (4) is a convex function, which looks like an inverted pyramid. It is smooth in all its domain except the set $M = \{\boldsymbol{x} \,|\, |x_1| = |x_2|\}$. We use multiple-starting-point algorithm with BFGS method for local search to optimize the function. 100 starting points were randomly generated. The function contour, all the starting points and all the finally converged points are shown in Fig 1.

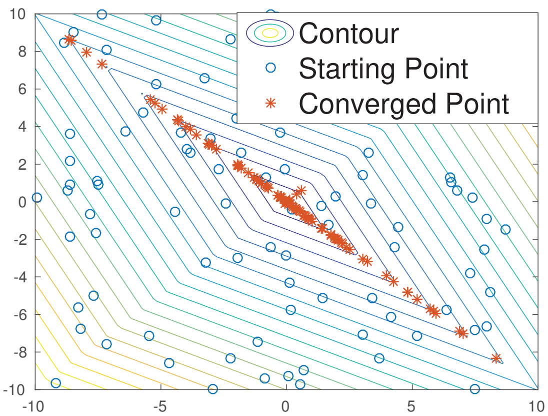

Fig. 1. Multiple-starting-point algorithm with BFGS method for local search on $3|x + y| + |x - y|$.

As shown in Fig 1, most starting points stuck at line $y = -x$ and do not converge to the local minimum. Actually, the function minimum is $f(0, 0) = 0$, while only about less 30% of the optimized results are less than 1.

Several attempts to improve the gradient-based methods for non-smooth optimization have been made. For example, a special line search method was introduced in [16] to avoid visiting the points where the objective functions are non-smooth. It has been proven that the BFGS method could still converge for 1-D functions in [16]. An alternative approach to deal with the non-smooth functions is to approximate the non-smooth functions by smooth function. For example, the log-sum-exp approximation [13] has been widely used in the placement of VLSI, which approximate the max function $\max(x_1, x_2, \ldots, x_D)$ by log-sum-exp approximations $\frac{1}{\alpha}\log(\sum_{i=1}^{D} e^{\alpha x_i})$. However, we need to manually select the parameter α to control the smoothness of approximated function, and the calculation of exponential function may cause numerical instability. More importantly, the smooth approximation would deviate the real non-smooth objective functions and thus degrade the optimization quality.

Smooth relaxation is another option for minimax optimization. We can convert the non-smooth optimization problem into a constrained smooth optimization problem. For example, if we want to minimize $F(\boldsymbol{x}) = \max(f_1(\boldsymbol{x}), \ldots, f_N(\boldsymbol{x}))$, we can convert it to the following equivalent formulation.

$$\begin{aligned} \text{minimize} \quad & \lambda \\ \text{s.t.} \quad & f_i(\boldsymbol{x}) < \lambda \\ & \forall i \in 1, \ldots, N. \end{aligned} \tag{5}$$

III. Subgradient Based Multiple-Starting-Point Algorithm for Non-Smooth Optimization

In this section, the subgradient based multiple-starting-point algorithm for non-smooth optimization of analog circuits will be presented. We follows the basic idea of multiple-starting-point algorithm but propose to use a subgradient based Shor's R Algorithm for local search. A two-stage optimization strategy is proposed to deal with the constraints during the optimization.

A. Subgradient and Shor's R Algorithm

A subgradient [17] of a function $h(\boldsymbol{x})$ at point $\boldsymbol{x}$, is define as any vector $\boldsymbol{g}$ that satisfies

$$h(\boldsymbol{z}) \le h(\boldsymbol{x}) + \boldsymbol{g}^T(\boldsymbol{z} - \boldsymbol{x}) \;\forall \boldsymbol{z} \in \operatorname{dom} h, \tag{6}$$

where $\operatorname{dom} h$ denotes the definition domain of function h.

The subdifferential ∂h of the function at point $\boldsymbol{x}$ is defined as the convex hull formed by all its subgradients. For example, for function $h(x) = |x|$, for all $x < 0$, $\partial h(x) = \{-1\}$; for all $x > 0$, $\partial h(x) = \{1\}$; at point $x = 0$, $\partial h(0)$ is in the interval $[-1, 1]$. Any value in that interval is a subgradient.

Subgradient has the following properties. Firstly, if a function $h(\boldsymbol{x})$ is smooth at point $\boldsymbol{x}$, then its only subgradient in $\partial h(\boldsymbol{x})$ is its gradient. Secondly, a function h is differentiable at point $\boldsymbol{x}$ if and only if its gradient is its only subgradient. Thirdly, a point $\boldsymbol{x}$ is a minimum of f if and only if $\boldsymbol{0} \in \partial h(\boldsymbol{x})$.

For the max function in our worst-case optimization (3), the subgradient can be obtained as follows. Consider the objective function $F(\boldsymbol{x}) = \max(f_1(\boldsymbol{x}), \ \ldots, \ f_M(\boldsymbol{x}))$ in (3). If $F(\boldsymbol{x}) = f_i(\boldsymbol{x})$, which means $f_i(\boldsymbol{x})$ is the only maximum among $f_1(\boldsymbol{x}), \ \ldots, \ f_M(\boldsymbol{x})$, the subgradient of $F(\boldsymbol{x})$ equals to the gradient of $f_i(\boldsymbol{x})$. If $F(\boldsymbol{x}) = f_{i_1}(\boldsymbol{x}) = f_{i_2}(\boldsymbol{x}) = \cdots = f_{i_s}(\boldsymbol{x})$, which means $f_{i_1}(\boldsymbol{x}), f_{i_2}(\boldsymbol{x}), \cdots, f_{i_s}(\boldsymbol{x})$ are s maximums among $f_1(\boldsymbol{x}), \ \ldots, \ f_M(\boldsymbol{x})$, the convex hull formed by those gradients is the subdifferential of $F(\boldsymbol{x})$. Any vector inside the convex hull can be selected as the subgradient for $F(\boldsymbol{x})$.

Subgradient based method [18] can be used to optimize the non-smooth functions. It is similar to the gradient descending method but differs in the following two aspects. It uses subgradients instead of gradients to guide the search. It can thus be applied to non-smooth functions. As the search direction is not always a descending direction and the traditional amoji-wolfe line search condition [19] is not well defined for the non-smooth cases. It usually uses a pre-defined step length sequence instead of incorporating line search method. Basic subgradient based method for minimizing $f(x)$ is described in Algorithm 1.

The step length α_k in Algorithm 1 is a predefined sequence. It had been proven that subgradient method would converge to the minimum as long as diminishing [18] condition is satisfied, i.e., the step length converges to 0 as the iteration steps. The frequently used sequence of a_k are $a_k = \frac{1}{k}$ and $a_k = \frac{1}{\sqrt{k}}$ [18].

Algorithm 1 Basic Subgradient Method for Non-Smooth Optimization

Input: $\boldsymbol{x}_1$ as the starting points
$k \leftarrow 1$
while not terminated **do**
  calculate subgradient $\boldsymbol{g}_k \in \partial f(\boldsymbol{x}_k)$
  $\boldsymbol{x}_{k+1} \leftarrow \boldsymbol{x}_k - \alpha_k \boldsymbol{g}_k$
  $k \leftarrow k + 1$
end while
return best $f(\boldsymbol{x})$ recorded during iterations

Although subgradient method [17] has a theoretical guarantee to converge, the convergence rate would be relatively slow in some cases. In [20], a space dilation method called Shor's R was proposed to improve the efficiency of the original subgradient method. At each iteration, it calculates the difference between a subgradient at the current point and that calculated at the previous step. The direction obtained is used to perform dilation of the space with a priori given coefficient.

Algorithm 2 Shor's R Algorithm

Input: $\boldsymbol{x}_0 \in R^n,\ \beta \in (0, 1)$
1: $B \leftarrow I$
2: calculate subgradient $\boldsymbol{g} \in \partial f(\boldsymbol{x}_0)$
3: $\tilde{\boldsymbol{g}} \leftarrow B^T \boldsymbol{g}$
4: calculate α from line search
5: $\boldsymbol{x} \leftarrow \boldsymbol{x}_0 - \alpha B \tilde{\boldsymbol{g}}$
6: **while** not terminated **do**
7:   calculate subgradient $\boldsymbol{g} \in \partial f(\boldsymbol{x})$
8:   $\boldsymbol{g}^* \leftarrow B^T \boldsymbol{g}$
9:   $\boldsymbol{r} \leftarrow \boldsymbol{g}^* - \tilde{\boldsymbol{g}}$
10:   **if** $|\boldsymbol{r}| = 0$ **or** $|\tilde{\boldsymbol{g}}| \leq 10^{-15}|\boldsymbol{g}|$ **then**
11:     $B \leftarrow I$ {Reset B for ill-conditioned problem}
12:   **else**
13:     $\boldsymbol{s} \leftarrow \frac{\boldsymbol{r}}{|\boldsymbol{r}|}$
14:     $B \leftarrow B(I + (\beta - 1)\boldsymbol{s}\boldsymbol{s}^T)$
15:   **end if**
16:   $\tilde{\boldsymbol{g}} \leftarrow B^T \tilde{\boldsymbol{g}}$
17:   calculate α from line search
18:   $\boldsymbol{x} \leftarrow \boldsymbol{x} - \alpha B \tilde{\boldsymbol{g}}$
19: **end while**
20: **return** best $f(\boldsymbol{x})$ recorded during iterations

The implementation would significantly affect the efficiency the algorithm. An efficient implementation of Shor's R algorithm is presented in [21], which is described in Algorithm 2. In Algorithm 2, line 9 calculates the difference between the subgradient at the current point and that calculated at the previous step. Line 13-14 calculate the space dilation matrix. Line 16-18 calculate the update of x with the space dilation matrix and line search.

The implementation of Shor's R algorithm in [21] improves the original Shor's R algorithm in the following aspects, i.e., the initial step choice, the reinitialization strategy and the line search method. In Algorithm 2, once a starting point is sampled and its subgradinet g_0 is calculated, the initial trial step for the initial line search in line 4 of Algorithm 2 used by [21] is expressed as

$$h_0 = \frac{1}{\log_2(|g_0| + 1)}. \tag{7}$$

Another modification, as described in line 11 of algorithm Algorithm 2, is that once the space transformation matrix B became ill-conditioned, B is reinitialized to identity matrix.

Thirdly, instead of using pre-defined step length sequence, [21] adopted a special line search approach. Once a search direction is specified, it tries to find a point in that direction such that the minimizer in that direction locates between the current point and the next point, the details of the line search strategy can be found in [21].

B. Two-Stage Subgradient Based Local Search for Non-Smooth Optimization

The analog circuit optimization problem is formulated as a constrained optimization problem as shown in (3). However, Shor's R algorithm does not consider the constraints. On the other hand, these constraints are very important for analog circuit optimization. They are remarkably tight to guarantee the performances of the analog circuits. During optimization, most of the time were spent on finding the first feasible solution. We thus propose a two-stage optimization strategy for the constrained non-smooth optimization problem in (3). In the first stage, we move the solution to the feasible regions. In the second stage, the objective function is minimized while keeping the constraints satisfied.

Once a starting point is selected, constraint function $c(\boldsymbol{x})$ in (3) is tackled firstly. We use Shor's R algorithm to minimize $c(\boldsymbol{x})$, the iteration stops once $c(\boldsymbol{x})$ is minimized to a feasible value or a local minimum of $c(\boldsymbol{x})$ is found.

If $c(\boldsymbol{x})$ converged to a infeasible local minimum, the local search stops. If a feasible point is found, we turn to optimize the objective function. In this stage, the constraints are used as a logarithmic barrier function [22]. Assume the feasible point we found after the first stage is $\boldsymbol{x}_c$ and $c(\boldsymbol{x}_c) = c_0 < 0$, the following penalty function $p(\boldsymbol{x})$ is constructed.

$$p(\boldsymbol{x}) = \begin{cases} -\log(-\frac{c(\boldsymbol{x})}{|0.1c_0|}) & c(\boldsymbol{x}) < 0 \\ +\infty & otherwise. \end{cases} \tag{8}$$

As shown in (8), if constraints are violated at point $\boldsymbol{x}$, the penalty would be ∞, which restricts the solutions inside the feasible region.

The feasible point $\boldsymbol{x_c}$ is used as the starting point of second stage optimization. In the second stage, the new objective function $F(\boldsymbol{x}) + p(\boldsymbol{x})$ is minimized. The two-stage Shor's R algorithm is described in Algorithm 3.

Algorithm 3 Two-Stage Shor's R Optimization

while not terminated **do**
  Find $\boldsymbol{x}_c$ where $c(\boldsymbol{x}_c) < 0$.
  if $\boldsymbol{x_c}$ is found **then**
    Minimize $F(\boldsymbol{x}) + p(\boldsymbol{x})$, $\boldsymbol{x}_c$ is used as starting point.
  end if
end while
return best $F(\boldsymbol{x})$ recorded during iterations

C. Stop criteria

As the local minimizer could locate in a non-smooth point, and Shor's R algorithm allows non-descending search direction, the following stop criteria are used in our local search algorithm.

- For a smooth point, check whether the gradient is zero or below a threshold ϵ_g.
- For non-smooth point, check whether zero is inside the subdifferential.
- Change of x in one iteration is less than a tolerance Tol_x.
- Objective function improvement in the last 15 iterations is less than a tolerance Tol_f.

If any of the aforementioned conditions is satisfied, the local search stops.

IV. Numerical Experiment

We tested our proposed subgradient based multiple-starting-point algorithm on 2 circuits: a three-stage amplifier, a rail-to-rail amplifier. For all the 2 test circuits, HSPICE 2012.06 is used as the circuit simulator. Our optimization method is implemented in C++. The experiments are conducted on a Linux workstation with 2 Intel Xeon CPUs and 128G memory.

We compared our proposed subgradient based multiple-starting-point algorithm (RAlgo for short) with the following algorithms.

1. Multiple-starting-point algorithm with gradient-based SQP for local search, where the gradients are obtained by numerical difference of the original non-smooth functions (Nonsmooth for short).
2. Multiple-starting-point algorithm with gradient-based SQP for local search, where the non-smooth functions are approximated by log-sum-exp smoothing (Smooth for short).
3. Multiple-starting-point algorithm with gradient-based SQP for local search, where the non-smooth functions are tackled by smooth relaxation (Relaxation for short).
4. Differential Evolution method (DE for short).

It should be noted that the implementation would significantly affect the efficiency the algorithm.

A. Three-Stage Amplifier

The first example is a three-stage amplifier [23]-[24] as shown in Fig 2. This circuit is implemented in standard $0.35\,\mu m$ CMOS process.

5 process corners are considered, each has its own specification values, as listed in Table I. In this experiment, we select the widths and lengths of transistors, resistance of resistors, bias current, and compensation capacitors as design variables. The number of design parameters is 24. The maximum Iq of 5 corners is selected as the objective function, while all other design specifications are considered as constraints. We limit the number of simulations to 10000 for all the algorithms. We run all the methods for 10 times to average the random fluctuations. The comparison of all algorithms for their averaged results is shown in Table II.

As shown in Table II, our algorithm RAlgo clearly achieved the best optimization result with the least number of simulations. Actually, 7 out of 10 runs of our algorithm finally get less than $40\mu A$ maximal Iq over 5 corners, while none of other algorithms reached $40\mu A$ even for once in their 10 runs.

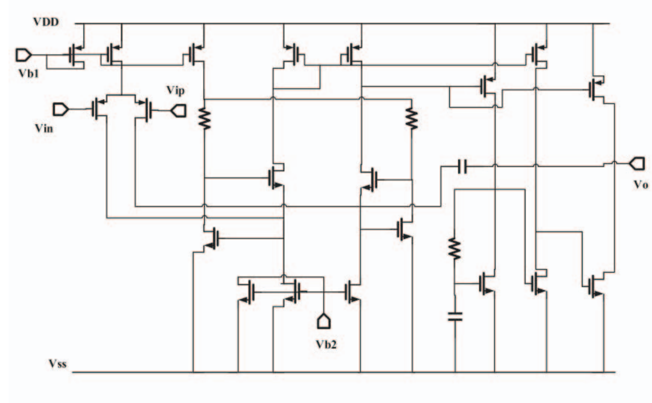

Fig. 2. Three-stage amplifier.

TABLE I
DESIGN SPECIFICATIONS OF THE THREE-STAGE AMPLIFIER.

Corner	TT	FF	SS	SF	SF	Type
GBW(MHz)	0.92	1.08	0.78	1.04	0.82	Maximize
PM(Degree)	52.5	50.3	55.6	50.6	55.2	Maximize
GM(dB)	19.5	20.2	19.3	19.1	20.3	Maximize
SR+(V/us)	0.18	0.26	0.14	0.21	0.16	Maximize
SR-(V/us)	0.20	0.27	0.15	0.16	0.25	Maximize
Iq(μA)	69.2	78.6	63.2	78.0	63.7	Minimize

B. Rail-to-Rail Amplifier

A rail-to-rail amplifier with class AB output stage is used as the second example. The circuit schematic is shown in Fig 3, The circuit is implemented by $0.35\,\mu m$ standard CMOS process. The supply voltage V_{dd} equals to $3.3\,V$.

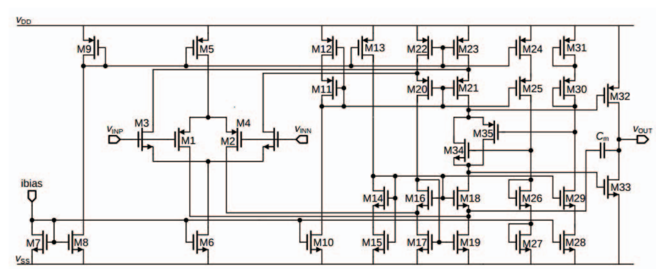

Fig. 3. Rail-to-rail amplifier.

The design specifications are listed in Table III. We aim to maximize the DC-Gain of the circuit, while the Unit Gain Frequency (UGF), Phase Margin (PM) and Power constraints should be satisfied.

In this experiment, we select the widths and lengths of transistors, the resistance of resistors, the capacitance of capacitors and the bias current as design variables. The number of the design variables is 34. To meet the rail-to-rail requirement, the amplifier should satisfy all specs with input common mode voltage varying from 0 to V_{dd}. 5 representative common mode input voltages 0, $0.825\,V$, $1.65\,V$, $2.475\,V$, $3.3\,V$ are selected. We optimize the worst-case performance of the circuit under these representative input voltage modes.

We compare the optimization results of each algorithm with a limited number of simulations. The limit for our

TABLE II
COMPARISON OF DIFFERENT ALGORITHMS ON THREE-STAGE AMPLIFIER.

	RAlgo	Nonsmooth	Smooth	Relaxation	DE
Iq	**38.31**	44.91	47.27	50.18	48.51
Feasible	**10/10**	10/10	10/10	10/10	10/10
Spec Satisfied	**10/10**	10/10	10/10	10/10	10/10
# Simulations	**5337**	5801	5934	5796	7747

TABLE III
DESIGN SPECIFICATIONS OF THE RAIL-TO-RAIL AMPLIFIER.

Name	Spec	Type
DC-Gain	$>70\,\text{dB}$	Maximize
UGF	$>200\,\text{MHz}$	Constraints
PM	$>60°$	Constraints
Power	$< 6.6\,mW$	Constraints

algorithm (RAlgo) is set to 25000, while the limits for all other algorithms are set to 50000. We run all the methods for 10 times to average out the random fluctuations.

TABLE IV
COMPARISON OF DIFFERENT ALGORITHMS ON RAIL-TO-RAIL AMPLIFIER.

	RAlgo	Nonsmooth	Smooth	Relaxation	DE
DC Gain	**83.95**	59.84	Fail	46.78	Fail
Feasible	**10/10**	2/10	0/10	2/10	0/10
Spec Satisfied	**10/10**	0/10	0/10	0/10	0/10
# Simulations	**13818**	41212	50000	45467	50000

As shown in Table IV. Our algorithm (RAlgo) finds out the feasible solutions for all 10 runs with much less number of simulations, and all the feasible solutions satisfy the design specifications. The multiple starting point algorithm with traditional SQP for local search (Nonsmooth) and the smooth relaxation method (Relaxation) for local search find feasible solutions in only 2 runs. But the DC-gains of these 4 feasible solutions are less than $70\,dB$, which means they do not meet the specification requirement as shown in Table III. Smooth approximation (Smooth) and DE performed poorly for this circuit. They failed to find feasible solutions in all 10 runs.

V. CONCLUSION

In this paper, we proposed an efficient algorithm for nonsmooth optimization of analog circuit. Shor's R algorithm with multiple starting points is used for non-smooth global optimization. A two-stage strategy is used to deal with constraints. Experiments on two test cases show that our proposed method can achieve better optimization result compared with the traditional gradient-guided methods and the differential evolution for non-smooth worst-case optimization of analog circuits.

ACKNOWLEDGMENT

This research was supported partly by National Science Foundation of China (NSFC) research projects 61376040, 61474026, 61574046, 61274032, 61574044 and 91330201, partly by Chen Guang project supported by Shanghai Municipal Education Commission and Shanghai Education Development Foundation, partly by the fund of CAEP-SCNS(R2016-03-010), and partly by the Recruitment Program of Global Experts (the Thousand Talents Plan).

REFERENCES

[1] W. Daems, G. Gielen, and W. Sansen, "Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 22, no. 5, pp. 517–534, 2003.

[2] M. Hershenson, "Analog circuit design via geometric programming," *IEICE transactions on fundamentals of electronics communications and computer sciences*, no. 2, pp. 298–310, 2004.

[3] Wikipedia, "Posynomial — wikipedia, the free encyclopedia," 2015, [Online; accessed 3-September-2016]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=Posynomial&oldid=662464608

[4] Y. Wang, C. Caramanis, and M. Orshansky, "Polygp: Improving gp-based analog optimization through accurate high-order monomials and semidefinite relaxation," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 1423–1428.

[5] C.-R. Hwang, "Simulated annealing: theory and applications," *Acta Applicandae Mathematicae*, vol. 12, no. 1, pp. 108–111, 1988.

[6] J. B. Grimbleby, "Automatic analogue circuit synthesis using genetic algorithms," *IEE Proceedings-Circuits, Devices and Systems*, vol. 147, no. 6, pp. 319–323, 2000.

[7] R. Vural and T. Yildirim, "Analog circuit sizing via swarm intelligence," *AEU-International Journal of Electronics and Communications*, vol. 66, no. 9, pp. 732–740, 2012.

[8] R. Storn and K. Price, "Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces," *Journal of global optimization*, vol. 11, no. 4, pp. 341–359, 1997.

[9] G. Huang, L. Qian, S. Saibua, D. Zhou, and X. Zeng, "An efficient optimization based method to evaluate the drv of sram cells," *IEEE Transactions on Circuits and Systems I: Regular Papers*, vol. 60, no. 6, pp. 1511–1520, June 2013.

[10] D. Zhou and G. Huang, "Design automation of analog circuit considering the process variations," in *ASIC (ASICON), 2013 IEEE 10th International Conference on*, Oct 2013, pp. 1–4.

[11] C.-W. Lee, "On-chip benchmarking and calibration without external references," 2011.

[12] R. Hogervorst and J. Huijsing, *Design of low-voltage, low-power operational amplifier cells*. Springer Science & Business Media, 2013, vol. 374.

[13] Wikipedia, "Logsumexp — wikipedia, the free encyclopedia," 2016, [Online; accessed 5-September-2016]. [Online]. Available: https://en.wikipedia.org/w/index.php?title=LogSumExp&oldid=732488346

[14] F. Leyn, W. Daems, G. Gielen, and W. Sansen, "Analog circuit sizing with constraint programming modeling and minimax optimization," in *Circuits and Systems, 1997. ISCAS '97., Proceedings of 1997 IEEE International Symposium on*, vol. 3, Jun 1997, pp. 1500–1503 vol.3.

[15] P. T. Boggs and J. W. Tolle, "Sequential quadratic programming," *Acta numerica*, vol. 4, pp. 1–51, 1995.

[16] A. Lewis and M. Overton, "Nonsmooth optimization via BFGS," *Submitted to SIAM J. Optimiz*, pp. 1–35, 2009. [Online]. Available: http://www.cs.nyu.edu/ overton/papers/pdffiles/bfgs_inexactLS.pdf

[17] S. Boyd and L. Vandenberghe, "Subgradients," *Notes for EE364b, Stanford University, Winter 2006-07*, no. x, pp. 1–7, 2008.

[18] S. Boyd, L. Xiao, and A. Mutapcic, "Subgradient methods," *lecture notes of EE392o, Stanford ...*, vol. 1, no. May, pp. 1–21, 2003.

[19] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[20] N. Z. Shor, *Minimization methods for non-differentiable functions*. Springer Science & Business Media, 2012, vol. 3.

[21] F. Kappel and A. V. Kuntsevich, "An implementation of shor's r-algorithm," *Computational Optimization and Applications*, vol. 15, no. 2, pp. 193–205, 2000.

[22] K. Bryan and Y. Shibberu, "Penalty functions and constrained optimization," *Dept. of Mathematics, Rose-Hulman Institute of Technology. http:// www. rosehulman. edu/~ bryan/lottamath/penalty. pdf*, 2005.

[23] Z. Yan, P. I. Mak, M. K. Law, and R. P. Martins, "A 0.016-mm2 144-μW three-stage amplifier capable of driving 1-to-15 nF capacitive load with $>$ 0.95-MHz GBW," *IEEE Journal of Solid-State Circuits*, vol. 48, no. 2, pp. 527–540, 2013.

[24] B. Peng, F. Yang, C. Yan, X. Zeng, and D. Zhou, "Efficient Multiple Starting Point Optimization for Automated Analog Circuit Optimization via Recycling Simulation Data," in *Design Automation, Test in Europe Conference & Exhibition (DATE), 2016*, 2016, pp. 1417–1422.