

# Automated Design of Operational Transconductance Amplifiers using Reversed Geometric Programming

Johan P. Vanderhaegen  
jpv@eecs.berkeley.edu

Robert W. Brodersen  
rb@eecs.berkeley.edu

Berkeley Wireless Research Center  
Department of Electrical Engineering and Computer Science  
University of California at Berkeley  
2108 Allston Way, Suite 200, Berkeley, CA 94704

## ABSTRACT

We present a method for designing operational amplifiers using reversed geometric programming, which is an extension of geometric programming that allows both convex and non-convex constraints. Adding a limited set of non-convex constraints can improve the accuracy of convex equation-based optimization, without compromising global optimality. These constraints allow increased accuracy for critical modeling equations, such as the relationship between  $g_m$  and  $I_{DS}$ . To demonstrate the design methodology, a folded-cascode amplifier is designed in a  $0.18\mu\text{m}$  technology for varying speed requirements and is compared with simulations and designs obtained from geometric programming.

### Categories and Subject Descriptors:

B.7.2[Integrated Circuits]: Design Aids

**General Terms:** Algorithms, Design

**Keywords:** CMOS integrated circuits, operational transconductance amplifiers, reversed geometric programming

## 1. INTRODUCTION

The evolution of the microelectronics industry is characterized by an ever increasing level of integration and complexity. This trend has resulted from the industry's ability to exponentially decrease over time the minimum feature sizes used to fabricate integrated circuits. The greatest threat to the continuation of this evolution is the cost of design.

The recent trend towards systems-on-a-chip forces the rapid adoption of "digital" semiconductor technologies for analog/RF circuits and a shortening of the analog design cycle to coincide with the digital design cycle [7]. Analog/RF circuits account for an increasing portion of the design cost in mixed-signal chips, even though they typically occupy only a small fraction of the total chip area. This motivates analog design automation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

DAC'04, June 7–11, 2004, San Diego, California, USA.  
Copyright 2004 ACM 1-58113-828-8/04/0006 ...\$5.00.

An important component in automated analog design is circuit synthesis, which consists of topology selection and circuit sizing. The first attempts at the analog sizing problem tried to capture the analog design knowledge heuristically, which proved to be too inflexible. Sizing methods using general purpose optimization techniques, whereby a cost metric, such as power or area, is minimized under a set of performance constraints, have been the subject of active research since the late 1980s [7] and offer the advantage of being straightforward to evaluate, and hence suitable for large and complex designs, but have the limitation of possibly finding local minima.

However, by using convex optimization, a global minimum can be computed with great efficiency [12]. Geometric programming is one particular form of convex optimization, which traditionally has been used for engineering design [5, 1], and more recently also for analog circuit sizing [9].

But this great computational efficiency comes at a price: the objective and the constraints have to be convex functions of the design variables. In section 3, we will give several examples of constraints which occur in analog design and which are essentially non-convex. Non-convex constraints are hard to approximate accurately with convex constraints, and this approximation leads to poorly optimized circuits.

Instead of approximating non-convex constraints with convex constraints, we can trade some of the efficiency in computing the solution for modeling accuracy in the constraints. For example, reversed geometric programming [4], which is introduced in section 2.3, allows non-convex reversed constraints in addition to the regular convex constraints.

## 2. REVERSED GEOMETRIC PROGRAMMING

### 2.1 Geometric programming

Consider the following optimization problem [5]:

$$\begin{aligned} \text{GP : } & \underset{\mathbf{x} \in \mathbb{R}^n}{\text{minimize}} && g_0(\mathbf{x}) \\ & \text{subject to} && g_k(\mathbf{x}) \leq 1 \quad \text{for } k = 1, \dots, m \\ & && h_l(\mathbf{x}) = 1 \quad \text{for } l = 1, \dots, n \\ & && \mathbf{x} > 0 \end{aligned}$$

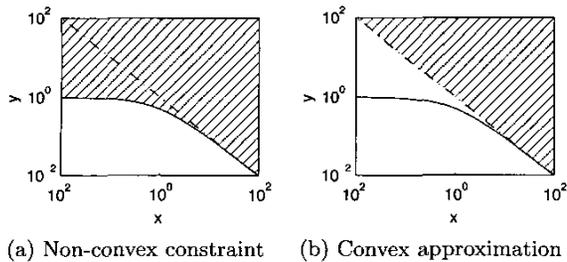


Figure 1: A non-convex constraint and a convex approximation.

$$\text{where } g_k(\mathbf{x}) \text{ is of the form } \sum_i c_i \prod_j x_j^{a_{i,j}}$$

$$h_l(\mathbf{x}) \text{ is of the form } c \prod_j x_j^{a_j}$$

The functions  $g_k(\mathbf{x})$  are called posynomials and the functions  $h_l(\mathbf{x})$  monomials; both posynomials and monomials can only have strictly positive coefficients  $c_i$ , but real exponents  $a_{i,j}$ . An optimization program of the form **GP** is called a geometric program.

Notice that the equality constraints can only have a single term, since they have to be monomials.

It is well-known that the optimization program **GP** can be cast in a convex form by transforming the variables according to  $y_j = \log x_j$  [5]. Hence, efficient computational techniques for general convex optimization problems, such as interior-point algorithms, can be applied to the transformed problem [12].

## 2.2 Convex constraints

Constraints which are not in a posynomial form can be approximated, e.g. by the techniques in [5, p. 98-101]. If the constraint is convex, it can be very well approximated by a set of posynomial constraints. However, if the original constraint is non-convex, the general behavior of the constraint will be lost by approximating it with a convex constraint. For example, consider the following non-convex constraint:

$$y \geq \frac{1}{1+x}$$

Figure 1 shows the constraint, together with a convex approximation:  $y \geq \frac{1}{x}$ . The shaded areas correspond to the sets of points satisfying the original non-convex constraint (figure 1(a)) and the convex approximation (figure 1(b)). The approximation is exact at one point ( $x = +\infty$ ), but is more restrictive than the original constraint everywhere else.

Replacing the non-convex constraint with a convex constraint inevitably leads to an over-design of the problem. Therefore, incorporating the non-convex constraint directly into the optimization program is necessary to make the correct trade-offs.

In section 3, we give several examples of constraints which occur in analog design and which are essentially non-convex.

## 2.3 Reversed geometric programming

Consider the following optimization problem [4]:

$$\begin{aligned} \text{RGP : } & \underset{\mathbf{x} \in \mathcal{R}^n}{\text{minimize}} && g_0(\mathbf{x}) \\ & \text{subject to} && g_k(\mathbf{x}) \leq 1 \quad \text{for } k = 1, \dots, m \\ & && g_k(\mathbf{x}) \geq 1 \quad \text{for } k = m+1, \dots, m+r \\ & && h_l(\mathbf{x}) = 1 \quad \text{for } l = 1, \dots, n \\ & && \mathbf{x} > 0 \end{aligned}$$

where again each of the functions  $g_k(\mathbf{x})$  are posynomials and the function  $h_l(\mathbf{x})$  are monomials. The constraints  $k = m+1, \dots, m+r$  with the inequality sign reversed are called reversed constraints and are non-convex. The functions  $g_k(\mathbf{x})$  are convex functions of the variables  $\mathbf{x}$ , but the set of points which satisfies  $g_k(\mathbf{x}) \geq 1$  is not a convex set.

Any well-posed algebraic program (i.e. a program involving real-valued functions that are generated solely by addition, subtraction, multiplication, division and the extraction of roots) can be transformed into an equivalent reversed geometric program [4].

Since the optimization program **RGP** is not convex anymore, the efficient computational techniques used for regular geometric programming cannot be used directly in this case. However, it is still possible to compute a globally optimal solution and to do better than using a general purpose optimization technique.

## 2.4 Algorithms for reversed geometric programming

We divide the constraints in convex and reversed convex constraints. Appropriate techniques can be used for each category.

We use a branch-and-bound algorithm for the reversed constraints, similar to the one presented in [8]. This approach guarantees convergence to the global optimum, but is slow [10]. At every step, each reversed convex constraint is approximated by a set of convex constraints which cover the original solution space completely. A sequence of geometric programs is solved to find the global optimum.

As pointed out in section 2.2, it is not possible to approximate a non-convex constraint accurately with a convex constraint. However, it is possible to approximate a non-convex constraint with a set of convex constraints. Figure 2 shows a set of 2 convex approximations to the non-convex constraint in figure 1(a), indicated by the shaded regions. The union of the sets of points satisfying these constraints contains all the points satisfying the original non-convex constraints. In addition, the set contains points which do not satisfy the original constraint.

Each of the constraints in figure 2 can be improved by replacing it with multiple convex constraints that provide a tighter approximation to the original constraint. The new approximations will include less points which do not satisfy the original constraint. Iteratively refining the constraints will give approximations arbitrarily close to the original constraint. In order to get the global optimum, a geometric program needs to be solved for each one of the convex approximations. The minimum of all these **GP**'s will be the global optimum.

The branch-and-bound algorithm selectively refines the sets of convex approximations by expanding those reversed constraints that are violated most. In the case of the con-

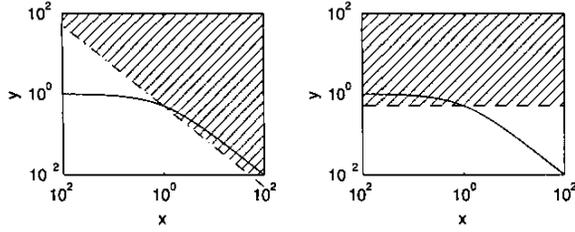


Figure 2: A set of convex approximations.

straint from figure 1, the algorithm will start by replacing it with the two convex constraints from figure 2 and solving the 2 geometric programs. The constraint which yields the lowest minimum will be selected for further refinement. The other constraint is not discarded, as it might be refined later on.

For the synthesized examples in section 5, the branch-and-bound algorithm executed about 330 geometric programs, each of them taking 1 to 2 seconds. It is possible to reduce the computational effort, by exploiting the fact that, after a couple of steps, the branch-and-bound optimizer solves a series of only slightly perturbed problems, so that the previous solution can be used as a good starting point for the next refinement. Unfortunately, our GP solver currently does not allow us to specify a starting point.

An interior point optimizer is used for the underlying geometric program.

### 3. EXAMPLES OF NON-CONVEX CONSTRAINTS IN EQUATION-BASED ANALOG DESIGN

#### 3.1 MOSFET model

A MOSFET has 3 independent variables. A typical choice of variables is  $W$ ,  $L$  and a bias variable.  $I_{DS}$  is preferred over  $V_{GS} - V_T$  as the bias variable, since, for a long channel MOSFET, all small-signal parameters, from weak inversion to strong inversion, can be expressed as a function of  $I_{DS}$  [3]. Expressing the same small-signal parameters analytically as a function of  $V_{GS} - V_T$  is not possible, since there is no analytical solution for the surface potential as a function of the terminal voltages [15, p. 134].

Since both  $I_{DS}$  and  $1/I_{DS}$  and both  $g_m$  and  $1/g_m$  are used in the design equations for the optimization of operational transconductance amplifiers [9],  $I_{DS}$  and  $g_m$  have to be monomials (i.e. single term posynomials). The relationship between  $I_{DS}$  and  $g_m$  cannot be specified as an inequality constraint, since it cannot be guaranteed that the constraint will be active for transistors which will be biased with a low  $g_m/I_D$  ratio and for transistors which will be biased with a high  $g_m/I_D$  ratio. The  $g_m-I_{DS}$  ratio has to be specified as an equality constraint. This constraint is probably the most important device characteristic for analog design [14].

Geometric programming only allows monomial equality constraints, which automatically leads to a MOSFET model valid in one region of operation: either the weak inversion, strong inversion or velocity saturated region. A more complex  $g_m-I_{DS}$  model can only be specified using a combination of regular and reversed convex constraints.

For example, the long-channel small-signal model in [6] and [3] can be expressed as:

$$g_m = \frac{2I_0}{\phi_{th}} (\sqrt{1 + I_{DS}/I_0} - 1)$$

where  $I_0$  is a normalization current and  $\phi_{th}$  is the thermal voltage. This expression can be reformulated as:

$$\left(g_m \frac{\phi_{th}}{2I_0}\right)^2 + 2\left(g_m \frac{\phi_{th}}{2I_0}\right) = \frac{I_{DS}}{I_0} \quad (1)$$

The first term in equation (1) is dominant in strong inversion, the second term in weak inversion. Both terms are necessary to find the correct trade-off between speed and power in OTA's. This constraint can be split into a convex and a reversed convex constraint, and thus fits into the RGP formulation:

$$\left(g_m \frac{\phi_{th}}{2I_0}\right)^2 + 2\left(g_m \frac{\phi_{th}}{2I_0}\right) \leq \frac{I_{DS}}{I_0} \quad (2a)$$

$$\left(g_m \frac{\phi_{th}}{2I_0}\right)^2 + 2\left(g_m \frac{\phi_{th}}{2I_0}\right) \geq \frac{I_{DS}}{I_0} \quad (2b)$$

Meeting both the convex posynomial constraint (2a) and the reversed posynomial constraint (2b) is equivalent to meeting (1).

Non-zero numerical tolerances in the algorithms guarantee that only one constraint will be active. The set of equations:

$$\left(g_m \frac{\phi_{th}}{2I_0}\right)^2 + 2\left(g_m \frac{\phi_{th}}{2I_0}\right) \leq \frac{I_{DS}}{I_0} (1 + \epsilon_c) \quad (3a)$$

$$\left(g_m \frac{\phi_{th}}{2I_0}\right)^2 + 2\left(g_m \frac{\phi_{th}}{2I_0}\right) \geq \frac{I_{DS}}{I_0} (1 - \epsilon_r) \quad (3b)$$

defines a band of numerically acceptable solutions ( $\epsilon_c$  and  $\epsilon_r$  are the tolerances).

#### 3.2 Settling

In switched-capacitor circuits, the OTA is typically put in a capacitive feedback configuration and is required to settle in about half a clock cycle. The settling typically consists of a slewing phase and a linear settling phase.

The slewing part of the settling time is given by [11],[2]:

$$T_{slewing} = \begin{cases} 0 & \text{if } \Delta V \leq \frac{I_{sat}}{\beta g_m} \\ \tau_{GBW} \left( \frac{\beta g_m \Delta V}{I_{sat}} - 1 \right) & \text{otherwise} \end{cases} \quad (4)$$

where  $\tau_{GBW}$  is the time constant corresponding to the gain-bandwidth product,  $\beta$  is the feedback factor,  $I_{sat}$  is the maximum current the OTA can deliver, and  $\Delta V$  is the maximum voltage step at the output of the OTA.

The linear part of the settling time is often approximated by  $\tau_{GBW} \log \frac{1}{\epsilon}$ , where  $\epsilon$  is a required upper bound on the settling error. However, this is only accurate for first order systems [2] [16].

For higher order systems, the settling error might show ringing, so that it makes more sense to look at the envelope of the settling error than at the settling error itself. This envelope decreases exponentially with time, and can be modeled with an exponential time constant. Figure 3 shows the time constant of the envelope of the settling error versus the position of the non-dominant pole, which is clearly non-convex. A first order approximation  $\tau_{GBW} \log \frac{1}{\epsilon}$  would make an error of a factor of 2 for  $\tau_{GBW}/\tau_{nd} = 4$ . Notice how for  $\tau_{GBW}/\tau_{nd} = 2$ , which corresponds to a phase margin of approximately 65 degrees [13], the simple first order approximation is exact.

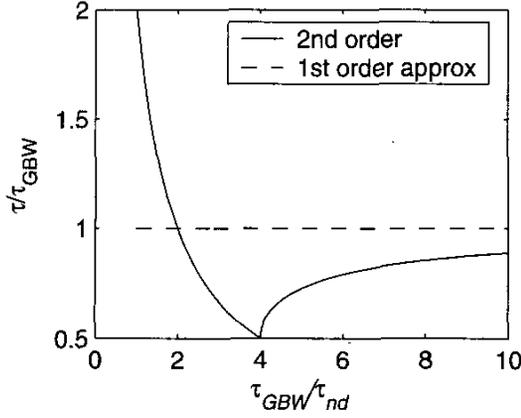


Figure 3: Time constant of the envelope of the settling error versus the position of the non-dominant pole.

Table 1: High level constraints

$V_{pp}$	$= 1\text{ V}$
$f_s$	varies from 10 MHz to 100 MHz
$v_{n,i}^2$	$\leq \frac{1}{12} \left(\frac{V_{pp}}{2^n}\right)^2$
$\epsilon$	$\leq \frac{1}{5} \frac{1}{2^n}$
$n$	$= 12$
$A_0$	$\geq 500$ (54 dB)

The total settling time is the sum of the slewing time and the linear settling time. This settling time will be a non-convex function of the time-constants. To size an OTA for minimum settling time, it is necessary to model this non-convex behavior.

## 4. MODELS

### 4.1 Operational Transconductance Amplifier

As an example, we designed a folded-cascode transconductance amplifier with different speed requirements. The schematic of the amplifier is shown in figure 4. The operational transconductance amplifier is embedded in a capacitive feedback network, as shown in figure 5. The actual circuit configuration that was used is a fully differential version of figure 5.

The high level constraints on the operational transconductance amplifier (OTA) are given in table 1, where  $V_{pp}$  is the peak-to-peak voltage swing,  $v_{n,i}^2$  is the input-referred thermal noise,  $\epsilon$  is the settling accuracy, and  $A_0$  is the low-frequency gain.

$A_0$  is quite relaxed compared to the other specifications; the topology has much more influence on  $A_0$  than the sizing does. Requiring a high gain from a folded-cascode amplifier will make the design problem impossible to satisfy.

The model equations used for the OTA and the capacitive feedback are well known equations [13].

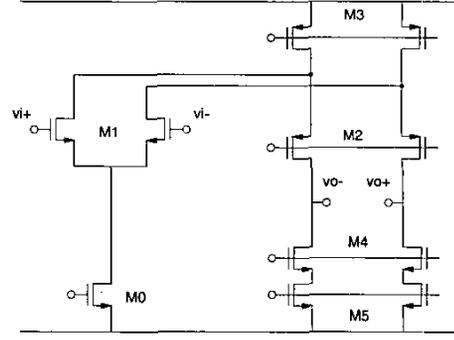


Figure 4: Folded cascode amplifier

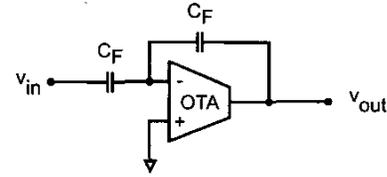


Figure 5: OTA in capacitive feedback configuration (single-ended version)

### 4.2 Device Models

The MOSFET model used in this synthesis approach is based on a charge-sheet model, which can provide expressions for the small signal parameters as a function of the current in all regions of operation (weak inversion - moderate inversion - strong inversion). The model is similar to the one presented in [3], but includes short channel and noise parameters.

Some of the ideal equations from the MOSFET model, together with their posynomial form, are given in table 2.  $q$  is the normalized charge density at the source and  $i$  is the inversion level; the normalization constants  $I_0$ ,  $g_{m0}$  and  $C_{gs0}$  are given in table 3. The actual model equations used in the optimization have some additional fitting parameters and short-channel corrections. Table 4 shows some of the modifications for short channel effects.  $q_{sat}$  models velocity saturation;  $L_E$  is the effective channel length, and  $\gamma_i$  and  $\gamma_2$  are fitting parameters.

The inversion level  $i = I_{DS}/I_0$  is a dimensionless metric for the region of operation of MOSFET transistor (weak inversion - moderate inversion - strong inversion). Weak inversion corresponds to  $i \ll 1$ , while strong inversion corresponds to  $i \gg 1$ . In strong inversion, the equation in table 2 for  $q$  can be approximated by  $q^2 \approx i$  (since  $i \gg 1$  implies  $q \gg 1$ ), so that  $g_m \approx g_{m0} \sqrt{I_{DS}/I_0}$ . In weak inversion,  $q \approx i$  and  $g_m \approx g_{m0} I_{DS}/I_0$ .

The model parameters used in this paper were derived from fitting to SPICE models for ST Microelectronics'  $0.18\ \mu\text{m}$  process. Figures 6 and 7 shows the simulation (SPICE) model and design (RGP) model for respectively  $L = 0.18\ \mu\text{m}$  and  $L = 0.7\ \mu\text{m}$ . The SPICE models use the Philips MOS Model 9.

Table 2: Ideal long-channel device model

original form	posynomial form
$q = \sqrt{1 + \frac{I_{DS}}{I_0}} - 1$	$q^2 + 2q \leq I_{DS}/I_0$
$i = I_{DS}/I_0$	$q^2 + 2q \geq I_{DS}/I_0$
$g_m = g_{m0} q$	$g_m = g_{m0} q$
$C_{gs} = C_{gs0} \frac{q(q+3)}{(q+2)^2}$	$C_{gs} = C_{gs0} \frac{q^3(q+3)}{i^2}$
$V_{DSsat} = \phi_{th}(q+2)$	$V_{DSsat} = \phi_{th} \frac{i}{q}$
$\gamma = \frac{1}{2} + \frac{1}{6} \frac{q}{q+2}$	$\gamma = \frac{1}{2} + \frac{1}{6} \frac{q}{i}$

Table 3: Normalization constants for the device model

$I_0 = \frac{W}{L} \frac{n_v \mu}{2} C_{ox} \phi_{th}^2$
$g_{m0} = \frac{1}{n_v} \frac{2I_0}{\phi_{th}}$
$C_{gs0} = \frac{2}{3} W L C_{ox}$

Table 4: Some short-channel modifications to the device model

long-channel equations	short-channel equations
$g_m = \frac{1}{n_v} \frac{2I_0}{\phi_{th}} q$	$g_m = \frac{1}{n_v} \frac{2I_0}{\phi_{th}} \frac{q}{1+q/q_{sat}}$
$\gamma = \frac{1}{2} + \frac{1}{6} \frac{q}{q+2}$	$\gamma = \frac{\gamma_1}{2} + \frac{1+\gamma_2/L}{6} \frac{q}{q+2}$

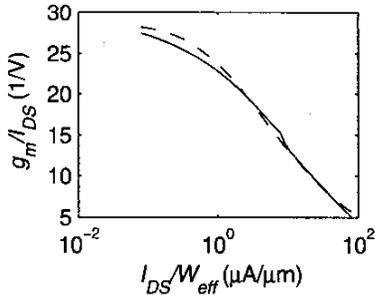


Figure 6:  $g_m/I_D$  for the simulation model (—) and the design model(- -) for  $L = 0.18 \mu\text{m}$

The device model used for optimization and the device model used for simulation match to within 10 %, which is an acceptable level of error in that the optimization solution found is sufficiently close to simulation.

## 5. SYNTHESIS AND SIMULATION RESULTS

We sized a folded cascode amplifier for the requirements give in table 1 for sampling speeds ranging from 10 MHz to 100 MHz. Table 5 shows the size of the optimization problem for the folded cascode amplifier. Table 6 shows the solution for  $f_s = 25 \text{ MHz}$ .

Table 7 compares the target specifications with the performance of the simulated circuit for  $f_s = 25 \text{ MHz}$ . Since

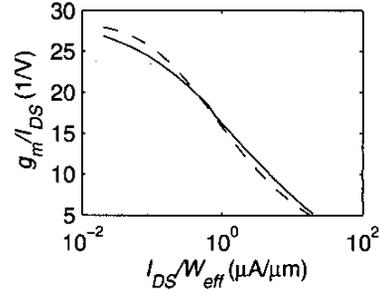


Figure 7:  $g_m/I_D$  for the simulation model (—) and the design model(- -) for  $L = 0.70 \mu\text{m}$

Table 5: Optimization problem size for the folded-cascode amplifier

# variables	52
# terms	210
# convex constraints	55
# reversed constraints	17
# equality constraints	11

Table 6: Optimization solution for  $f_s = 25 \text{ MHz}$

$C_F$	4.4 pF	$I_{BIAS}$	115 $\mu\text{A}$
$W_0$	6.3 $\mu\text{m}$	$L_0$	6.89 $\mu\text{m}$
$W_1$	430 $\mu\text{m}$	$L_1$	0.21 $\mu\text{m}$
$W_2$	28.5 $\mu\text{m}$	$L_2$	0.25 $\mu\text{m}$
$W_3$	114 $\mu\text{m}$	$L_3$	1.73 $\mu\text{m}$
$W_4$	4.1 $\mu\text{m}$	$L_4$	0.31 $\mu\text{m}$
$W_5$	6.9 $\mu\text{m}$	$L_5$	6.89 $\mu\text{m}$

Table 7: Simulated performance for  $f_s = 25 \text{ MHz}$

	target	simulation
$A_0$	$\geq 500$	898
$GBW$	$\geq 70 \text{ MHz}$	79.1 MHz
$v_{n,o}^2$	$\leq 1.99 \cdot 10^{-8} \text{ V}^2$	$1.6 \cdot 10^{-8} \text{ V}^2$
$t_{settle}$	$\leq 20 \text{ ns}$	21 ns

the transistor model is only accurate to about 10 %, it is reasonable to expect the constraints to be met with a 10 % accuracy. All the constraints are met, except for the settling constraint, which is violated by only 5 %.

We simulated several designs to check the accuracy of the design equations over a range of specifications and we compared the results with simulations of designs from regular geometric programming (GP), as shown in table 8. The settling constraints are violated by a small amount, which we attribute to the small difference between the device model used in the simulation and the device model used in the optimization.

To compare with GP, all reversed constraints were approximated with convex constraints; for example, the  $g_m-I_{DS}$  model from table 2 was approximated by the quadratic model  $q = \sqrt{i}$ . Since this approximation predicts a higher

**Table 8: Simulated performance and relative power consumption versus sampling speed**

target		RGP	GP	GP*	
$f_s$ (MHz)	$t_{settle}$ (ns)	$t_{settle}$ (ns)	$t_{settle}$ (ns)	$t_{settle}$ (ns)	$\frac{P_{GP*}}{P_{RGP}}$
10	50	53.0	66.6	54.7	3.5
16	31.5	33.5	41.5	34.3	3.5
25	20	21.0	25.1	21.7	3.4
40	12.5	12.9	16.2	13.6	3.4
63	8	8.27	10.1	8.6	3.1
100	5	5.2	6.25	5.4	2.8

$g_m$  for the same  $I_{DS}$  compared to the full model, it underestimates the settling time, as can be seen from table 8. This modeling error is bias-dependent: it is small in the strong-inversion regime, but quite large in the weak-inversion region.

To make **GP** more accurate, we added extra constraints to keep the transistors in the strong inversion region, and the results are shown in the columns labeled **GP\***. The extra constraints improve the design accuracy, but at the price of highly increased power consumption. For these examples, **GP\*** gives reasonably accurate solutions, violating the settling constraint by only 10 %, which is within the error between the simulation device model and the optimization device model. However, the power consumption is about 3 times higher compared to the solution from **RGP**. The difference is larger for slower, lower-power designs.

This example clearly shows that convex approximations to critical non-convex model equations can give either inaccurate solutions or inefficient solutions to the design problem.

## 6. CONCLUSION

Operational transconductance amplifiers can be designed with accurate models using reversed geometric programming, an extension of geometric programming. Reversed geometric programming allows both convex and non-convex constraints. Adding a limited set of non-convex constraints allows accurate modeling of critical design equations and yields more efficient solutions.

## 7. ACKNOWLEDGMENTS

The authors would like to thank DARPA, the C2S2 MARCO Focus Center, STMicroelectronics, and all the BWRC industrial sponsors for their support. The authors also would like to thank D. Sobel and A. Vladimirescu for their helpful comments.

## 8. REFERENCES

- [1] C. S. Beightler and D. T. Phillips. *Applied Geometric Programming*. Wiley, New York, 1976.
- [2] C. T. Chuang. Analysis of the settling behavior of an operational amplifier. *IEEE Journal of Solid-State Circuits*, SC-17(1):74–80, 1982.
- [3] A. I. A. Cunha, M. C. Schneider, and C. Galup-Montoro. An MOS transistor model for analog circuit design. *IEEE Journal of Solid-State Circuits*, 33(10):1510–19, 1998. English.
- [4] R. J. Duffin and E. L. Peterson. Geometric programming with signomials. *Journal of Optimization Theory & Applications*, 11(1):3–35, 1973.
- [5] R. J. Duffin, E. L. Peterson, and C. Zener. *Geometric Programming: theory and applications*. Wiley, New York, 1967.
- [6] C. C. Enz, F. Krummenacher, and E. A. Vittoz. An analytical MOS transistor model valid in all regions of operation and dedicated to low-voltage and low-current applications. *Analog Integrated Circuits and Signal Processing*, 8(1):83–114, 1995.
- [7] G. Gielen and R. Rutenbar. Computer-aided design of analog and mixed-signal integrated circuits. *Proceedings of the IEEE*, 88(12):1825–1854, 2000.
- [8] W. Gochet and Y. Smeers. A branch-and-bound method for reversed geometric programming. *Operations Research*, 27(5):982–96, 1979.
- [9] M. Hershenson, S. Boyd, and T. Lee. Optimal design of a CMOS op-amp via geometric programming. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 20(1):1–21, 2001.
- [10] R. Horst and H. Tuy. *Global optimization: deterministic approaches*. Springer-Verlag, Berlin, third edition, 1990.
- [11] B. Y. T. Kamath, R. G. Meyer, and P. R. Gray. Relationship between frequency response and settling time of operational amplifiers. *IEEE Journal of Solid-State Circuits*, SC-9(6):347–52, 1974.
- [12] K. O. Kortanek, X. J. Xu, and Y. Y. Ye. An infeasible interior-point algorithm for solving primal and dual geometric programs. *Mathematical Programming*, 76(1):155–181, 1997.
- [13] K. R. Laker and W. M. Sansen. *Design of Analog Integrated Circuits and Systems*. McGraw-Hill, New York, 1994.
- [14] F. Silveira, D. Flandre, and P. G. A. Jespers. A  $g_m/I_D$  based methodology for the design of CMOS analog circuits and its application to the synthesis of a silicon-on-insulator micropower OTA. *IEEE Journal of Solid-State Circuits*, 31(9):1314–19, 1996.
- [15] Y. Tsididis. *Operation and Modeling of the MOS Transistor*. WCB/McGraw-Hill, Boston, 2nd edition, 1999.
- [16] H. C. Yang and D. J. Allstot. Considerations for fast settling operational amplifiers. *IEEE Transactions on Circuits & Systems*, 37(3):326–34, 1990.