CMOS analog circuit design via geometric programming

Maria del Mar Hershenson

***Abstract*— This talk presents a new method for optimizing and automating component sizing in CMOS analog circuits. It is shown that a wide variety of circuit performance measures have a special form, *i.e.*, they are posynomial functions of the design variables. As a result, circuit design problems can be posed as geometric programs, a special type of convex optimization problem for which very efficient global optimization methods exist. The synthesis method is therefore fast, and determines the globally optimal design; in particular, the final solution is completely independent of the starting point, and infeasible specifications are unambiguously detected. Also, because the method is highly efficient, in practice it can be used to carry out robust designs and quickly explore the design space. We show that despite the restricted form of geometric programs, a large variety of circuit design problems can be posed as geometric programs.**

I. INTRODUCTION

In current mixed-mode integrated circuits, the analog circuits represent only a small part of the total area. However, their design is very complex: a wide range of specifications have to be met and sensitivity to process variations is very high. Unlike the sophisticated synthesis CAD tools available for digital circuits, analog synthesis CAD tools are practically non existent and the vast majority of analog circuits are still designed ”manually” by experts [1]. Thus, the development time for analog blocks is far out of proportion to the die area that they consume. This issue is exacerbated by the tremendous shortage of analog designers [2].

In this paper, we describe an efficient approach to analog circuit design automation. The circuit performance is described with a set of analytical design equations that have a special form (*i.e.*, they are convex) and the design problem is cast as a convex optimization problem, which is then solved very efficiently and globally by a numerical algorithm.

This paper is organized as follows. In §II, we describe geometric programming, the optimization problem which is the basis of the method. In §III, we describe the transistor and circuit models used. In §IV, we give several op-amp and analog-to-digital converter (ADC) design examples and show some measured results.

II. GEOMETRIC PROGRAMMING

Let x be a vector $(x_1, \dots, x_n)$ of n real, positive variables. A function f is called a *posynomial* function of x if it has the form

$$f(x_1, \dots, x_n) = \sum_{k=1}^{t} c_k x_1^{\alpha_{1k}} x_2^{\alpha_{2k}} \cdots x_n^{\alpha_{nk}}$$

where $c_j \geq 0$ and $\alpha_{ij} \in \mathbf{R}$. When $t = 1$, we call f a *monomial* function. A *geometric program* (GP) is an optimization problem of the form

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x) \leq 1, \quad i = 1, \dots, m, \\ & g_i(x) = 1, \quad i = 1, \dots, p, \\ & x_i > 0, \quad i = 1, \dots, n, \end{array} \tag{1}$$

where f_i are posynomial functions and g_i, are monomial functions. A GP can be reformulated as a convex optimization problem, by changing variables and considering the logs of the functions involved [3].

Although there are several options for solving geometric programs, probably one of the most efficient GP special purpose solvers is the one proposed by Kortanek *et al.* where they have shown how the most sophisticated primal-dual interior-point methods used in linear programming can be extended to GP, resulting in an algorithm with efficiency approaching that of current interior-point linear programming solvers [4].

III. POSYNOMIAL MODELS

Our goal is to pose circuit design problems as geometric programs. To do this we describe posynomial circuit models hierarchically and modularly (see [5]). Even though the formulation is hierarchical, the resulting geometric program is solved in a *flat* manner, *i.e.*, design equations at all levels are solved simultaneously. This can *only* be done because of the great efficiency with which we can solve geometric programs. If we were using a general-purpose optimization method, a flat solve would probably not be feasible because of the large size of the problem at question (tens of thousands of variables and hundreds of thousands of constraints for a robust design).

A. Process dependent models

There are several process-dependent devices: transistors, capacitors, resistors, *etc.* Here we describe how to model a CMOS transistor but the methodology is generic and can be extended to other devices and other process technologies. A CMOS transistor is implemented with a parameterizable layout cell. The dimensions of this cell constitute the problem design variables (width W, length L and number

This work was not supported by any organization

M. Hershenson is a co-founder of Barcelona Design, Inc. 39677 Eureka Drive, Newark, CA 94560 `mar@stanfordalumni.org`

of copies M). The transistor model contains both physical and electrical properties as a function of the design variables and process dependent parameters (*e.g.*, minimum poly-poly spacing).

1) Physical model: We can model several physical parameters, for example: cell height and width, source/drain area and perimeters, length of intra-cell routing, *etc*. The important thing is that they can all be modeled in a form compatible with GP.

2) Electrical model: Here we show the a type of transistor model called *GP1 model* [6].

- The overdrive voltage, $V_{\mathrm{gs}} - V_{\mathrm{TH}}$, is a monomial function of transistor length L, transistor width W and transistor drain current I,

$$V_{\mathrm{gov}} = \beta_{v,1} W_T^{\alpha_{v,1}} L^{\alpha_{v,2}} I^{\alpha_{v,3}}, \qquad (2)$$

  where $W_T = M \cdot W$, and $\beta_{v,1}$, $\alpha_{v,1}$, $\alpha_{v,2}$, $\alpha_{v,3}$, are process dependent constants that can be found by simple data fitting techniques [3].
- The transconductance, g_{m}, is a monomial function in L, W_T, and I,

$$g_{\mathrm{m}} = \beta_{gm,1} W_T^{\alpha_{gm,1}} L^{\alpha_{gm,2}} I^{\alpha_{gm,3}}, \qquad (3)$$

  where $\beta_{gm,1}$, $\alpha_{gm,1}$, $\alpha_{gm,2}$, $\alpha_{gm,3}$, are process dependent constants.
- The output conductance, g_{o}, is given by $\gamma g_{\mathrm{o,m}}$ where $g_{\mathrm{o,m}}$ is monomial in L, W_T, and I, and γ is a constant that can take two different values depending on whether the transistor in question typically operates with large or small V_{ds}.
- Capacitances between the terminals and bulk are posynomial in L, W, M, and I.

Obtaining the coefficients for the *GP1 model* is a straightforward task. The first step consists on generating the data for the parameters we wish to model. This data can come directly from laboratory measurements or can be computer generated by simulating existing device models (*e.g.*, BSIM [7]) with a circuit simulator like SPICE. The second step consists on solving an optimization problem to find the *GP1 model* coefficients. This optimization problem minimizes some norm of the modeling error [3]. One can use a general non-linear optimization package to solve this problem or special purpose algorithms like the one described in [8]. We have been able to obtain *GP1 models* that have good agreement with SPICE higher order models (BSIM models), over large ranges of length, width, and bias currents [6] (see Fig. 1). To model deep sub-micron devices, one can develop more sophisticated and accurate models that are still compatible with GP based design.

B. Circuit block models

Although it is difficult to prove mathematically that all analog circuits can be modeled using posynomial equations, it is not hard to understand why this could be possible. In this section we just show how we model some of the electrical performance metrics of a simple two-stage op-amp (see [6] for a detailed description).

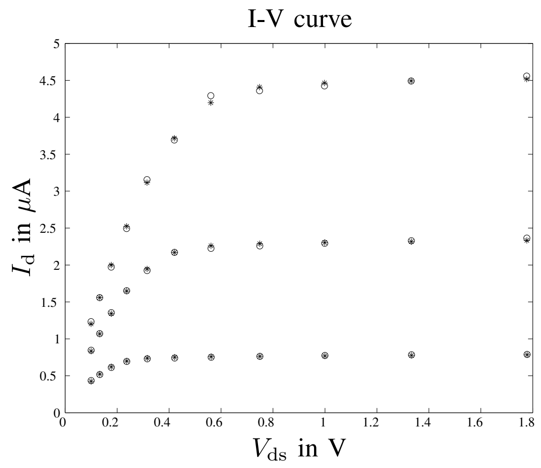

Fig. 1. GP-SPICE modeling of I-V curve comparison for an NMOS device in a 0.18μm CMOS process. "$*$" and "$\circ$" represent the SPICE model and posynomial model respectively.

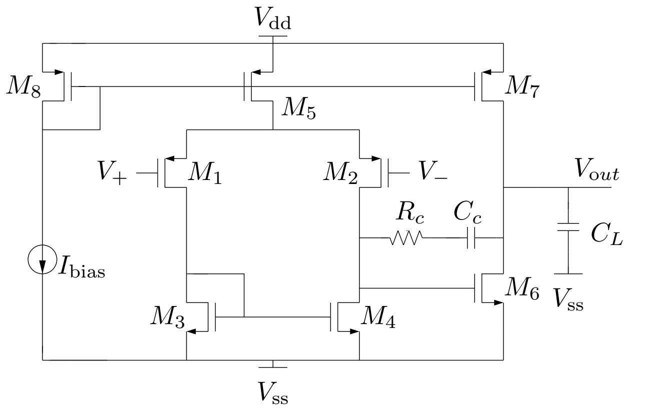

Fig. 2. Two stage op-amp.

Fig. 2 shows the schematic for a simple two-stage op-amp. There are twenty seven design variables: length, width and number of copies of each transistor, capacitor value, resistor value and reference bias current. There are more than a dozen of specifications: gain, bandwidth, power, phase margin, maximum area, slew rate, *etc*. Even though the performance specifications are highly nonlinear functions of the design variables, we show the amazing result that they are in fact, posynomial functions of the design variables.

1) Electrical model:

- *Quiescent power:* It is given by the product of the supply and the total consumed current,

$$P = (V_{\mathrm{dd}} - V_{\mathrm{ss}})\,(I_{\mathrm{bias}} + I_5 + I_7)\,, \qquad (4)$$

where I_5 and I_7 are just mirrored copies of the reference bias current given by monomial expressions.

$$I_5 = \frac{W_5 L_8}{L_5 W_8} I_{\text{bias}}, \quad I_7 = \frac{W_7 L_8}{L_7 W_8} I_{\text{bias}}. \tag{5}$$

Thus, power is posynomial in the design variables and we can upper bound it, or minimize it.

- *Open-loop DC gain:* It is given by the product of gains of each stage. The inverse-gain is given by the posynomial,

$$\frac{1}{A_v} = \left(\frac{g_{\text{o}2} + g_{\text{o}4}}{g_{\text{m}2}}\right)\left(\frac{g_{\text{o}6} + g_{\text{o}7}}{g_{\text{m}6}}\right), \tag{6}$$

  Thus we can minimize the inverse-gain (or maximize the gain) or impose a minimum required gain.
- *Input-referred noise power:* The equivalent input-referred noise power spectral density $S_{\text{in}}(f)^2$ (in V^2/Hz), can be expressed as

$$S_{\text{in}}^2 = S_1^2 + S_2^2 + (g_{\text{m}3}/g_{\text{m}1})^2 (S_3^2 + S_4^2), \tag{7}$$

  where S_k^2 is the input-referred noise power spectral density of transistor M_k, given by the sum of the thermal noise and the $1/f$ noise,

$$S_k(f)^2 = \left(\frac{2}{3}\right)\frac{4kT}{g_{\text{m,k}}} + \frac{K_{\text{f}}}{C_{\text{ox}} W_k L_k f}. \tag{8}$$

  Thus, $S_{\text{in}}(f)^2$ is a posynomial function of the design parameters at each frequency and we can impose a maximum input-referred noise at a certain frequency (or at several frequencies). By integrating over f (which results in another posynomial) we can limit the total RMS noise in an arbitrary frequency band.
- *Unity-gain bandwidth:* It is given by

$$\omega_c = g_{\text{m}1}/C_c, \tag{9}$$

  which is monomial. Therefore we can impose lower bounds on, or maximize, the unity-gain bandwidth. Note that in order to make the op-amp have this dominant pole, some stability constraints such as phase margin must be added to the model.

2) Physical model: Typically we are also interested in physical objectives and specifications like area, aspect ratio, length and width. We can include floorplanning information in posynomial form. To do this we specify the relative positioning of the circuit components by using a slicing tree representation [9]. The positioning constraints turn out to be posynomial constraints and thus, can be handled in geometric programming.

As an example consider the two-stage op-amp of Fig. 2. A possible floorplan is given in Fig. 3 whose slicing tree representation is given in Fig. 4. According to the slicing tree, starting from the main cell, we first perform a vertical split. (The v and h represent vertical or horizontal slicings for the subcells respectively.) The cell on the right becomes C_{c} and the cell on the left is sliced again, but this time horizontally. Now, the top subcell of this split, is split vertically, *etc.*

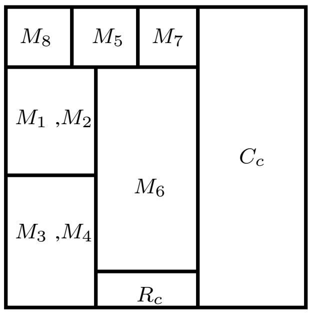

Fig. 3. Floorplan of two-stage op-amp.

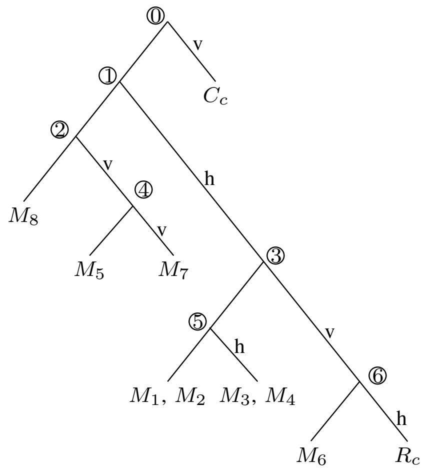

Fig. 4. Slicing tree of floorplan of two-stage op-amp.

Suppose that (x_i, y_i) are the widths and heights of the ith cell corresponding to the ith node of the slicing tree. Given the slicing tree, one can write the inequality constraints relating the (x_i, y_i)s. At a node with vertical split, the sum of the widths of the sibling nodes are less than or equal to the width of the parent node, and the heights of the sibling nodes are each less than or equal to the height of the parent node. At a node with a horizontal split, the sum of the heights of the sibling nodes are less than or equal to the height of the parent node, and the widths of the sibling nodes are each less than or equal to the width of the parent node. For example, at node 0:

$$\begin{aligned} &x_1 + x_{\text{cap}}(C_c) \le x_0, \\ &y_1 \le y_0, \\ &y_{\text{cap}}(C_c) \le y_0, \end{aligned} \tag{10}$$

where $x_{\text{cap}}(C_{\text{c}})$ and $y_{\text{cap}}(C_{\text{c}})$ are the width and height of the capacitor C_{c} respectively. As can be seen, constraints such as (10) are posynomial in the cell sizes or variables

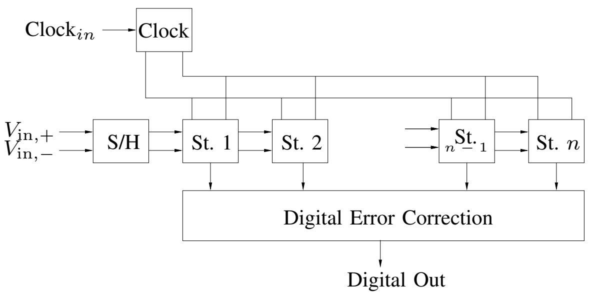

Fig. 5. Top-level schematic of pipeline ADC.

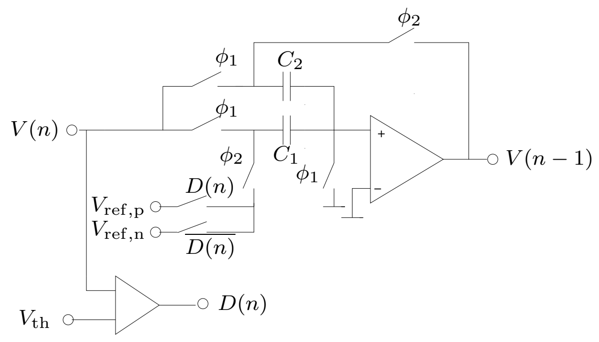

Fig. 6. Single-ended implementation of a pipeline ADC stage.

(x_i, y_i). They are also posynomial in the circuit variables (e.g., M, W, L of transistors, value of C_c, etc.) because the functions $x_{\text{cap}}(C_c)$, $y_{\text{cap}}(C_c)$ are all posynomial in the design variables.

A constraint on the total area of the circuit to be less than A_{spec} is simply given by the monomial:

$$x_0 y_0 \leq A_{\text{spec}}. \tag{11}$$

A constraint on the aspect ratio of the circuit to be less than κ_{spec} is given by:

$$x_0/y_0 \leq \kappa_{\text{spec}}, \quad y_0/x_0 \leq \kappa_{\text{spec}}. \tag{12}$$

The smallest aspect ratio can be found by minimizing $\max(x_0/y_0, y_0/x_0)$, which can then be converted into a GP (by introducing a slack variable). Hence, by mixing the layout constraints, such as (10)-(12) with the electrical circuit design constraints, it is possible to optimally the circuit and place it in a single step.

C. System models

The extension from circuit block level models to higher levels of hierarchy models is relatively straight forward. One just needs to write posynomial design equations at each hierarchy level in terms of the input and output variables and the design variables of sub-blocks. Here we show a simple example, how to model in GP the signal-to-noise-ratio (SNR) specification for a standard pipeline ADC. Details about the complete ADC GP formulation can be found in [5].

In Fig. 5 we show a top level schematic of the converter. It consists of a set of stages connected in series that perform the conversion in decreasing level of resolution in a pipelined manner. Fig. 6 shows a single-ended implementation of a one bit per stage resolution stage. Each stage consists of two capacitors, a digital-to-analog converter, an operational amplifier and six switches. We define four levels of hierarchy:

1) *Top level.* The SNR is given by (see [10]),

$$SNR = 10 \log \frac{\left(2^{N-1}\Delta\right)^2/2}{n_p}, \tag{13}$$

where Δ is the quantization step size and n_p , the noise power at the input of the converter, is

$$n_p = n_Q + \sum_{i=1}^{M} \frac{n_{stage_i}}{G^{2(i-1)}}, \tag{14}$$

where n_{stage_i} is the input referred noise of the ith stage, G is the gain of a stage, and n_Q is the quantization noise given by

$$n_Q = \frac{\Delta^2}{12} = \frac{(V_{\text{ref}}/2^N)^2}{12}. \tag{15}$$

Therefore if we want to impose a condition on a maximum allowed dynamic range, we would impose the following posynomial constraint,

$$n_Q + \sum_{i=1}^{M} \frac{n_{stage_i}}{G^{2(i-1)}} < \frac{\left(2^{N-1}\Delta\right)^2/2}{10^{\frac{DR_{\max}}{10}}}. \tag{16}$$

Note that the right hand side of (16) is a constant determined by the user specifications. On the left hand side, n_Q is a constant for a given number of bits of resolution, G is fixed for a given number of bits per stage and n_{stage_i} is an output variable of this hierarchy level.

2) *Stage level.* The noise for the i^{th} stage is given by the sum of the thermal noise in the switches and the amplifier noise (see [10]),

$$\overline{e_i^2} = \overline{e^2_{\text{switches},i}} + \overline{e^2_{\text{amp},i}}, \tag{17}$$

where $\overline{e^2_{\text{amp},i}}$ is the input-referred op-amp noise and $\overline{e^2_{\text{switches},i}}$ is the noise contributed by the switches. Note that (17) is posynomial in the op-amp and the switches noise which are output variables of this level of hierarchy.

3) *Block level.* This level contains the information regarding the switches and the op-amp. The noise of the switches is given by the posynomial

$$\overline{e^2_{\text{switches},i}} = 2kT\left(\frac{2}{C_1} + \frac{2}{C_2}\right), \tag{18}$$

Performance measure	Spec	GP	SPICE	Lab
Power (mW)	≤ 8	8	8.7	8
DC gain (dB)	≥ 70	70	74	71
Unity-gain BW (MHz)	Max.	47	46	44
Phase margin (°)	≥ 63	63	68	58
Slew rate (V/μs)	≥ 30	83	66	64

TABLE I
TWO-STAGE OP-AMP DESIGN EXAMPLE.

where k is Boltzman's constant, T is the absolute temperature and C_1 and C_2 are output variables of this level of hierarchy and represent the capacitance values of the sampling capacitors. The input-referred op-amp noise, $\overline{e^2_{\mathrm{amp},i}}$, is given by

$$\overline{e^2_{\mathrm{amp},i}} = \frac{S_o}{4\tau}, \tag{19}$$

where S_o is the op-amp input-referred white noise density (given by a posynomial, as shown in (7)) and τ is the closed loop time constant given by a monomial for a fixed number of bits per stage (see [5]). Since S_o is a posynomial and τ a monomial, (17) is a posynomial.

4) *Process dependent level.* At the lowest level of hierarchy, the device noise (see (8)), device transconductance (see (3)) and capacitance values are posed as posynomials or monomials of the device dimensions.

One can solve (16)-(19), and the process dependent equations simultaneously using GP since they are all posynomial. Although we have not talked about other electrical specifications like power, area or sampling frequency or about any physical specifications, those can all be represented in a form compatible with GP [5].

IV. DESIGN EXAMPLES

A. Op-amp example

We provide some sample designs for the two-stage op-amp described in the last section. We use a 0.35μm CMOS process. The load capacitance is 5pF and the supply voltages are $V_{\mathrm{dd}} = 3.3$V and $V_{\mathrm{ss}} = 0$V.

Table I describes the sample design problem. The objective was to maximize the unity gain bandwidth subject to the other given constraints. The first column in Table I identifies the performance measure (and its units); the second gives the specification (showing whether it is an upper or lower bound). The third column, labeled GP, shows the performance of the design obtained. The fourth column, labeled SPICE, shows the value of the specification as simulated by BSIM3v1 models. The fifth column shows the value of the specifications as measured in the laboratory.

We make several observations. First, we can see that there is very close agreement between the predicted results from GP and the SPICE simulations and the measured results.

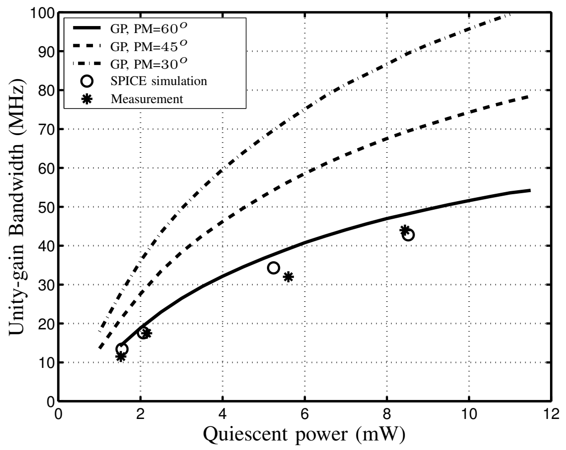

Fig. 7. Optimal trade-off curve for two-stage op-amp.

This shows that even though we are restricted to using posynomial equations, these are adequate to model real circuit behavior accurately. Second, we can see that many of the constraints are tight (power, gain and phase margin). This is in contrast with general-purpose methods that generally stop searching as soon as one of the constraints is tight. Finally, we notice that the time required to obtain the design was less than a minute in a 400MHz, 0.5GB workstation (and this was without providing any sort of initial starting point).

We can use all this design efficiency in several ways by generating *globally optimal* trade-off curves between competing objectives (with the others fixed). To do this, we repeatedly solving the design problem while varying one of the specifications. In Fig. 7 we plot a trade-off curve for the two-stage op-amp, obtained by maximizing unity-gain bandwidth, while varying the maximum allowed power dissipation for different values of phase margin, and leaving the rest of the specifications to the values given in Table I. The curve is obtained in only a few minutes. Any analog designer could have predicted that more power means more bandwidth. However, it would have been difficult for him to predict exactly what the optimal trade-off curve looks like and how it changes for different values of phase margin.

B. ADC example

Fig. 8 shows the trade-off curve power versus sampling frequency for a 12 bit ADC and a 10 bit ADC. We impose an SNR of 70dB for the 12 bit ADC and of 58dB for the 10 bit ADC . Again, it is no surprise that the higher the required sampling frequency the more power is needed. The usefulness of this curve resides in the quantification of this trade-off. It is also interesting to observe what the optimum power scaling per stage is (see Fig. 9). From the fifth stage on, all stages consume the same power and scaling only takes place in the initial stages. The reason is that after

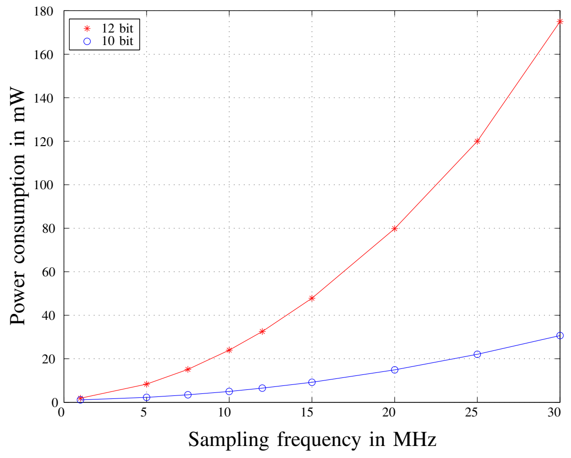

Fig. 8. Minimum power for 12 bit and 10 bit ADC.

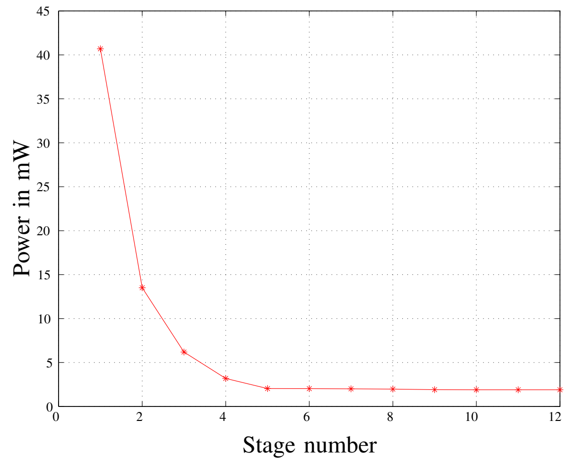

Fig. 9. Power scaling for 12 bit ADC, 20MHz.

the fifth stage, the op-amp behavior is determined by its own parasitics rather than by the switching capacitances. Note that this optimum scaling depends on all the ADC specifications and cannot be assumed constant.

Additional examples of mixed-signal systems designed with GP can be found in [11], where detailed silicon results are given for a clock generation and synthesis phase-locked loop.

V. Conclusions

We have described a method for analog circuit design based on geometric programming. The method consists on formulating the circuit design problem as a convex optimization problem. This, of course, requires expertise and effort. However, once this convex formulation is completed the the circuit is truly *synthesizable*. This has many advantages: we can easily change specifications and create a custom instance of the circuit in a very short time; we can easily port a design to a new process (by only updating the transistor data); we can perform what-if analysis that allow us to understand the engineering design trade-offs; we can hand-craft each tool for a specific architecture, mapping each design in a form that maximizes accuracy, *etc*. An important advantage of the formulation is that it encapsulates the experienced designer's knowledge. Once a model is complete, a novice designer can use it to create custom circuit instances. The designer only needs to input the circuit specifications. He or she does not need to provide a starting point, a set of simulation tests or a search plan. The circuit block is customized rapidly and independently of the experience of the designer customizing it.

Before deciding if one is going to spend the effort in formulating the problem in convex form, one needs to perform a simple return-on-investment calculation. If the circuit to be modeled is going to be used in more than one process or if several instances of it are needed (*e.g.*, same circuit with different specifications), then the time invested in making it synthesizable is worth it.

References

[1] P. Gray. Analog ICs in the submicron era: trends and perspectives. In *Proceedings IEEE International Electron Devices Meeting*, pages 5–9, 1987.

[2] S. Kariya. Where the jobs are. *IEEE Spectrum*, 40(1), January 2003.

[3] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2003.

[4] K. O. Kortanek, X. Xu, and Y. Ye. An infeasible interior-point algorithm for solving primal and dual geometric programs. *Math Programming*, 76:155–181, 1996.

[5] M. Hershenson. Design of pipeline analog-to-digital convertes via geometric programming. In *Proceedings of the 2002 IEEE/ACM International Conference on Computer Aided Design*, November 2002.

[6] M. Hershenson, S. Boyd, and T. H. Lee. Optimal design of a CMOS op-amp via geometric programming. *IEEE Transactions on Computer-Aided Design*, 20:1–21, January 2001.

[7] http://www-device.eecs.berkeley.edu/b̃sim3.

[8] W. Daems, G. Gielen, and W. Sansen. An efficient optimization-based technique to generate posynomial performance models for analog. In *39th Design Automation Conference*, pages 431–436, 2002.

[9] D.F. Wong and C.L. Liu. A new algorithm for floorplan design. In *Design Automation Conference*, pages 101–017, 1986.

[10] B. Wooley and K. Vleugels. EE315 course notes. Stanford University, CA, April 2003.

[11] D. M. Colleran, C. Portmann, A. Hassibi, C. Crusius, S. S. Mohan, S. Boyd, T. H. Lee, and M. Hershenson. Optimization of phase-locked loop circuits via geometric porgramming. September 2003.