

Simulation-based Reusable posynomial models for MOS transistor parameters

Varun Aggarwal
Massachusetts Institute of Technology
Cambridge, MA
varun_ag@mit.edu

Una-May O'Reilly
Massachusetts Institute of Technology
Cambridge, MA
unamay@csail.mit.edu

Abstract

We present an algorithm to automatically design posynomial models for parameters of the MOS transistors using simulation data. These models improve the accuracy in the Geometric Programming flow for automatic circuit sizing. The models are reusable for multiple circuits on a given Silicon technology and hence don't adversely affect the scalability of the Geometric Programming approach. The proposed method is a combination of genetic algorithms and Quadratic Programming. It is the only approach for posynomial modeling with real-valued exponents and easily extensible to different error metrics. We compare the proposed technique with all state-of-art posynomial/monomial modeling techniques.

1. Introduction

Automatic sizing of analog circuits continues to be a research focus for the EDA industry. For SOC (System-On-Chip) design, digital synthesis is automated to a large extent. However, the manual sizing of analog blocks become a bottleneck and governs the time-to-market. The technology evolution is guided by digital circuits (lower area and power) and the behavior of the transistor with respect to analog circuits is somewhat ignored as technology is scaled. The analog designer has to live with and learn to design circuits with new unrelenting transistor models. Automatic sizing frees the designer to work on new architectures and study system-level tradeoffs. It aims for better designs and shorter time to market.

Several techniques for sizing have been proposed and implemented. In the late 80's, knowledge based approaches [9, 7] were proposed. These techniques captured the expert knowledge of a designer and translated it into a set of rules which then automatically sized a circuit for a given set of specifications. These approaches were not very useful, since for every new circuit topology and technology, a new set of rules had to be created by manual labor. In the 90's,

simulation based approaches [18, 17] became very popular, where SPICE was used inside the loop of optimization. Stochastic black-box optimization techniques were used in this approach, which didn't require the knowledge of structure of the function being optimized. In parallel, equation-based sizing [8] was also proposed, which used similar optimization algorithms, but invoked symbolic equations derived from the circuit instead of SPICE.

The equation-based approaches have limitations. It is not possible to conduct the symbolic analysis of very large circuits and the symbolic expressions are approximate. Also, symbolic expressions can only be derived for small signal specifications, not for large signal transient specifications. On the other hand, simulation-based techniques though accurate, take an awful lot of time due to computationally expensive SPICE simulations in the loop of optimization. One of the greatest limitations of both these approaches is scalability. Simulation-based techniques provide no guarantee that the optima obtained is close to the real optima due to the high dimensionality of circuit problems. For instance, NSGA-II used in [3, 19] has been only benchmarked for 10 dimensional problems [6], where it is shown to be able to find the global optima. However, a simple folded-cascode opamp has more than 20 parameters to be optimized. It is well known that these algorithms do not scale well with the size of the problem. They become extremely slow (specially for the simulation-based approaches) and sub-optimal.

The approach of using Geometric Programming to size circuits simultaneously invented by Hershenson, et.al. [11] and Mandal, et.al. [14] comes as a fresh alternative. In [11, 12, 4], it is shown that many circuits (e.g. opamps, PLLs) can be expressed as a geometric program using first-order transistor models. The circuit DC equations and specifications are expressed as posynomials, which comprise the geometric program (GP). A GP is non-convex, but can be converted in to a convex optimization problem by taking the logarithm of the input variables, objectives and constraints. This allows global optimization of the GP, which implies that given the equations are correct, the algorithm finds the absolute maxima, that too in a few seconds.

This approach doesn't suffer from the problem of sub-optimality or scalability (large size of circuit), since a GP with 1000 variables and 10000 constraints is solved in less than a minute on a small desktop computer [2]. The advantage of scalability is an extremely important consideration not just with regard to the large size of the circuit, but for designing robust designs. Robustness is becoming more and more important with shrinking technology. It would need simulation of circuit on several design-corners (or statistical models) making simulation-based approaches highly sub-optimal or infeasible. GP's promise will deliver here, since this would just mean a few additional equations for GP.

This makes GP a nice alternative to simulation-based or equation-based optimization techniques discussed above. However, it has similar limitations as equation-based approach, which is that of inaccuracy of the posynomial equations and very simple hand-written equations for large signal transient performance measures.

In this paper, we have taken a first step to decrease the inaccuracy of the posynomial equations. We show how posynomial models for MOS parameters can be automatically designed using simulation data. We design an algorithm which uses a genetic algorithm with quadratic programming to automatically derive posynomial models. Earlier, GP formulations have used first-order models i.e. the square-law for MOS parameters [11] and it is well-known that the transistor doesn't accurately follow it. Moreover, with the shrinking technology, transistor behavior has become more unpredictable. The transition from weak inversion to strong inversion adds another dimension of complexity obviously not captured in the square law. The only other strategy to derive models for MOS is restricted to monomial models [2] which have less expressive power than posynomials and are more inaccurate, as will be exemplified in our results.

Section 2 introduces the basics of GP, while Section 3 discusses the different approaches used for GP formulation of analog circuits. In Section 4, we discuss the different approaches for posynomial modeling. Section 5 discusses our algorithm in detail and Section 6 contains our experiments and results. Section 7 concludes the paper.

2. Geometric Programming

A geometric program is a non-linear optimization problem, which can be transformed into a convex form and solved globally. Let \bar{x} be a vector of n real positive variables. A function f is called a posynomial function of \bar{x} if it has the following form:

$$f(x_1, \dots, x_n) = \sum_{k=1}^t c_k x_1^{\alpha_{1,k}} x_2^{\alpha_{2,k}} \dots x_n^{\alpha_{n,k}}, \quad c_j > 0, \quad \alpha_{i,j} \in \mathbb{R}$$

Note that posynomials exclude the expression of a negative term but can express a fraction. When $t = 1$, the expression is called a monomial. Geometric programming solves an optimization problem of the following form:

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

Here f_i and f_0 are posynomials while g_i are monomials. A geometric program can be solved for the global optimum in a few seconds using interior point methods.

3. Expressing circuit as a GP

Two approaches have been proposed for expressing the circuit as a GP. Hershenson, Mandal [14, 11] formulates the circuit specifications as a function of MOS parameters. AC specifications are expressed as posynomial functions of MOS small signal specifications, while DC specifications are posynomial functions of large-signal parameters such as threshold voltage, effective voltage and current. The large signal transient specifications are simple hand-written equations. Subsequently, the MOS parameters are expressed as monomial functions of the width, length and current of the transistor. Since posynomials are closed under addition and multiplication, the final equations for specifications and constraints turn out to be posynomials in term of the circuit W, L and I. The convex solver solves to give the optimal value of W, L and I of all devices and other parameters such as resistor and capacitor values.

The second approach by Daems [5], et.al. follows a different strategy. It simulates the circuit at a set of points chosen using Design-of-Experiments to get the value of circuit specifications from SPICE. It uses black-box fitting techniques to directly fit the circuit specifications in posynomial functions of W, L and I. These equations are then optimized using GP. This approach allows to get more accurate models of transient specifications, which in the first approach were hand-written first-order approximations.

However, the second approach suffers from a big disadvantage. It sacrifices the scalability of GP, the very reason for which it is attractive. In this approach, every new circuit topology or the same topology on a new technology needs to be re-simulated to collect modeling points. As the size of the circuit increases, the dimensionality of the problem will increase. Increase of dimensionality would need exponentially more number of modeling points for accurate models. This phenomenon is well-known in Machine Learning literature [10] as the 'curse of dimensionality'. Thus, the approach is not scalable because it uses exponentially increasing simulations as the circuit size increase. Moreover the approach calls for design-centering, which will need

even more simulations. This makes the approach similar to simulation-based sizing methods with respect to high computational cost.

Secondly, the modeling approach used in [5] allows only integer exponents for variables. There is clear intuition that the circuit specifications depend on non-integer exponents for width, length and current. Consider, for instance, gain for a single-stage common source amplifier is the product of gm (transconductance) and ro (mos output resistance). This makes the gain proportional to the square-root of width, length and inverse square root of current in the transistor (assuming transistor in strong inversion). Integer-valued exponents completely disregard this observation and this explains for the inaccurate generalization of posynomial models in [5].

These issues tip the scale toward the first approach of formulating a GP as in [11]. The approach provides scalability but remains hindered by the inaccuracy of the MOS parameter models. The MOS parameter models that require remediation are either first-order analytical approximations or trained monomials. Our contribution is to build more accurate posynomial models for MOS parameters. We use simulation data yet we do not forfeit scalability because, for a given technology, the posynomial models have to be designed only once and they could be *reused* multiple times for all circuits in the technology.

4. Posynomial Modeling Approaches

There are a few approaches for generating posynomial models in the literature. In [2], the monomial modeling problem is handled by using the logarithmic form of the monomial function. This reduces the problem to a linear regression problem which can be solved globally. There is no clean way to extend this approach for posynomial modeling. In [5], a constructive constrained linear-regression approach is used to fit posynomial models, however only integer exponents are found. Clearly, MOS parameters can have fractional exponents, for instance, consider the case of effective voltage or transconductance. Using only integers would mean disregard to this information, which should inform our prior for modeling. A third approach [13] fits a function into a set of piece-wise monomial functions. Piece-wise monomial functions can be used in a GP instead of a posynomial and hence one could model the MOS parameters in this form as well. Our algorithm automatically designs posynomial models with much more expressive real-valued exponents. The algorithm is a combination of a genetic algorithm and quadratic programming. It is a general algorithm to fit posynomials, regardless of the type of data. In Section 6, we compare the accuracy of our algorithm to all these other methods.

5. Algorithm to generate posynomial models

5.1. Modeling Objective

The objective of the modeling algorithm is to create posynomial models for each of the following 9 MOS parameters in terms of W (width), L (length) and I (current): gm , gds , ro , V_{eff} , V_t , V_{dsat} , C_{gs} , C_{gd} , C_{gs} . The input to the algorithm are training points, which have been derived from actual SPICE simulation of the transistors. The algorithm derives a posynomial which minimizes the squared error between actual simulation data and model results. The modeling algorithm is rerun for every MOS parameter.

5.2. Genetic Algorithms

Genetic algorithms [16] are a class of algorithms inspired by natural evolution. They have been widely used in analog CAD for real-valued optimization [18, 3, 19]. However, they are much broader than this and have been used for solving traveling salesman problem, combinatorial optimization and structural synthesis of circuits [1]. Here, we show how they can be combined with quadratic programming to evolve posynomial models. The basic flow of a genetic algorithm is the following: Each solution is expressed as a genotype. The algorithm builds an initial population of possible solutions, i.e. genotypes. It evaluates the performance of each and assigns its a corresponding value referred as fitness. It then selects the better solutions (e.g. circuits) from the population, applies variation operators to them to create a new population for the next generation. We now discuss how all these operators were encoded for posynomial modeling.

5.3 Posynomial Representation: Genotype

Figure 1 shows how the posynomial is expressed as the genotype. The genotype is a matrix of real numbered values as shown in Figure 1. Each row represents a term of the posynomial. The number of rows is fixed. A choice parameter associated with each row decides whether the row is actually used or not (1:used, 0:don't care). This allows posynomials with varying number of terms in the population. The number of rows is equivalent to the maximum number of possible terms in the posynomial. Each column is associated with one of the 3 input variables. The value in a cell encodes the exponent of the variable (represented by the column) for the term (represented by the row). All cell values are in a specified range $[minVal, maxVal]$. The coefficient of each term is not a part of the genotype.

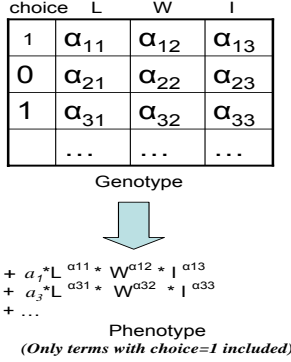


Figure 1. Genotype to Posynomial mapping

5.4 Fitness Evaluation

The GA evolves the exponents of all variables for each term. To determine the complete posynomial form of the candidate solution, the coefficient of each term must be determined. This is done deterministically, given the specific values of the exponents, with a minimization of mean square error (MSE) objective. We formulate a Quadratic Programming (QP) problem from the MSE objective function (because it is second degree) along with linear constraints that all coefficients are positive to ensure posynomial formulation. The coefficients found by QP are the global optima for the given exponents. A QP solver within Matlab is used to solve the QP problem. The minimum value of the error (minimum MSE) is a measure of the accuracy of the posynomial.

Our complete simulation data-set is substantially large (about 70000 points) and the formulation and solving of QP becomes computationally expensive for the whole dataset. Therefore, we only use a small uniformly sampled fraction of the dataset. Using this smaller fraction requires that the evolved model does not overfit the sampled points. To ensure this, we use 2-fold cross validation [10] on the sampled data set and use the cross validation MSE as the fitness of the individual.

To summarize, the candidate solution is derived by evolution of its exponents and QP optimization of its coefficients. It is evaluated on a fixed randomly sampled small fraction of the complete data set and the cross-validation error is used as the candidate's fitness.

Noteworthy is an effect arising from using QP to find the coefficients. The QP problem has the constraints that all coefficients should be more than zero. This formulation pushes some of the coefficients to exact zero. Thus QP implicitly performs *feature selection* on the evolved terms by setting the coefficients of useless terms to zero.

5.5 Variation Operators

The genotype information provides the exponents for a sum-of-products solution. The first variation operator we use is called crossover, which combines two genotypes to form a new set of exponents. We combine two genotypes by exchanging their monomial terms. Two solutions are selected from the current population and the new individual for the next generation is created by choosing each row from one of the two parents randomly. This is called uniform crossover. The second operator we use is the mutation operator. The mutation operator is used to perturb the real-numbered values in the cells of the matrix. A normal distribution centered at zero with a given variance (λ) is added to the real numbered value. The variance is adaptively decreased as the algorithm proceeds.

The choice of the real-value to be mutated is decided in the following way. A term with a zero coefficient is mutated by a probability (p_{zero_term}), the operation being reinitialization of the term randomly. A term with a non-zero coefficient is mutated by a different probability ($p_{non_zero_term}$). Each cell in the chosen row is mutated by a given probability (p_{cell}). The variation operators are informed by the building-block hypothesis of genetic algorithms.

6 Experiments

6.1 Simulation Data

We built posynomial models for mosfet technology TSMC 0.18 μ . We simulated and logged values of the 9 MOS parameters from SPICE simulation for an NMOS transistor. We chose the range for width and length from 0.18 μ to 20 μ with grid size of 0.1 μ . The current ranged from 0 to 5mA (equivalent Vgs and Vds). All points out-of-saturation were filtered and removed. Vds is an extra degree of freedom not included in the models as of now. Thus, only points at the edge of saturation were retained and rest were filtered out. This left approximately 70,000 points which were used for modeling.

6.2 Experimental Setup

We used the proposed genetic algorithm to evolve posynomial models for 9 mos model output variables in terms of its W, L and I. We sampled 2000 points uniformly from the complete set for fitness evaluation. Each genotype of generation 0 is initialized using a uniform random distribution bounded by $[-3, 3]$ for each cell element. The number of rows in the genome is 5. The choice parameter is randomly initialized to 1 or 0 such that the average number of terms per individual in the initial generation is 3. We use a generation based GA with tournament selection [16]. The

population size is 50 and tournament size is 6. The genetic algorithm parameters are given in Table 1.

Parameter	Value
Initial λ	1
λ rate	Halved every 20 generations
$p_{crossover}$	0.5
Mixing Ratio	0.7
p_{cell}	0.5
p_{zero_term}	0.7
$p_{non_zero_term}$	0.3

Table 1. GA Parameters

6.3 Results

A comparison of the normalized root mean-square (RMS) error (in percent) [15] of models created by compared approaches is shown in Table 2.¹ The results of our algorithm are in the 5th column. The sixth column of the table shows the percentage improvement of our algorithm over the best of the three compared approaches. We evolved posynomials for all 9 mos output variables using the proposed genetic algorithm. We ran 2 runs for each output variable for 1000 generations. In each generation, the coefficients of the best individual and its error were re-determined according to the complete data set. The posynomial expression which gave the least error for complete data over all runs and generations was used as the output of the algorithm.

For comparison, we designed monomials and piecewise monomials (PWM) using two published methods (reported in columns 2 and 3 of Table 2 respectively): 1. Log regression [2] was done to find a set of coefficients and exponents. The exponents and coefficients were re-tuned to minimize MSE using a gradient-descent method in the second step; 2. Piecewise monomial modeling approach [13] (code was received from the author) was used to build a model for the 2000 sampled points and the error was calculated on all the 70,000 points. We used an initial partition size of 2, 5, 10 and 15 and gave 10 trials for each of these setting. The model with the highest accuracy of all these runs is reported here. In the third approach, we improved upon the monomials created using log regression [2]. We fix the exponents of the model, however re-learn the coefficient of the monomial and an extra constant by a QP formulation (constrained linear regression). This results in a degenerate two-term posynomial (referred as $2P$) containing a monomial term and a constant.

¹Test and train error are not reported. The 70,000 simulation points cover the whole MOSFET operating range. There is no unseen data here.

Param	Mon.	PWM.	$2P$	$Posy.$	Imp
gm	4.62	31.64	3.15	1.82	42
gds	1.92	1.74	1.67	0.31	85
V_{eff}	1.69	6.30	1.69	0.71	58
Vt	6.05	2.60	6.03	3.01	–
V_{dsat}	1.56	0.86	1.56	0.81	6
Cgd	0.37	0.27	1.33	0.06	78
Cgs	2.09	45.15	2.07	1.97	5
Cdb	2.81	0.16	2.36	0.14	12
Ro	0.47	0.08	0.24	0.05	37

Table 2. Comparison of Normalized Root MSE (%) for different models

It can be seen that the posynomial models consistently outperform all the other modeling approaches for all parameters. Only in case of Vt , the piece-wise monomial gives better accuracy than posynomial models. Between 2-term posynomials and piece-wise monomials, there is no clear better approach and they outperform each other for different parameters. Our posynomial modeling approach brings the normalized RMS error under 3%.

The above results indicate that the monomial/posynomial models are highly accurate with error being less than 10% in all cases. However, this isn't completely true due to the following reason. The MOS parameters span several orders of magnitude, for instance in the given data, gm , Ro and Cgs span 4, 6 and 3 orders of magnitude respectively. This implies that the mean square error metric may sacrifice the accuracy of lower-values heavily for better higher values. This can be remedied by using the mean relative absolute error (MRAE) formulation for fitting the model. MRAE is defined as $mean(|y - y_{fitted}|/|y|)$. This error metric sums the percent error for each point, giving accuracy of lower and higher-valued point equal weights. We modified our algorithm by modifying the fitness evaluation to optimize coefficients for MRAE using a linear program. The results are compiled in Table 3. The other approaches are not easily extensible to MRAE measure since they use log transformation before fitting.

The results show that the error for V_{dsat} is alarmingly high. Also, gm , gds and ro have moderately high error. Other errors are under acceptable range. The algorithm needs to be further improved to bring all errors in 5% accuracy. An alternative is to limit the range of parameters to only the known useful areas, however this comes at the price of optimality.

Parameter	MRAE (%)
gm	13.00
gds	7.21
V_{eff}	66.20
V_t	0.35
V_{dsat}	1.09
C_{gd}	0.28
C_{gs}	4.32
C_{db}	0.18
R_o	9.96

Table 3. MRAE for evolved posynomials

7 Conclusion and future work

The present work proposes an algorithm for automatic posynomial modeling of MOSFET parameters for use in GP. The approach is a generic way for posynomial modeling irrespective of the kind of data. It is the only approach which can automatically design posynomials with real-valued exponents. We show how the given approach is superior to other published approaches for monomial or piece-wise monomial modeling.

This is our first step towards reducing the inaccuracy in circuit optimization using GP. The models are based on simulation data and are reusable multiple times for a given technology. The algorithm can be used for building models for different devices such as MESFETS, Si-Ge devices, RF devices, etc. The created models though considerably accurate and always better in terms of RMS error than any existing models, need some more innovation in algorithms to be completely satisfactory. GP, if made accurate, offers scalability. The power of this scalability is immense should it prove able to address robustness, which is infeasible in the simulation based approach. Beyond reducing model error accuracy further, the next larger challenge in GP is to automatically derive accurate models of circuit performance parameters in terms of MOS parameters. These are currently hand-written and inaccurate. The final challenge, which rounds out our GP research agenda is to be able to express transient specifications through reusable models.

References

- [1] H.-G. Beyer and U.-M. O'Reilly, editors. *Genetic and Evolutionary Computation Conference, GECCO 2005, Proceedings, Washington DC, USA, June 25-29, 2005*. ACM, 2005.
- [2] S. Boyd, S.-J. Kim, L. Vaudenberghe, and A. Hassibi. A tutorial on geometric programming. In *Technical report, EE Department, Stanford University*, 2004.
- [3] M. Chu, D. J. Allstot, J. M. Huard, and K. Y. Wong. NSGA-based parasitic-aware optimization of a 5GHz low-noise VCO. *asp-dac*, 00:169–173, 2004.

- [4] 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 programming. In *IEEE Custom Integrated Circuits Conference*, pages 377–380, 2003.
- [5] W. Daems and G. Gielen. Simulation-based generation of posynomial performance models for the sizing of analog integrated circuits. *CAD of Integrated Circuits and Systems, IEEE Trans.*, 22(5):517–534, 2003.
- [6] K. Deb, S. Agrawal, A. Pratap, and T. Meyarivan. A fast elitist non-dominated sorting genetic algorithm for multi-objective optimisation: NSGA-II. In *PPSN*, pages 849–858, 2000.
- [7] F. M. El-Turky and E. E. Perry. BLADES: an artificial intelligence approach to analog circuit design. *IEEE Trans. on CAD of Int. Circuits and Systems*, 8(6):680–692, 1989.
- [8] G. Gielen, K. Swings, and W. Sansen. An intelligent design system for analogue integrated circuits. In *EURO-DAC '90*, pages 169–173, Los Alamitos, CA, USA, 1990. IEEE Computer Society Press.
- [9] R. Harjani, R. A. Rutenbar, and L. R. Carley. OASYS: a framework for analog circuit synthesis. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 8(12):1247–1266, 1989.
- [10] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning*. Springer, 2001.
- [11] M. Hershenson, S. Boyd, and T. Lee. GPCAD: A tool for CMOS op-amp synthesis. In *Int. Conf. on CAD*, pages 296–303. *IEEE/ACM, November 1998.*, 1998.
- [12] M. Hershenson, S. S. Mohan, S. P. Boyd, and T. H. Lee. Optimization of inductor circuits via geometric programming. In *DAC*, pages 994–998, New York, NY, USA, 1999. ACM Press.
- [13] A. Magnani and S. Boyd. Convex piecewise-linear fitting. In *Submitted for publication by authors*, 2006.
- [14] P. Mandal and V. Visvanathan. CMOS op-amp sizing using a geometric programming formulation. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 20(1):22–38, 2001.
- [15] T. McConaghy, T. Eeckelaert, and G. Gielen. CAFFEINE: Template-free symbolic model generation of analog circuits via canonical form functions and genetic programming. In *Proc. DATE*, volume 2, pages 1082–1087, Munich, 2005.
- [16] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, Cambridge, MA, USA, 1998.
- [17] P. Siarry, G. Berthiau, F. Durdin, and J. Haussy. Enhanced simulated annealing for globally minimizing functions of many-continuous variables. *ACM Trans. Math. Softw.*, 23(2):209–228, 1997.
- [18] B. D. Smedt and G. Gielen. WATSON: Design space boundary exploration and model generation for analog and RF IC design. *IEEE Trans. on CAD of Int. Circuits and Systems*, 22(2):213–224, 2003.
- [19] S. K. Tiwary, P. K. Tiwary, and R. A. Rutenbar. Generation of yield-aware pareto surfaces for hierarchical circuit design space exploration. In *DAC*, pages 31–36, 2006.

Acknowledgements

We would like to thank — for their assistance with test data and helpful discussions