Analog Circuit Simulation Using Range Arithmetics

Darius Grabowski, Markus Olbrich, and Erich Barke

Institute of Microelectronic Systems
Leibniz Universität Hannover
30419 Hannover, Germany
e-mail: {dg,mo,eb}@ims.uni-hannover.de

Abstract— The impact of parameter variations in integrated analog circuits is usually analyzed by Monte Carlo methods with a high number of simulation runs. Few approaches based on interval arithmetic were not successful due to tremendous overapproximations. In this paper, we describe an innovative approach computing transient and DC simulations of nonlinear analog circuits with symbolic range representations that keeps correlation information, and hence has a very limited overapproximation. The methods are based on affine and quadratic arithmetic. Ranges are represented by unique symbols so that linear correlation information is preserved. We demonstrate feasibility of the methods by simulation results using complex analog circuits.

I. Introduction

The impact of parameter variations in microelectronic circuits is increasing due to downsizing. The verification of circuits under presence of uncertainties like process variations, drift, aging, noise, etc. is still a challenge. Consider the illustration in Fig. 1 showing a specified tolerance scheme in time domain. The desired objective is to prove that under presence of given parameter deviations the circuit's outputs do not exceed the specified tolerances.

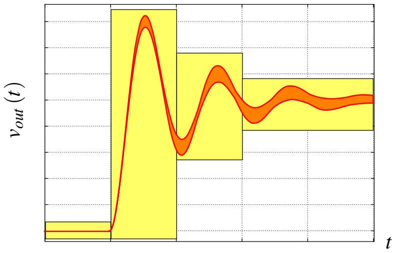

Fig. 1. Tolerance Scheme in Time Domain

Monte Carlo or corner case simulations [13] need a large number of simulation runs to compute output signals. Nevertheless, they only provide limited reliability. Sensitivity analysis [8] gives some information about the impact of parameter variations on circuit behavior, however, the inclusion of all possible results is not guaranteed.

Range arithmetics offer an alternative to explore systems under parameter variations computing bounds for possible output signals as shown in Figure 1. In this context, interval arithmetic (IA) [11] plays a key role. Hansen [7] proposes a methodology to compute outer solutions for linear interval equations, whereas Neumaier [12] describes how to solve general equation systems using interval arithmetic. Unfortunately, IA leads to very high overapproximations that restrict applicability of this approach.

A promising area has been entered by the introduction of affine arithmetic (AA) [1]. In contrast to IA, affine arithmetic preserves linear correlation information throughout the computations. Hence, the overapproximation of the results – especially in the case of longer computation chains – becomes smaller. In [9, 14] some approaches based on IA and AA for the worst case analysis of linear analog circuits are presented. Femia [4] reviews some small analog circuits with parameter variations assuming an explicit mathematical formulation for the output of interest. In [5, 6], simulation of mixed-signal systems at block level is described. Nevertheless, the approach is restricted to almost linear systems with explicit formulations.

In this paper, we present a novel approach for affine simulation of analog circuits at transistor level in DC and transient domain. The main benefit of the new approach is the capability to solve nonlinear implicit equation systems with affine arithmetic. Parameter variations are described by affine forms using a symbolic representation. The simulation results contain a combination of those symbols indicating the contribution of each parameter's variation. Furthermore, we present new extensions that reduce overapproximation significantly by replacing AA evaluations by quadratic arithmetic. The experimental results show that the proposed algorithms cope with analog circuits (FET-circuits) of relatively high complexity.

The remainder of this paper is organized as follows: In Section II we give an overview of affine arithmetic. Section III describes the implemented algorithm in detail and gives some characteristics how to analyze circuits in DC and transient domain. Section IV presents the enhancement considering quadratic arithmetic which additionally reduces overapproximation. The experimental results are presented and discussed in Section V.

978-1-4244-1922-7/08/$25.00 ©2008 IEEE

II. Affine Arithmetic

Affine arithmetic defines operations on affine forms (AF) which are first degree polynomials (Eq. 1). Affine forms and operations will by denoted by (ˆ) in this paper. An affine form is made up by a central value and a sum of deviations $x_i \in \mathbf{R}$ scaled by formal symbols ε_i, also called noise symbols. To avoid misunderstandings in relation to noise definitions in circuit theory, we call the symbols deviation symbols.

$$\hat{x} := x_0 + \sum_{i \in \mathcal{N}_{\hat{x}}} x_i \varepsilon_i \tag{1}$$

Each ε_i corresponds to a distinct property throughout all computations and might accept a real value within $[-1,1]$. $\mathcal{N}_{\hat{x}}$ is a set of positive integers.

The radius is the total deviation of an affine form defined by the following formula:

$$rad(\hat{x}) \quad := \quad \sum_{i \in \mathcal{N}_{\hat{x}}} |x_i| \,. \tag{2}$$

Mathematical operations on affine forms return a new affine form and can be separated into two classes: linear $\hat{f}^{[l]}$ and nonlinear operations $\hat{f}^{[nl]}$. Linear operations include addition, subtraction and scalar multiplication. They provide exact results and do not generate further deviation terms.

$$\begin{aligned} \hat{f}^{[l]}(\hat{x} \pm \hat{y}) \quad &:= \quad \hat{x} \pm \hat{y} = x_0 \pm y_0 \\ &+ \sum_{i \in \left(\mathcal{N}_{\hat{x}} \cup \mathcal{N}_{\hat{y}}\right)} (x_i \pm y_i)\varepsilon_i \end{aligned} \tag{3}$$

$$\hat{f}^{[l]}(c\hat{x}) \quad := \quad c\hat{x} = cx_0 + \sum_{i \in \mathcal{N}_{\hat{x}}} cx_i \varepsilon_i \tag{4}$$

In contrast, each nonlinear operation is approximated by a linear operation introducing a new (independent) deviation term $y_k \varepsilon_k$ indicating the approximation error.

$$\hat{y} = \hat{f}^{[nl]}(\hat{x}) = \hat{f}^{[l]}(\hat{x}) + y_k \varepsilon_k \quad k \notin \mathcal{N}_{\hat{x}} \tag{5}$$

Note that each computation chain can be expressed by a sequence of fundamental functions (e.g. $e^x, log(x), x^n, ...$), so that only implementations of those functions are required.

There are several ways to approximate nonlinear operations. Geometrically, these methods enclose the nonlinear curves by parallelograms as shown in Fig. 2. The properties of the parallelograms are extracted by the monotony information of the operation. The methods guarantee the inclusion of the exact solution:

$$\hat{f}^{[nl]}(\hat{x}) \supseteq f(\hat{x}). \tag{6}$$

The parallelogram's area corresponds to the approximation error which depends on the nonlinearity of the operation and the radius of $\hat{x}$ affecting the convergence of the algorithm presented in the next chapter.

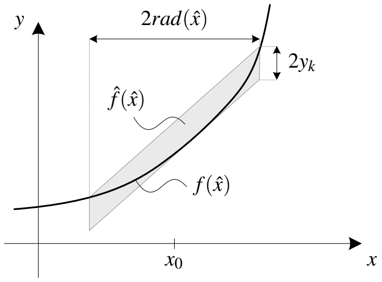

Fig. 2. Affine Approximation

A. Affine Parameter Description

For a better understanding we point out the advantage of affine arithmetic compared to interval arithmetic. Describing circuit parameters with deviations can be split up into two categories. The first category deals with various parameter deviations which are not correlated (e.g. two discrete resistors with different process parameters). In this case, we use formulations with different symbols ε_1 and ε_2:

$$\hat{R}_1 = R_{1,0} + R_{1,1}\varepsilon_1 \quad \hat{R}_2 = R_{2,0} + R_{2,1}\varepsilon_2. \tag{7}$$

The second category deals with linearly correlated parameter variations. For example, the width of (linearly) matching transistors can be described as follows:

$$\hat{W}_+ = W_0 + W_1\varepsilon_1 \quad \hat{W}_- = W_0 + W_1\varepsilon_1. \tag{8}$$

More general, manufacturing tolerances are modeled by a normal probability density function *pdf* defined by the mean value and the covariance matrix [3]. The equidensity contours of a normal pdf are n-ellipsoids defining a tolerance body (e.g. 3σ-body). Since superposition holds, affine descriptions permit combinations of (linearly) correlated deviations describing a n-dimensional convex hull which approximates the n-ellipsoids.

Note that conventional interval arithmetic provides only the possibility to describe uncorrelated parameter deviations (n-hypercubes). Consequently, a differential operation of the W, as it often appears implicitly in analog circuits, provides a result containing $2W_1$ in contrast to affine arithmetic yielding 0.

III. Circuit Simulation with AA

The idea of affine simulation was initially applied to block diagrams describing almost linear control loops [6]. Theory of linear systems allows us to give an explicit formulation of the output quantities as functions of input quantities and inner states in case of dynamical systems. This approach has been extended to blocks with small nonlinearities - but still including only explicit formulations.

A. Mathematical Circuit Model

At transistor level, analog circuits are ordinarily described by nonlinear differential algebraic equation systems (DAEs). Quantities like node potentials and branch currents are determined implicitly. This property requires an alternative approach to solve equations with affine arithmetic providing an outer inclusion of all possible results with respect to parameter variations.

For DC-analysis all time derivatives are eliminated, so that the DAE-System is reduced to a nonlinear equation system:

$$\underline{F}(\underline{x},\underline{p}) = \underline{0}. \tag{9}$$

Variables are denoted by $\underline{x}$, whereas parameters are described by the vector $\underline{p}$. Circuit description for transient (TR) simulation extends the nonlinear equation system to a DAE by time derivatives. Generally, a closed-form solution can not be obtained due to nonlinearities. Therefore, TR analysis uses numerical methods. These methods, such as implicit Euler or trapezoidal method, substitute the time derivatives by difference quotients mapping the DAE to a nonlinear equation system.

$$\underline{F}(\underline{x}(t),\underline{\dot{x}}(t),\underline{p},t) \rightarrow \underline{F}(\underline{x}_n,\underline{x}_{n-1},\underline{x}_{n-2},...,\underline{p},t) \tag{10}$$

Handling of previous time step solutions $\underline{x}_{n-k}$ in affine simulation is an open issue. We propose the introduction of an extended parameter vector $\underline{p}_x$ combining circuit parameters and all $\underline{x}_{n-k}$:

$$\underline{p}_x^T := \left[\underline{p},\underline{x}_{n-1},\underline{x}_{n-2},...\right]^T. \tag{11}$$

For simplicity, we use $\underline{p}$ for the extended parameter vector in the following.

B. Affine Solutions

Starting point is an equation system which contains variables $\underline{x}$ and parameters $\underline{p}$. Note that usually, since at least one parameter is range valued, the solution contains range valued variables as well.

$$\underline{F}(\underline{x},\underline{p}) = 0 \tag{12}$$

Besides the equation system, the Jacobian $\mathbf{J}$ and the derivatives with respect to the parameters $\mathbf{P}$ are required.

$$\mathbf{J} = \frac{\partial \underline{F}(\underline{x},\underline{p})}{\partial \underline{x}} \qquad \mathbf{P} = \frac{\partial \underline{F}(\underline{x},\underline{p})}{\partial \underline{p}} \tag{13}$$

In Alg. 1 the proposed algorithm for nonlinear equation systems is presented. In the first steps (1-4), the initial affine solution is computed. First, the nominal solution, usually obtained by Newton-Raphson method, is computed and assumed to be the central value of the affine solution. Therefore, we replace all parameters with their central values. In Step 3, the equation system is linearized in the operating point with respect to the variables and parameters. Considering the partial derivatives, we get the linear dependency (sensitivity) of the variables according to the deviations of the parameters. Note that, if the equation system contains at least affine expressions (linear systems with constant parameters and variable inputs) the initial affine solution is exact and the algorithm ends. As the equation system contains nonlinear expressions, the initial affine solution is usually an underestimation of the exact result, thus it has to be extended to preserve the inclusion of the exact solution area (shaded area). This idea is comparable to the affine extension used for nonlinear approximations shown in Sec. II. To guarantee the inclusion condition, each variable is extended by one (independent) extended deviation (ED). The initial guess is assumed to be 10 % of the radius of the initial affine solution. In the following steps the EDs are computed iteratively. In Step 11, the equation system is evaluated using affine arithmetic. The affine residual vector includes the origin since the nominal solution is part of the affine solution computed so far. Furthermore, $\underline{\hat{r}}$ contains linear combinations of deviations corresponding to the ED introduced in Step 10 and further deviation terms arising from nonlinear operations. Based on this result, we can map the residual on an n-dimensional polytope spanned onto the Cartesian Space described by $\underline{\varepsilon}_{\mathrm{ED}}$. For simplicity, the graphical interpretation is shown in Figure 3 for a single dimension. The parallelogram represents an overapproximation of all possible residuals due to the actual solution. The exact manifold is indicated by the shaded area inside the parallelogram. Consequently, all zeros are included in the parallelogram as well. In order to find the entire set of zeros, we have to resize the parallelogram computing the maximum or minimum zeros respectively

Algorithm 1 AAFSolve

```
 1: $\underline{x}_0 \leftarrow \underline{F}(\underline{x}_0,\underline{p}_0) = \underline{0}$
 2: $\underline{\hat{x}} \leftarrow \underline{x}_0$
 3: $\mathbf{J}|_{\underline{x}_0,\underline{p}_0} \cdot (\underline{\hat{x}} - \underline{x}_0) + \mathbf{P}|_{\underline{x}_0,\underline{p}_0} \cdot (\underline{\hat{p}} - \underline{p}_0) = \underline{0}$
 4: $\underline{\hat{x}} \leftarrow \underline{\hat{x}} - \mathbf{J}^{-1} \cdot \mathbf{P} \cdot (\underline{\hat{p}} - \underline{p}_0)$
 5: if $\underline{\hat{F}}(\underline{\hat{x}}) = \underline{0}$ then
 6:     BREAK
 7: else
 8:     $\underline{\hat{x}}_{init} \leftarrow \underline{\hat{x}}$
 9:     Compute initial guess: $\underline{\hat{x}}_{ED}$
10:     repeat
11:         $\underline{\hat{x}} \leftarrow \underline{\hat{x}}_{init} + \underline{\hat{x}}_{ED}$
12:         $\underline{\hat{r}} = \underline{\hat{F}}(\underline{\hat{x}},\underline{\hat{p}}) \quad (\neq \underline{0})$
13:         Solve: $\mathbf{M}_{ED} \cdot \underline{\hat{\lambda}} = \underline{\hat{r}}'$
14:         for all $\lambda_i \in \underline{\lambda}$ do
15:             if $\max(\underline{\hat{\lambda}}_i) \geq -\min(\underline{\hat{\lambda}}_i)$ then
16:                 $\lambda_i \leftarrow \max(\underline{\hat{\lambda}}_i)$
17:             else
18:                 $\lambda_i \leftarrow -\min(\underline{\hat{\lambda}}_i)$
19:             end if
20:         end for
21:         Update: $\forall\lambda_i : \hat{x}_{ED,i} \leftarrow \lambda_i \cdot \hat{x}_{ED,i}$
22:     until $\forall\lambda_i : \lambda_i \rightarrow 1.$
23: end if
```

(denoted as possible zeros in Fig. 3). Therefore, solving the linear affine equation system (Step 14) is required. The entries of the real valued matrix $\mathbf{M}_{ED}$ are the deviations of the residual vector $\underline{\hat{r}}$ corresponding to the EDs. The right-hand-side vector $\underline{\hat{r}}'$ contains the central values and all deviations of $\underline{\hat{r}}$ except for the EDs. As the right-hand-side is an affine vector, the scaling vector $\underline{\hat{\lambda}}$ is also affine. The maximum/minimum zeros are computed by the maximum/minimum of $\underline{\hat{\lambda}}$ (Lines 16-19). Hence, the algorithm ends as all entries of $\underline{\lambda}$ converge to unity.

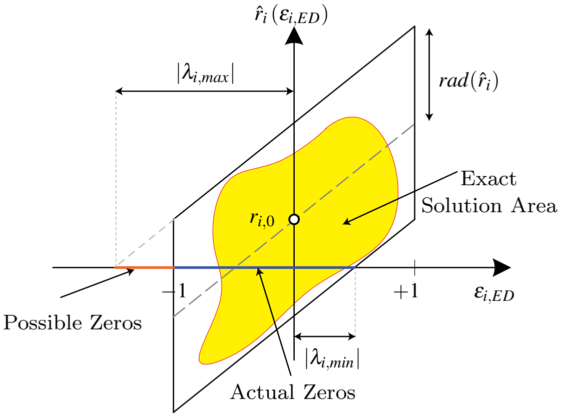

Fig. 3. Polytope in Case of Single Variable

IV. Quadratic Arithmetic

In this chapter, we present an extension of the affine arithmetic which we call quadratic arithmetic (QA) [10]. Quadratic affine forms (QF) will be denoted by (˜) in this paper. A QF consists of a central value, a sum of linear terms representing linear deviations scaled by formal symbols and a double sum of quadratic terms representing quadratic deviations.

$$\tilde{x} := x_0 + \sum_{i\in\mathcal{N}_{\tilde{x}}} x_i\varepsilon_i + \sum_{i\in\mathcal{N}_{\tilde{x}}} \sum_{j\in\mathcal{N}_{\tilde{x}}}^{j\geq i} x_{ij}\varepsilon_{ij} \tag{14}$$

Similar to affine arithmetic each ε_i corresponds to a distinct property throughout all computations and might accept a real value within $[-1,1]$ whereas $\mathcal{N}_{\tilde{x}}$ is a set of positive integers. For quadratic deviation symbols we make the following assumption:

$$\begin{aligned} \varepsilon_{ij} &\in [-1,1] \quad \forall(i\neq j) \\ \varepsilon_{ij} &\in [0,1] \quad \forall(i=j). \end{aligned} \tag{15}$$

Computation of the exact total deviation is more complex because the QF is not symmetrically anymore. Furthermore, the linear terms are correlated with the quadratic terms which provides solving a combinatorial problem to get the exact total minimum and maximum. For simplicity we use a straightforward computation neglecting the correspondence between ε_i and ε_{ij} and the modified interval of ε_{ii}:

$$\mathrm{rad}(\tilde{x}) := \sum_{i\in\mathcal{N}_{\tilde{x}}} |x_i| + \sum_{i\in\mathcal{N}_{\tilde{x}}} \sum_{j\in\mathcal{N}_{\tilde{x}}}^{j\geq i} |x_{ij}| \geq \mathrm{rad}^{exact}(\tilde{x}) \tag{16}$$

In contrast to AA, the multiplication and the square function of AFs are exact in case of quadratic arithmetic. Other nonlinear functions are approximated by quadratic polynomials. Taylor series expansion (Fig. 4) is a well-established method but it is not ideal in terms of approximation error. To reduce overapproximation we formulate the optimization problem of finding the best quadratic approximation for fundamental functions which is known as the mini-max approximation. Given a function $f(x)$, $x \in [x_{lo}..x_{hi}]$, we find the quadratic approximation $q(x)$ with the minimum approximation error $r(x)$:

$$\min\left[\max_{x\in I}(|r(x)|)\right]. \tag{17}$$

Finding the mini-max approximation is expensive. Therefore, we propose the Chebyshev approximation based on the Chebyshev polynomials. The problem can be mapped on finding the optimal grid points z_0, z_1 and z_2 inside the given interval reducing the approximation error. The error is bounded by the function:

$$r_2(x) = \frac{(x-z_0)(x-z_1)(x-z_2)}{3!} f^3(\xi),\ \xi \in [-1,1] \tag{18}$$

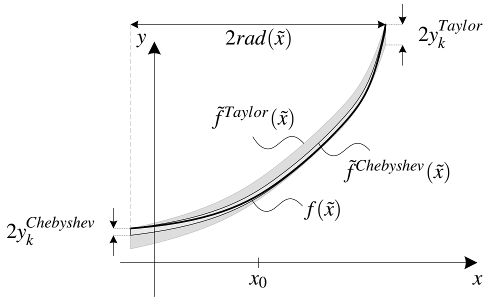

Fig. 4. Quadratic Approximation

Since we do not know the value ξ, we take the monic polynomial expression into account trying to minimize it. Modified Chebyshev polynomials are monic. They are computed recursively in the following manner:

$$\begin{aligned} &T_{n+1}(x) = x\cdot T_n(x) - \frac{1}{2}\cdot T_{n-1}(x), \qquad n\geq 1 \\ &T_0(x) = 1 \quad T_1(x) = x. \end{aligned} \tag{19}$$

The most important property of modified Chebyshev polynomials is that the resulting range for $x \in [-1..1]$ is minimal. In this case we consider the modified Chebyshev polynomial of third degree having 3 roots at $0, \sqrt{3}/2$ and $-\sqrt{3}/2$. The maximum value of the polynomial on the unity interval is 1/4. Therefore, the maximum error is approximated by:

$$r_2(x) = \frac{1}{24} f^3(\xi),\, \xi \in [-1,1]. \tag{20}$$

As we take into account the monotony properties of the fundamental functions and their derivatives it can be shown that the maximum error occurs at the borders of the interval making error computation less difficult.

We extended Alg. 1 by quadratic arithmetic to reduce the overapproximation in the following manner: The computation of initial affine solution remains the same as in affine simulation. Consequently, the solution vector is still affine. Quadratic arithmetic has only been applied to the iterative ED computation as shown in Alg. 2.

Algorithm 2 QPFSolve

```
repeat
    $\underline{\hat{x}} \leftarrow \underline{\hat{x}}_{init} + \underline{\hat{x}}_{ED}$
    $\underline{\tilde{r}} = \underline{\tilde{F}}(\underline{\hat{x}}, \underline{\hat{p}}) \quad (\neq \underline{0})$
    Solve: $\mathbf{M}_{ED} \cdot \underline{\tilde{\lambda}} = \underline{\tilde{r}}'$
    for all $\lambda_i \in \underline{\lambda}$ do
        if $\max(\underline{\tilde{\lambda}}_i) \geq -\min(\underline{\tilde{\lambda}}_i)$ then
            $\lambda_i \leftarrow \max(\underline{\tilde{\lambda}}_i)$
        else
            $\lambda_i \leftarrow -\min(\underline{\tilde{\lambda}}_i)$
        end if
    end for
    Update: $\forall \lambda_i : \hat{x}_{ED,i} \leftarrow \lambda_i \cdot \hat{x}_{ED,i}$
until $\forall \lambda_i : \lambda_i \rightarrow 1$.
```

Quadratic approximation involves errors of third degree (Eq. 20), whereas affine approximation leads to quadratic errors.

V. Experimental Results

The proposed algorithms (Alg. 1-2) have been tested with several circuits of different complexities. In this contribution, we present a mixed-level simulation of a control loop with a DC motor. We show how the proposed algorithms can be used to check specification tolerances. Finally, we compare both algorithms in terms of computation time and overapproximation. As a reference, corner case simulations are used.

A. Control Loop with DC Motor

The system consists of a control loop as shown in Fig. 5. The PID controller is built up by three stages representing the proportional, integral and derivative operation. The DC motor is described by a linear behavioral model. The mechanical quantity of interest is the actual motor position $\phi(t)$ which is controlled by the input voltage $v(t)$. We assume, that the position is proportional to the voltage occurring in the feedback of the loop.

The tolerance scheme of the system is specified for the step response at the set point by e.g. settling time or overshoot restrictions. Several parameter variations for the PID controller have been assumed. The resistors and capacitors were assumed to have deviations of 3 % each without any correlations. Figure 6 shows the schematic of the Miller op-amp which is used for the realization of the PID controller. It consists of 8 transistors and the Miller capacitor. All transistors are described by the Schichman-Hodges MOS model [2]. Matching transistor pairs – indicated by the shaded areas – are assumed to have 5% deviation of W.

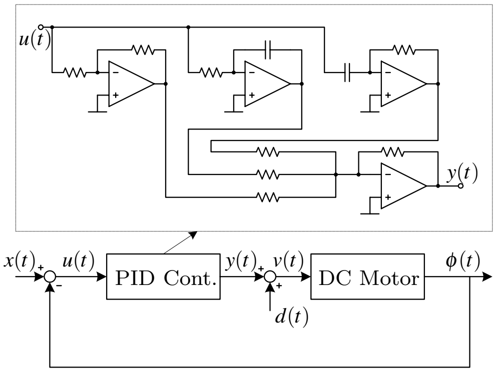

Fig. 5. Control Loop

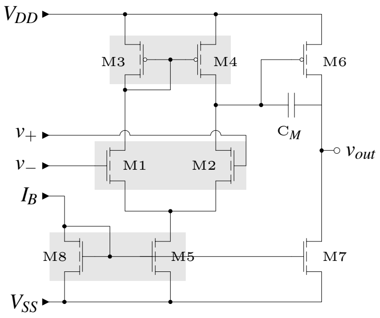

Fig. 6. Miller Operational Amplifier

Fig. 7 shows the tolerance scheme and the simulation results obtained by Algorithms 1 and 2. The tolerance scheme is composed of rectangles given for several specifications. Additionally, the nominal solution and results for corner cases are shown. The tubes are only simplified representations of the affine values by their upper and lower bounds. In fact, much more information is available. Alg. 2 reduces the overapproximation compared to Alg. 1 by about 7%.

B. Simulation Performance

The simulation results were obtained on a 2.6-GHz-AMD-Opteron workstation. First, we scheduled the relative simulation time (time per affine iteration) using the Miller op-amp (Fig. 6) and increased the number of parameters with deviations consecutively. The simulation time

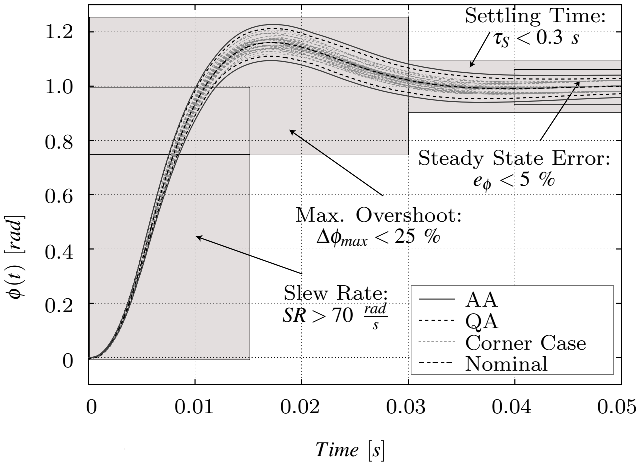

Fig. 7. Step Response and Tolerance Specification

with Alg. 1 remains almost constant whereas simulation with QA depends linearly on the number of parameters. We can figure out that the simulation time depends generally on the structure of the equation system. The absolute simulation times and radius reductions for various circuits are presented in Tab. I. Although simulation using Alg. 2 is more time-consuming, it comes along with the reduced overapproximation which yield better conclusions for tolerance specification analysis. Nevertheless, both approaches still yield a good performance compared to an exhaustive corner case simulation.

Circuit	n	τ_{AA}	τ_{QA}	Δr_{av}	τ_{CC}
Miller Op-amp	36	0.6 s	1.0 s	-32 %	14.4 s
Exp-Generator	128	4.7 s	26.7 s	-15 %	1.5 h
Control Loop	142	2 min	18 min	-7 %	1.4 h
Bandpass Filter	242	11 min	1.1 h	-40 %	8 h

TABLE I
Simulation Performance

n : # of Equations
τ_{AA} : Simulation Runtime (Alg. 1)
τ_{QA} : Simulation Runtime (Alg. 2)
Δr_{av} : Average Radius Reduction
τ_{CC} : Simulation Runtime (Corner Case)

VI. Conclusion and Future Work

To explore the impact of parameter variations in analog circuits, we developed a new approach based on range arithmetics. So far, the implementation provides analysis of nonlinear analog circuits in DC and transient domain. We consider affine forms to describe parameter variations within distinct bounds, and use affine arithmetics to compute affine solutions of circuit quantities including the exact results. The affine solutions contain symbols providing information about contributions of parameter variations to circuit quantities. For this reason, affine simulation is more reliable than corner case or Monte Carlo analysis.

Although affine computations yield better inclusions than interval arithmetic, we additionally improved the approximations by quadratic arithmetic. Therefore, quadratic forms have been introduced. The reduced overapproximation involves an increased runtime.

Circuit examples actually demonstrate feasibility, comparing affine and quadratic simulation to Monte Carlo or corner case analysis. The fact that the results are conservative motivates applications such as Model Checking or Property Refinement. The symbolic representation can initiate further work such as yield analysis and optimization. At the same time affine simulation lays the foundations for reliable model generation with respect to parameter variations promising faster simulations and reduced overapproximation.

References

[1] M. Andrade, J. Comba, and J. Stolfi. Affine Arithmetic (Extended Abstract). In *INTERVAL '94, St. Petersburg, Russia*, 1994.

[2] D. A. Divekar. *FET Modeling for Circuit Simulation.* Kluwer Academic Publishers, 1988.

[3] K. S. Eshbaugh. Generation of correlated parameters for statistical circuit simulation. *IEEE Trans. on CAD of Integrated Circuits and Systems*, 11(10):1198–1206, 1992.

[4] N. Femia and G. Spagnuolo. True Worst-Case Circuit Tolerance Analysis Using Genetic Algorithms and Affine Arithmetic. *IEEE Transactions on Circuits and Systems*, 47(9), 2000.

[5] D. Grabowski, C. Grimm, and E. Barke. Semi-Symbolic Modeling and Simulation of Circuits and Systems. In *International Symposium on Circuits and Systems (ISCAS 2006)*, Kos, Greece, 2006. IEEE Press.

[6] C. Grimm, W. Heupke, and K. Waldschmidt. Refinement of Mixed-Signal Systems with Affine Arithmetic. In *Design and Test in Europe 2004 (DATE '04)*, Paris, France, 2004.

[7] E. R. Hansen. Bounding the solution of interval linear equations. *SIAM J. Numer. Anal.*, 29(5):1493–1503, 1992.

[8] D. E. Hocevar, P. Yang, T. N. Trick, and B. D. Epler. Transient Sensitivity Computation for MOSFET Circuits. In *IEEE Transactions on CAD, Vol. 4, Issue 4*, pages 609–620. IEEE press, Oct 1985.

[9] L. Kolev. Worst-Case Tolerance Analysis of Linear DC and AC Electric Circuits. *IEEE Transactions on Circuits and Systems*, 49(12), 2002.

[10] F. Messine. Extensions of affine arithmetic: Application to unconstrained global optimization. In *Journal of Universal Computer Science, Vol. 8, Issue 11*, pages 992–1015, 2002.

[11] R. E. Moore. *Interval Analysis.* Prentice-Hall, Englewood Cliffs, NJ, 1966.

[12] A. Neumaier. *Interval Methods for Systems of Equations.* Cambridge University Press, 1990.

[13] R. Y. Rubinstein. *Simulation and the Monte Carlo Method.* John Wiley & Sons, Inc., New York, NY, USA, 1981.

[14] M. W. Tian and C.-J. Shi. Worst Case Tolerance Analysis of Linear Analog Circuits Using Sensitivity Bands. *IEEE Transactions on Circuits and Systems*, 47(8), 2000.