A comparison of interval methods in symbolic circuit analysis applications

B. Thanigaivelan[1] T. J. Hamilton[2] A. Postula[3]

(Received 3 February 2011; revised 13 December 2011)

Abstract

Symbolic circuit analysis involves deriving symbolic expressions for performance measures, such as voltage gain, input impedance, and evaluating them to obtain more insight into the behaviour of a circuit. In modern semiconductor technologies, it is more useful to evaluate the symbolic expressions using interval methods in order handle variations in parameter values. We compare the performance of different interval methods in evaluating symbolic expressions. Our experiments show that Generalised Interval Arithmetic is the most efficient method in affine form for our application. However, this method should be modified to suit long chains of computation. Our modification yields tighter interval bounds compared with other interval methods.

http://journal.austms.org.au/ojs/index.php/ANZIAMJ/article/view/3981 gives this article, © Austral. Mathematical Soc. 2012. Published January 16, 2012. ISSN 1446-8735. (Print two pages per sheet of paper.)

Contents

1 Introduction

Symbolic circuit analysis is a method for analysing electric circuits by deriving transfer functions in which some or all of the circuit parameters are retained as symbols. A fully expanded symbolic circuit expression in the Laplace domain can be represented as

$$G(s) = \frac{a_0 + a_1 \cdot s + a_2 \cdot s^2 + a_3 \cdot s^3 + a_4 \cdot s^4 + a_5 \cdot s^5 + \cdots + a_m \cdot s^m}{b_0 + b_1 \cdot s + b_2 \cdot s^2 + a_3 \cdot s^3 + a_4 \cdot s^4 + a_5 \cdot s^5 + \cdots + b_n \cdot s^n}, \quad (1)$$

where s is the complex frequency variable. The coefficients of s-powers (such as $\{a_0, a_1, \ldots a_m\}$ and $\{b_0, b_1, \ldots b_n\}$) are usually represented in sum-of-product form, consisting of symbolic circuit parameters. In modern semiconductor technologies, the circuit parameter values vary due to process variations, temperature and supply voltage fluctuations, and so on. The symbolic circuit expressions need to be evaluated or processed further in many applications, such as circuit sizing, optimization, and pole-zero analysis [1], under parameter variations. The use of interval methods helps to

evaluate symbolic expressions efficiently, taking parameter variations into account. This article is concerned with using interval methods—such as Affine Arithmetic (AA) [6], Revised Affine Arithmetic (RAA) [5], Extension to Affine Arithmetic (EAA) [4], and Generalized Interval Arithmetic (GIA) [2, 3]—to evaluate the expressions with interval values and compare their results to identify the most suitable method for this application.

Since the arithmetic operations in the coefficients of the s-powers in (1) are predominantly multiplication, the interval methods are compared based on their performance in multiplication, which is a non-affine operation. We show that the GIA yields optimal multiplication results, but its size in affine form grows with the number of multiplication operations. In order to restrict the growth of the affine form, we also present our modifications to the GIA multiplication rules suggested by Kolev [2, 3].

2 Intervals in affine form

An interval number in its affine form is

$$\tilde{x} = x_0 + x_1 \cdot \epsilon_1 + x_2 \cdot \epsilon_2 + x_3 \cdot \epsilon_3 + \cdots + x_n \cdot \epsilon_n \,, \tag{2}$$

where x_0 and x_i are known real numbers and ϵ_i is a symbolic variable whose value lies in the interval $[-1, +1]$. The value x_0 is the central value of $\tilde{x}$, whereas each product $x_i\epsilon_i$ quantifies an uncertainty which, when algebraically added to x_0, yields $\tilde{x}$.

Whenever two affine forms share common noise symbols, they are said to be dependent on each other. For example, if

$$\tilde{x} = 6 + 0.6\epsilon_1 + 0.1\epsilon_2 + 0.3\epsilon_4 \quad \text{and} \quad \tilde{y} = 7 - 0.4\epsilon_1 + 0.1\epsilon_3 - 0.5\epsilon_4 \,, \tag{3}$$

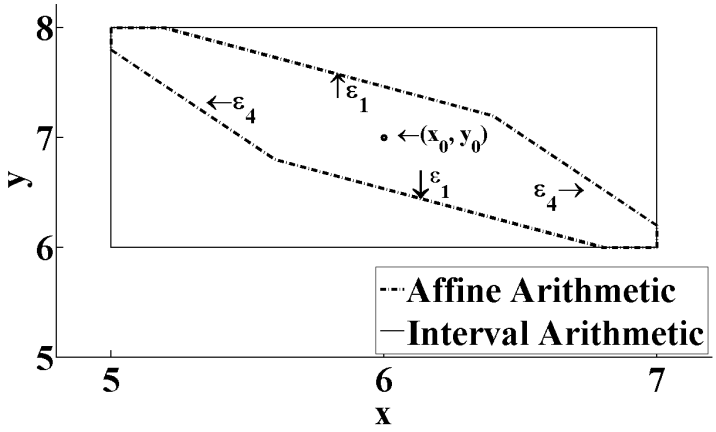
then the affine forms $\tilde{x}$ and $\tilde{y}$ share the noise symbols ϵ_1 and ϵ_4. The benefits of sharing noise variables is understood better by using a graphical

representation called a *joint range* plot, which is a convex polygon symmetric around a central point (x_0, y_0). The common noise symbols determine the size of the joint range polygon, as shown by the dotted line in Figure 1(a). Each noise symbol coefficient determines the length of two edges in the polygon as indicated in the figure. In this example, coefficients of ϵ_1 and ϵ_4 are significantly larger than the coefficients of other noise terms such as ϵ_2 and ϵ_3, resulting in a small size polygon. This information is completely lost when the two ranges are represented in interval form. The joint range of two intervals is a rectangle, as shown by the solid line in Figure 1(a), and has a larger area than the enclosed polygon. Therefore, by preserving the dependency between parameters, all those combinations of values in the operand intervals that do not yield a valid result are eliminated, resulting in tighter interval bounds.

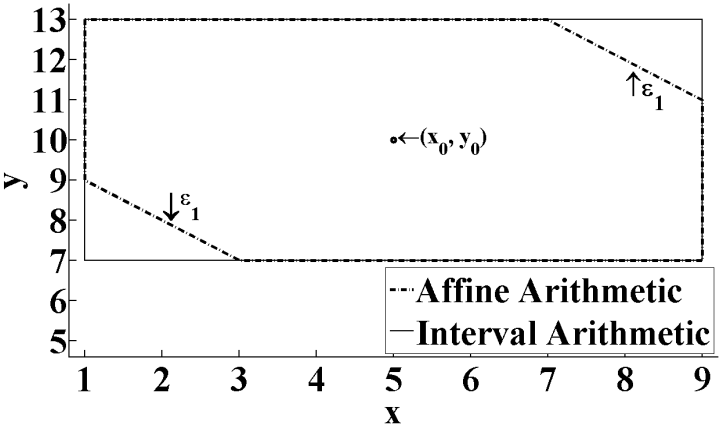
The arithmetic operations between two affine forms are broadly classified into affine and non-affine operations. The interval methods [2, 3, 4, 5, 6] vary in the way non-affine operations (for example, multiplication, division, exponentiation) are handled. We compare these methods to each other using a quantitative measure for overestimation in their results, called the *Relative Accuracy* (RA). The RA measure is defined as the ratio between the size of the actual range and that of the computed range [7]. The actual range is the smallest interval that contains only the possible values of the evaluated expression when the value of each variable in the expression is set to vary over its given range. If the width of the computed range is w_c and the width of the actual range is w_a, then the relative accuracy is

$$\text{RA} = w_a / w_c \,. \tag{4}$$

The actual range is not a known quantity, especially in a long computation chain. So the relative accuracy measure is computed based on Interval Arithmetic (IA) results [7]. The ratio of the width of the IA result to the width of the AA result represents the relative accuracy of AA when compared against IA. Therefore, RA should not be less than one for a good interval method using affine forms. The comparison is based on IA, because it is the fundamental interval method and assumes all the variables in the computation



(a) Affine numbers shown in (3)



(b) Affine numbers shown in (9)

Figure 1: Joint range plots of two affine numbers.

to be independent of each other. The methods using affine forms preserve the dependencies between variables during computations so as to achieve tighter bounds, albeit with additional computational effort.

3 Multiplication of intervals in affine form

The multiplication of two affine forms is classified as a non-affine operation, because the result cannot be expressed again in affine form. Therefore, the exact result is approximated and expressed in affine form,

$$\tilde{z}_{\text{exact}} = \tilde{x} \times \tilde{y} = \tilde{z}_{\text{approx}} + z_{\text{err}} \cdot \epsilon_{\text{err}} \,. \tag{5}$$

The $\tilde{z}_{\text{approx}}$ and z_{err} computed using AA [6] for the example in (3) are

$$\tilde{z}_{\text{approx}} = 42 + 1.8\epsilon_1 + 0.7\epsilon_2 + 0.6\epsilon_3 - 0.9\epsilon_4 \quad \text{and} \quad z_{\text{err}} = +1 \,, \tag{6}$$

$$\tilde{z}_{\text{exact}} = \tilde{x} \times \tilde{y} = 42 + 1.8\epsilon_1 + 0.7\epsilon_2 + 0.6\epsilon_3 - 0.9\epsilon_4 + \epsilon_{\text{err}} = [37, 47]. \tag{7}$$

The product of $\tilde{x}$ and $\tilde{y}$, when expressed in interval form, is $[30, 56]$. Therefore, the RA for this example is $(56 - 30)/(47 - 37) = 2.6$. The RA measure is consistent with the expectation from the joint range plot in Figure 1(a) that AA helps to achieve tighter bounds than IA.

The multiplication rules defined previously [2, 3, 4, 5, 6] attempt to compute the best estimate for $\tilde{z}_{\text{approx}}$ and z_{err} in (5). To gain insight into the performance of these rules, consider the following two examples

$$\tilde{x} = 5 + \epsilon_1 - 3\epsilon_3 \,, \quad \tilde{y} = 10 - \epsilon_1 + 2\epsilon_2 \,; \tag{8}$$

$$\tilde{x} = 4.5 + 0.5\epsilon_1 \,, \quad \tilde{y} = 10 + 2\epsilon_2 \,. \tag{9}$$

These two examples represent two different cases during the multiplication of affine forms. Example (8) represents multiplication between affine forms when they share common noise symbols. Figure 1(b) shows their joint range.

The products, obtained using AA [6], EAA, RAA [5] and GIA [2], are

$$\begin{aligned}
(\tilde{x} \times \tilde{y})_{AA} &= 50 + 5\epsilon_1 + 10\epsilon_2 - 30\epsilon_3 + 12\epsilon_{err} = [-7, 107], \\
(\tilde{x} \times \tilde{y})_{EAA} &= 50 + 5\epsilon_1 + 10\epsilon_2 - 30\epsilon_3 + 11\epsilon_{err1} + \epsilon_{err3} = [-7, 107], \\
(\tilde{x} \times \tilde{y})_{RAA} &= 49.5 + 5\epsilon_1 + 10\epsilon_2 - 30\epsilon_3 + 11.5\epsilon_{err} = [-7, 106], \\
(\tilde{x} \times \tilde{y})_{GIA} &= 54 + 6\epsilon_1 + 2\epsilon_2 - 21\epsilon_3 + 16\epsilon_4 = [9, 99].
\end{aligned} \tag{10}$$

(In the case of EAA, we use the multiplication rule for Affine Form 2, AF2, suggested by Messine and Touhami [4].) The product of $\tilde{x}$ and $\tilde{y}$ is $[7, 117]$ in interval form. As can be seen from (10), the AA, RAA and EAA (RA ≈ 0.97) results are overestimated, where as the GIA (RA $= 1.22$) result is consistent with the joint range plot in Figure 1(b). In addition, this example was chosen to illustrate that the overestimation can result in unacceptable bounds. The negative lower bound in (10) is unacceptable because the intervals of the two multiplicands contain only positive numbers. Therefore the interval of their product is expected to at least have positive bounds.

Example (9) illustrates multiplication of two affine forms when they do not share a noise symbol and are said to be independent. The products of these two affine forms obtained using AA, RAA, EAA and GIA [3] are

$$\begin{aligned}
(\tilde{x} \times \tilde{y})_{AA,RAA} &= 45 + 5\epsilon_1 + 9\epsilon_2 + \epsilon_{err} = [30, 60], \\
(\tilde{x} \times \tilde{y})_{EAA} &= 45 + 5\epsilon_1 + 9\epsilon_2 + \epsilon_{err1} + 0\epsilon_{err2} + 0\epsilon_{err3} = [30, 60], \\
(\tilde{x} \times \tilde{y})_{GIA} &= 46 + 4\epsilon_1 + 8\epsilon_2 + 2\epsilon_3 = [32, 60].
\end{aligned} \tag{11}$$

The product of $\tilde{x}$ and $\tilde{y}$ is $[32, 60]$ in interval form. Since IA assumes all the intervals in a computation to be independent of each other, the joint range plot for this example is a rectangle for both interval and affine form. In contrast, the AA, RAA and EAA (RA $= 0.93$) results are overestimated, whereas GIA (RA $= 1$) results are as expected. This is because the AA, RAA and EAA use the same conservative multiplication rules for both dependent and independent affine forms, whereas the GIA suggests separate multiplication rules for independent and dependent affine forms [2]. While the improvement

in RA from 0.93 to 1 appears small, it is important that RA is equal to or greater than one to avoid additional computation effort.

The GIA provides consistent results with no overestimation in both the scenarios discussed in this section and proves to be the method of choice for symbolic analysis applications. However, the difficulty in using GIA is that there is always an extra noise term (ϵ_{m+1}) added to the result after each multiplication. Thus, in symbolic circuit analysis the number of noise terms grows dramatically and hence the multiplication rules of Kolev [2, 3] must be modified to improve the feasibility of GIA for this application.

4 GIA in symbolic analysis application

Let $\tilde{x}$ and $\tilde{y}$ be the results of two separate multiplications in a computation chain obtained using the GIA multiplication rules shown in (12) [2, 3]. If m is the total number of noise variables in the entire computation, then

$$\begin{aligned} \tilde{x} &= \tilde{x}_{approx} + x_{m+1}\epsilon_{m+1} = x_0 + \sum_{i=1}^{n_1} x_i\epsilon_i + x_{m+1}\epsilon_{m+1}\,, \\ \tilde{y} &= \tilde{y}_{approx} + y_{m+2}\epsilon_{m+2} = y_0 + \sum_{j=C}^{n} y_j\epsilon_j + y_{m+2}\epsilon_{m+2}\,. \end{aligned} \tag{12}$$

Assuming $n > n_1$, ($C = n_1 + 1$) if $\tilde{x}$ and $\tilde{y}$ are independent, and $C = 1$ if $\tilde{x}$ and $\tilde{y}$ are dependent. Let $\tilde{z}$ be the result of multiplying $\tilde{x}$ and $\tilde{y}$ in a chain,

$$\tilde{z} = \tilde{x} * \tilde{y} = z_0 + \sum_{i=1}^{m} z_i\epsilon_i + z_{m+1}\epsilon_{m+1} + z_{m+2}\epsilon_{m+2} + z_{m+3}\epsilon_{m+3}\,. \tag{13}$$

In (12) and (13), the three new noise symbols, ϵ_{m+1}, ϵ_{m+2} and ϵ_{m+3}, are generated from three separate multiplication operations. The growth in the number of noise symbols with the number of multiplications poses significant

implementation issues. This problem does not occur in AA or RAA because the approximation errors in products are accumulated in a single noise term. The product $\tilde{z}$ in (16) needs to be generalized to the form shown in (13) in order to overcome this difficulty:

$$\tilde{z} = \tilde{x} * \tilde{y} = z_0 + z_1\epsilon_1 + z_2\epsilon_2 + \cdots + z_m\epsilon_m + B\,, \quad \text{where } B = [b_l, b_h]. \tag{14}$$

Here B is an interval that represents the accumulated approximation error in the product. The bounds of interval B are obtained from (13) by converting the terms with subscripts $m+1$, $m+2$ and $m+3$ to interval form:

$$b_l = -(|z_{m+1}| + |z_{m+2}| + |z_{m+3}|); \quad b_h = +(|z_{m+1}| + |z_{m+2}| + |z_{m+3}|). \tag{15}$$

In our modifications to the GIA multiplication rules, the accumulated approximation error terms are identified using a common symbol ϵ_{err}. The coefficient of ϵ_{err} is obtained from the radius of B in (15). Since the central value of B in (15) is zero, the product $\tilde{z}$ is computed as

$$\tilde{z} = z_0 + \sum_{i=1}^{m} z_i\epsilon_i + z_{err}\epsilon_{err}\,, \quad z_{err} = |z_{m+1}| + |z_{m+2}| + |z_{m+3}|\,. \tag{16}$$

The multiplication rules stated as theorems [2, 3] were modified accordingly to obtain the results in the form shown in (16).

In the case of independent intervals, the multiplication rule stated by Kolev [3, Theorem 3.1] is used. The affine approximation error in that theorem is modified to obtain the product as shown in (16).

Theorem 1 (Multiplication of independent affine forms)**.** *Let* $\tilde{x}$ *and* $\tilde{y}$ *be two independent affine forms as given in* (12)*. Let* $\mathbf{x}$ *and* $\mathbf{y}$ *be their respective interval forms. The product*

$$\tilde{z} = \tilde{z}_{approx} + z_{err}\epsilon_{err} = z_0 + z_1\epsilon_1 + z_2\epsilon_2 + \cdots + z_m\epsilon_m + z_{err}\epsilon_{err} \tag{17}$$

of $\tilde{x}$ *and* $\tilde{y}$ *reduces to* $\mathbf{z} = \mathbf{xy}$*, if the components of* $\tilde{z}$ *are determined as*

$$\begin{aligned} &z_0 = -\alpha\beta + \alpha x_0 + \beta y_0 + b_0\,, \quad z_{err} = b' + |\alpha x_{m+1}| + |\beta y_{m+2}|, \\ &z_i = \alpha x_i \text{ for } i = 1, \ldots, n_1\,; \quad z_i = \beta y_i \text{ for } i = n_1 + 1, \ldots, m\,. \end{aligned} \tag{18}$$

(Parameters α, β, b_0 and b' are real; their values were obtained by Kolev [3, Theorem 2.1].)

Proof: From Kolev [3, Theorem 2.1], the product in affine form is expressed in general as

$$\tilde{z} = -\alpha\beta + \alpha\tilde{x} + \beta\tilde{y} + B \quad \text{where } B = [b_l, b_h] = b_0 + b'\epsilon_{m+3}\,. \tag{19}$$

Substituting from (12) in (19) we get

$$\tilde{z} = z_0 + z_1\epsilon_1 + z_2\epsilon_2 + \cdots + z_m\epsilon_m + \alpha x_{m+1}\epsilon_{m+1} + \beta y_{m+2}\epsilon_{m+2} + b'\epsilon_{m+3} \tag{20}$$

where

$$z_0 = -\alpha\beta + \alpha x_0 + \beta y_0 + b_0\,, \tag{21}$$

$$z_i = \alpha x_i \text{ for } i = 1, \ldots, n_1\,, \quad \text{and} \quad z_i = \beta y_i \text{ for } i = n_1 + 1, \ldots, m\,. \tag{22}$$

The product in (20) is equivalent to the result obtained from Kolev [3, Theorem 3.1]. Since b_0 and b' are known from Kolev [3, Theorem 2.1], the terms with subscripts $m+1$, $m+2$ and $m+3$ in (20) are accumulated as done in (14)–(16) to obtain

$$z_{\text{err}} = b' + |\alpha x_{m+1}| + |\beta y_{m+2}|, \tag{23}$$

which completes the proof. ♠

In the case of dependent intervals the optimal multiplication rule of Kolev [2] is used. The first step in this rule is to compute (as described in section 2 [2]) the narrowest possible interval $\mathbf{z}^*$ $([z_l^*, z_h^*])$, to which the product $\tilde{z}$ in (13) reduces. The coefficients x_{m+1} and y_{m+2} are treated independently during this computation. The product in affine form is then computed using Kolev's result [3, Theorem 2.1]. This theorem needs to be modified to obtain $\tilde{z}$ as shown in (16). Our modifications are presented as follows.

Theorem 2 (Multiplication of dependent affine forms)**.** *Let* $\tilde{x}$ *and* $\tilde{y}$ *be two dependent affine forms as given in* (12)*. Let* $\mathbf{x}$ *and* $\mathbf{y}$ *be their respective interval forms satisfying property (1.5) [2]. Further, assume that the end points of* $\mathbf{z}^*$ *have been computed as described by Kolev [3, section 2] with* x_{err} *and* y_{err} *treated as independent noise coefficients. The product*

$$\tilde{z} = \tilde{z}_{approx} + z_{err}\epsilon_{err} = z_0 + z_1\epsilon_1 + z_2\epsilon_2 + \cdots + z_m\epsilon_m + z_{err}\epsilon_{err} \tag{24}$$

of $\tilde{x}$ *and* $\tilde{y}$ *reduces to* $\mathbf{z}^*$*, if the components of* $\tilde{z}$ *are determined as*

$$\begin{aligned} &z_0 = \frac{z_h^* + z_l^*}{2}\,, \quad z_{err} = \frac{z_h^* - z_l^*}{2} - R \quad \textit{where} \quad R = \sum_{i=1}^{n} |z_i|, \\ &z_i = y_l x_i \textit{ for } i \in M_x \textit{ and } i \neq err \\ &z_i = x_l y_i \textit{ for } i \in M_y \textit{ and } i \neq err; \quad z_i = y_l x_i + x_l y_i \textit{ for } i \in M_c\,. \end{aligned} \tag{25}$$

(Each of the sets M_x *and* M_y *contain the subscripts of independent noise symbols from* $\tilde{x}$ *and* $\tilde{y}$ *respectively along with this subscript, (err), indicating that the approximation errors are treated independently similar to the original* GIA *rule. The value* R*, which is the radius of* $\tilde{z}_{approx}$*, does not contain the magnitudes corresponding to* ϵ_{err} *terms.)*

Proof: From Kolev [3, equation (2.1)], the product $\tilde{z}$ is in general

$$\tilde{z} = -x_l y_l + y_l\tilde{x} + x_l\tilde{y} + B\,, \quad \text{where } B = [b_l, b_h]. \tag{26}$$

Substituting (12) in (26) we get

$$\tilde{z} = z_0' + z_1\epsilon_1 + z_2\epsilon_2 + \cdots + z_m\epsilon_m + y_l x_{m+1}\epsilon_{m+1} + x_l y_{m+2}\epsilon_{m+2} + B \tag{27}$$

where

$$z_0' = -x_l y_l + y_l x_0 + x_l y_0 \quad \text{and} \quad z_i = y_l x_i + x_l y_i\,. \tag{28}$$

The objective is to find the interval B, such that the product $\tilde{z}$ reduces to an optimal interval $\mathbf{z}^*$, $[z_l^*, z_h^*]$. By equating the centre and radius of $\tilde{z}$ and $\mathbf{z}^*$,

$$\begin{aligned} &(z_h^* + z_l^*)/2 = z_0' + (b_h + b_l)/2 \quad \text{and} \\ &(z_h^* - z_l^*)/2 = R + |y_l x_{m+1}| + |x_l y_{m+1}| + (b_h - b_l)/2\,, \end{aligned} \tag{29}$$

where R is the radius of $\tilde{z}_{\text{approx}}$. Since $\mathbf{z}^*$ is a known quantity, obtained as described by Kolev [3, section 2], the centre z_0 and the accumulated error z_{err} is derived from (29) as

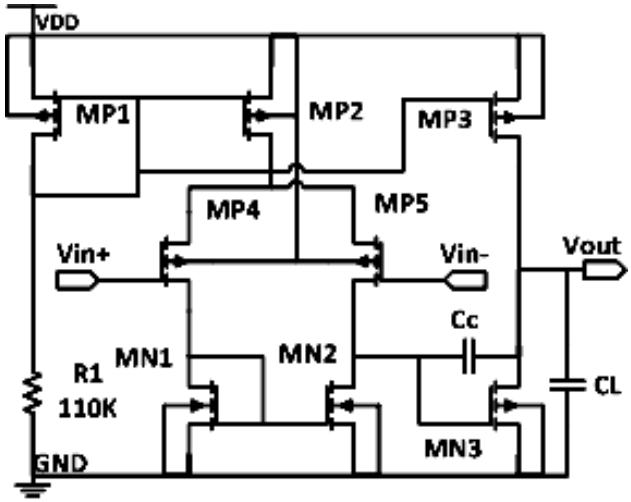
$$z_0 = (z_{\text{h}}^* + z_{\text{l}}^*)/2 \quad \text{and} \quad z_{\text{err}} = (z_{\text{h}}^* - z_{\text{l}}^*)/2 - \text{R}\,. \tag{30}$$

Therefore (27) is now rewritten as in (24) and (25) using (28) and (30), which completes the proof. ♠

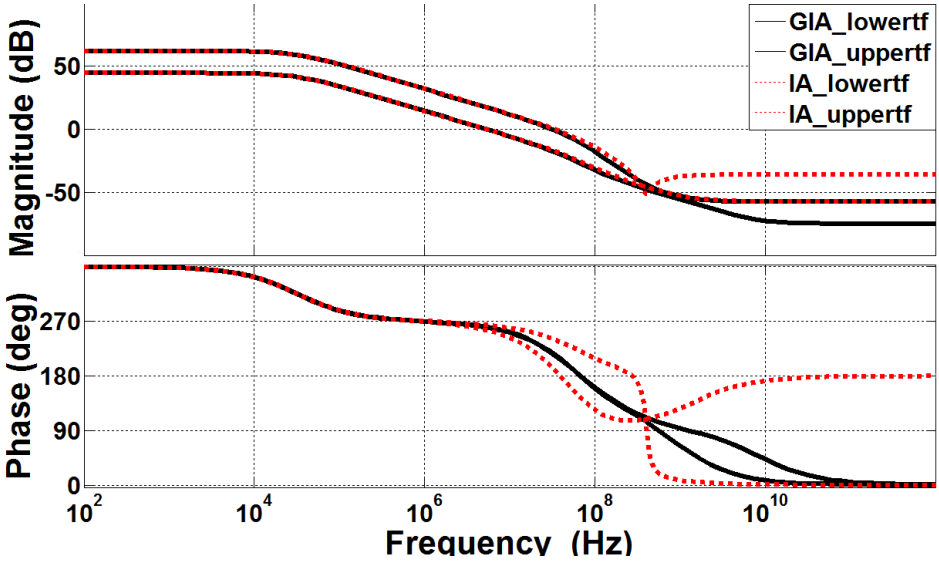
Thus the errors in a chain of computations are now accumulated under a single error coefficient z_{err} for both dependent and independent cases.

5 Experimental results and discussion

The interval methods (AA, RAA, EAA and GIA) were implemented as a MATLAB toolbox with all basic arithmetic operations. The IA results were computed using INTLAB [10]. The GIA toolbox was implemented along with the modifications presented in section 4. The symbolic expressions for different circuit examples presented by Thanigaivelan et al. [8] were used for our experiments. The experimental results for an Operational Transconductance Amplifier (OTA) circuit are presented and discussed. Figure 2 shows the circuit schematic along with frequency response plots for the voltage gain. The specifications for the elements in the circuit (transistors, resistor and capacitors) were obtained from Kaminska et al. [9] and the nominal values of its transistor model parameters (conductance, transconductance and capacitance) were obtained from SPICE DC operating point simulation results. The conductance, transconductance and capacitance parameters were then varied ±10% of their nominal values. The interval values of parameters in the expression were then converted to affine form. The symbolic expression in sum of product form for the voltage gain of the OTA was obtained using a determinant technique. Table 1 lists the numerical coefficients of the s-powers in the lower and upper



(a) OTA Schematic



(b) Voltage Gain Bode Plot

Figure 2: OTA circuit schematic and voltage gain frequency response plot.

bound transfer functions for GIA and IA. Table 2 compares RA measures for each coefficient in the circuit expression for different interval methods.

Figure 2(b) compares the GIA with IA using magnitude and phase plots. The GIA results are plotted using solid lines and the IA results are plotted using dotted lines as indicated by the legends in the plot. As seen from this figure, the IA results are overestimated by a large margin from medium to high frequencies, starting from 1 MHz. The plots for the GIA results are always enclosed by the plots for the IA results upto 1 MHz. The upper bound curve for IA in the magnitude plot is misleading from 100 MHz because the bounds for coefficients of higher powers of s in the numerator (especially a_3, a_4 and a_5 shown in Table 1) are wider than necessary. The IA results being wider than necessary is because the dependencies are not preserved during computations. The affine small signal model used for the transistors extracts the dependencies between model parameters [8] and the GIA uses special rules for multiplying affine numbers that are well correlated [2]. The dependency between parameters is high in the case of capacitance elements (as they share two or more parameters) and low or absent altogether in the case of conductance elements (as they share the parameters depending on circuit connections) [8]. The coefficients a_3, a_4 and a_5 are dominated by capacitance elements resulting in overestimation.

The IA and GIA results in Table 1 show the upper and lower bounds of s-power coefficients in (1) for the OTA example. The coefficients of lower powers of s are numerically the same or nearly the same compared with coefficients of higher powers because of the dependencies among parameters at higher powers of s. The bounds of a_3, a_4 and a_5 in the numerator of IA result are not only wider than the GIA result, but are of opposite sign. This is of particular concern in applications where rational expressions in s are used in another loop such as optimization and automated circuit sizing.

The interval methods AA, RAA and EAA are compared against each other using the RA measures as shown in Table 2. Although their average RA scores are high, the overestimation in individual coefficients being worse than IA is a

Table 1: Coefficients of s-powers in the transfer function for OTA gain: L denotes Lower Bound, and U denotes Upper Bound.

		a_5	a_4	a_3	a_2	a_1	a_0
GIA	L	9.4E−64	−5.3E−53	−1.6E−44	2.6E−35	1.7E−26	1.1E−18
	U	2.5E−63	−1.9E−53	−7.6E−46	8.2E−35	4.7E−26	2.9E−18
IA	L	−8.7E−63	−8.7E−53	−9.4E−44	−6.7E−36	1.5E−26	1.1E−18
	U	1.2E−62	1.3E−53	7.6E−44	1.1E−34	4.9E−26	2.9E−18
		b_5	b_4	b_3	b_2	b_1	b_0
GIA	L	1.9E−60	2.7E−51	1.3E−42	2.4E−34	1.1E−26	2.3E−21
	U	5.3E−60	7.3E−51	3.6E−42	6.7E−34	3.2E−26	6.5E−21
IA	L	7.8E−61	1.5E−51	9.9E−43	2.2E−34	1.1E−26	2.3E−21
	U	6.4E−60	8.5E−51	3.9E−42	6.9E−34	3.2E−26	6.5E−21

Table 2: A comparison of interval methods for the OTA circuit example: N denotes the Numerator, and D denotes the Denominator.

		s^5	s^4	s^3	s^2	s^1	s^0	Average RA
AA	N	3.6	1.9	3.2	1.5	1.0	0.9	1.52
	D	1.3	1.2	1.0	0.9	0.9	0.9	
EAA	N	3.6	1.9	3.2	1.5	1.0	0.9	1.52
	D	1.3	1.2	1.0	0.9	0.9	0.9	
RAA	N	10.9	2.5	7.8	1.8	1.0	0.9	2.61
	D	1.4	1.3	1.1	0.9	0.9	0.9	
GIA	N	12.8	3.0	10.7	2.2	1.1	1.0	3.20
	D	1.7	1.5	1.3	1.1	1.0	1.0	

concern as it indicates that computational effort will be below optimal when using these methods. The high RA scores for each coefficient and high average accuracy for the GIA indicates that it is the best method for circuit analysis applications. The GIA scores are high mainly because non-affine operations are handled separately for dependent and independent affine forms.

6 Conclusion

The use of interval arithmetic leads to overestimation and therefore can be regarded as unsuitable in symbolic circuit analysis applications. A comparison between different interval methods in affine form presented in this article shows that the Generalized Interval Arithmetic (GIA) of Kolev [2] is suitable for symbolic circuit analysis applications. This is because the GIA defines separate optimal rules for multiplying dependent and independent affine numbers. However, the GIA reported has to be modified because the number of noise symbols grows dramatically with the number of multiplications. Our experiments show that our modifications still yield optimal results, while restricting the growth in the number of noise symbols and thus improving computational efficiency.

References

[1] Francisco Fernandez et al., Symbolic Analysis Techniques Applications to Analog Design Automation *IEEE Press*, 1998. C1085

[2] L. Kolev, Optimal Multiplication of G-intervals *Reliable Computing*, **13**, pp.399–408, 2007. C1086, C1087, C1089, C1090, C1091, C1092, C1093, C1094, C1097, C1099

[3] L. Kolev, New Formulae for Multiplication of Intervals, *Reliable Computing*, **12**, pp.281–292, 2006. C1086, C1087, C1089, C1090, C1091, C1092, C1093, C1094, C1095

[4] F. Messine and A. Touhami, A General Reliable Quadratic Form: An Extension of Affine Arithmetic, *Reliable Computing*, **12**, pp.171–192, 2006. C1086, C1087, C1089, C1090

[5] Xuan-Ha Vu, Rigourous solution techniques for numerical constraints satisfaction problems, PhD thesis no. 3155 (2005), Swiss Federal Institute of Technology, Lausanne, Switzerland 2005. C1086, C1087, C1089, C1090

[6] G. Manson, Calculating frequency response functions for uncertain systems using complex affine analysis, *J. Sound and Vibration*, **288**, pp.487–521, 2005. C1086, C1087, C1089, C1090

[7] L. H. d. Figueiredo and J. Stolfi, Self-Validated Numerical Methods and Applications, *IMPA, Rio de Janeiro, Brazil*, July 1997. C1087

[8] B. Thanigaivelan et al., A modified MOSFET small-signal model based on Affine Arithmetic concepts, *Proceedings of Asia Pacific Conference on Postgraduate Research In Microelectronics and Electronics*, Shangai, China, 2009. C1095, C1097

[9] Bozena Kaminska et al. Analog and Mixed-Signal Benchmark Circuits-First Release *Proceedings of ITC*, pp.183–190, 1997 C1095

[10] S. M. Rump. (July 2010). Interval Laboratory. http://www.ti3.tu-harburg.de/rump/intlab/ C1095

Author addresses

1. **B. Thanigaivelan**, School of Information Technology and Electrical Engineering, The University of Queensland, Queensland, AUSTRALIA.

mailto:velan@itee.uq.edu.au

2. **T. J. Hamilton**, School of Electrical Engineering and Telecommunications, The University of New South Wales, New South Wales, Australia.

3. **A. Postula**, School of Information Technology and Electrical Engineering, The University of Queensland, Queensland, Australia.