Fast Interval-Valued Statistical Modeling of Interconnect and Effective Capacitance

James D. Ma and Rob A. Rutenbar, *Fellow, IEEE*

***Abstract*—Correlated interval representations of range uncertainty offer an attractive solution to approximating computations on statistical quantities. The key idea is to use finite intervals to approximate the essential mass of a probability density function (pdf) as it moves through numerical operators; the resulting compact interval-valued solution can be easily interpreted as a statistical distribution and efficiently sampled. This paper first describes improved interval-valued algorithms for asymptotic wave evaluation (AWE)/passive reduced-order interconnect macromodeling algorithm (PRIMA) model order reduction for tree-structured interconnect circuits with correlated resistance, inductance, and capacitance (*RLC*) parameter variations. By moving to a much faster interval-valued linear solver based on path-tracing ideas, and making more optimal tradeoffs between interval- and scalar-valued computations, the delay statistics roughly 10× faster than classical Monte Carlo (MC) simulation, with accuracy to within 5% can be extracted. This improved interval analysis strategy is further applied in order to build statistical effective capacitance (C_{eff}) models for variational interconnect, and show how to extract statistics of C_{eff} over 100× faster than classical MC simulation, with errors less than 4%.**

***Index Terms*—Affine arithmetic, effective capacitance, interconnect, modeling, statistical analysis.**

I. Introduction

At 90-nm technology and below, it is no longer realistic to regard device or interconnect parameters as deterministic. The increasingly atomic scale of manufacturing means that all important design parameters are statistically distributed with complex correlations. The new problem is how to efficiently analyze critical devices, interconnects, and layouts in this new regime. We see the first practical solutions in recent progress on statistical static timing tools [1]–[5]. By representing all circuit delays and arrival times as correlated Gaussian distributions, it is possible to perform delay analysis by "pushing" these statistics through the machinery of static timing. The problem is that static timing requires only a very limited portfolio of basic operations: addition and maximum/minimum of two distributions. We cannot apply the statistical ideas successful in this application to other, arbitrary problems which require richer palettes of computations; e.g., if our interest is in interconnect analysis, these ideas do not tell how to invert a matrix of Gaussians, or find its statistically distributed eigenvalues.

Statistical modeling of resistance, inductance, and capacitance (*RLC*) interconnects in particular, and of circuits in general, is a vigorous new research area. The most common approach is Monte Carlo (MC) sampling/simulation, which is general, flexible, straightforward, and widely trusted. It provides the maximum accuracy, but at an unattractive cost for large designs and large numbers of random parameters. Other approaches are specifically developed and tailored for statistical linear-interconnect analysis, and can be further categorized as through: 1) model order reduction [6]–[9] and 2) performance parameter (e.g., delay) extraction [10]–[12]. Lee *et al.* [6] calculated the sensitivities of poles and zeros found by asymptotic wave evaluation (AWE) [13] with respect to variational circuit elements. The analysis is tightly coupled with the original AWE algorithm, and does not handle explicit correlations among the random process parameters, which may lead to overpessimism in estimating delay impact. Liu *et al.* [7] showed how the pole analysis via congruence transformations (PACT) [14] and passive reduced-order interconnect macromodeling algorithm (PRIMA) [15] interconnect model reduction methods can be rendered in variational forms via modest sampling. The goal of the multiparameter moment-matching algorithm proposed in [8] was to synthesize compact "parameterized" model that predicts interconnect effects from wire structures with geometry one does not know yet. Wang *et al.* [9] built an explicit stochastic circuit model and solved it via a Galerkin technique in an appropriate Hilbert space. It is unclear, however, how complex correlations are handled in this framework, for circuits larger than a few tens of nodes. In [10], the random parameters are represented in a linear form and are used to derive low-order analytical formulas for variational delay metrics. Li *et al.* [11] proposed an asymptotic probability extraction methodology for nonnormal distributions of circuit performance (e.g., delay) via high-order moment matching. Finally, Harkness *et al.* [12] used classical intervals on the real line to approximate each uncertain circuit parameter as a range. The idea is appealing, but has yet to be shown to be practical. Classical interval calculations are notoriously pessimistic, and attempts to date have been restricted to simplistic Elmore-style models that have a small closed-form.

Our interest is to find more general and flexible techniques for representing correlated statistics "inside" a wide range of algorithms. We believe that recent advances in correlated interval representations of range uncertainty [16] offer an attractive solution that can offer not only a workable level of numerical accuracy, but also more importantly, the flexibility we seek to

Manuscript received June 23, 2005; revised September 20, 2005. This paper was supported in part by C2S2, the Microelectronics Advanced Research Corporation (MARCO) Focus Center for Circuit and System Solutions, under MARCO contract 2003-CT-888. This paper was recommended by Associate Editor L. Scheffer.

The authors are with the Department of Electrical and Computer Engineering, Carnegie Mellon University, Pittsburgh, PA 15213 USA (e-mail: jdma@ece.cmu.edu; rutenbar@ece.cmu.edu).

Digital Object Identifier 10.1109/TCAD.2006.870067

allow us to deploy these techniques in a wide range of applications. Our idea, first proposed in [17] and [18], is to approximate each input range uncertainty as a finite interval, and to use an appropriate algebra of interval arithmetic to replace each conventional floating-point calculation (hereafter referred to as scalar calculation, to distinguish from interval-valued calculations) in a given numerical algorithm. The essential assumption is that the mechanics of range calculation for each finite interval are a reasonable—though clearly imperfect—surrogate of how statistics actually move through the same computations.

In [17] and [18], we showed how the complete numerical algorithms for AWE [13] and PRIMA [15] linear model order reduction could be recast in this interval-valued form, and used to predict variational interconnect delay with errors between 5%–10% for correlated *RLC* parameter variations up to 35%. In this new paper, we extend on our ideas from [19] and show first how to improve upon our earlier results to build a new interval-valued model order reduction strategy (for statistical delay estimation) that is: 1) 10× faster than our tools from [17] and [18] for the common case of tree-structured interconnect and 2) accurate to within 5% of classical MC simulation-based analysis.

The first key idea is to understand that intervals, like floating-point representations, inevitably accrue more errors as they push through deep chains of calculation. Thus, we replace the interval-valued modified-nodal analysis (MNA) formulation and lower-upper-triangular-matrix (LU) decomposition of [17] and [18] with interval-valued versions of the standard path-tracing algorithms [20] for (orthonormalized) moment computation, which significantly reduces both the number of operations and the overall estimation error.

The second key idea is to recognize that it is not essential to employ intervals in every step of our algorithms. In [17] and [18], we used intervals pervasively—from *RLC* circuit inputs to pole/residue outputs—to prove the feasibility of our approach. In this paper, we switch from interval-based computation to scalar-valued MC sampling right after the original variational system is reduced to a low-order model, but before the reduced system requires further nonlinear solves (e.g., root finding and/or eigendecomposition) for poles and residues. The right tradeoff between interval- and scalar-valued computations is a new degree of freedom in these algorithms; we show how to exploit this to make the best tradeoff between accuracy and efficiency.

Furthermore, as an obvious next step of this improved interval-valued interconnect model reduction strategy, we build interval-valued statistical "effective capacitance" (C_{eff}) models for variational interconnect. The standard scalar-valued C_{eff} model [21] has been crucial for both static gate and interconnect timing analysis. We demonstrate how to retarget two different scalar-valued C_{eff} modeling algorithms [21], [22] to extract statistics of C_{eff} with over 100× speedup and less than 4% errors compared to classical MC simulation.

This paper is organized as follows. Section II reviews the interval model we use to represent correlated statistical variations. Section III describes our statistical AWE and PRIMA algorithms based on interval-valued path tracing. Section IV applies the improved interval-valued interconnect modeling strategy to build statistical C_{eff} models. Section V shows experimental results comparing our interval methods against conventional MC analysis using a fast linear-interconnect simulator (rapid interconnect circuit evaluation (RICE) [20], [23]) in the loop. Concluding remarks are offered in Section VI.

II. Background

A. Basics of Affine Intervals and Arithmetic

For clarity, we review here the basic derivation of the affine interval ideas from [17] and [18]. This is the starting point for our statistical interpretation of the affine interval idea. Classical-interval arithmetic was invented by Moore [24] to solve range estimation problems in the presence of uncertainties. The uncertainty of a variable x is represented by an interval $\overline{x} = [\overline{x}.lo, \overline{x}.hi]$. The true value of x is only known to satisfy $\overline{x}.lo \leqslant x \leqslant \overline{x}.hi$. Basic arithmetic is redefined to yield interval solutions from interval operands. However, due to the lack of information about operand dependencies, a serious problem is overestimation. For instance, suppose $\overline{x} = [-1, 1]$, $\overline{y} = [-1, 1]$, and that x and y have the relationship as $y = -x$. If we compute $\overline{z} = \overline{x} + \overline{y}$, we can only obtain $\overline{z} = [-2, 2]$, while in reality $z = x + y = 0$. This is a classical example of range explosion, when interval computations are pushed through long chains of calculations.

This situation was not improved until a novel range arithmetic model—affine arithmetic—was proposed by Stolfi and de Figueiredo [16] to preserve first-order correlations among uncertainties. In this model, the uncertainty of a variable x is represented as a range in an affine form $\widehat{x}$, given by

$$\widehat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2 + \cdots + x_n\varepsilon_n \qquad (-1 \leqslant \varepsilon_i \leqslant 1, \qquad i = 1, 2, \ldots, n). \quad (1)$$

Each uncertainty symbol ε_i stands for an independent component of the total uncertainties of the variable x; the corresponding coefficient x_i gives the magnitude of that component. In contrast to traditional interval methods defined by their endpoints, affine intervals are defined by their central point x_0, and a set of symmetric excursions about this point. For affine addition, if $\widehat{x} = x_0 + x_1\varepsilon_1 + x_2\varepsilon_2$ and $\widehat{y} = y_0 + y_1\varepsilon_1 + y_3\varepsilon_3$

$$\widehat{z} = \widehat{x} + \widehat{y} = (x_0 + y_0) + (x_1 + y_1)\varepsilon_1 + x_2\varepsilon_2 + y_3\varepsilon_3 \quad (2)$$

which is again in an affine form. One symbol ε_i may contribute to the uncertainties of two or more variables, indicating dependence among them. When these variables are combined, uncertainty terms may actually be canceled.

Returning to the previous example, suppose that x and y have affine forms $\widehat{x} = 0 + 1\varepsilon$ and $\widehat{y} = -\widehat{x} = 0 - 1\varepsilon$. In this case, the affine form of the sum $\widehat{z} = \widehat{x} + \widehat{y} = 0$ perfectly coincides with the actual range of the variable z. This is the unique feature of the model, and its central advantage over earlier interval methods.

Multiplication of affine intervals $\widehat{x}$ and $\widehat{y}$ gives the product

$$\begin{aligned}\widehat{z} &= \widehat{x}\widehat{y}\\ &= \left(x_0+\sum_{i=1}^{n} x_i\varepsilon_i\right)\left(y_0+\sum_{i=1}^{n} y_i\varepsilon_i\right)\\ &= x_0y_0+\sum_{i=1}^{n}(y_0x_i+x_0y_i)\varepsilon_i+\left(\sum_{i=1}^{n} x_i\varepsilon_i\right)\left(\sum_{i=1}^{n} y_i\varepsilon_i\right) \qquad (3)\end{aligned}$$

which is not in affine form. It can still be approximated in an affine form

$$\begin{aligned}\widehat{z} &\approx x_0y_0+\sum_{i=1}^{n}(y_0x_i+x_0y_i)\varepsilon_i\\ &\quad+\left[R\left(\sum_{i=1}^{n} x_i\varepsilon_i\right)\right]\left[R\left(\sum_{i=1}^{n} y_i\varepsilon_i\right)\right]\zeta\\ &= z_0+\sum_{i=1}^{n} z_i\varepsilon_i+R(\widehat{x}\widehat{y})\zeta \qquad (4)\end{aligned}$$

where ζ is a new uncertainty symbol that is also in [−1, 1], but distinct from all the other uncertainty symbols $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$ that have already appeared in the same computation. R is the "radius operator," defined as $R(\sum_{i=1}^{n} x_i\varepsilon_i) = \sum_{i=1}^{n} |x_i|$, which computes the upper error bound of an affine variable. Note that the substitution of $R(\widehat{x}\widehat{y})\zeta$ for the quadratic terms implies a loss of information because ζ is assumed to be independent from $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$, while it is in fact a function of them. Equation (4) is a conservative approximation in the sense that it always bounds the original true affine product given by (3).

The twin goals of the fundamental work in [16] were: 1) to create a practical set of interval arithmetic operators with which to replace their scalar counterparts in standard numerical codes and 2) to guarantee conservative bounding for the range uncertainty represented by each affine interval computation. In our paper, we use 1), but we need to abandon 2) to move from intervals to statistics.

B. From Intervals to Statistics

As we introduced in [17] and [18], the essential idea is to use intervals to model the range uncertainty that results from each atomic arithmetic calculation. We cannot afford the expense necessary to calculate the exact statistics resulting from arbitrary arithmetic operations, but the same computations on intervals can be done quite efficiently as long as we have some mapping from the finite intervals we use during computations, to the statistics we seek at the end of the process.

In their traditional development, the uncertainty symbols $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$ are independent and arbitrarily distributed within [−1, 1], thus creating a set of (random) excursions about each interval's midpoint. Following [17] and [18], we reinterpret each ε_i as a normal random variable with $\mu = 0$ and $\sigma = 1$. That is, an algorithm computes with interval-valued operators and delivers its outputs in the form $\widehat{z} = z_0 + \sum_{i=1}^{n} z_i\varepsilon_i$. We interpret this as a symbolic form combining a set of independent zero-mean unit-variance normal random variables. We assume that the careful mechanics of bound estimation, computed for each affine interval operation, can serve as a workable stand in for how the statistics would actually move through the same computations, i.e., how the essential bulk of the distribution moves through these computations. Note that, while our calculations are done on conservative finite ranges, we evaluate the final interval-valued formula by sampling it with values that may actually extend outside each of these ranges. In other words, we have abandoned the idea that any range uncertainty has finite support, but adopt a heuristic interpretation that the uncertainty terms each model the important contributing mass of an infinite, continuous distribution, between $\pm\sigma$. In general, we must also note that this is not the only statistical interpretation of the affine form that one can choose. In this paper, we: 1) use a simple statistical interpretation that emphasizes speed over accuracy and 2) explore carefully the boundary for where to stop interval calculation, and proceed forward with standard MC sampling of a compact interval-valued intermediate result. It is also possible to alter the fundamental calculation and interpretation of the affine uncertainty terms to produce more accurate interval results, with a more elaborate model of the relationship between interval ranges and the statistics of the ε_i uncertainty terms, at the cost of additional CPU time. This is not the focus of this paper; see [25] for details.

C. New Affine Arithmetic Formulation

We do need to mention one important new way in which the work described here diverges from the classical bounding formulation for the affine model, and from the interval-valued methods implemented in [17] and [18]. Refer again to the multiplication example [see (3) and (4)]. Observe that any operation other than addition/subtraction will create a new "linearized" uncertainty term ζ to account for the higher order combinations of uncertainty symbols we cannot represent in affine form. Large problems need to create and manage a large number of these linearized uncertainty terms, which is inefficient. Can we do this faster?

The answer is "yes," and to do this we take some inspiration from statistical static timing methods [3], [4] which "match variance" across a fixed set of normal random variables as delay is computed gate by gate. Thus, our idea is not to create the standard lumped linearized uncertainty ζ, but instead, to distribute its "effect" across the existing uncertainty symbols. Returning to the multiplication example of (4), we still compute the coefficient $R(\widehat{x}\widehat{y})$ of ζ, but distribute its effect proportionally among the original $\varepsilon_1, \varepsilon_2, \ldots, \varepsilon_n$. Specifically

$$\widehat{z} \approx z_0 + \sum_{i=1}^{n} z_i\left[1+\frac{z_i}{\sum_{i=1}^{n}|z_i|}R(\widehat{x}\widehat{y})\right]\varepsilon_i \qquad (5)$$

which is also a conservative approximation that always bounds the true range of $\widehat{z}$. Moreover, this new formulation also offers the advantage that at the end of any computation, the final affine form can be immediately interpreted as a nominal value

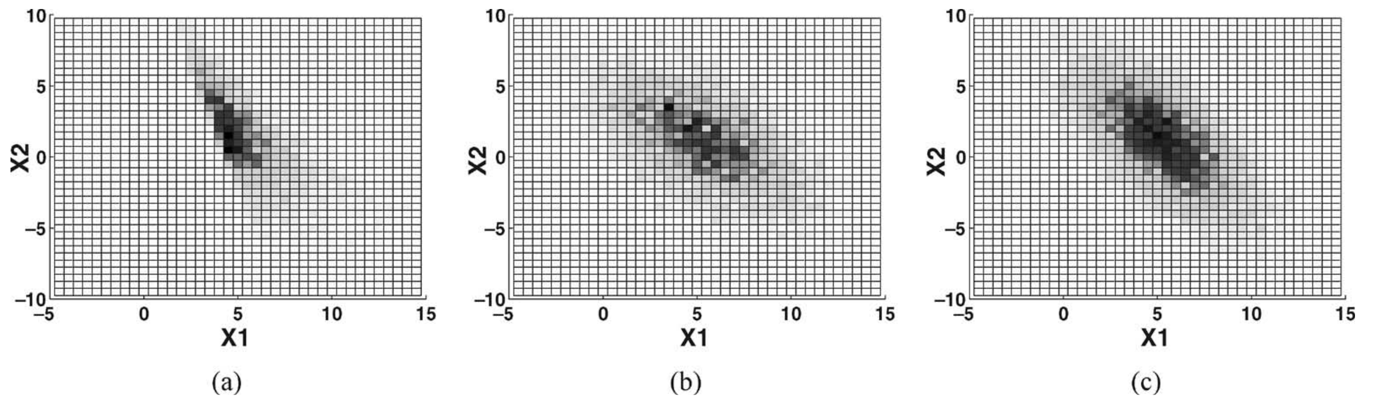


Fig. 1. Comparison of the joint distributions of (x_1, x_2) solution points in false color shading among (a) MC simulation, (b) affine interval linear solution as in this paper (average error: 12.1%), and (c) affine interval linear solution as in [18] (average error: 10.8%).

plus a set of first-order "sensitivities" to the original, principal uncertainty sources with which we have modeled the problem.

If $R(\widehat{x}\widehat{y})$ is much less than $\sum_{i=1}^{n}|z_i|$, we can simply truncate the quadratic terms in (3) and obtain the most efficient approximation of affine multiplication as below

$$\widehat{z} \approx z_0 + \sum_{i=1}^{n} z_i \varepsilon_i. \tag{6}$$

In reality, a threshold ratio of $R(\widehat{x}\widehat{y})$ over $\sum_{i=1}^{n}|z_i|$ can be set to determine whether to use (6). Note that this approximation is no longer conservative and does not always bound the true range of $\widehat{z}$. However, since our own statistical interpretation does not require perfect conservatism on the bounds, and abandons the finite support assumption altogether, this turns out to be an attractive tradeoff. We have implemented similar simplifications for division and for the other useful nonlinear operations we need to perform on our intervals.

D. Concrete Example of Interval-Valued Computation

It is useful at this point to show a concrete example of interval-valued computation based on our new model. We revisit the simple example of solving three interval-valued linear equations in [18], but unlike that earlier example, we no longer introduce any new uncertainty symbol as a result of each basic nonaffine arithmetic operation. The matrix linear-solve problem and its solution via interval-valued backward-substitutions appear in (7)

$$\begin{bmatrix} 3+\varepsilon_1+\varepsilon_2 & 0 & 4-\varepsilon_1+2\varepsilon_3 \\ 0 & 2+\varepsilon_1 & 3+\varepsilon_1-\varepsilon_3 \\ 0 & 0 & -4+\varepsilon_1-\varepsilon_3 \end{bmatrix} \begin{bmatrix} \widehat{x}_1 \\ \widehat{x}_2 \\ \widehat{x}_3 \end{bmatrix} = \begin{bmatrix} -1 \\ -10 \\ 16 \end{bmatrix}$$

$$\begin{bmatrix} \widehat{x}_1 \\ \widehat{x}_2 \\ \widehat{x}_3 \end{bmatrix} = \begin{bmatrix} 5-2.4\varepsilon_1-2.4\varepsilon_2+1.9\varepsilon_3 \\ 1+3.7\varepsilon_1+0.8\varepsilon_3 \\ -4-\varepsilon_1+\varepsilon_3 \end{bmatrix}. \tag{7}$$

We compare the joint distributions of (x_1, x_2) between MC simulation and the affine interval-valued linear solve. For MC simulation, we randomly sample over the uncertainty symbols in the affine interval-valued coefficient matrix in (7), assuming normal distributions with zero mean and unit variance for the symbols. For each sample, we solve the resultant three scalar-valued linear equations. Fig. 1(a) shows the real joint distribution of solution points on the $x_1 - x_2$ plane using false color shading, and 1000 samples are used. More (x_1, x_2) solution points are distributed in darker regions. For the affine interval-valued linear solve, we randomly, normally sample 1000 times over the uncertainty symbols ε_1 to ε_3 of the affine interval-valued solutions in (7) directly, and plot the joint distribution of (x_1, x_2) in Fig. 1(b). To compare, Fig. 1(c) shows the joint distribution of (x_1, x_2) obtained in [18], i.e., using the affine arithmetic that introduces (and later samples over) new uncertainty symbols. To quantify the accuracy of affine interval linear solve, we first define the "distribution error" for a grid as the difference between the number of solution points in the grid obtained by affine interval linear solve and that obtained by MC simulation. Then, the average error of the affine interval linear solve is the square root of the sum of the squared distribution errors over all the grids, divided by the total number of samples. It is 12.1% for the affine interval linear solve as in this paper [Fig. 1(b)] and 10.8% for the affine interval linear solve as in [18] [Fig. 1(c)]. We can see that both sets of affine arithmetic do a workable job of modeling the most likely part of the real distribution. As explained in Section II-B, however, the new affine arithmetic is more efficient to compute and maintains the first-order sensitivity information of the final affine result with respect to the original uncertainties.

E. Modeling RLC Parameter Variations

Similar to [17] and [18], we classify the variations into two categories: global variation and local variation [7], [26]. Global variations can be assumed to affect all the devices and interconnects in a similar way within the same chip. Local variations often exhibit spatial correlations, i.e., the device and interconnect parameters are affected similarly by a common source of variation when these physical elements are close enough to each other. This paper also considers a linear-interconnect circuit where the input-driver voltage source is assumed to be scalar and deterministic, and the resistance (R_i), capacitance (C_i), and inductance (L_i) are subject to linear combinations of global

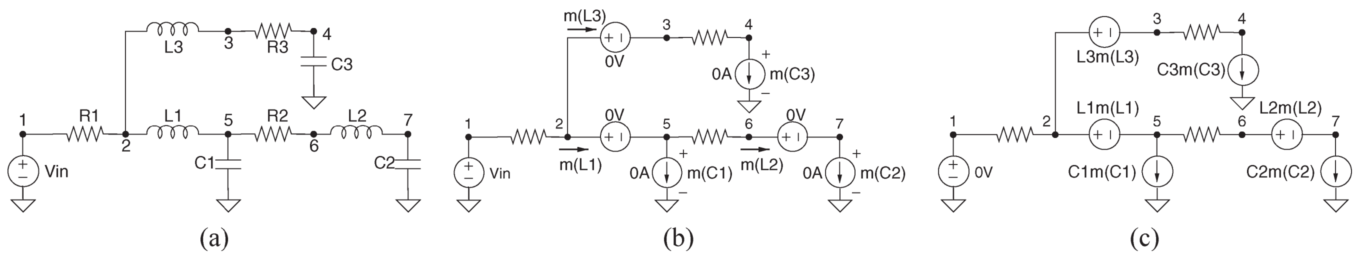


Fig. 2. (a) *RLC* interconnect circuit example. (b) Equivalent dc circuit for first moment generation. (c) Equivalent dc circuit with source values changed for second moment generation.

and local variations with correlations in the basic affine forms introduced earlier

$$R_i = R_{i,0} + \sum_{j=1}^{l} \Delta R_{i,j}\varepsilon_j + \sum_{j=l+1}^{m} \Delta R_{i,j}\varepsilon_j + \sum_{j=m+1}^{n} \Delta R_{i,j}\varepsilon_j \tag{8}$$

$$C_i = C_{i,0} + \sum_{j=1}^{l} \Delta C_{i,j}\varepsilon_j + \sum_{j=l+1}^{m} \Delta C_{i,j}\varepsilon_j - \sum_{j=m+1}^{n} \Delta C_{i,j}\varepsilon_j \tag{9}$$

$$L_i = L_{i,0} + \sum_{j=1}^{l} \Delta L_{i,j}\varepsilon_j - \sum_{j=l+1}^{m} \Delta L_{i,j}\varepsilon_j - \sum_{j=m+1}^{n} \Delta L_{i,j}\varepsilon_j. \tag{10}$$

$R_{i,0}$, $C_{i,0}$, and $L_{i,0}$ are the nominal parameter values. Each unique source of global or local variation is modeled by an uncertainty symbol ε_j. $\Delta R_{i,j}$, $\Delta C_{i,j}$, and $\Delta L_{i,j}$ are the magnitude of the linearized parameter variations due to ε_j, a particular source of variation [27]. Any individual source of uncertainty may contribute to more than one parameter and thus lead to direct correlations among these parameters. The equations shown above simply emphasize the fact that any uncertainty symbol ε_j can appear with positive or negative proportional impact in any of the linearized formulas for any R, C, and L. For example, when metal width increases from its nominal value, the metal ground capacitance may be increased while the metal resistance is decreased. This is a simple model, but it can capture global and local variations and correlations, and it maps perfectly onto our preferred affine interval model of computation.

III. Fast Interval-Valued Statistical Interconnect Model Reduction

In this section, we develop efficient interval-valued versions of the standard path-tracing algorithms [20], [23] for (orthonormalized) moment computation. We represent variational circuit element values as affine intervals, in the forms of (8)–(10), and then replace the entire "recipe" of the path-tracing algorithm by pushing these interval values through the computation steps. The result is a reduced, small set of either interval-valued moments (for interval AWE [13]) or interval-valued reduced system matrices and vector (for interval PRIMA [15], to be defined later on), compared with the large number of parameters in the original, unreduced circuits. Then, we statistically sample these "reduced" intervals to complete each model-order-reduction algorithm and produce an estimate of the delay distribution for the original variational interconnect.

A. Interval-Valued Path Tracing for Moment Generation

In [17] and [18], we started with an interval-valued MNA formulation for variational interconnect circuits of general topologies, and the interval-valued moments were computed by interval-valued LU decomposition. This speaks well for the general utility of the interval approach. However, MNA formulation and LU decomposition for moment generation is not the preferred approach for one extremely important class of circuits: tree-structured circuits. For such circuits, the moment computation can be realized by a much more efficient algorithm: path tracing [20]. Indeed, the fastest state-of-the-art AWE/PRIMA implementations (most notably RICE 4 [20] and RICE 5 [23]) achieve some of their speedup by using path tracing for treelike interconnects. Our goal is to create an interval-valued counterpart of path tracing.[1]

In principle, the interval-valued moments are computed through successive dc analyses of the interconnect circuit where capacitors are substituted with interval-valued dc current sources and inductors with interval-valued dc voltage sources [20]. The dc analyses are based on a sequence of reverse and forward depth-first search (DFS). Initially, the input driver is set to a dc source of its final (deterministic) value, all the capacitors to zero-valued current sources, and all the inductors to zero-valued voltage sources. After the first dc analysis, the resultant interval-valued voltages across each capacitor-current source form the first generation of interval-valued capacitor moments, and the interval-valued currents through each inductor-voltage source form the first generation of interval-valued inductor moments. The succeeding generations of interval-valued moments are computed by setting the driver to 0 and replacing each capacitor or inductor source with the product of its previous interval-valued moment and respective interval value of capacitance or inductance. Fig. 2(b) and (c) illustrate how to compute the first two generations of interval-valued moments of a simple interconnect circuit example shown in Fig. 2(a).

[1]Ratzlaff and Pillage [20] and Odabasioglu and Pileggi [23] also showed how to attack nontree circuits using these ideas; we restrict ourselves here to treelike circuits.

```
/⋆ interval-valued path-tracing ⋆/
1. for $k = 1, 2, \cdots, 2q–1+s$
       1.1 reverse DFS to compute inductor moments;
       1.2 forward DFS to compute capacitor moments;
   end
2. obtain the moments at the output node of interest;

/⋆ Sample moments and compute delay ⋆/
3. repeat: normally sample moment intervals ($\mu$=0, $\sigma$=1);
4.     for each set of scalar samples
       4.1 set up Hankel matrix and vector;
       4.2 coefficients of transfer function's denominator
               = – (Hankel matrix)$^{-1}$·(Hankel vector);
       4.3 find roots of denominator polynomial as poles;
       4.4 set up Vandemonde matrix;
       4.5 residues = – (Vandemonde matrix)$^{-1}$·(moments);
       4.6 compute interconnect delay;
       end
   until pre-specified number of samples are generated
5. obtain delay distribution;
```

Fig. 3. Fast interval-valued AWE algorithm.

The above "path tracing" procedure is stopped when the required number of interval-valued moments is obtained.[2] Note that the topology of an interconnect circuit does not change over successive moment computations. Therefore, the DFS of the circuit tree needs only to be performed once, and the tree elements can be "remembered" in the proper computational order, which makes all path-tracing algorithms very fast. Furthermore, just as in the scalar case, interval-valued path tracing requires many fewer (interval valued) calculations than a full LU step, and is thus both faster and more accurate.

B. Fast Interval-Valued AWE

In [17] and [18], affine interval arithmetic was employed exclusively, that is, from *RLC* variational circuit elements, all the way through until a final, reduced set of interval valued poles/residues was obtained. Let us revisit the assumption that we must use the interval mechanics exclusively. Consider stopping the interval calculation as soon as we have generated the required $2q + s$ interval-valued moments. At this point, we randomly, normally ($\mu = 0$ and $\sigma = 1$) sample the interval-valued moments. Then, each set of scalar-valued moment samples is used to construct the scalar-valued transfer function and compute the corresponding scalar-valued poles and residues, just like a standard AWE algorithm [13]. The scalar-valued interconnect circuit delay can be obtained via time-domain transient analysis. Enough random samples produce a delay distribution for the variational interconnect. MC sampling over moment intervals is also quite efficient, since it primarily involves scalar computations for the reduced interconnect model. Fig. 3 describes the overall fast interval-valued AWE algorithm with path tracing. It will become more clear in Section III-D why we push affine interval arithmetic only to interval-valued moments rather than interval-valued poles and residues, a later step in the chain of numerical computations.

[2]For AWE, if q is the order of the reduced interconnect model, $2q + s$ moments are needed, where s is the order of moment shifting, as in the scalar case [20].

C. Fast Interval-Valued PRIMA

For clarity, we shall reserve the "$\widehat{\ }$" symbol for interval-valued quantities. In [17] and [18], the interval-valued PRIMA algorithm started with an interval-valued MNA formulation

$$(\widehat{\boldsymbol{G}} + s\widehat{\boldsymbol{C}})\widehat{\boldsymbol{x}} = \boldsymbol{b} \tag{11}$$

where the interval-valued conductance matrix $\widehat{\boldsymbol{G}}$, interval-valued susceptance matrix $\widehat{\boldsymbol{C}}$, and scalar-valued excitation vector $\boldsymbol{b}$ are defined as

$$\widehat{\boldsymbol{G}} = \begin{bmatrix} \widehat{\boldsymbol{N}} & \boldsymbol{E} \\ -\boldsymbol{E}^{\mathrm{T}} & \boldsymbol{0} \end{bmatrix}, \quad \widehat{\boldsymbol{C}} = \begin{bmatrix} \widehat{\boldsymbol{Q}} & \boldsymbol{0} \\ \boldsymbol{0} & \widehat{\boldsymbol{H}} \end{bmatrix}, \quad \boldsymbol{b} = -\begin{bmatrix} \boldsymbol{i} \\ \boldsymbol{v} \end{bmatrix}. \tag{12}$$

$\widehat{\boldsymbol{N}}$, $\widehat{\boldsymbol{Q}}$, and $\widehat{\boldsymbol{H}}$ are the interval-valued matrices containing the stamps for variational resistors, capacitors, and inductors, respectively. $\boldsymbol{E}$ consists of constants ones, negative ones, and zeros, which correspond to the variables of current induced by inductance and voltage source. The interval-valued solution vector $\widehat{\boldsymbol{x}}$ contains node voltages appended by inductance and voltage source current. The output-selecting vector is denoted as $\widehat{\boldsymbol{l}}$. Interval-valued LU decomposition was used to construct an orthonormal interval-valued matrix $\widehat{\boldsymbol{X}}$ that spans the interval-valued Krylov subspace defined as

$$\mathrm{Kr}(\widehat{\boldsymbol{A}}, \widehat{\boldsymbol{r}}, q) = \mathrm{colsp}(\widehat{\boldsymbol{r}}, \widehat{\boldsymbol{A}}\widehat{\boldsymbol{r}}, \widehat{\boldsymbol{A}}^2\widehat{\boldsymbol{r}}, \ldots, \widehat{\boldsymbol{A}}^{\mathrm{q}-1}\widehat{\boldsymbol{r}}) \tag{13}$$

where $\widehat{\boldsymbol{A}} = -\widehat{\boldsymbol{G}}^{-1}\widehat{\boldsymbol{C}}$ and $\widehat{\boldsymbol{r}} = -\widehat{\boldsymbol{G}}^{-1}\widehat{\boldsymbol{b}}$. Then, the original-variational-interconnect system (characterized by $\widehat{\boldsymbol{G}}$, $\widehat{\boldsymbol{C}}$, and $\boldsymbol{b}$) is projected into a reduced-order interval-valued system (characterized by $\widehat{\boldsymbol{G}}'$, $\widehat{\boldsymbol{C}}'$, and $\widehat{\boldsymbol{b}}'$) in the interval-valued Krylov subspace spanned by $\widehat{\boldsymbol{X}}$ via congruence transformation: $\widehat{\boldsymbol{G}}' = \widehat{\boldsymbol{X}}^{\mathrm{T}}\widehat{\boldsymbol{G}}\widehat{\boldsymbol{X}}$, $\widehat{\boldsymbol{C}}' = \widehat{\boldsymbol{X}}^{\mathrm{T}}\widehat{\boldsymbol{C}}\widehat{\boldsymbol{X}}$, and $\widehat{\boldsymbol{b}}' = \widehat{\boldsymbol{X}}^{\mathrm{T}}\boldsymbol{b}$.

Again, instead of interval-valued LU factorization, a much faster interval-valued version of the standard path-tracing algorithm in [23] can be used to successively generate the interval-valued moment vectors $\widehat{\boldsymbol{x}}_k$ $(k = 0, 1, \ldots, q-1)$, and filling in the columns of $\widehat{\boldsymbol{X}}$ while maintaining its orthonormality (by Gram–Schmidt Orthonormalization). According to [23], the interval-valued path-tracing algorithm in Section III-A can be augmented, only minimally, to implicitly compute the interval-valued reduced system without any explicit matrix construction or manipulation. And the congruence transformation is performed concurrently with interval-valued moment generation and orthonormalization. See [19] and [23] for details. Most of the computations here are matrix-vector operations, easily handled by basic affine interval arithmetic.

For fast interval-valued PRIMA, we again normally sample ($\mu = 0$ and $\sigma = 1$) each uncertainty symbol of the interval-valued elements of $\widehat{\boldsymbol{G}}'$, $\widehat{\boldsymbol{C}}'$, and $\widehat{\boldsymbol{b}}'$ to produce a scalar-valued reduced system (in terms of $\boldsymbol{G}'$, $\boldsymbol{C}'$, and $\boldsymbol{b}'$). Then, a standard version of the PRIMA algorithm [15] can be applied to eigendecompose this scalar-valued reduced system, extract the poles and residues, and finally analyze the interconnect circuit delay. Fig. 4 gives the details of the fast interval-valued PRIMA algorithm.

```
/⋆ interval-valued path-tracing ⋆/
1. for $k = 1, 2, \cdots, q-1$
      1.1 reverse DFS to compute inductor moments;
      1.2 forward DFS to compute capacitor moments;
      1.3 append $\widehat{\boldsymbol{x}}_k$ to $\widehat{\boldsymbol{X}}^{(k-1)}$;
      1.4 orthonormalize $\widehat{\boldsymbol{X}}^{(k)}$;
      1.5 compute $\widehat{\boldsymbol{C}}\widehat{\boldsymbol{x}}_k$ and $\widehat{\boldsymbol{G}}\widehat{\boldsymbol{x}}_k$;
   end
2. compute $\widehat{\boldsymbol{G}}' = \widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{G}}\widehat{\boldsymbol{X}}$ and $\widehat{\boldsymbol{C}}' = \widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{C}}\widehat{\boldsymbol{X}}$;
   compute $\widehat{\boldsymbol{b}}' = \widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{b}}$ and $\widehat{\boldsymbol{l}}' = \widehat{\boldsymbol{X}}^T\widehat{\boldsymbol{l}}$;

/⋆ Sample reduced system and compute delay ⋆/
3. repeat: normally sample $\widehat{\boldsymbol{G}}'$, $\widehat{\boldsymbol{C}}'$, and $\widehat{\boldsymbol{b}}'$ ($\mu$=0, $\sigma$=1);
4.    for each set of scalar samples
      4.1 let $\boldsymbol{A}' = -\boldsymbol{G}'^{-1}\boldsymbol{C}'$ and eigendecompose $\boldsymbol{A}'$;
      4.2 let column vectors of $\boldsymbol{S}$ be $\boldsymbol{A}'$'s eigenvectors;
            $\boldsymbol{\Lambda} = diag(\lambda_1, \lambda_2, \cdots, \lambda_q)$;
      4.3 invert $\lambda_1, \lambda_2, \cdots, \lambda_q$ to obtain poles;
      4.4 solve $\boldsymbol{G}'\boldsymbol{w} = \boldsymbol{b}'$ for $\boldsymbol{w}$;
      4.5 residues = $-\boldsymbol{S}^T\boldsymbol{l}'\boldsymbol{S}^{-1}\boldsymbol{w}$·(poles);
      end
   until pre-specified number of samples are generated
5. obtain delay distribution;
```

Fig. 4. Fast interval-valued PRIMA algorithm.

D. Interval/Scalar Tradeoffs

Fig. 5(a) and (c) summarize the flows of the interval-valued AWE and PRIMA algorithms, respectively, presented in [17] and [18]. Fig. 5(b) and (d) summarize the flows of the fast interval-valued AWE and PRIMA algorithms, respectively, presented in Section III of this paper. Compared to our earlier approaches proposed in [17] and [18], faster basic interval operations (that avoid new uncertainty terms), and replacing MNA formulation and LU decomposition with tree path tracing provide much of the speed improvement. But the other source of improvement for better accuracy is the use of a hybrid interval/scalar computation strategy.

To understand why, recall first that approximation of interval endpoints and correlations creates errors rather like floating-point roundoff errors, but more macroscopic and not so easy to "ignore." Also, as with basic floating-point calculations, the longer the chain of computation is, the more errors may be accrued. Therefore, replacing LU decomposition with path tracing for (orthonormalized) moment computation increases not only efficiency but also accuracy, by greatly reducing the number of operations that need approximate affine interval arithmetic, e.g., multiplication. Replacing scalar-valued LU decomposition by scalar-valued path tracing does not have this significant accuracy boost as an additional considerable benefit.

Second, it is not essential to employ intervals in every step of our algorithms. The right "boundary" between interval- and scalar-valued computations is a new degree freedom in these algorithms. Recall that the affine model is fundamentally a linear model defined by each interval's midpoint, and a statistical interpretation on the uncertainty symbols creates a set of symmetric excursions about each interval's central point. A pair of correlated affine intervals by construction define a range bounded by a two-dimensional "central-symmetric convex polytope" [16]. This turns out to be an efficient approximation to uncertainties moving through numerical computations, and does an excellent job for computations dominated by linear operations—additions and subtractions. This may provide a more pessimistic overestimation on the bound, however, if the computation is highly nonlinear and the real distribution is asymmetric. For example, the root locus of a polynomial with correlated varying coefficients can have too high a degree of curvature for any central-symmetric convex polytope to bound tightly (see [17]). Thus, one remedial tradeoff is stopping the interval-valued computations as soon as the resulting "intermediate" problem representation is small enough that standard, scalar MC sampling can be done efficiently to prevent more interval estimation errors. This is especially attractive for AWE/PRIMA-style interconnect reduction algorithms, which employ a "front-end" of large-scale, near-linear matrix computations, and a "back-end" of small-scale, yet more nonlinear operations (e.g., root finding, eigendecomposition) on reduced representations of the problem. Thus, in the development of the algorithms for fast interval-valued AWE and PRIMA in Section III, we made the following tradeoffs.

1) As shown in Fig. 5(b), for fast interval-valued AWE, conduct interval-valued computations to reduce the original large variational system to a much smaller set of interval-valued moments; stop interval-valued computation here, and switch to scalar-valued MC sampling over the moments to avoid the errors of affine approximation of nonlinear polynomial root finding for poles (Step 4.3 in Fig. 3).
2) As shown in Fig. 5(d), for fast interval-valued PRIMA, conduct interval-valued computations to reduce the original large variational system to a much smaller set of interval-valued system matrices and vector; stop interval-valued computation here, and switch to scalar-valued MC sampling over the reduced system to minimize the errors of affine approximation of nonlinear eigendecomposition (Step 4.1 in Fig. 4).

IV. Fast Interval-Valued Statistical Effective Capacitance Modeling

Having developed fast interval-valued interconnect model reduction algorithms, we study in this section the next obvious problem: effective capacitance (C_{eff}) models for interconnect. We first briefly review the idea of C_{eff} modeling for interconnect with driving gate,[3] and then apply the proposed interval-valued computation strategy to build two statistical C_{eff} models.

A. From Interconnect to Effective Capacitance

It is not adequate any more to compute gate delay assuming the total interconnect capacitance for the load. It is crucial for a successful static timing analysis tool, deterministic or statistical, to accurately handle the interactions between the gate and

[3]For simplicity, we assume a single CMOS inverter for the driving gate.

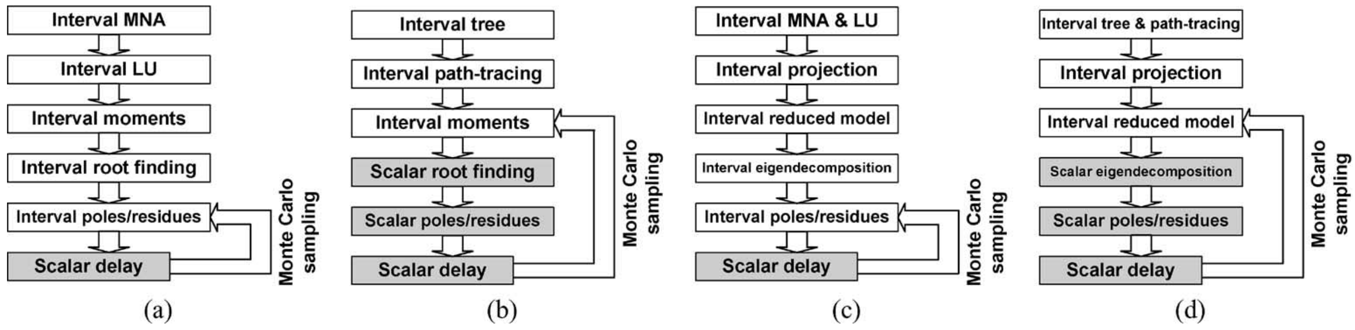


Fig. 5. Flow of (a) interval-valued AWE algorithm, (b) fast interval-valued AWE algorithm, (c) interval-valued PRIMA algorithm, and (d) fast interval-valued PRIMA algorithm.

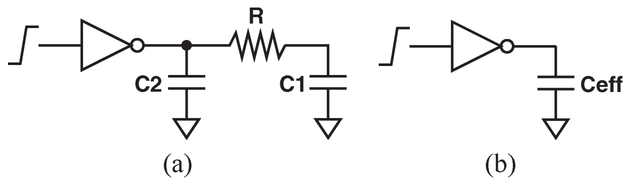


Fig. 6. (a) π-circuit model for interconnect. (b) C_{eff} model for interconnect.

interconnect-delay models, because the "shielding effect" due to increasing wire resistance can no longer be ignored. That is, the wire resistance shields some of the interconnect capacitance such that the actual load capacitance "seen" by the driving gate is less than the total interconnect capacitance. To account for this, one can approximate the driving point admittance by synthesizing a π-circuit model [see Fig. 6(a)] to replace the original interconnect [28]. π-model is generally regarded as an accurate enough approximation for interconnect. However, it is not compatible with most existing timing analysis methodologies, which often require a lumped interconnect capacitive load for the gate. Therefore, an effective capacitance (C_{eff}) model for the interconnect [see Fig. 6(b)] was developed in [21].

As pointed out in Section I, circuit (including gate and interconnect) timing analysis tools that can handle manufacturing variations are being developed. But what is still missing is the statistical interaction between the variational gate and interconnect models. In what follows, we build the missing link by developing statistical C_{eff} models, based on affine interval representations.

B. Statistical Iterative Effective Capacitance

The first three interval-valued moments of the driving point admittance are readily computed by our interval-valued path-tracing algorithm. After that, we stop interval-valued computation and switch to sampling over the interval moments. Each set of scalar-valued moment samples (m_1, m_2, and m_3) is used to compute one set of scalar-valued π-model parameters

$$R = -\frac{m_3^2}{m_2^3} \tag{14}$$

$$C_1 = \frac{m_2^2}{m_3} \tag{15}$$

$$C_2 = m_1 - \frac{m_2^2}{m_3}. \tag{16}$$

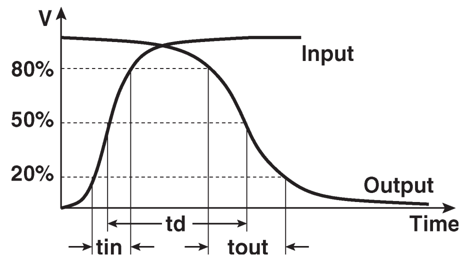


Fig. 7. Scalar-valued gate delay notations.

Then, by equating the average current flowing into the π-circuit with the average current flowing into the C_{eff} (following the development in [21]), we can estimate the scalar-valued C_{eff} as

$$C_{\mathrm{eff}} = C_2 + C_1\left[1 - \frac{RC_1}{t_{\mathrm{D}} - \frac{t_{\mathrm{x}}}{2}} + \frac{(RC_1)^2}{t_{\mathrm{x}}\left(t_{\mathrm{D}} - \frac{t_{\mathrm{x}}}{2}\right)} e^{\frac{-(t_{\mathrm{D}}-t_{\mathrm{x}})}{RC_1}}\left(1 - \mathrm{e}^{\frac{-t_{\mathrm{x}}}{RC_1}}\right)\right]. \tag{17}$$

In other words, the scalar-valued C_{eff} load is so much so that it has the same total charge transfer as the scalar-valued π-circuit. Note that the value of C_{eff} is between C_2 and $C_1 + C_2$ (i.e., the total interconnect capacitance). t_{D} and t_{x} are computed by

$$t_{\mathrm{D}} = t_{\mathrm{d}} + \frac{t_{\mathrm{in}}}{2} \tag{18}$$

$$t_{\mathrm{x}} = t_{\mathrm{d}} + \frac{t_{\mathrm{in}}}{2} - 0.5t_{\mathrm{out}} \tag{19}$$

where the scalar-valued gate delay notations (i.e., t_{in}, t_{out}, and t_{d}) are illustrated in Fig. 7. t_{in} denotes the scalar gate input transition time from 20% to 80% of the final input voltage, t_{out} the scalar output transition time from 20% to 80% of the final output voltage, and t_{d} the scalar delay between the time to reach 50% of the final input voltage point and the time to reach 50% of the final output voltage point.

Typically, the scalar gate delay t_{d} and output transition time t_{out} can be empirically precharacterized with respect to scalar input transition time t_{in} and scalar load capacitance C_{L} (currently being modeled as C_{eff}), using either lookup tables

```
1. compute the first three interval-valued moments;

/⋆ Sample moments and compute $C_{eff}$ ⋆/
2. repeat: normally sample moment intervals;
3.   for each set of scalar samples
        compute $R_1$, $C_1$, and $C_2$;
        $C_L = C_1 + C_2$;
        repeat:
           use $C_L$ and $t_{in}$ to compute $t_d$ and $t_{out}$;
           use $t_d$ and $t_{out}$ to compute $C_{eff}$;
           $C_L = C_{eff}$;
        until $C_{eff}$ changes within a pre-specified tolerance
     end
   until pre-specified number of samples are generated
4. obtain $C_{eff}$ distribution;
```

Fig. 8. Statistical iterative C_{eff} modeling algorithm.

or k-factor equations [29]. For simplicity, we adopt the same scalar-valued k-factor equation templates as in [21] and [29][4]

$$t_{\text{d}} = (k_{\text{d1}} + k_{\text{d2}}C_{\text{L}})t_{\text{in}} + k_{\text{d3}}C_{\text{L}}^3 + k_{\text{d4}}C_{\text{L}} + k_{\text{d5}} \quad (20)$$

$$t_{\text{out}} = (k_{\text{o1}} + k_{\text{o2}}C_{\text{L}})t_{\text{in}} + k_{\text{o3}}C_{\text{L}}^3 + k_{\text{o4}}C_{\text{L}} + k_{\text{o5}}. \quad (21)$$

We can see that the scalar-valued C_{eff} is a function of the scalar gate delay t_{d} and output transition time t_{out} [i.e., (17)–(19)] and vice versa [i.e., (20) and (21)]. Therefore, for each scalar sample, C_{eff}, t_{d}, and t_{out} must be calculated altogether iteratively. The procedure is stopped when the difference between the scalar C_{eff} values in two consecutive iterations is less than a given threshold. One complete pass of iterations (mostly in less than five iterations, according to [21]) produces one set of scalar C_{eff}, t_{d}, and t_{out}.

Enough random sampling over the interval-valued moments and then scalar-valued computations for each sample as outlined above eventually produce a distribution of the variational C_{eff}, as well as those of the gate delay and output transition time. MC sampling over the only three moment intervals is very efficient. Fig. 8 describes the complete statistical iterative C_{eff} algorithm.

C. Statistical Direct Effective Capacitance

It has also recently been suggested in [22] that the C_{eff} can be calculated by direct formulas. Different from the approach in [21], Nassif and Li [22] modeled the gate by a constant linear resistor R_{eff}, as shown in Fig. 9, such that the overall gate-interconnect is equivalent to a simple *RC* circuit. The value of the equivalent output resistance R_{eff} is determined by

$$R_{\text{eff}} = \frac{\ln\left(\frac{3V_{\text{dd}}-4V_{\text{tn}}}{V_{\text{dd}}}\right) + \frac{2V_{\text{tn}}}{V_{\text{dd}}-V_{\text{tn}}}}{(\ln 2)k_{\text{n}}(V_{\text{dd}} - V_{\text{tn}})} \quad (22)$$

through basic inverter delay analysis [29] for a falling output transition. V_{dd} is the supply voltage and V_{tn} is the threshold voltage of the NMOS transistor, and we assume $V_{\text{dd}} > 2V_{\text{tn}}$. k_{n} is the "gain factor" of the NMOS transistor, which depends on the mobility μ_n, gate oxide permittivity ε_{ox} and thickness t_{ox}, and aspect ratio W_n/L_n of the NMOS transistor.

[4]It is still under investigation whether the interval model can be well applied to table lookup, which is different from other affine arithmetic operations we presented before.

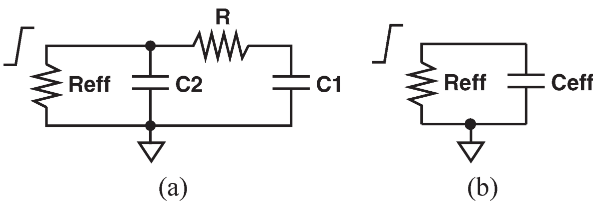


Fig. 9. Constant linear resistor model for the gate with (a) π-circuit and (b) C_{eff} models for the interconnect, respectively.

```
1. compute the first three interval-valued moments;

/⋆ Sample moments and compute $C_{eff}$ ⋆/
2. repeat: normally sample moment intervals;
3.   for each set of scalar samples
        compute $R_{eff}$;
        compute $R_1$, $C_1$, and $C_2$;
        compute $\alpha$ and $\beta$;
        compute $C_{eff}$;
     end
   until pre-specified number of samples are generated
4. obtain $C_{eff}$ distribution;
```

Fig. 10. Statistical direct C_{eff} modeling algorithm.

TABLE I
NUMBERS OF *RCL* ELEMENTS FOR THREE BENCHMARK CIRCUITS

	R	C	L
design0	41	41	40
design1	638	637	0
design2	1121	1120	0

Then, instead of matching the average current between the π-circuit and the C_{eff} as in [21], Nassif and Li [22] matched the time to reach 50% of the final gate output voltage between the R_{eff}-π circuit [see Fig. 9(a)] and the R_{eff}-C_{eff} circuit [see Fig. 9(b)]. The solution to the matching equation is the C_{eff}, which can be further approximated in a simple closed-form as

$$C_{\text{eff}} = \frac{3\alpha + \beta^2}{3 + \beta^2}(C_1 + C_2) \quad (23)$$

with less than 1% error compared to the exact yet nonanalytical solution. α and β are

$$\alpha = \frac{C_2}{C_1 + C_2} \quad (24)$$

$$\beta = \frac{R_{\text{eff}}}{R}. \quad (25)$$

And the π-model parameters R, C_1, and C_2 are still computed by (14)–(16).

Note that in general, the C_{eff} values given by the standard, iterative, and direct scalar-valued models are somewhat different, as we shall see in Section V-B. But this presents no difficulty to our interval-valued statistical analysis framework, which is flexible enough to accommodate a wide variety of

TABLE II
COMPARISON BETWEEN MC-AWE AND statAWE

1	2	3	4	5	6	7	8	9	10	11
global/local	number	MC-AWE			*stat*AWE			error		run time
variation	of ε	mean (ps)	std (ps)	run time (s)	mean (ps)	std (ps)	run time (s)	mean	std	speed-up
design1										
20% / 10%	6	1313.2	184.8	196	1347.8	189.8	19	2.6%	2.7%	10
	11	1316.6	185.3	189	1347.2	189.7	18	2.3%	2.4%	11
	21	1314.5	185.0	185	1347.8	189.8	18	2.5%	2.6%	10
10% / 20%	6	1343.1	189.0	197	1346.6	189.6	18	0.3%	0.3%	11
	11	1350.0	190.0	185	1350.4	189.3	17	0.0%	0.4%	11
	21	1352.5	190.3	202	1354.0	189.8	20	0.1%	0.3%	10
5% / 30%	6	1345.3	189.3	199	1352.0	190.3	19	0.5%	0.5%	10
	11	1352.1	190.2	193	1348.8	189.9	18	0.2%	0.2%	11
	21	1358.1	191.1	191	1357.9	190.4	20	0.0%	0.4%	10
design2										
20% / 10%	6	847.4	119.3	302	879.3	123.2	28	3.8%	3.3%	11
	10	845.1	118.9	305	875.0	123.2	29	3.5%	3.6%	11
	19	845.2	118.9	297	878.0	123.6	31	3.9%	4.0%	10
10% / 20%	6	864.1	121.6	304	878.6	123.6	27	1.7%	1.6%	11
	10	866.8	122.0	299	866.3	121.9	27	0.1%	0.1%	11
	19	866.1	121.9	292	870.8	122.5	26	0.5%	0.5%	11
5% / 30%	6	864.1	121.6	301	892.9	125.6	28	3.3%	3.3%	11
	10	869.2	122.3	298	892.5	125.6	27	2.7%	2.7%	11
	19	875.1	123.1	297	867.9	122.1	29	0.8%	0.8%	10

TABLE III
COMPARISON BETWEEN MC-PRIMA AND statPRIMA

1	2	3	4	5	6	7	8	9	10	11
global/local	number	MC-PRIMA			*stat*PRIMA			error		run time
variation	of ε	mean (ps)	std (ps)	run time (s)	mean (ps)	std (ps)	run time (s)	mean	std	speed-up
design1										
20% / 10%	6	1307.3	184.0	184	1336.5	188.1	17	2.2%	2.2%	11
	11	1310.3	184.4	178	1348.1	190.1	18	2.9%	3.1%	10
	21	1312.3	184.7	176	1348.6	190.2	17	2.8%	3.0%	10
10% / 20%	6	1269.9	178.7	189	1338.1	188.6	16	5.4%	5.5%	12
	11	1283.7	180.7	175	1341.8	189.2	17	4.5%	4.7%	10
	21	1291.3	181.7	184	1340.5	189.0	19	3.8%	4.0%	10
5% / 30%	6	1290.3	185.1	177	1339.1	188.5	18	3.8%	1.8%	10
	11	1307.1	179.9	180	1298.2	182.3	16	0.7%	1.3%	11
	21	1316.8	181.3	179	1337.0	188.2	17	1.5%	3.8%	11
design2										
20% / 10%	6	845.9	119.0	269	869.9	122.3	25	2.8%	2.8%	11
	10	845.1	118.9	265	870.7	122.6	25	3.0%	3.1%	11
	19	847.1	119.2	267	871.6	122.7	24	2.9%	2.9%	11
10% / 20%	6	863.5	121.5	254	872.4	122.8	22	1.0%	1.1%	12
	10	870.2	122.5	279	861.5	121.2	27	1.0%	1.1%	10
	19	866.5	121.9	260	845.7	119.0	24	2.4%	2.4%	11
5% / 30%	6	864.2	121.6	262	877.0	123.4	23	1.5%	1.5%	11
	10	865.1	121.7	256	874.3	123.0	26	1.1%	1.1%	10
	19	867.8	122.1	263	852.2	120.0	22	1.8%	1.7%	12

numerical computation schemes. The above scalar-valued direct C_{eff} modeling algorithm can be easily retargeted to an interval-valued statistical direct C_{eff} algorithm, as shown in Fig. 10. Just like the approach proposed in Section IV-B, we first compute the first three interval-valued moments of the driving point admittance, based on our interval-valued path-tracing algorithm. We stop interval computation here, and switch to efficient MC sampling over the three moment intervals. For each sample, we then perform scalar-valued direct C_{eff} computations, as introduced earlier in this section. Again, enough random scalar samples eventually produce a distribution of the variational C_{eff}. Fig. 10 describes the complete statistical direct C_{eff} algorithm.

To summarize, following the discussions in Section III-D, it is easy to understand that in the development of both iterative and direct statistical C_{eff} algorithms, we made the following straightforward tradeoff in the hybrid interval/scalar computation framework.

1) Conduct interval-valued computations to reduce the original large variational system to three interval-valued moments; stop interval-valued computation here.
2) Now switch to scalar-valued MC sampling over the moments to avoid the errors of affine approximation for the nonlinear equations for π-model or C_{eff} computation.

V. EXPERIMENTAL RESULTS

A. Results of Interconnect Model Reduction

The fast interval-valued statistical algorithms for AWE and PRIMA of Figs. 3 and 4 have been implemented as a pair of tools called statAWE and statPRIMA, respectively. The tools, together with the underlying affine arithmetic library, were implemented in C/C++ and benchmarked on a 1.0 GHz UNIX machine, using three *RC*(*L*) treelike interconnect circuits (design 0, design 1, and design 2). The numbers of *RLC* elements of the three circuits are given in Table I. It is worth pointing out that design 2 is a synthetic unbuffered balanced H-shape clock tree similar to that in [30]. We choose a number of uncertainty symbols ranging from 6 to 21 for (8)–(10). Among these symbols, one is assumed to originate from global variation and is shared by all *RLC* elements. The rest of the symbols are assumed for local wire width variations, linearized by first-order Taylor series expansion, and are only shared by a cluster of *RLC* elements that are "close enough" to one another. In other words, for the case of, say, six uncertainty terms, we partition each net-list into six groups, and assign the same uncertainty term to every element in one group. Our algorithms and implementation can accommodate any number of global and local uncertainty symbols that originate from most types of variation sources. Furthermore, we assume three combinations for the relative σ of global and local variations: 20%/10%, 10%/20%, and 5%/30%, given in the first columns of Tables II and III.

We find 50% delay distribution of fourth-order statAWE and statPRIMA for design 1 and design 2. For comparison of accuracy and speed, we use a fast interconnect simulator based on standard model order reduction (RICE 4 for AWE [20] and RICE 5 for PRIMA [23]) in a simple MC loop.[5] That is, randomly varying *RLC* circuit elements according to the specified variations at the very beginning and repeating scalar deterministic RICE 4/5. For the sake of fairness and efficiency, we determine how many samples for each tool, parameter setting, and experiment using standard confidence interval methods [31]. We use samples sufficient to guarantee a 99% confidence level with 1% accuracy. For almost all our experiments, this turns out to be between 3000 and 3500 samples. We show mean and standard deviation (std) of the resulting delay distributions, and overall run times, comparing our interval approaches statAWE and statPRIMA with the straightforward MC simulation runs (MC-AWE and MC-PRIMA) in Tables II and III, respectively.

In columns 9–11 of Tables II and III, the errors of mean delay and standard deviation, and the run-time speedup are defined as the difference between the result of our interval-valued approach and that of MC simulation, normalized to the latter. It is easy to see that both the mean delay and standard deviation errors for all the experiments are less than 6%, even with variations up to 35%. On average, the mean delay error is 1.7% for statAWE and 2.5% for statPRIMA; the standard deviation error is 1.8% for statAWE and 2.6% for statPRIMA. Both statAWE and statPRIMA achieve an average of 10× run-time speedup over the simple MC simulation. Stated differently, a typical interval-valued variational analysis conducted with our tools takes the same time as simulating about 300 different interconnect configurations (samples) in a brute-force manner, with a very fast interconnect simulator. We see more speedup

[5]Note that we use the RICE tools in a simple simulator-in-a-loop MC experiment and not the extensions of [7], which can compute variational models directly, but only for uncorrelated uncertainties.

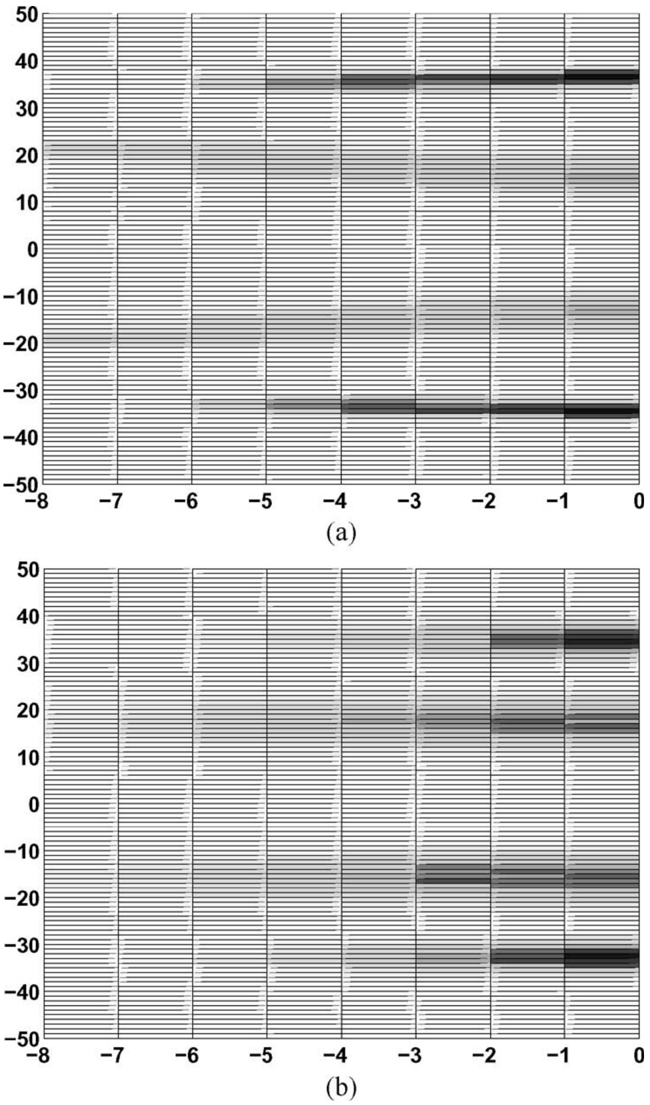


Fig. 11. Comparison of pole distribution in false color shading between MC-AWE and statAWE for design 0. (a) Pole distribution of MC-AWE. (b) Pole distribution of statAWE. For both figures, x-axis is real axis and y-axis is imaginary axis. Average errors are 7.1%, 3.6%, 4.2%, and 7.2%, respectively, for four poles (from top to bottom).

for larger circuits. The run time for the interval-valued path-tracing component of our tools is less than 5 s for all cases; MC sampling of the much smaller, interval-valued reduced system, and the subsequent scalar-valued computations on these samples, comprise the majority of the total run time.

Figs. 11 and 12 offer some visual evidence on the quality of results of our interval-based algorithms. Fig. 11 plots the distributions of the four dominant poles for the *RLC* circuit design 0, assuming 5% global variation, 30% local variation, nine uncertainty terms, and eighth-order AWE reduction. False color shading indicates greater pole probability with a darker color. We compare the pole density between the MC simulation [Fig. 11(a)] and our own statAWE extraction [Fig. 11(b)]. The visual correspondence is good. We further quantify the correspondence in the same way as in Section II-D. The average errors of pole distributions predicted by statAWE with respect to MC-AWE are 7.1%, 3.6%, 4.2%, and 7.2%, respectively, for the four poles (from top to bottom). Similar observations hold

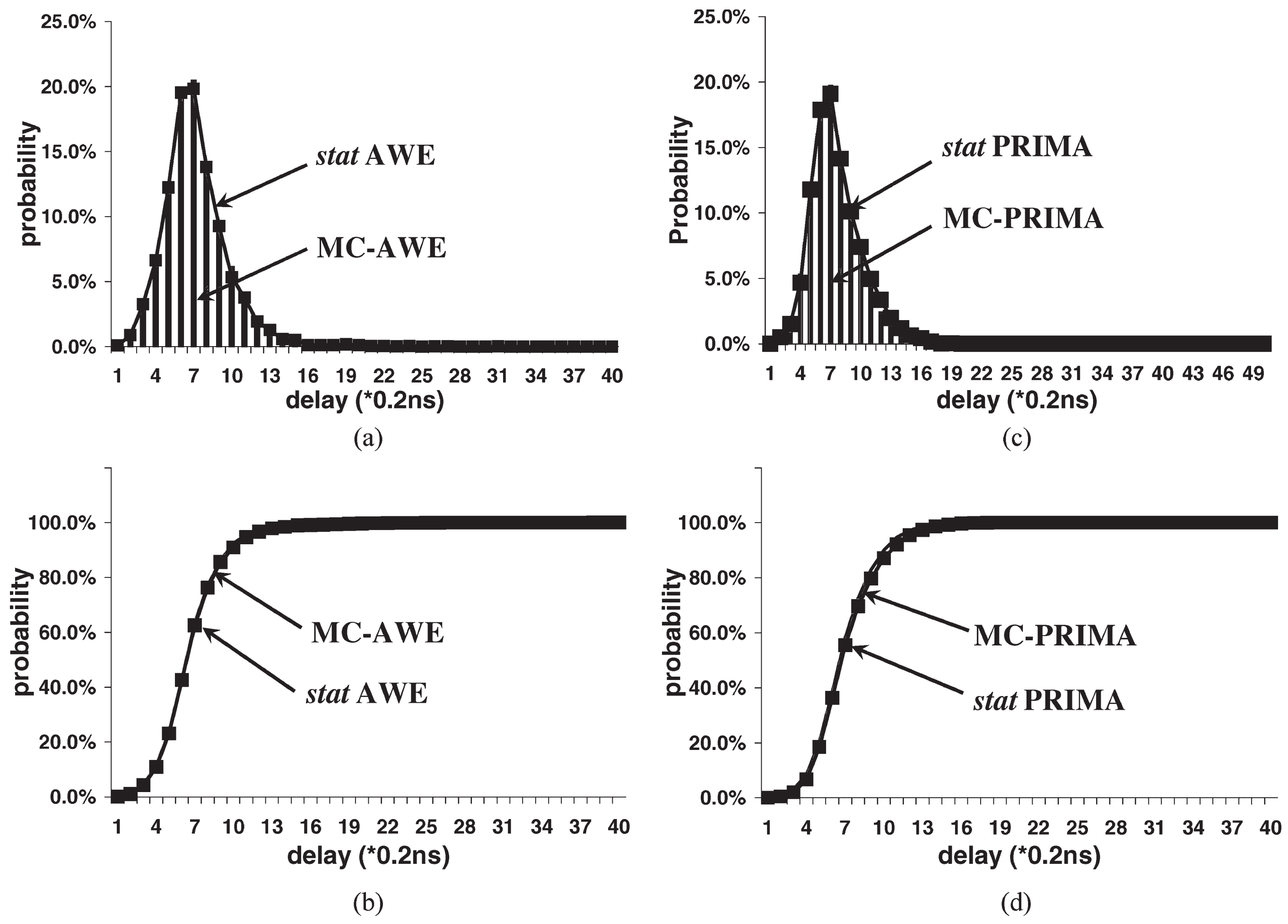


Fig. 12. Comparison between interval-valued approaches and MC simulation for design 1. (a) pdf: AWE (average error: 4.7%). (b) cdf: AWE. (c) pdf: PRIMA (average error: 6.3%). (d) cdf: PRIMA.

for PRIMA reduction. As is obvious from the figure, unstable right half-plane "false poles" as a result of AWE are pruned in the standard manner [17], [18], [20]. Assessing the stability/passivity of interval valued computation remains a challenging problem, though there are some earlier useful results for the simpler case of classical nonaffine intervals [32]. Fig. 12 plots the probability density function (pdf) and cumulative density function (cdf) for design 1 with 5% global variation, 30% local variation, and 11 initial uncertainty symbols. The pdf and cdf given by our interval-valued approaches match well with those given by MC simulation. For this test case, we also quantify the accuracy of our approaches in the following sense: For each MC sample over εs, the relative error is the difference between the delay result of our approach and that of straightforward MC simulation, normalized to the latter. Then, the average error of the delay distribution predicted by our approach is defined as the square root of the sum of the squared relative errors over all the MC samples, divided by the number of samples. The average error is 4.7% for statAWE [Fig. 12(a)] and 6.3% for statPRIMA [Fig. 12(c)]. Note that the distribution of 50% delay is asymmetrical, with a positive skew (mean > median) and a long tail at the far end of distribution histogram [see Fig. 12(a) and (c)]. Although each individual interval-valued quantity is, in our heuristic interpretation, modeled as a sum of normal random variables and thus normal itself, the final delay values are not simple intervals. They are nonlinear functions of intervals, which we evaluate via MC methods. These final nonlinear formulas are fairly compact, so the MC simulations are quick. But because of these critical, final nonlinearities, the delay pdf and cdf can have arbitrary, nonnormal form.

To further demonstrate the efficiency and effectiveness of the interval-valued statistical approaches, as well as the tradeoffs of interval- and scalar-valued computations, we compare the following four AWE-style approaches to statistical interconnect reduction.

1) statAWE proposed in this paper: Interval-valued tree formulation, interval-valued path tracing for interval-valued moments, and then MC sampling.
2) intAWE proposed in [17] and [18]: Interval-valued MNA formulation, LU-decomposition-based interval-valued computation for interval-valued poles and residues, and then MC sampling.
3) Interval-valued tree formulation, path-tracing-based interval-valued computation for interval-valued poles and residues, and then MC sampling.
4) Interval-valued MNA formulation, LU-decomposition-based interval-valued computation for interval-valued moments, and then MC sampling.

We test the above four approaches on the same combinations of global variation, local variation, and number of uncertainty terms for circuit design 1 as those in Table II. Note that for approaches 1) and 3), the mean delay and standard deviation errors are with respect to RICE-based MC simulations, and for approaches 2) and 4), the errors are with respect to Matrix Laboratory (MATLAB)-based MC simulations, using MNA formulation and LU decomposition. We use the same confidence interval techniques [31] to determine the proper number of samples for each test case. Fig. 13 plots the results of run time versus mean delay and standard deviation errors for all the approaches. We can see that all the errors are less than 10%, which proves the feasibility of our interval-valued approaches to statistical interconnect reduction. Furthermore, statAWE provides the best efficiency-accuracy tradeoff among all the four approaches: it achieves far less errors for both mean delay and standard deviation estimation than all the other approaches, while its run time is only slightly longer than that of approach 3) and much shorter than those of approaches 2) and 4). In particular, approach 1) (i.e., statAWE) achieves more than 3× reductions in both mean delay and standard deviation errors, and is an order of magnitude faster, compared to approach 2) (i.e., intAWE). Similar observations hold for PRIMA-style interval-valued statistical approaches. Again, as discussed in Section III-D, this is attributable to: 1) many fewer interval operations in interval-valued path tracing versus interval-valued LU decomposition and 2) smarter choices on where to switch from interval computation to scalar MC sampling—after model order reduction yet before highly nonlinear solve for the reduced system. Finally, it is worth pointing out that approach 3) (i.e., pervasive interval-valued computations based on path tracing) is also attractive, yielding the best efficiency, and accuracy to within 8%.

B. Results of Effective Capacitance Modeling

We have also implemented the two statistical C_{eff} modeling algorithms, denoted as var-iterC_{eff} and var-dirC_{eff}, respectively. They are benchmarked on design 1 and design 2 with the same settings of the manufacturing uncertainties as in Section V-A. We compare the mean and standard deviation of the distribution of variational C_{eff} (in pF) between our proposed approaches and MC simulations of scalar-valued C_{eff} modeling algorithms ([21] and [22]) using RICE 4 [20] function libraries (denoted as MC-iterC_{eff} and MC-dirC_{eff}, respectively). Again, for each case, we use MC samples sufficient to guarantee a 99% confidence level with 1% accuracy. (For almost all our experiments, this turns out to be between 5000 and 6000 samples.)

We use Taiwan Semiconductor Manufacturing Company (TSMC) 0.18 μm technology parameters [33] to fit the coefficients of (20) and (21), and to calculate R_{eff} in (22). Furthermore, to take into account process variations induced on the driving gate, we randomly, normally vary the threshold voltage, gate oxide thickness, W, and L of its transistors, as well as its input transition time, during both the complete MC simulations and the sampling steps in our approaches. Equations (20) and (21) were extended to accommodate these parameters, and R_{eff} was evaluated differently for each sample of these parameters according to (22). Obviously, it is more straightforward for the direct C_{eff} model than for the iterative C_{eff} model to consider variations of device parameters which are explicitly present in (22). And we choose 20% and 30% relative σ, respectively, for the variations induced on the gate. Tables IV and V show all the results.

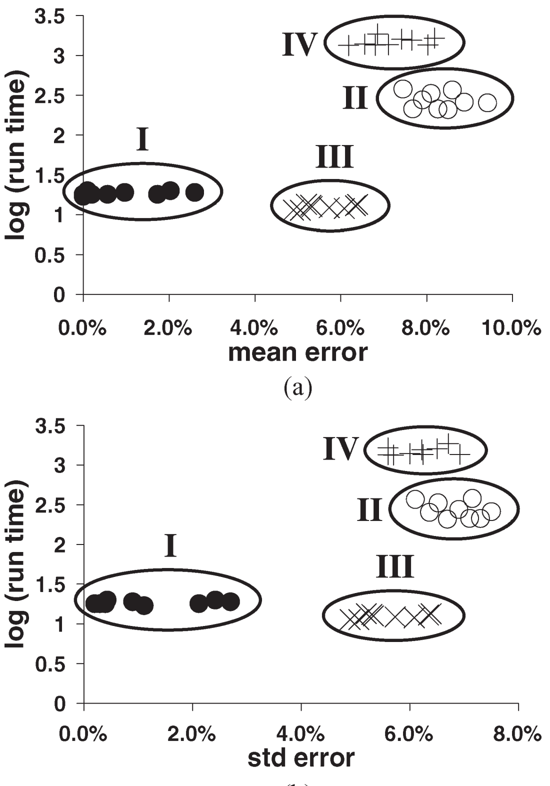


Fig. 13. Comparison among four approaches in terms of (a) run time and mean delay error and (b) run time and standard deviation error. For both figures, x axis is error and y axis is run time in logarithmic scale.

It is easy to see that both the mean C_{eff} and standard deviation errors for all the experiments are less than 4%. On average, the mean error is 0.8% for var-iterC_{eff} and 1.2% for for var-dirC_{eff}; the standard deviation error is 0.7% for var-iterC_{eff} and 0.9% for var-dirC_{eff}. The run time results are not shown in Tables IV and V. They are less than 4 s for all the runs of our proposed approaches, more than 400 s for all the MC-dirC_{eff} runs, and more than 500 s for all the MC-iterC_{eff} runs. Our proposed approaches achieve more than 100× run time speedup over the simple MC simulation. Stated differently, our interval-valued statistical C_{eff} modeling algorithms take about the same time as a brute-force MC simulation using just 50 samples.

VI. Conclusion

The affine interval model, which allows us to represent and preserve first-order correlations among range uncertainties, can be applied to classical model order reduction techniques for variational linear-interconnect analysis. A simple statistical interpretation of the resulting intervals allows us to statistically

TABLE IV
C_{eff} RESULTS, ASSUMING 20% VARIATIONS FOR GATE PARAMETERS AND INPUT TRANSITION TIME

1	2	3	4	5	6	7	8	9	10	11	12	13	14
global/local variation	number of ε	MC-*iter*C_{eff}		*var-iter*C_{eff}		**error**		MC-*dir*C_{eff}		*var-dir*C_{eff}		**error**	
		mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
						design1							
20% / 10%	6	2.21	0.511	2.23	0.512	0.9%	0.2%	0.974	0.337	0.969	0.336	0.5%	0.3%
	11	2.20	0.510	2.21	0.511	0.5%	0.2%	0.972	0.337	0.966	0.336	0.6%	0.3%
	21	2.19	0.509	2.20	0.511	0.5%	0.4%	0.973	0.337	0.968	0.336	0.5%	0.3%
10% / 20%	6	2.23	0.515	2.24	0.517	0.4%	0.4%	0.973	0.350	0.980	0.346	0.7%	1.1%
	11	2.20	0.510	2.21	0.512	0.5%	0.4%	0.964	0.336	0.949	0.334	1.6%	0.6%
	21	2.19	0.508	2.20	0.511	0.5%	0.6%	0.941	0.333	0.930	0.331	1.2%	0.6%
5% / 30%	6	2.35	0.533	2.37	0.539	0.9%	1.1%	1.08	0.352	1.10	0.358	1.9%	1.7%
	11	2.32	0.528	2.34	0.532	0.9%	0.8%	1.06	0.350	1.08	0.355	1.9%	0.3%
	21	2.33	0.528	2.36	0.535	1.3%	1.3%	1.05	0.346	1.08	0.352	2.7%	1.7%
						design2							
20% / 10%	6	6.21	1.69	6.26	1.77	0.8%	0.5%	10.6	3.39	10.8	3.44	1.9%	1.5%
	10	6.17	1.67	6.22	1.68	0.8%	0.6%	10.4	3.36	10.6	3.39	1.9%	0.9%
	19	6.16	1.67	6.19	1.68	0.5%	0.6%	10.4	3.38	10.3	3.35	1.0%	0.9%
10% / 20%	6	6.39	1.73	6.26	1.69	2.1%	2.4%	9.76	3.38	9.84	3.42	0.8%	1.1%
	10	6.28	1.69	6.21	1.68	1.1%	0.6%	9.71	3.37	9.84	3.39	1.3%	0.6%
	19	6.20	1.68	6.16	1.67	0.6%	0.6%	9.72	3.37	9.67	3.36	0.5%	0.3%
5% / 30%	6	6.68	1.90	6.60	1.93	1.2%	1.6%	10.7	3.51	10.6	3.51	0.9%	0.0%
	10	6.64	1.84	6.59	1.83	0.8%	0.5%	10.5	3.48	10.6	3.49	1.0%	0.3%
	19	6.51	1.82	6.44	1.81	1.1%	0.5%	10.2	3.43	10.3	3.45	1.0%	0.6%

TABLE V
C_{eff} RESULTS, ASSUMING 30% VARIATIONS FOR GATE PARAMETERS AND INPUT TRANSITION TIME

1	2	3	4	5	6	7	8	9	10	11	12	13	14
global/local variation	number of ε	MC-*iter*C_{eff}		*var-iter*C_{eff}		**error**		MC-*dir*C_{eff}		*var-dir*C_{eff}		**error**	
		mean	std	mean	std	mean	std	mean	std	mean	std	mean	std
						design1							
20% / 10%	6	2.21	0.512	2.23	0.514	1.0%	0.4%	1.04	0.346	1.06	0.350	0.2%	0.1%
	11	2.20	0.511	2.23	0.514	1.4%	0.6%	1.03	0.346	1.06	0.349	2.9%	0.9%
	21	2.20	0.510	2.22	0.513	0.9%	0.6%	1.05	0.349	1.06	0.349	1.0%	0.0%
10% / 20%	6	2.23	0.514	2.25	0.516	0.9%	0.4%	1.03	0.348	1.04	0.350	1.0%	0.6%
	11	2.21	0.512	2.22	0.513	0.5%	0.2%	1.03	0.346	1.04	0.347	1.0%	0.3%
	21	2.21	0.512	2.22	0.513	0.5%	0.2%	1.03	0.345	1.02	0.343	1.0%	0.6%
5% / 30%	6	2.38	0.535	2.38	0.537	0.0%	0.4%	1.15	0.361	1.18	0.367	2.6%	1.7%
	11	2.36	0.532	2.36	0.533	0.0%	0.2%	1.13	0.360	1.17	0.365	3.5%	1.4%
	21	2.34	0.529	2.36	0.533	0.9%	0.8%	1.14	0.356	1.17	0.365	2.6%	2.5%
						design2							
20% / 10%	6	6.22	1.68	6.29	1.70	1.1%	1.2%	11.2	3.49	11.2	3.55	0.0%	1.7%
	10	6.20	1.67	6.26	1.68	1.0%	0.6%	11.0	3.44	10.9	3.53	0.9%	2.6%
	19	6.17	1.67	6.30	1.69	2.1%	1.2%	11.0	3.44	10.9	3.53	0.9%	2.6%
10% / 20%	6	6.36	1.76	6.31	1.78	0.8%	1.1%	10.6	3.50	10.5	3.48	0.4%	0.5%
	10	6.26	1.68	6.25	1.68	0.2%	1.2%	10.5	3.48	10.5	3.47	0.0%	0.3%
	19	6.19	1.67	6.26	1.68	1.1%	0.6%	10.4	3.48	10.3	3.45	1.0%	0.9%
20% / 10%	6	6.72	1.88	6.62	1.87	1.5%	0.5%	11.2	3.58	11.3	3.60	0.9%	0.6%
	10	6.70	1.85	6.63	1.84	1.0%	0.5%	11.1	3.56	11.1	3.57	0.0%	0.3%
	19	6.56	1.83	6.55	1.82	0.2%	0.5%	10.7	3.51	10.9	3.54	1.9%	0.9%

sample the small, reduced model and estimate the delay distribution for the original variational interconnect circuit. We have extended our paper in [17] and [18] by developing more efficient and accurate interval-valued path tracing algorithms, and have answered a key question posed in [17] and [18] on the optimal "boundary" between where it is advantageous to compute with intervals, and where it is more accurate yet still efficient enough to stop and sample the intervals to produce a set of scalar-valued models. Our new algorithms are roughly 10× faster than our first generation efforts, yet still able to achieve accuracies to within 5% of classical MC simulation. We have further extended the interval analysis strategy to build statistical effective capacitance models for variational interconnect, with over 100× speedup and less than 4% errors compared to classical MC simulation. It is still an open question which, among the many competing strategies being pursued for efficient and accurate variational analysis for circuits and systems will be the winner. Based on the results in this paper, we believe that interval-valued models will find a practical and useful role in this important and evolving new area.

ACKNOWLEDGMENT

The authors would like to thank C. Fang of Carnegie Mellon University for many useful insights into affine arithmetic and L. Pileggi and X. Li of Carnegie Mellon University for providing the RICE 4/5 software package and many useful insights into fast model reduction. They also thank S. Nassif of International Business Machines Corporation for helpful discussions on the direct C_{eff} model and the anonymous reviewers for valuable comments.

References

[1] A. Devgan and C. Kashyap, "Block-based static timing analysis with uncertainty," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2003, pp. 607–614.

[2] S. Bhardwaj, S. B. Vrudhula, and D. Blaauw, "τAU: Timing analysis under uncertainty," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2003, pp. 615–620.

[3] H. Chang and S. Sapatnekar, "Statistical timing analysis considering spatial correlations using a single PERT-like traversal," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2003, pp. 621–625.

[4] C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan, "First-order incremental block-based statistical timing analysis," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 331–336.

[5] J. Le, X. Li, and L. T. Pileggi, "STAC: Statistical timing analysis with correlation," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 343–348.

[6] J. Y. Lee, X. Huang, and R. A. Rohrer, "Pole and zero sensitivity calculation in asymptotic waveform evaluation," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 11, no. 5, pp. 586–597, May 1992.

[7] Y. Liu, L. T. Pileggi, and A. J. Strojwas, "Model order-reduction of RC(L) interconnect including variational analysis," in *Proc. Design Automation Conf.*, New Orleans, LA, 1999, pp. 201–206.

[8] L. Daniel, O. C. Siong, L. S. Chay, K. H. Lee, and J. White, "Multi-parameter moment-matching model-reduction approach for generating geometrically parameterized interconnect performance models," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 23, no. 5, pp. 678–693, May 2004.

[9] J. M. Wang, P. Ghanta, and S. Vrudhula, "Stochastic analysis of interconnect performance in the presence of process variations," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2004, pp. 880–886.

[10] K. Agarwal, D. Sylvester, D. Blaauw, F. Liu, S. Nassif, and S. Vrudhula, "Variational delay metrics for interconnect timing analysis," in *Proc. Design Automation Conf.*, San Diego, CA, 2004, pp. 381–384.

[11] X. Li, J. Le, P. Gopalakrishman, and L. T. Pileggi, "Asymptotic probability extraction for non-normal distributions of circuit performance," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2004, pp. 2–9.

[12] C. L. Harkness and D. P. Lopresti, "Interval methods for modeling uncertainty in RC timing analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 11, no. 11, pp. 1388–1401, Nov. 1992.

[13] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 9, no. 4, pp. 352–366, Apr. 1990.

[14] K. J. Kerns and A. T. Yang, "Stable and efficient reduction of large, multiport RC networks by pole analysis via congruence transformations," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 16, no. 7, pp. 734–744, Jul. 1997.

[15] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 17, no. 8, pp. 645–654, Aug. 1998.

[16] J. Stolfi and L. H. de Figueiredo, "Self-validated numerical methods and applications," in *Brazilian Mathematics Colloquium Monograph*. Rio De Janeiro, Brazil: IMPA, 1997.

[17] J. D. Ma and R. A. Rutenbar, "Interval-valued reduced order statistical interconnect modeling," in *Proc. Int. Conf. Computer Aided Design*, San Jose, CA, 2004, pp. 460–467.

[18] ——, "Interval-valued reduced order statistical interconnect modeling," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.* [Online]. Available: http://ece.cmu.edu/~dazhuanm/docs/tcad_intmor.pdf

[19] ——, "Fast interval-valued statistical interconnect modeling and reduction," in *Proc. Int. Symp. Physical Design*, San Jose, CA, 2005, pp. 159–166.

[20] C. L. Ratzlaff and L. T. Pillage, "RICE: Rapid interconnect circuit evaluation using AWE," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 6, pp. 763–776, Jun. 1994.

[21] J. Qian, S. Pullela, and L. Pillage, "Modeling the 'effective capacitance' for the RC interconnect of CMOS gates," *IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst.*, vol. 13, no. 12, pp. 1526–1535, Dec. 1994.

[22] S. Nassif and Z. Li, "A more effective C_{eff}," in *Proc. Int. Symp. Quality Electronics Design*, San Jose, CA, 2005, pp. 648–653.

[23] A. Odabasioglu and L. T. Pileggi, "RICE 5: Rapid interconnect circuit evaluation version 5," in *Reference Manual*. Pittsburgh, PA: Carnegie Mellon Univ., 1997.

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

[25] C. F. Fang, "Probabilistic interval-valued computation: Representing and reasoning about uncertainty in DSP and VLSI designs," Ph.D. dissertation, Dept. Elect. Comp. Eng., Carnegie Mellon Univ., Pittsburgh, PA, 2005.

[26] S. Nassif, "Modeling and analysis of manufacturing variations," in *Proc. Custom Integrated Circuits Conf.*, San Diego, CA, 2001, pp. 223–228.

[27] V. Mehrotra, S. Nassif, D. Boning, and J. Chung, "Modeling the effects of manufacturing variation on high-speed microprocessor interconnect performance," in *Proc. IEEE Int. Electron Devices Meeting*, San Francisco, CA, 1998, pp. 767–770.

[28] P. R. O'Brien and T. L. Savarino, "Modeling the driving-point characteristic of resistive interconnect for accurate delay estimation," in *Proc. Int. Conf. Computer Aided Design*, Santa Clara, CA, 1989, pp. 512–515.

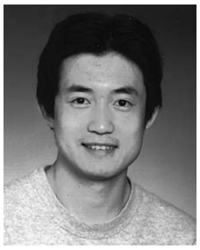
[29] N. H. E. Weste and K. Eshraghian, *Principles of CMOS VLSI Design: A System Perspective*. Reading, MA: Addison-Wesley, 1993.

[30] Y. Liu, S. R. Nassif, L. T. Pileggi, and A. J. Strojwas, "Impact of interconnect variations on the clock skew of a gigahertz microprocessor," in *Proc. Design Automation Conf.*, Los Angeles, CA, 2000, pp. 168–171.

[31] R. Burch, F. N. Najm, P. Yang, and T. N. Trick, "A Monte Carlo approach for power estimation," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 1, no. 1, pp. 63–71, Mar. 1993.

[32] L. V. Kolev, *Interval Methods for Circuit Analysis*. Singapore: World Scientific, 1993.

[33] [Online]. Available: http://www.tsmc.com/ and [Online]. Available: http://www-device.eecs.berkeley.edu/~bsim3/



James D. Ma received the B.S. degree in electrical engineering from Fudan University, Fudan, China, in 2000, and the M.S. degree in computer engineering from the University of Wisconsin, Madison, in 2002. He is currently working toward the Ph.D. degree in computer engineering from the Department of Electrical and Computer Engineering, Carnegie Mellon University (CMU), Pittsburgh, PA.

He was a summer intern with Synopsys, in 2003, and an intern with Cadence Berkeley Labs, from 2005 to 2006. His research interests include variational analysis, design-for-manufacturing, and physical design automation.

Rob A. Rutenbar (S'77–M'84–SM'90–F'98) received the Ph.D. degree in computer engineering from the University of Michigan, Ann Arbor, in 1984.

He subsequently joined the faculty at Carnegie Mellon University (CMU), Pittsburgh, PA, where he is currently the Stephen J. Jatras Professor of electrical and computer engineering and (by courtesy) computer science. His research pioneered much of today's commercial analog circuit and layout synthesis technology. From 1992 to 1996, he chaired the Analog Technical Advisory Board for Cadence Design Systems. He cofounded Neolinear Inc., in 1997, and served as its Chief Scientist until its acquisition by Cadence in 2004. He is the founding Director of the Microelectronics Advanced Research Corporation (MARCO) Focus Center for Circuit and System Solutions (called "C2S2"), a consortium of American universities chartered by the U.S. semiconductor community and U.S. government to address future circuit design challenges. His research interests include circuit and layout synthesis algorithms for mixed-signal application-specified integrated circuits (ASICs) and for large digital ICs.

Dr. Rutenbar received a Presidential Young Investigator Award from the National Science Foundation in 1989. He has won Best/Distinguished Paper Awards from the Design Automation Conference in 1987 and 2002, the International Conference on CAD in 1991, and the Semiconductor Research Corporation TECHCON in 1993, 2000, 2003, and 2005. He has been on the program committees for the IEEE International Conference on CAD, the Association of Computing Machinery (ACM)/IEEE Design Automation Conference, the ACM International Symposium on FPGAs, and the ACM International Symposium on Physical Design. He was General Chair of the 1996 International Conference on Computer Aided Design (ICCAD). He also served on the Editorial Board of IEEE Spectrum. He is a 2001 winner of the Semiconductor Research Corporation (SRC) Aristotle Award for excellence in education and 2004 cowinner of the SRC Technical Excellence Award for contributions to tools for analog circuit design. In 2002, he was honored with a University of Michigan Alumni Society Merit Award. He is a member of the ACM and Eta Kappa Nu.