Library Compatible Variational Delay Computation

Luis Guerra e Silva[1], Zhenhai Zhu[2], Joel R. Phillips[2], and L. Miguel Silveira[1]

[1] Cadence Laboratories / INESC-ID
IST / Tech. U. Lisbon
Lisbon, Portugal
{lgs,lms}@inesc-id.pt
[2] Cadence Berkeley Laboratories
Cadence Design Systems
San Jose, CA 95134, U.S.A.
{zhzhu,jrp}@cadence.com

Abstract. With technology steadily progressing into nanometer dimensions, precise control over all aspects of the fabrication process becomes an area of increasing concern. Process variations have immediate impact on circuit performance and behavior and standard design and signoff methodologies have to account for such variability. In this context, timing verification, already a challenging task due to the sheer complexity of todays designs, becomes an increasingly difficult problem. Statistical static timing analysis has been proposed as a solution to this problem, but most of the work has focused in the development of timing engines for computing delay propagation. Such tools rely on the availability of delay formulas accounting for both cell and interconnect delay that take into account unpredictable variability effects. In this paper, we concentrate on the impact of interconnect on delay and propose an extension to the standard modeling strategies that is variation-aware and compatible with such statistical engines. Our approach, based on a specific type of perturbation analysis, allows for the analytical computation of the quantities needed for statistical delay propagation. We also show how perturbation analysis can be performed when only the standard delay table lookup models are available for the standard cells. This makes the proposed approach compatible with existing timing analysis frameworks. Results from applying our proposed modeling strategy to computing delays and slews in several instances accurately match similar results obtained using electrical level simulation.

1 Introduction

The impact of process variation on circuit performance is an area of increasing concern, both in the semiconductor industry, as well as academic research. In the research community, considerable work has been devoted to the development of statistical static timing analysis [1, 2]. Nowadays, designers spend a considerable amount of their verification budget trying to make sure that their circuits will

work under all possible settings. To achieve this, they target the worst possible scenarios by considering so-called pessimistic conditions, and design in order to ensure that such corner cases are accounted for. This analysis is usually based on assuming worst-case conditions on all possible variations simultaneously. Such an scenario is pessimistic and may lead to considerable over-design.

Improving this situation requires tools that are better suited to handle realistic variations and the complex inter-relations that exist between those variations. Not only should those tools directly make use of realistic process information, thus making them better suited to model the unpredictability of process parameter variations, but they should be able to implicitly determine how such variations affect the circuit behavior. Such a formulation makes it possible to compute on a single analysis the circuit behavior not only due to a given parameter setting, but to a variety of settings. The recent development and availability of statistical timers that are based on a parametric description of delay in terms of random process variables is an example of movement in this direction [1]. Other approaches targeting direct determination of the worst parameter settings with respect to delay also follow the same trend [3].

A timing analyzer consists of several component pieces. In a statistical context, the most well-studied part of the timing engine is the timing graph traversal, which manages the calculation of arrival times and slews at the level of abstraction of a timing graph. An equally important, if more mundane, component is the delay calculation engine. The delay calculator takes as input the cell and interconnect models and produces a delay expression in a form that can be consumed by the graph engine. This paper is concerned with a portion of the delay calculation step, the impact of interconnect on delay. We explore how commonly used interconnect modeling strategies can be extended to be compatible with the most recent generation of statistical timing analysis tools [1]. Specifically, we wish to produce cell and interconnect delay as affine functions of process parameters. We assume that one of several recently proposed approaches for interconnect reduction under process variation is available to generate tractably sized reduced order models [4–6]. The key technology in our approach is a specific type of perturbation analysis. While digital circuits are strongly nonlinear with respect to the circuit inputs, cell delays are often close to linear *with respect to process parameters*. In this paper we adapt the general development of *linear time-varying* (LTV) perturbation theory [7, 8] for extraction of variation-aware delay models to the specific needs of delay calculation for precharacterized standard cells. LTV perturbation theory has been widely used in RF analysis with great success [9] and is at the heart of many interesting new developments. The advantage of this type of approach over, for example, differencing repeated delay calculation runs is that it is essentially an analytic method. Differencing type approaches can suffer from severe robustness problems that make them difficult to use reliably. In addition, our technique can potentially be made very fast, handling parametric models with ten to twenty parameters at minimal penalty relative to a non-variational calculation.

The outline of this paper is as follows: in Section 2, we review the basics of delay computation, the general mechanics of the procedure including cell and interconnect delay, assuming no variations are taken into account. Then, in Section 3, we introduce the general perturbation formulation and discuss the specific specialization of the more general technique to cell-level interconnect-related delay. In Section 4, we also discuss how perturbation analysis can be performed when only delay table lookup models are available for the standard cells. A key point is that *analytic* expressions for delay sensitivities can be obtained without having to have closed-form expressions for the cell delay elements (however, see [10] for such closed-form expressions). Finally in Section 5 we discuss the utilization of adjoint methods to accelerate the computation of timing models when large numbers of parameters are present. Results of using our proposed approach are shown in Section 6 and conclusions are drawn in Section 7.

2 Nominal Delay Calculation

Timing verification is an enabling methodology for optimizing performance and ensuring that circuits satisfy certain timing and frequency requirements. To that end, timing verifiers determine approximate but safe estimates of the worst-case delay through a circuit: for every input and output signal, there are many possible paths through the circuit, each path consisting of a set of interconnected network cells. Timing verification deals with the identification and analysis of the critical paths, also known as the longest delay paths in the circuit. In addition to finding critical-path delays, timing verifiers can also be used to do miscellaneous static analysis, like finding high-speed components off the critical path that can be slowed down to save power and several other relevant tasks. However, the most common usage is indeed to determine the worst case paths. Computing the delay along a path requires the computation of the delay of every cell along that path, plus the added delay due to interconnect between the cells. In this section we review the standard computation of cell and interconnect delay.

2.1 Mechanics of Delay Computation

Timing analysis constitutes the foundation of any timing verification methodology. The typical timing analysis methodology consists in *arrival time* computation, which is concerned with computing the time instants at which signal transitions reach "interesting" nodes in the circuit, often corresponding to primary outputs or register inputs, where specific timing constraints must be enforced.

Two main approaches have been proposed for timing analysis: block-based and path-based. In the block-based approach, characterized by linear runtime, arrival times are pushed through the circuit in a levelized fashion, performing sum operations with cell or interconnect delays and min/max operations to compute the arrival times in the outputs of multi-input cells, assuming that the earliest/latest input transition determines the output transition. The alternative path-based approach consists in individually computing the delay of each

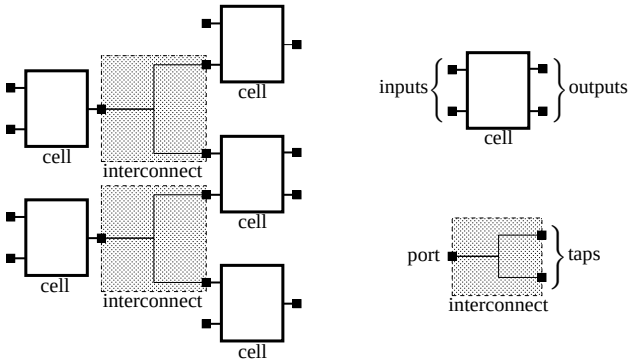

Fig. 1. Typical partition of a digital IC topology for timing analysis.

path in the circuit by adding the delay of all the cells and interconnect along that path. Even though more accurate, this approach is computationally much more expensive than the former, since the number of paths is known to grow exponentially with the number of nodes. Clearly any timing analysis approach requires the computation of cell and interconnect delays.

For timing analysis purposes, the digital IC topology is usually partitioned into cells and interconnect nets, as illustrated in Figure 1. Primary inputs and outputs are usually represented by the corresponding *pads*, which are a particular type of cells. Cell input and output pins are connected by interconnect nets. Each interconnect net can be seen as distributing the signal injected in its input, designated by *port*, to each of its outputs, designated by *taps*, that are connected to cell input pins. For typical cell and interconnect delay models, the slew of the input signal(s) is a required parameter. Accordingly, the slew of the output signal(s) is a result produced by the model. Therefore, once the circuit is properly partitioned and all the cell and interconnect delay models are in place, the task of the delay computation engine is to forward propagate the slews and invoke the appropriate delay models that will compute delays and output slews given the input slews and output loads.

2.2 Cell Delay and Cell Loading

Mainly for historical reasons, the most common modeling strategy for cell library characterization is based in delay look-up tables (LUTs) sometimes referred to as *dot-lib (.lib)* tables. This is a simplified model where delay and power information is maintained in the form of a few parameters. In this simplified model the timing behavior of a cell is usually characterized by a set of lookup tables that, for each input/output pin pair, describe the delay and output slew of the cell as a function of the input slew and output load. Such a model is illustrated in Figure 2 where the standard definitions are also used, namely input and output slews are defined as $s = t_H - t_L$, where t_L and t_H are the time instants at which the respective voltage waveform reaches some pre-defined values, V_L and V_H, related to the

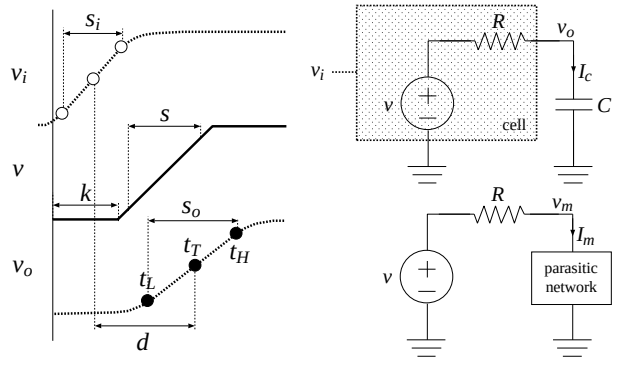

Fig. 2. Voltage source based cell model, loaded by the effective capacitance (top right), and by the parasitic network (bottom right) and corresponding waveforms (left).

definition of noise margins. In a similar manner, delay is defined as the time it takes the output of a cell to reach its transition midpoint, from the time the cell input waveform reached its own midpoint. Cell characterization is performed by simulating the cell behavior as a function of input slew and loading capacitances. These results are then stored in look-up tables as mentioned, which are accessed to determine delay and slew in specific instances.

The outlined delay modeling strategy assumes a voltage source model for the cell characterization, as illustrated in Figure 2, since delay and slew values implicitly characterize the output voltage waveforms of the cell. However, in recent years, current source models are gaining more prominence, since they are more effective in handling complex interconnect loading effects. Even though throughout this paper we assume voltage source delay models, the proposed techniques can also be directly applied when using current source delay models.

In Figure 2, the output load is assumed to be a single lumped capacitance that somehow models the capacitive effects introduced by the interconnect and by the input pins of the cells connected to same net. In reality, however, the interconnect attached to the driver cell is a complex RC network that in deep submicron processes is very poorly modeled by a lumped capacitance. The loading effect of interconnect on the cell, i.e. the impact of downstream interconnect on the cell delay itself, cannot be accurately obtained simply by looking at the total capacitance on the net. To try to account for the effects of complex interconnect, while still preserving table-based cell models, the concept of *effective capacitance* [11, 10] has been widely adopted. For the remainder of this paper we will consider that the C shown in Figure 2 is such an effective capacitance.

The idea behind the effective capacitance consists in determining the value of C that in a certain sense approximates as accurately as possible the behavior of the original parasitic network. In Figure 2. the output stage of a cell (or more accurately, of an output pin of a cell) is modeled by a voltage source, producing a voltage ramp v, with slew s, and a series resistor, with resistance R, that models the output resistance of the pin. The figure depicts the output stage of a cell

loaded by the effective capacitance C (top right), and by the original parasitic RC network, obtained by layout extraction (bottom right). In the following, without loss of generality, in order to simplify the description, we restrict ourselves to the case of rising output waveforms for non-inverting cells. Clearly any other case can be derived in a similar manner.

The simple RC circuit on the top of Figure 2 is an approximated model of the output stage of a cell connected to an effective capacitance, that is itself an approximation of the interconnect load. For a given input slew s_i and a given effective capacitance C, we can compute the estimated cell delay d and the estimated output slew s_o, by a table lookup in the timing characterization of the cell. Using this information, we can compute the three time instants at which the waveform of the output voltage v_o should cross V_L, V_T and V_H, respectively,

$$t_L = \frac{s_i}{V_H - V_L} V_T + d - \frac{s_o}{V_H - V_L}(V_T - V_L) \tag{1}$$

$$t_T = \frac{s_i}{V_H - V_L} V_T + d \tag{2}$$

$$t_H = \frac{s_i}{V_H - V_L} V_T + d + \frac{s_o}{V_H - V_L}(V_H - V_T) \tag{3}$$

Assuming the voltage v to be a rising ramp of slew s, shifted in time by k,

$$v(s,k,t) = \begin{cases} 0 & if \;\; 0 \le t < k \\ \frac{V_H - V_L}{s} t & if \;\; k \le t < k + \frac{sV_{DD}}{V_H - V_L} \\ V_{DD} & if \;\; t \ge k + \frac{sV_{DD}}{V_H - V_L} \end{cases} \tag{4}$$

the output voltage, v_o, produced by the simple RC circuit presented in Figure 2, with time constant $\tau = RC$, is given by,

$$v_o(s,k,\tau,t) = \begin{cases} 0 & if \;\; 0 \le t < k \\ \frac{V_H - V_L}{s}(-\tau + t - k + \tau e^{-\frac{t-k}{\tau}}) & if \;\; k \le t < k + \frac{sV_{DD}}{V_H - V_L} \\ V_{DD} - \frac{V_H - V_L}{s}(e^{\frac{sV_{DD}}{V_H - V_L}} - 1)\tau e^{-\frac{t-k}{\tau}} & if \;\; t \ge k + \frac{sV_{DD}}{V_H - V_L} \end{cases} \tag{5}$$

In order to simplify our notation, in the following we will assume,

$$\phi = \langle s, k, R, C \rangle. \tag{6}$$

Using Eqn. (5), we can compute a waveform for v (e.g. s and k) and a resistance R, such that the waveform of the response v_o crosses (t_L, V_L), (t_T, V_T) and (t_H, V_H), thus matching the tabulated behavior of the cell and its output response. This problem can be stated by the following three equations,

$$v_o(t_L, \phi) = V_L \tag{7}$$

$$v_o(t_T, \phi) = V_T \tag{8}$$

$$v_o(t_H, \phi) = V_H \tag{9}$$

The waveform of v can be seen as the "ideal" output voltage of the cell, under a zero output load. We should not lose track of the fact that our goal is to

determine an appropriate value for the effective capacitance C. The previous derivations assumed that such a value was somehow known. However, all that is required is that C should approximate the behavior of the original parasitic network as accurately as possible. Several criteria [12] can be used when defining what effective capacitance provides a good approximation of the behavior of the original parasitic network. In this work we consider that the effective capacitance that better approximates the behavior of the original parasitic network is the one that draws the same average current, over the transition period (e.g. when the output voltage switches from V_L to V_H). Formally,

$$\langle I_c \rangle = \langle I_m \rangle \Leftrightarrow \frac{1}{s_o} \int_{t_L}^{t_H} I_c \, dt = \frac{1}{t_H^{'} - t_L^{'}} \int_{t_L^{'}}^{t_H^{'}} I_m \, dt \tag{10}$$

where $v_m(t_L^{'}) = V_L$ and $v_m(t_H^{'}) = V_H$. An analytical expression for $\langle I_c \rangle$ can be derived. On the other hand, $\langle I_m \rangle$ must be computed by numerically integrating the port current, obtained by interconnect simulation, as detailed in Section 2.3.

From Eqns. (7), (8), (9), and (10) we can compute the value of ϕ that both matches the output waveform v_o with the tabulated timing information at t_L, t_T and t_H, and also that matches the average current drawn by the original parasitic network and the effective capacitance. Since Eqns. (7), (8), (9) and (10) contain nonlinear terms, an implicit iterative method must be used to solve them. We have used Newton's method in this work. Once the value of the effective capacitance C is known, we can compute the delay d and output slew s_o of the cell by a simple lookup in the timing characterization of the cell. This completely characterizes the cell output waveform within the constraints of the simple model. Such a waveform constitutes the input to the interconnect model.

2.3 Interconnect Delay

Assuming that the cell output voltage waveform has been computed, signals are then propagated along the path through an interconnect net. The input of such nets, the port, is tied to the output of a cell, and the net outputs, the taps, connect to the inputs of several other cells. At the timing level, the difference in the timing of the transition at the cell output (port) and next cell inputs (taps) we refer to as intrinsic interconnect delay. There are various methods of computing the interconnect delay ranging from closed-form expressions, descendants of the Elmore delay formula, to numerical solution of the underlying interconnect equations. In this work we assume that the circuit equations of the cell driver plus interconnect network are solved numerically, either via direct integration or an equivalent process like recursive convolution. Likewise the slew at the output nodes must be computed to be used in the analysis of the following cell.

The general state-space representation of a parasitic RC network (either in its original of reduced form) is

$$C\frac{d}{dt}x(t) + Gx(t) = u(t) \tag{11}$$

$$y(t) = N^T x(t) \tag{12}$$

where $x \in \mathbb{R}^n$ is the vector of circuit state variables, u is the input excitation, y is the output response, C and G are the matrices describing the reactive (capacitances) and dissipative (conductances) parts of the circuit and N selects the output response.

Assuming a cell characterization in terms of voltage source models, as illustrated in Figure 2, the input excitation is the voltage waveform, v_m, and the output response are the voltage waveforms in the taps, v_{tap}. Therefore, we have,

$$u(t) = Bv_m \tag{13}$$
$$v_{tap}(t) = L^T x(t) \tag{14}$$

where B is a matrix describing the node where the input voltage is injected, and L is an incidence-type matrix describing which voltage nodes are monitored (taps). In the particular case of voltage source models, the current drawn by the parasitic network, I_m, is also relevant, both for computing the effective capacitance and the input voltage waveform. Hence, an additional equation should be added,

$$I_m(t) = M^T x(t) \tag{15}$$

where M selects the output current out of the state vector x.

3 Variation-Aware Methodology

3.1 General Perturbation Formulation

In this section, we will discuss the parametric analysis of the intrinsic interconnect delay itself. The impact of the interconnect parameters on the cell delay (i.e. variation in cell loading effects) is taken up in the next section.

The starting point of our analysis is the general formulation of time-varying linear perturbation theory (see [8] for details). We assume the existence of a set of nonlinear differential-algebraic equations whose topology is fixed, but whose constitutive relations depend on a continuous way on a set of parameters. Without loss of generality the basic circuit equations can be written as

$$\frac{d}{dt}q(x, \lambda) + i(x, \lambda) = u(t) \tag{16}$$

where x again represents the circuit state variables, for example, node voltages, $q \in \mathbb{R}^n$, the dynamic quantities such as stored charge, $i \in \mathbb{R}^n$, the static quantities such as device currents, t, time, and $u(t) \in \mathbb{R}^n$, the independent inputs such as current and voltage sources. In departure from the usual case, we introduce a p-element parameter vector $\lambda \in \mathbb{R}^p$. These parameters represent properties of the circuit, such as wire width or thickness, that induce variation in the circuit behavior through the q and i functions.

The perturbation approach to modeling the parameter variation treats the parameters as fluctuations $\Delta\lambda$ around a nominal value λ_0, and assumes the

circuit response x can be treated similarly, i.e.

$$\lambda = \lambda_0 + \Delta\lambda \tag{17}$$
$$x(t) = x_0(t) + \Delta x(t). \tag{18}$$

Expanding i and q as function of x, λ and keeping the first order variations, we get

$$q(x, \lambda) = q(x_0, \lambda_0) + \frac{\partial q}{\partial \lambda}\Delta\lambda + \frac{\partial q}{\partial x}\Delta x \tag{19}$$
$$i(x, \lambda) = i(x_0, \lambda_0) + \frac{\partial i}{\partial \lambda}\Delta\lambda + \frac{\partial i}{\partial x}\Delta x. \tag{20}$$

Assuming a solution to the nominal case, $x_0(t)$ is obtained, that is

$$\frac{d}{dt}q(x_0, \lambda_0) + i(x_0, \lambda_0) = u(t) \tag{21}$$

then substituting the perturbation expansions (19) and (20) into Eqn. (16) and using (21) to eliminate the nominal-case terms, we obtain the equations for the first-order perturbation expansion as

$$\frac{d}{dt}\left[\frac{\partial q}{\partial x}\Delta x\right] + \frac{\partial i}{\partial x}\Delta x = -\left[\frac{d}{dt}(\frac{\partial q}{\partial \lambda})\Delta\lambda + \frac{\partial i}{\partial \lambda}\Delta\lambda\right] \tag{22}$$

The simplest way to compute waveform sensitivities from Eqn. (22) is by solving it once for each parameter in turn, as

$$\text{for each } k: \quad \frac{d}{dt}\left[\frac{\partial q}{\partial x}\frac{\partial x}{\partial \lambda_k}\right] + \frac{\partial i}{\partial x}\frac{\partial x}{\partial \lambda_k} = -\left[\frac{d}{dt}(\frac{\partial q}{\partial \lambda_k}) + \frac{\partial i}{\partial \lambda_k}\right]. \tag{23}$$

This gives the final expression

$$x(t, \lambda) = x_0(t) + \sum_{k=1}^{p} \frac{\partial x}{\partial \lambda_k}(t)\Delta\lambda_k. \tag{24}$$

Once the sensitivities in the waveforms are known, the next step is to translate to sensitivity of delay. As discussed, delay can be computed as $d = t_2 - t_1$ where t_2, t_1 are the crossing times of the two waveforms of interest. The sensitivity in a crossing time can be related to the sensitivity of the waveform value $x(t)$ at that point via the slew, $\partial x/\partial t$. Suppose there is a small change ΔT in the crossing time of a given waveform. With a linear model, the corresponding change in the voltage is

$$\Delta X = \frac{\partial x}{\partial t}\Delta T. \tag{25}$$

Conversely, if the perturbation in the waveform ΔX can be computed, the change in crossing time is given by

$$\Delta T = \frac{\Delta X}{\frac{\partial x}{\partial t}}. \tag{26}$$

Therefore we can compute the sensitivity of the delay as

$$\frac{\partial d}{\partial \lambda_k} = \frac{\frac{\partial x}{\partial \lambda_k}\Big|_{t_2}}{\frac{\partial x}{\partial t}\Big|_{t_2,\lambda_0}} - \frac{\frac{\partial x}{\partial \lambda_k}\Big|_{t_1}}{\frac{\partial x}{\partial t}\Big|_{t_1,\lambda_0}} \tag{27}$$

Note that for this computation, the waveform sensitivity is only needed at a few points in time, a fact that can be used to speedup computations (see Section 5).

This is the formulation for a general first-order perturbation analysis. In the following we restrict ourselves to the problem at hand, namely modeling the linear interconnect sub-circuits assuming variations in parameters affecting the interconnect elements.

3.2 Specialization to Interconnect

Our concern in this document is with the special case of interconnect parameters, so simplifications of the general theory are possible. On-chip cell-level interconnect models are usually written in terms of capacitances and resistances, or equivalently, capacitances and conductances. Inductance is typically neglected at this level and for the sake of simplicity we will proceed likewise; it is however easy to see that the derivation is quite similar when inductance is involved. Therefore, in this case,

$$q(x,\lambda) = C(\lambda)x \qquad i(x,\lambda) = G(\lambda)x \tag{28}$$

so that

$$\frac{\partial k}{\partial \lambda_k} = \frac{\partial G}{\partial \lambda_k}x \qquad \frac{\partial q}{\partial \lambda_k} = \frac{\partial C}{\partial \lambda_k}x \tag{29}$$

Let us then assume, for now, that for every element in the parasitic network (resistor or capacitor), a linear variational model is available. Such a model contains the nominal values for the elements and also the sensitivities to each parameter. Therefore, the conductance and the capacitance matrices have the form:

$$G = G_0 + \sum_{k=1}^{p} \left(G_k \Delta\lambda_k\right), \qquad C = C_0 + \sum_{k=1}^{p} \left(C_k \Delta\lambda_k\right) \tag{30}$$

where G_0 and C_0 are the nominal values of the elements in the interconnect network and the sensitivities $\frac{\partial G}{\partial \lambda_k}$ and $\frac{\partial C}{\partial \lambda_k}$ to each parameter λ_k are given by

$$\frac{\partial G}{\partial \lambda_k} = G_k, \qquad \frac{\partial C}{\partial \lambda_k} = C_k. \tag{31}$$

The nominal value corresponds to the solution of the equations with each $\Delta\lambda_k = 0$, that is $\lambda = \lambda_0$. Assuming the variational formulation for G presented in Eqn. (30), and for x presented in Eqn. (18) we obtain, for instance for $i(x,\lambda)$:

$$i(x,\lambda) = \left[G_0 + \sum_{k=1}^{p} \left(G_k \Delta\lambda_k\right)\right] \left(x_0 + \Delta x\right) \tag{32}$$

Simplifying and eliminating the (non-linear) cross-product terms, we obtain:

$$i(x,\lambda) \approx G_0 x_0 + G_0 \Delta x + \sum_{k=1}^{p} (G_k x_0 \Delta\lambda_k) \tag{33}$$

implying that:

$$i_0 \equiv i(x_0, 0) = G_0 x_0, \quad \frac{\partial i}{\partial x} = G_0, \quad \frac{\partial i}{\partial \lambda_k} = G_k x_0. \tag{34}$$

An identical procedure can be applied to $q(x,\lambda)$ leading, as expected, to:

$$q(x,\lambda) \approx C_0 x_0 + C_0 \Delta x + \sum_{k=1}^{p} (C_k x_0 \Delta\lambda_k) \tag{35}$$

and therefore, that:

$$q_0 \equiv q(x_0, 0) = C_0 x_0, \qquad \frac{\partial q}{\partial x} = C_0, \qquad \frac{\partial q}{\partial \lambda_k} = C_k x_0 \tag{36}$$

Eqns. (21) and (22) which describe the general perturbation analysis framework, can therefore, in the specialization of parameter-varying interconnect, be written as:

$$C_0 \frac{d}{dt} x_0(t) + G_0 x_0(t) = u(t) \tag{37}$$

$$C_0 \frac{d}{dt} [\Delta x] + G_0 \Delta x = -\sum_{k=1}^{p} \left[\frac{d}{dt} (C_k x_0(t)) \Delta\lambda_k + G_k \Delta\lambda_k \right] \tag{38}$$

The delay modeling problem is completed by adding the notion of inputs and outputs to form state-space models. In the case of cell-level interconnect, the inputs are represented by drivers, the output stages of cells. If the cell library is characterized using current source models, then the input is a fixed current source,

$$u(t) = B i_{drv}(t) \tag{39}$$

where B is simply an incidence matrix indicating at which node each driver is connected to. Similarly, if the cell library is characterized using voltage source models (as in the case under study), we have

$$u(t) = B v_{drv}(t) \tag{40}$$

as in Eqn. (13), where $v_{drv} = v_m$. Other models may be used, like nonlinear current source models [13, 14].

Recalling Eqn. (14), the full set of equations is now

$$C_0 \frac{d}{dt} x_0(t) + G_0 x_0(t) = u(t) \tag{41}$$

$$v_{0,tap}(t) = L^T x_0(t) \tag{42}$$

$$C_0 \frac{d}{dt} [\Delta x] + G_0 \Delta x = -\sum_{k=1}^{p} \left[\frac{d}{dt} (C_k x_0(t)) \Delta\lambda_k + G_k \Delta\lambda_k \right] \tag{43}$$

$$\Delta v_{tap} = L^T \Delta x \tag{44}$$

These equations can be written more compactly if we define

$$s_k(t) = -\left[C_k \frac{d}{dt} x_0(t) + G_k x_0(t)\right] \tag{45}$$

where $x_0(t)$ is the nominal solution computed above. s_k can be interpreted as the "equivalent source" that will allow determination of the sensitivity to the kth interconnect parameter. With this definition, the final, complete set of equations is then rewritten as

$$C_0 \frac{d}{dt} x_0(t) + G_0 x_0(t) = u(t) \tag{46}$$

$$v_{0,tap} = L^T x_0(t) \tag{47}$$

$$C_0 \frac{d}{dt} [\Delta x] + G_0 \Delta x = \sum_{k=1}^{p} s_k(t) \Delta\lambda_k \tag{48}$$

$$\Delta v_{tap} = L^T \Delta x \tag{49}$$

3.3 Interconnect Sensitivity Calculation

The process of sensitivity calculation can now be concisely stated. First, solve Eqns. (46) and (47) to get the nominal case responses. Then, for each parameter k, solve

$$C_0 \frac{d}{dt} \left[\frac{\partial x}{\partial \lambda_k}\right] + G_0 \left[\frac{\partial x}{\partial \lambda_k}\right] = s_k(t) \tag{50}$$

$$\frac{\partial v_{tap}}{\partial \lambda_k} = L^T \left[\frac{\partial x}{\partial \lambda_k}\right] \tag{51}$$

to get the sensitivity of the response waveforms. From the sensitivity waveforms, the delay sensitivity can be computed using Eqn. (27) at the appropriate time-points. Of course, in practice, it is useful to diagonalize the state-space model above, i.e. to put the C_0, G_0 matrices into pole-residue form, as numerical solution of the multiple systems is much more efficient.

4 Cell Delay Sensitivity Calculation

In the preceding section, we have seen how to perform variation-aware delay computation, by computing the sensitivities of the response waveforms in interconnect blocks. However, it is also necessary to show that similar sensitivities can be computed at the output of cells, in particular assuming that cell delay computation is still based on delay table models.

To show this, we refer back to the derivation in Section 2 and in particular to Eqns. (7), (8), (9) and (10). If we perform an expansion around a nominal point ϕ_0, keeping the first order variations, and eliminating the nominal-case terms, we obtain,

$$\Delta v_o(t_L, \Delta\phi) = 0 \tag{52}$$

$$\Delta v_o(t_T, \Delta\phi) = 0 \tag{53}$$

$$\Delta v_o(t_H, \Delta\phi) = 0 \tag{54}$$

$$\langle \Delta I_c \rangle(t_L, t_H, \Delta\phi) = \langle \Delta I_m \rangle \tag{55}$$

Noticing the dependence of t_L, t_T and t_H, on d and s_o, and their dependence on s_i and C, we obtain the generic equation,

$$\begin{aligned} &\frac{\partial v_o}{\partial s}\Delta s + \frac{\partial v_o}{\partial k}\Delta k + \frac{\partial v_o}{\partial R}\Delta R \\ &+ \left(\frac{\partial v_o}{\partial C} + \frac{\partial v_o}{\partial t_X}\frac{dt_X}{dC}\right)\Delta C + \frac{\partial v_o}{\partial t_X}\frac{dt_X}{ds_i}\Delta s_i = 0 \end{aligned} \tag{56}$$

where

$$\frac{dt_X}{dC} = \frac{\partial t_X}{\partial s_o}\frac{\partial s_o}{\partial C} + \frac{\partial t_X}{\partial d}\frac{\partial d}{\partial C}, \qquad \frac{dt_X}{ds_i} = \frac{\partial t_X}{\partial s_i} + \frac{\partial t_X}{\partial s_o}\frac{\partial s_o}{\partial s_i} + \frac{\partial t_X}{\partial d}\frac{\partial d}{\partial s_i}. \tag{57}$$

t_X can be replaced by t_L, t_T or t_H to obtain Eqns. (52), (53), and (54), and all derivatives are computed at time t_X. For Eqn. (55) a similar expansion can be performed,

$$\begin{aligned} &\left(\frac{\partial \langle I_c \rangle}{\partial s} - \frac{\partial \langle I_m \rangle}{\partial s}\right)\Delta s + \frac{\partial \langle I_c \rangle}{\partial k}\Delta k \\ &+ \left(\frac{\partial \langle I_c \rangle}{\partial R} - \frac{\partial \langle I_m \rangle}{\partial R}\right)\Delta R + \frac{d\langle I_c \rangle}{dC}\Delta C + \frac{d\langle I_c \rangle}{ds_i}\Delta s_i = \langle \Delta I_m \rangle \end{aligned} \tag{58}$$

where

$$\frac{d\langle I_c \rangle}{dC} = \frac{\partial \langle I_c \rangle}{\partial C} + \frac{\partial \langle I_c \rangle}{\partial t_L}\frac{dt_L}{dC} + \frac{\partial \langle I_c \rangle}{\partial t_H}\frac{dt_H}{dC} \tag{59}$$

$$\frac{d\langle I_c \rangle}{ds_i} = \frac{\partial \langle I_c \rangle}{\partial t_L}\frac{dt_L}{ds_i} + \frac{\partial \langle I_c \rangle}{\partial t_H}\frac{dt_H}{ds_i} \tag{60}$$

Δs_i and $\langle \Delta I_m \rangle$ are related to the parameter variation vector, $\Delta\lambda$, by the following expressions,

$$\Delta s_i = \frac{\partial s_i}{\partial \lambda}\Delta\lambda \tag{61}$$

$$\langle \Delta I_m \rangle = \frac{\partial \langle I_m \rangle}{\partial \lambda}\Delta\lambda \tag{62}$$

where $\frac{\partial s_i}{\partial \lambda}$ and $\frac{\partial \langle I_m \rangle}{\partial \lambda}$ are the sensitivity vectors. Resorting to Eqns. (56), (58), (61), and (62), we can now represent Eqns. (52), (53), (54), and (55) in matrix form as,

$$J\Delta\phi = \left(Q\frac{\partial s_i}{\partial \lambda} + W\frac{\partial \langle I_m \rangle}{\partial \lambda}\right)\Delta\lambda \tag{63}$$

where J, Q and W are given by

$$J = \begin{bmatrix} \frac{\partial v_o}{\partial s}\Big|_{t_L} & \frac{\partial v_o}{\partial k}\Big|_{t_L} & \frac{\partial v_o}{\partial R}\Big|_{t_L} & \frac{\partial v_o}{\partial C}\Big|_{t_L} + \frac{\partial v_o}{\partial t_L}\frac{dt_L}{dC} \\ \frac{\partial v_o}{\partial s}\Big|_{t_T} & \frac{\partial v_o}{\partial k}\Big|_{t_T} & \frac{\partial v_o}{\partial R}\Big|_{t_T} & \frac{\partial v_o}{\partial C}\Big|_{t_T} + \frac{\partial v_o}{\partial t_T}\frac{dt_T}{dC} \\ \frac{\partial v_o}{\partial s}\Big|_{t_H} & \frac{\partial v_o}{\partial k}\Big|_{t_H} & \frac{\partial v_o}{\partial R}\Big|_{t_H} & \frac{\partial v_o}{\partial C}\Big|_{t_H} + \frac{\partial v_o}{\partial t_H}\frac{dt_H}{dC} \\ \frac{\partial \langle I_c \rangle}{\partial s} - \frac{\partial \langle I_m \rangle}{\partial s} & \frac{\partial \langle I_c \rangle}{\partial k} & \frac{\partial \langle I_c \rangle}{\partial R} - \frac{\partial \langle I_m \rangle}{\partial R} & \frac{d\langle I_c \rangle}{dC} \end{bmatrix} \tag{64}$$

$$Q = \begin{bmatrix} -\frac{\partial v_o}{\partial t_L}\frac{dt_L}{ds_i} \\ -\frac{\partial v_o}{\partial t_T}\frac{dt_T}{ds_i} \\ -\frac{\partial v_o}{\partial t_H}\frac{dt_H}{ds_i} \\ -\frac{d\langle I_c\rangle}{ds_i} \end{bmatrix} \quad , \quad W = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \tag{65}$$

$\frac{\partial s_i}{\partial \lambda}$ results from the variational timing analysis on the interconnect of the input net, as described in Section 3. $\frac{\partial \langle I_m \rangle}{\partial \lambda}$can be computed by integrating the sensitivities of the port current, I_m, for the transition period and dividing by its width. All the derivatives in J, Q and W can either be computed analytically or by accessing the timing characterization of the cell.

If $N_C = [0\ 0\ 0\ 1]$ is a vector that "selects" the capacitance row of $\Delta\phi$,

$$\Delta C = N_C \Delta\phi = N_C J^{-1} \left(Q\frac{\partial s_i}{\partial \lambda} + W\frac{\partial \langle I_m \rangle}{\partial \lambda} \right) \Delta\lambda \tag{66}$$

Acknowledging the dependence of the delay d and the output slew s_o on the input slew s_i and the capacitance C, the following expressions can be derived,

$$\Delta d = \frac{\partial d}{\partial s_i}\Delta s_i + \frac{\partial d}{\partial C}\Delta C \tag{67}$$

$$\Delta s_o = \frac{\partial s_o}{\partial s_i}\Delta s_i + \frac{\partial s_o}{\partial C}\Delta C \tag{68}$$

where $\frac{\partial d}{\partial s_i}$, $\frac{\partial d}{\partial C}$, $\frac{\partial s_o}{\partial s_i}$ and $\frac{\partial s_o}{\partial C}$ can be computed by direct analysis of the lookup table that contains the timing characterization of the cell. Substituting Eqns. (61) and (66) in Eqns. (67) and (68), we can derive the sensitivities of the delay and output slew to the parameters,

$$\frac{\partial d}{\partial \lambda} = \frac{\partial d}{\partial s_i}\frac{\partial s_i}{\partial \lambda} + \frac{\partial d}{\partial C} N_C J^{-1} \left(Q\frac{\partial s_i}{\partial \lambda} + W\frac{\partial \langle I_m \rangle}{\partial \lambda} \right) \tag{69}$$

$$\frac{\partial s_o}{\partial \lambda} = \frac{\partial s_o}{\partial s_i}\frac{\partial s_i}{\partial \lambda} + \frac{\partial s_o}{\partial C} N_C J^{-1} \left(Q\frac{\partial s_i}{\partial \lambda} + W\frac{\partial \langle I_m \rangle}{\partial \lambda} \right) \tag{70}$$

5 Optimizations for Large Numbers of Parameters

In this section, we discuss how using adjoint methods [15] can accelerate the computation of the timing models when large numbers of parameters are present. Since the computation time is nearly independent of the number of parameters, the sensitivity to a larger number of parameters can be done simultaneously. Thus, if only a few crossing times are of interest, the computation is very cheap on an information-gained basis. When device mismatch is of interest, the sensitivities to multiple model parameters, for every device in a circuit, are needed. Mismatch could be caused by purely stochastic mechanisms, such as dopant fluctuations in MOSFET channels. Systematic effects such as optical proximity printing errors may also lead to device-by-device parameter variations.

Let us suppose the nominal system in Eqn. (46) has been discretized into M time-points and an operating point $x_0(t)$ has been computed. For a given timepoint t_k, $k = 1 \ldots M$, let us introduce the capacitance and conductance matrices C, G as follows:

$$C_k \equiv \left.\frac{\partial q}{\partial x}\right|_{x(t_k)} \qquad G_k \equiv \left.\frac{\partial i}{\partial x}\right|_{x(t_k)} \tag{71}$$

Similarly, define the "source" functions s as

$$s_k^{(q),(l)} \equiv -\frac{d}{dt}\left[\frac{\partial q}{\partial \lambda_l}\right]_{x(t_k)}, \quad s_k^{(i),(l)} \equiv -\left.\frac{\partial i}{\partial \lambda_l}\right|_{x(t_k)} \tag{72}$$

$$s_k^{(l)} = s_k^{(q),(l)} + s_k^{(i),(l)}. \tag{73}$$

We "pack" the time-varying quantities into matrices and vectors with a block structure. If there are N equations in (22) and M timepoints, then the vectors

$$\mathcal{X} = \begin{bmatrix} \Delta x_1 \\ \Delta x_2 \\ \cdots \\ \Delta x_M \end{bmatrix}, \quad s^{(l)} = \begin{bmatrix} s_1^{(l)} \\ s_2^{(l)} \\ \cdots \\ s_M^{(l)} \end{bmatrix} \tag{74}$$

have M sections, each section a vector of N entries. The vector $\mathcal{X}$ represents the waveforms of perturbations due to parameter fluctuation. The vectors $s^{(l)}$ will be used to form the p columns (one for each parameter) of the matrix $\mathcal{S}$,

$$\mathcal{S} = \begin{bmatrix} s^{(1)} & s^{(2)} \ldots s^{(p)} \end{bmatrix} \tag{75}$$

Likewise the matrix

$$\mathcal{G} = \begin{bmatrix} G_1 & & & \\ & G_2 & & \\ & & \ddots & \\ & & & G_M \end{bmatrix} \tag{76}$$

has $M \times M$ blocks, each block an $N \times N$ matrix.

After time-discretization, a composite capacitance matrix $\mathcal{C}$ may also be formed. The precise structure of this matrix depends on the discretization scheme used. For example, for a backward Euler discretization with timesteps $h_1, \ldots, h_M$, the matrix $\mathcal{C}$ becomes

$$\mathcal{C} = \begin{bmatrix} \frac{C_1}{h_1} & & & \\ -\frac{C_1}{h_2} & \frac{C_2}{h_2} & & \\ & & \ddots & \\ & & -\frac{C_{(M-1)}}{h_M} & \frac{C_M}{h_M} \end{bmatrix} \tag{77}$$

Eq. (22) can be written as one composite matrix equation[3]

$$\mathcal{C}\mathcal{X} + \mathcal{G}\mathcal{X} = \mathcal{S}\Delta\lambda. \tag{78}$$

To extract the sensitivity of the waveforms to the parameter λ_k, we solve

$$(\mathcal{C} + \mathcal{G})\mathcal{X}_k = \mathcal{S}e_k \tag{79}$$

where e_k denotes the kth unit vector (all zero, except entry k, where it is unity).

For the delay computation, the sensitivity of the waveform at a specific timepoint j and node k is needed. Construct the (block-structured) vector

$$E = \begin{bmatrix} E_1 \\ E_2 \\ \dots \\ E_M \end{bmatrix} \tag{80}$$

with the vectors E_l given by

$$E_l = e_k, l = j \quad E_l = 0, l \neq j. \tag{81}$$

Then the required sensitivity a_k is $a_k = E^T\mathcal{X}_k$.

Note that $(\mathcal{C}+\mathcal{G})$ is block-lower-triangular. This means that operations with $(\mathcal{C}+\mathcal{G})^{-1}$ are cheap to compute. Of course, the matrices $\mathcal{C}, \mathcal{G}$ are never written down explicitly, we only perform implicit operations as as multiplying $(\mathcal{C}+\mathcal{G})^{-1}$ times a vector. Clearly, to extract the full set of sensitivity information, we must perform p solves – one for each parameter. This is acceptable if p is small, but problematic if p is large. On the other hand, for one solve, we obtain the sensitivities of the waveforms at *all* nodes and all *all* timepoints. The computational complexity is $O(pNM)$ for the solution.

The idea of adjoint analysis is to obtain the sensitivity of a voltage waveform at a *single* timepoint and *single* node, to perturbations of all parameters simultaneously, at all timepoints. With the above notation, the notation of the procedure is simple. First we solve

$$(\mathcal{C} + \mathcal{G})^T\mathcal{U} = E. \tag{82}$$

Denoting the vector of sensitivities $\eta = [a_1, a_2, \dots, a_p]$, we have

$$\eta = \mathcal{U}^T\mathcal{S}. \tag{83}$$

If the sensitivities for multiple timepoints or nodes are to be computed, there is one solve of Eq (82) for each such observation point. The computation of $\mathcal{S}$ is done once, and shared across all solves. If t is the number of such terminal points, the computational complexity is $O(tNM)$ for matrix solution. Compared to the direct computation, savings is possible if $t < p$.

[3]We have omitted uncertainty in the initial condition, which will contribute an additional term to $s^{(q),(l)}$ above.

We have not yet discussed the computation time for constructing the matrix $\mathcal{S}$. At worst, this is $O(pDM)$ where D is the number of devices. However, usually either the number of parameters is small, p is $O(1)$, or each device depends on only a small number of parameters. In either case, the complexity becomes $O(DM)$ $O(NM)$ if the implementation is done so as to exploit such structure.

6 Experimental Results

A realistic circuit block was synthesized and mapped to an industrial 90nm technology. As process parameters, we considered the widths and thicknesses of the six metal layers needed to route the block. During parasitic extraction of the design, we computed the nominal values and sensitivities of each parasitic element (resistors and grounded capacitors), relative to each one of the 12 parameters.

In order to validate the interconnect delay and slew computations, we selected from the design 3671 nets, including nets in the internal logic, nets in the clock tree and nets in the pad wiring. For each of these nets, we computed the parametric delay and slew expressions for each of its taps (resulting in 13870 taps among all nets), while the port was excited by a rising voltage ramp. To assess the accuracy of the proposed methodology, the delay and slew sensitivities were compared to transistor-level simulations performed using the circuit simulator Spectre. In Figure 3 we present scatter plots of the sensitivities computed by both methods, for 4 parameters. In Figure 4 we present histograms of the relative errors for other 4 parameters. Both figures clearly show that the computed sensitivities accurately match those obtained by simulation.

In order to validate the cell delay and output slew computations we proceeded as follows. For a given standard cell of that same 90nm technology, and using Spice-level models, we generated a dotlib-style lookup table of size 7x7, for delay and output slew, as a function of input slew and load. Using these tables, and applying the proposed methodology, we computed the delay and output slew sensitivities for one of the cell instances in the previously mentioned design, considering its loading net obtained from extraction. Using the methodology proposed in Section 4 we generated the sensitivities of delay and output slew to all 12 parameters. Next, varying the parameter values, a similar set of sensitivities was also computed with Spectre, using accurate Spice-level models for the cell. The delay and output slew sensitivity values obtained using the proposed method were then assessed by computing its relative error versus the Spectre-generated data. These relative errors are shown in Figure 5 (left plot). As can be observed, the errors are in general small, usually in the low percentage range. The only exception to this rule is the pathological case of the slew sensitivity to parameter #2, whose absolute value is small, the smallest of all the sensitivities and near machine precision. In order to investigate this behavior, we introduced a variation in the input slew depending on parameter #2, so that the delay and output slew sensitivity values to this parameter would become larger. As a result we observed that when this happened the relative error dropped to the normal range, as shown in Figure 5 (right plot). Considering that the size

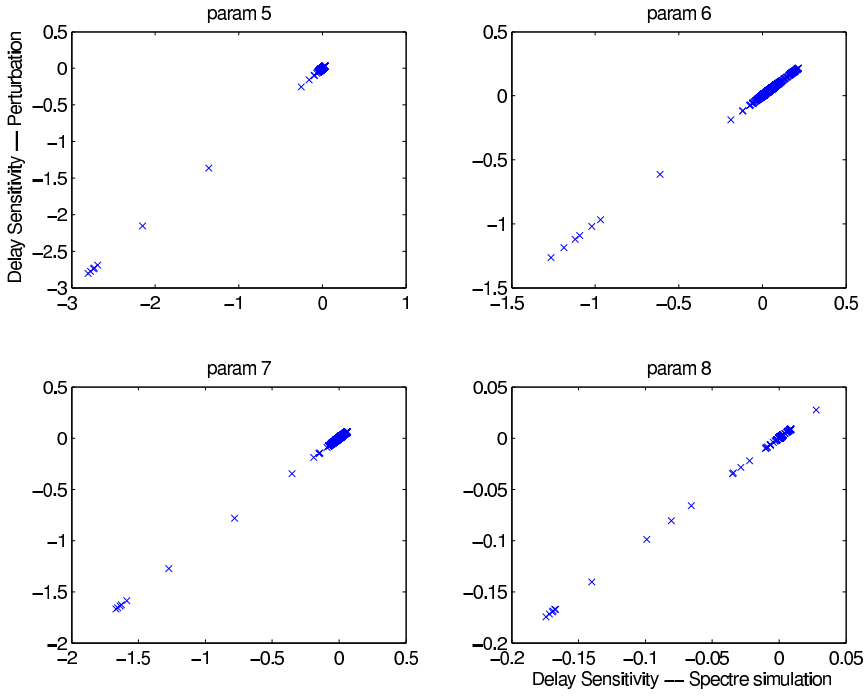

Fig. 3. Computed delay sensitivities vs. transistor-level simulation.

of the dotlib-style lookup table used was only 7x7 (typical value), providing a rough approximation of the behavior of the cell, and that the parasitic network was also approximated by a single lumped capacitance, we believe that the accuracy of the computed values is fairly good. Better accuracy should be obtained by using larger lookup tables, or by extending the proposed model for handling tables depending on other parameters.

7 Conclusions

In this paper we have developed an analytic delay calculation methodology suitable for use in a statistical static timing methodology. Our approach, based on a specific type of perturbation analysis, allows for the analytical computation of the quantities needed for statistical delay propagation. We also showed how perturbation analysis can be performed when only the standard cell delay table lookup models are available. The techniques proposed are robust and show good correlation with transistor level calculations. Furthermore, they can be directly applied when cell characterization is based either in voltage or current source models. Future work will show how to develop models that include nonlinear contributions from the process parameters.

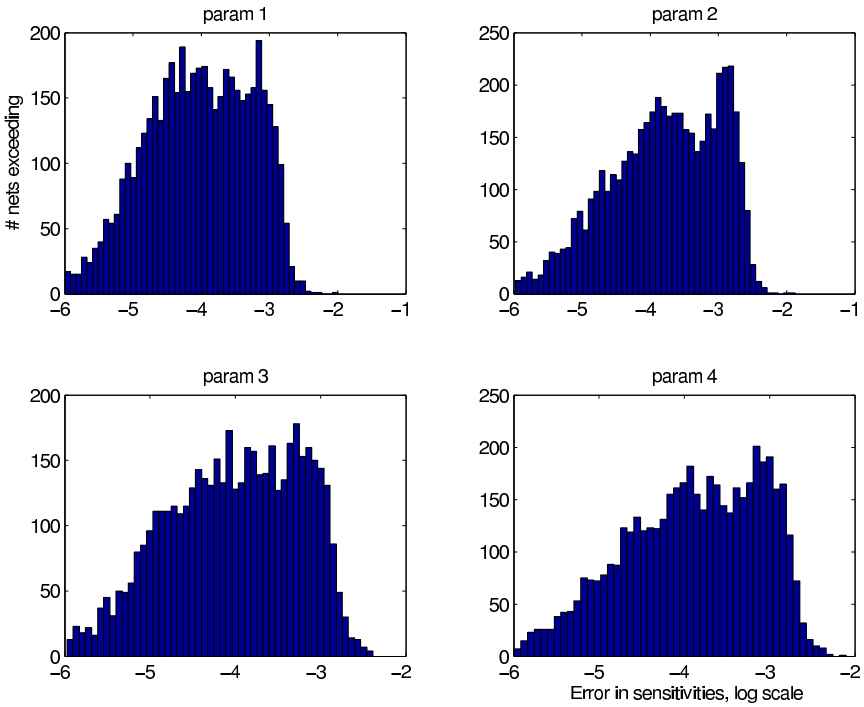

Fig. 4. Histograms of errors in computed delay sensitivities.

Acknowledgments

The authors want to thank Roberto Passerone for providing example circuits, and Vinod Kariat and Igor Keller for suggestions and discussions.

References

1. C. Visweswariah, K. Ravindran, K. Kalafala, S. G. Walker, and S. Narayan. First-Order Incremental Block-Based Statistical Timing Analysis. In *Proceedings of the Design Automation Conference*, pages 331–336, San Diego, CA, June 2004.
2. H. Chang and S. S. Sapatnekar. Statistical Timing Analysis Considering Spatial Correlations using a Single Pert-like Traversal. In *Proceedings of the International Conference on Computer Aided-Design*, pages 621–625, San Jose, CA, November 2003.
3. Luis Guerra e Silva, L. Miguel Silveira, and Joel R. Phillips. Efficient Computation of the Worst-Delay Corner. In *Proceedings of Design, Automation and Test in Europe, Exhibition and Conference*, Nice, France, April 2007.
4. J. Wang, P. Ghanta, and S. Vrudhula. Stochastic Analysis of Interconnect Performance in the Presence of Process Variations. In *Proceedings of the International Conference on Computer Aided-Design*, pages 880–886, San Jose, CA, November 2004.

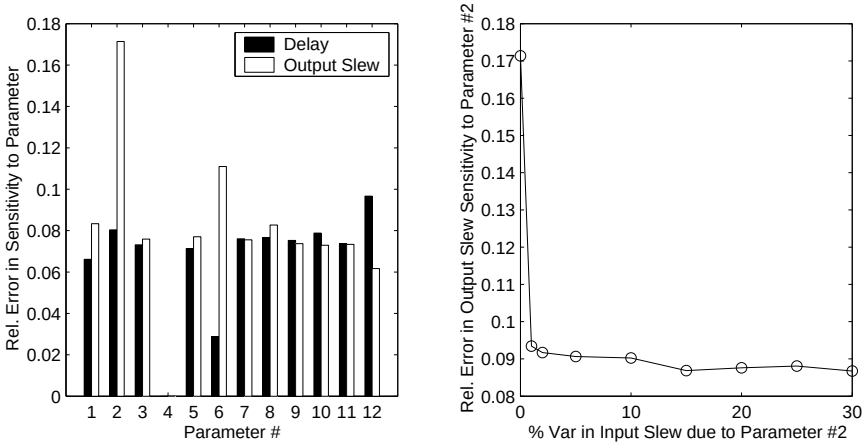

Fig. 5. Relative errors in computed cell delay and output slew sensitivities.

5. Joel R. Phillips. Variational Interconnect Analysis Via PMTBR. In *Proceedings of the International Conference on Computer Aided-Design*, pages 872–879, San Jose, CA, November 2004.
6. X. Li, P. Li, and L. Pileggi. Parameterized interconnect order reduction with Explicit-and-Implicit multi-Parameter moment matching for Inter/Intra-Die variations. In *Proceedings of the International Conference on Computer Aided-Design*, pages 806–812, San Jose, CA, November 2005.
7. Z. Wang, R. Murgai, and J. Roychowdhury. ADAMIN: Automated, accurate macomodelling of digital aggressors for power and ground supply noise prediction. *IEEE Transaction on CAD*, 24:56–64, January 2005.
8. Joel R. Phillips. Model Computation for Statistical Static Timing Analysis, May 2006. Cadence Internal Report.
9. R. Telichevesky, J. White, and K. Kundert. Efficient AC and Noise Analysis of Two-Tone RF Circuits. In *Proceedings of the Design Automation Conference*, June 1996.
10. Sani R. Nassif and Zhuo Li. A More Effective C_{EFF}. In *Proceedinge of the Sixth International Symposium on Quality of Electronic Design*, pages 654 – 661, San Jose, CA, March 2005.
11. J. Qian, S. Pullela, , and L. Pillage. Modeling the Effective Capacitance for the RC Interconnect of CMOS Gates. *IEEE Trans. on VLSI*, 13:1526 – 1535, 1994.
12. Florentin Dartu, Noel Menezes, and Lawrence T. Pileggi. Performance Computation for Precharacterized CMOS Gates with RC Loads. *IEEE Trans. on CAD*, 15(5):544 – 553, May 1996.
13. Igor Keller, Nishath Verghese, and Kenneth Tseng. A Robust Cell-Level Crosstalk Delay Change Analysis. In *Proceedings of the International Conference on Computer Aided-Design*, San Jose, CA, November 2004.
14. J.F. Croix and D.F Wong. Blade and Razor: Cell and Interconnect Delay Analysis using Current-Based Models. In *Proceedings of the Design Automation Conference*, pages 386 – 389, Anaheim, CA, June 2003.
15. S.W. Director and R. A. Rohrer. The Generalized Adjoint Network and Network Sensitivities. *IEEE Trans. on Circuit Theory*, CT-16(3):318–323, August 1969.