On Exponential Fitting for Circuit Simulation

Luís Miguel Silveira, *Student Member, IEEE,* Jacob K. White, *Associate Member, IEEE,*
Horácio Neto, and Luís Vidigal

***Abstract*—In this paper, the stability and accuracy properties of exponentially fit integration algorithms applied to the test problem $\dot{x} = -Ax$ are compared with the more standard backward-Euler and semi-implicit methods. For the analysis, $A \in \mathbb{R}^{n \times n}$ is assumed to be connectedly diagonally dominant with positive diagonals, as this models the equations resulting from the way MOS transistors and interconnect parasitics are treated in circuit-level timing simulation programs. Examples are used to demonstrate that all the exponential-fitting methods, and the semi-implicit methods, are much less accurate than backward-Euler for tightly coupled stiff problems, and an example is given which destabilizes one of the exponential-fitting methods. It is then proved that in the limit of large time steps, the more stable exponential-fitting methods become equivalent to a semi-implicit algorithm. Finally, it is shown that the backward-Euler, semi-implicit, and certain exponentially fit algorithms are *multirate* A-stable.**

I. Introduction

DESIGNERS of MOS digital circuits often use transistor-level simulation programs that are very fast but have limited accuracy compared with circuit simulation programs such as SPICE [1]. This reduction in computation time allows for entire designs, or at least whole critical paths, to be simulated, though only a rough idea of circuit performance can be derived. Programs of this type are referred to as timing simulators and typically are simplified circuit simulators with loosely controlled accuracy. Specifically, these programs use nodal analysis to derive a system of differential equations that describes the circuit. Then, by exploiting the assumption that each node has a capacitor to ground, the programs can use simplified semi-implicit multistep integration algorithms [2]–[4].

Recently, exponentially fit integration methods [5] have been applied in an attempt to improve the performance and accuracy of timing simulation, as in the programs CINNAMON [6], XPSim [7], ELOGIC [8] and ADEPT [9]. For the purposes of this paper, we define the first-order exponentially fit integration algorithm as

$$x(t_{m+1}) = x(t_m) + (x(\infty) - x(t_m))(1 - e^{-h/\tau}) \quad (1)$$

where $h = t_{m+1} - t_m$ is the time step and $x(\infty)$ and τ depend on the particular exponentially fit integration method being used. The interpretation of these parameters is that $x(\infty)$ is an estimate of the equilibrium or steady-state value of x, and τ is an estimate of the time constant of the approach to steady state.

Exponentially fit methods seem appealing because when applied to numerically integrating the scalar linear differential equation $\dot{x} + dx = b$, $d, b \in \mathbb{R}$, with an appropriate choice of τ and $x(\infty)$, the exact solution is produced. This accuracy for scalar problems has the practical consequence that when exponentially fit integration methods are applied to solving the differential equations associated with MOS digital circuits, they frequently retain reasonable accuracy for much larger time steps than standard multistep methods. In this paper, we investigate whether such approaches are still superior to implicit or semi-implicit schemes when used to simulate MOS circuits which include interconnect parasitic resistances.

It is difficult to draw any general conclusions about the relative merits of the above algorithms by comparing the performance of the related programs. All the programs mentioned above demonstrate speed advantages over SPICE of nearly two orders of magnitude, and their accuracies are sensitive to time step control heuristics. Instead, in this paper we attempt to compare the algorithms theoretically in an admittedly simplified setting, but one which we feel lends insight. In particular, we assume a matrix test problem:

$$\dot{\boldsymbol{x}} = -\boldsymbol{A}\boldsymbol{x} \quad \boldsymbol{x}(0) = \boldsymbol{x}_0 \neq \boldsymbol{o} \quad (2)$$

where $\boldsymbol{A} \in \mathbb{R}^{n \times n}$. In our case, $\boldsymbol{A}$ is assumed to be connectedly diagonally dominant [10] with positive diagonals, as this models the equations resulting from the way MOS transistors and interconnect parasitics are treated in timing simulation programs.

We start in the next few sections by comparing the several integration methods using fixed time steps and then analyze the multirate case. In particular, in the next section we show that consistency enforces a relation between $x(\infty)$ and τ of (1) and that the methods used in the programs CINNAMON and XPSim can be derived from (1). In Section III, stiff, tightly coupled examples are examined to demonstrate a stability problem with the XPSim

Manuscript received March 27, 1990; revised August 23, 1990, and April 1, 1991. This work was supported by a research initiation grant from AT&T, by the National Science Foundation, by the Defense Advanced Research Projects Agency (Contract N00014-87-K-825), by the Portuguese INVOTAN committee, and by CEC Esprit Project 1058. This paper was recommended by Associate Editor A. E. Ruehli.

L. M. Silveira and J. K. White are with the Research Laboratory of Electronics, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139.

H. Neto and L. Vidigal are with INESC—Instituto de Engenharia de Sistemas e Computadores, Lisboa, Portugal, and the Instituto Superior Técnico, Universidade Técnica de Lisboa, Lisboa, Portugal.

IEEE Log Number 9105729.

algorithm and an accuracy problem with the CINNAMON algorithm. A more stable semi-implicit version of the XPSim algorithm, denoted IPSim, is given, and it is proved that in the limit of large time steps the IPSim and CINNAMON style exponentially fit algorithms become equivalent to the semi-implicit algorithm originally used in the MOTIS program [2]. In Section IV, we prove that the CINNAMON, MOTIS, and backward-Euler algorithms are *multirate A-stable*. Conclusions are given in Section V, followed by acknowledgments.

II. Explicit Exponential Fitting

Not all values of the $x(\infty)$ and τ parameters introduced in (1) produce consistent integration methods, where by *consistency* we mean that the error introduced in one time step, h, is $O(h^2)$. Assuming that $x(t_m)$ is exact and using a Taylor series expansion of the exact solution about $x(t_m)$, we find that the exact solution for time t_{m+1} is given by

$$x^E(t_{m+1}) = x(t_m) + h\dot{x}(t_m) + O(h^2)$$

where the superscript E is used to denote the exact solution. The approximate solution computed from (1) is

$$x(t_{m+1}) = x(t_m) + h\frac{x(\infty) - x(t_m)}{\tau} + O(h^2).$$

Therefore, for consistency, $x(\infty)$ and τ must satisfy the condition

$$\frac{x(\infty) - x(t_m)}{\tau} = \dot{x}(t_m). \tag{3}$$

Using the relation in (3) we can rewrite (1) as

$$x(t_{m+1}) = x(t_m) + \tau(1 - e^{-(h/\tau)})\dot{x}(t_m), \tag{4}$$

which defines a family of consistent, one-step, first-order, explicit, exponentially fitted formulas parameterized by τ.

Also note that when $h \to 0$, and τ is bounded away from 0,

$$(1 - e^{-(h/\tau)}) \approx \frac{h}{\tau}, \tag{5}$$

which implies that formula (4) reduces to the well-known forward-Euler integration algorithm and therefore inherits its convergence properties. That is, when numerically integrating on a finite interval $[0, T]$ and in the limit as $h \to 0$,

$$\max_{0<m<M} \{|x(t_m) - x^E(t_m)|\} = 0, \tag{6}$$

where $t_m = mh$ and $t_M = T$.

For a system of N ordinary differential equations, (4) can be generalized as

$$\boldsymbol{x}(t_{m+1}) = \boldsymbol{x}(t_m) + \boldsymbol{D}^{-1}(\boldsymbol{I} - e^{-h\boldsymbol{D}})\dot{\boldsymbol{x}}(t_m), \tag{7}$$

where now $\boldsymbol{x} \in \mathbb{R}^n$ and $\boldsymbol{D}$ is an $n \times n$ diagonal matrix, with $d_i = 1/\tau_i$, $i \in \{1, \cdots, n\}$.

The approach to exponential fitting used in the circuit simulators CINNAMON and XPSim can be reduced to the above formulation, again assuming fixed time steps.

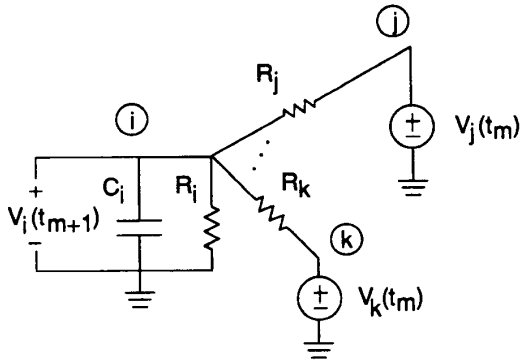


Fig. 1. Updating a node value with CINNAMON (circuit interpretation).

It is in the choice of the fitting matrix, $\boldsymbol{D}$, that the approaches differ significantly. We will illustrate the difference between the two exponential fitting methods for the test problem in (2). In CINNAMON, a simple approach is used: the τ_i is selected from the diagonal term of the $\boldsymbol{A}$ matrix as

$$\tau_i = \frac{1}{a_{ii}} \tag{8}$$

and the $x_i(\infty)$ is set according to the consistency relationship (3) to be

$$x_i(\infty) = x_i(t_m) + \tau_i\dot{x}_i(t_m) \tag{9}$$

for $i \in \{1, \cdots, n\}$.

There is a simple circuit interpretation of the CINNAMON algorithm. Each node in the circuit is updated to a new time point by computing the exact solution to that node, given that all the other nodes are treated as fixed voltage sources with values given from the previous time-point (see Fig. 1).

In the XPSim algorithm, the value selected for $\boldsymbol{x}(\infty)$ in (7) is, for a linear system, the *correct* steady-state value. Note that when applied to nonlinear circuits, the $\boldsymbol{x}(\infty)$ used is not necessarily the correct steady state, and may change at each time step, adding to the computational cost of the method [7]. However, if the intention is to simulate digital logic circuitry, the $\boldsymbol{x}(\infty)$ computation can possibly be avoided by exploiting the fact that eventually the circuit voltages approach the power supply or ground. For our problem, $\boldsymbol{x}(\infty) = \boldsymbol{0}$, and by consistency (3) reduces to

$$\tau_i = -\frac{x_i(t_m)}{\dot{x}_i(t_m)} \tag{10}$$

and

$$\begin{aligned} x_i(t_{m+1}) &= x_i(t_m)\exp\left(h\frac{\dot{x}_i(t_m)}{x_i(t_m)}\right) \\ &= x_i(t_m)\exp\left[-h\frac{\sum_{j=1}^{n} a_{ij}x_j(t_m)}{x_i(t_m)}\right] \end{aligned} \tag{11}$$

for $i \in \{1, \cdots, n\}$.

Clearly, the XPSim algorithm as described above cannot be used when the solution passes through 0. In that case τ is not bounded away from 0, (5) no longer applies, and the XPSim algorithm is not consistent. In fact, the method is positive invariant. That is, the XPSim algorithm can only produce a positive $x(t_{m+1})$ from a positive $x(t_m)$. For our examples, this difficulty can be mostly ignored if the initial condition is assumed to be a positive vector. Then, if the problem is as given in (2), where A is strictly or connectedly diagonally dominant, the exact solution and the solution computed by XPSim will be positive for all time [11]. The method is still useful since logic circuits with a single positive supply, no floating capacitors, and physically reasonable transistor models will have node voltages that are always nonnegative. If the voltages do change signs, a modified XPSim algorithm must be used, where, for example, a CINNAMON-like technique can be used when the solution approaches 0 [12].

III. Fixed Time Step Properties

Because exponential-fitting methods are tuned to scalar problems, they obviously are going to perform well when A of (2) is strongly diagonally dominant. In fact, some of the methods can be A-stable for the test problem. Nevertheless, their accuracy degrades surprisingly quickly when A is stiff but only weakly diagonally dominant [5]. In this section we consider the example in Fig. 2, a tightly coupled two-node circuit where the initial condition at each node is 1 V.

A. Stability Properties

An eigenvalue analysis shows that the example in Fig. 2 has two widely separated time constants, at $\tau = 2.0075$ and $\tau = 0.0075$, and is therefore a very stiff problem, as well as being tightly coupled.

A comparison of the computed results simulating this circuit using the backward-Euler (BE), the XPSim (XP), and the CINNAMON (CIN) algorithms with a 0.1 s time step is plotted in Fig. 3 (the algorithm IP in this figure is described below). As is clear from the picture, all algorithms produce roughly the same accurate results.

In Fig. 4, the results from simulating the same example, but using a 1.1 s time step, are plotted. For this case, disappointing results are achieved for the CINNAMON (CIN) and XPSim (XP) methods when compared with the standard backward-Euler (BE) algorithm. The solution computed with XPSim is unstable and the solution computed with CINNAMON is inaccurate in that it decays much too slowly. (The algorithms SI and IP will be described below.)

It is possible to stabilize the XPSim algorithm for diagonally dominant problems by making the computation semi-implicit. When applied to (2), a semi-implicit version of XPSim, which we denote as IPSim (IP), is

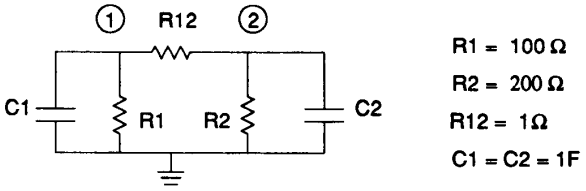


Fig. 2. Test circuit.

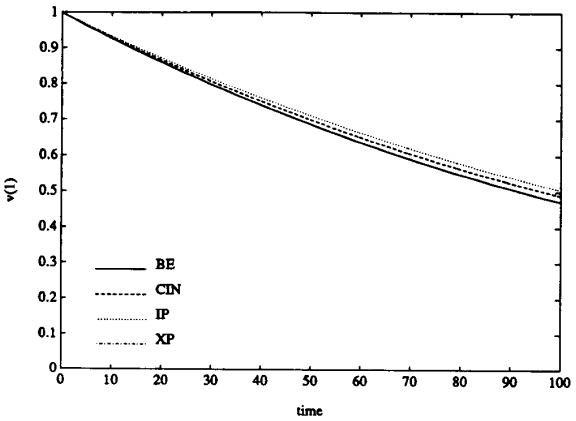


Fig. 3. $v_1(t)$ for $h = 0.1$.

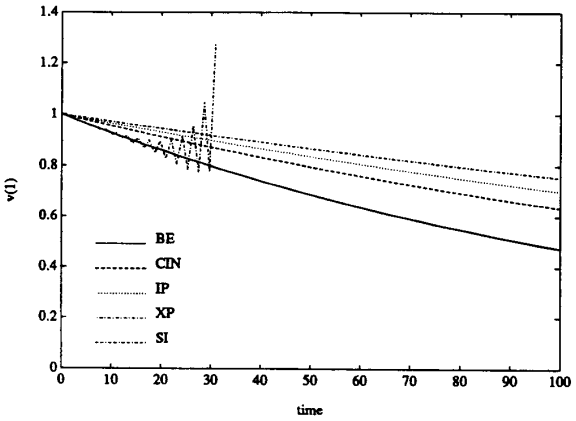


Fig. 4. $v_1(t)$ for $h = 1.1$.

$$x_i(t_{m+1}) = x_i(t_m) \exp\left[-h \frac{a_{ii}x_i(t_{m+1}) + \sum_{j \neq i} a_{ij}x_j(t_m)}{x_i(t_m)}\right] \tag{12}$$

for $i \in \{1, \cdots, n\}$. Note that the method is not implicit with respect to the denominator term in the exponential, as this does not enhance stability and makes for a harder nonlinear problem to solve at each step. The IPSim algorithm is similar to the Gauss–Jacobi semi-implicit al-

gorithm used in the MOTIS [2] program, which for our test problem is described by the update equation

$$x_i(t_{m+1}) = x_i(t_m) - h\left[a_{ii}x_i(t_{m+1}) + \sum_{j \neq i} a_{ij}x_j(t_m)\right] \quad (13)$$

for $i \in \{1, \cdots, n\}$. Note that in both (12) and (13) the update equation is made implicit with respect to the variable being updated.

The CINNAMON, MOTIS and IPSim algorithms have similar stability properties, as is made clear in the following theorem:

Theorem 1: When applied to solving (2), where A is assumed to be strictly or connectedly diagonally dominant with positive diagonals, the MOTIS and CINNAMON algorithms are A-stable, and if the initial condition is a positive vector, IPSim is A-stable.

We will not prove Theorem 1 here. The A-stability of MOTIS for these types of problems is well known [13]; the A-stability of the CINNAMON algorithm is a special case of Theorem 3, which follows (also see [14] and [15]); and the A-stability of IPSim requires a lengthy proof, which can be found in [15].

B. Accuracy Properties

If the time step is set to 10 s, as in Fig. 5, the result from the XPSim algorithm becomes too unstable to plot, and the result from the CINNAMON algorithm gets "stuck," that is, it decays much more slowly than the exact solution. This property of the CINNAMON algorithm, that of getting "stuck," is particularly insidious, as the slowly changing node may be misinterpreted as having achieved equilibrium. This can confuse an event-driven simulator based on the CINNAMON algorithm and may lead to large errors.

The results using the IPSim (IP) and MOTIS (SI) algorithms to simulate the circuit in Fig. 2 with 1.1 and 10 s time steps are plotted in Fig. 4 and Fig. 5 respectively. As the plots show, the IPSim algorithm is stable but produces results not significantly more accurate than the MOTIS or CINNAMON algorithms. This comparison is true in general, as is made clear in the following theorem.

Theorem 2: If A has positive diagonal entries, then in the limit of large time steps, the CINNAMON, IPSim, and MOTIS algorithms produce identical results.

Proof: Summarizing, the update equations for the MOTIS, CINNAMON, and IPSim algorithms are, respectively,

$$x_i(t_{m+1}) = x_i(t_m) - h\left[a_{ii}x_i(t_{m+1}) + \sum_{j \neq i} a_{ij}x_j(t_m)\right] \quad (14)$$

$$x_i(t_{m+1}) = x_i(t_m) - \frac{1 - e^{-ha_{ii}}}{a_{ii}}\left[\sum_{j=1}^{n} a_{ij}x_j(t_m)\right] \quad (15)$$

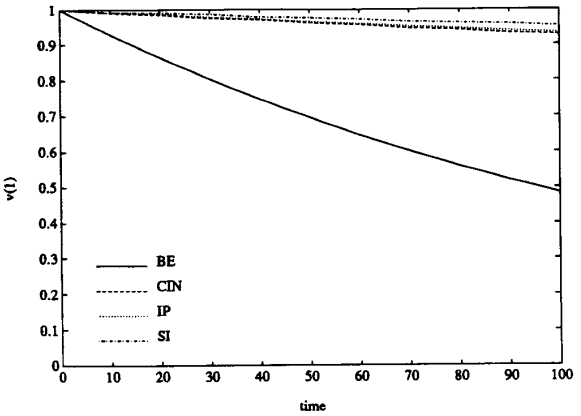


Fig. 5. $v_1(t)$ for $h = 10$.

$$x_i(t_{m+1}) = x_i(t_m) \exp\left[-h\,\frac{a_{ii}x_i(t_{m+1}) + \sum_{j \neq i} a_{ij}x_j(t_m)}{x_i(t_m)}\right]. \quad (16)$$

It is easy to see in the limit of large h, and given that a_{ii} is positive for all i, that $x_i(t_{m+1})$ for both the MOTIS and the CINNAMON algorithm is

$$x_i(t_{m+1}) = -\sum_{j \neq i} \frac{a_{ij}}{a_{ii}} x_j(t_m). \quad (17)$$

The result in (17) holds true for the IPSim algorithm as well. This can be seen by considering that in the limit of large h the argument of the exponential in IPSim's update equation, (16), cannot go to ∞. Therefore, the term from the numerator of the exponential's argument in (16),

$$a_{ii}x_i(t_{m+1}) + \sum_{j \neq i} a_{ij}x_j(t_m), \quad (18)$$

must approach zero as $h \to \infty$. ■

C. Ordering

In general, it is possible to improve the stability and accuracy of explicit or semi-implicit methods by ordering the equations being solved and using updated values when possible. Specifically, when calculating $x_i(t_{m+1})$ the use of the already computed values of $x_j(t_{m+1})$ for $j < i$ will improve the accuracy of the solution. This is exploited in most implementations of semi-implicit integration algorithms [3], [16] and also in the CINNAMON circuit simulator [17].

Using ordering in the XPSim algorithm leads to the following update equation for x_i:

$$x_i(t_{m+1}) = x_i(t_m) \cdot \exp\left[-h\,\frac{\sum_{j<i} a_{ij}x_j(t_{m+1}) + \sum_{j \geq i} a_{ij}x_j(t_m)}{x_i(t_m)}\right], \quad (19)$$

where subscript index also indicates the ordering.

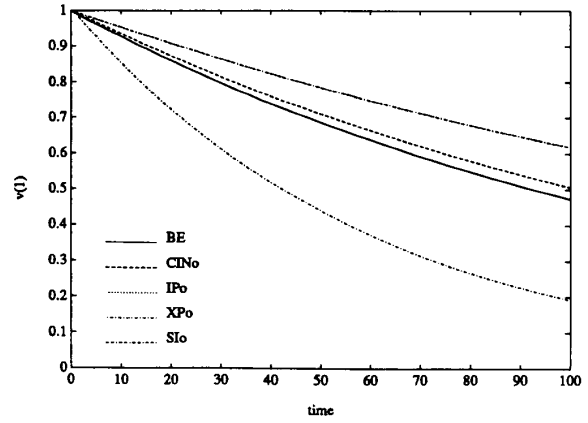


Fig. 6. $v_1(t)$ for $h = 1.1$.

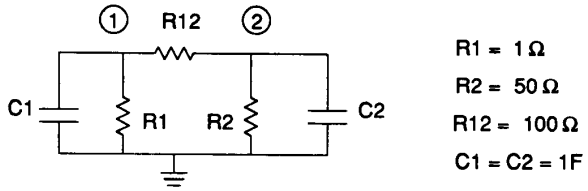


Fig. 7. Circuit to test possible stability of ordered XPSim.

In Fig. 6 we present the plots of computed solutions to the test circuit in Fig. 2, under the same conditions of Fig. 4, i.e., using a 1.1 s time step. The waveforms shown correspond to backward-Euler (BE) and ordered versions of CINNAMON (CINo), IPSim (IPo), XPSim (XPo), and the semi-implicit algorithm (SIo). Here, the ordering used always updates node 1 first. From the figure, and compared with the results shown in Fig. 4, one can see that ordering improved the accuracy of all methods, most notably that of XPSim, which is stabilized by ordering. Also from the plot we note that, with ordering, the solution produced with CINNAMON becomes quite accurate and that the solution produced with IPSim is almost the same as that obtained with the semi-implicit algorithm, albeit at the expense of more computation.

The results of the above plot are partially deceptive in that it is not always possible to stabilize the XPSim algorithm by ordering, as can be demonstrated by considering the example circuit in Fig. 7. Solutions for the initial condition of both nodes at 1 V were computed using node 1 first ordered XPSim (XPo), node 1 first ordered CINNAMON (CINo), and backward-Euler (BE), all with a time step $h = 6$, and are plotted in Fig. 8. Note that the solution computed with the ordered XPSim method shows oscillations of increasing magnitude. This plot is somewhat surprising since the system matrix is not far from diagonal. In addition, if the time step is made larger, say, to $h = 7$, then it is *impossible* to stabilize XPSim with any ordering, a fact that can be established by exhaustive search over all the possible orderings.

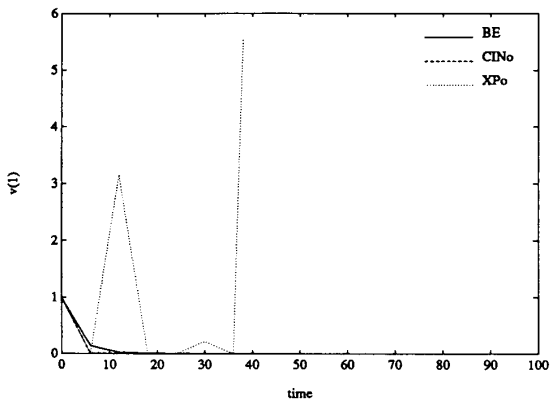


Fig. 8. $v_1(t)$ from the circuit in Fig. 7 for $h = 6$.

IV. Multirate Properties

Most practical circuit-level simulators use adaptive time step control to ensure accuracy in the computed solution, and timing simulators gain some of their speed advantage over SPICE-like simulators by independently adapting the time steps for individual circuit nodes. An integration method which allows different nodes in the circuit to use different time steps is referred to as a multirate integration method, and the stability theory for such methods is not as complete as for the fixed time step case [18]. In this section, we show that the MOTIS, CINNAMON, and backward-Euler algorithms are multirate A-stable for our test problem, which in turn implies that A-stability is preserved regardless of the time step selection strategy. Also, we note that for the tightly coupled stiff example in Section III, a time step control strategy based on ensuring that each circuit node changes by some fixed voltage increment cannot be used directly with CINNAMON- and MOTIS-style algorithms and does *not* make it possible to use large time steps with XPSim.

A. Multirate Methods

As mentioned above, a key advantage of timing simulators is that different time steps can be used for different circuit nodes. In particular, the update equations for the backward-Euler, MOTIS, and CINNAMON algorithms in this multirate case become

$$x_i(t_m^i) = x_i(t_{m-1}^i) - h_m^i \left[a_{ii} x_i(t_m^i) + \sum_{j \neq i} a_{ij} I_{t_m^i}^i(\{x_j\}) \right]$$

$$x_i(t_m^i) = x_i(t_{m-1}^i) - h_m^i \left[a_{ii} x_i(t_m^i) + \sum_{j \neq i} a_{ij} I_{t_{m-1}^i}^i(\{x_j\}) \right]$$

$$x_i(t_m^i) = x_i(t_{m-1}^i) - \frac{1 - e^{-h_m^i a_{ii}}}{a_{ii}} \cdot \left[a_{ii} x_i(t_{m-1}^i) + \sum_{j \neq i} a_{ij} I_{t_{m-1}^i}^i(\{x_j\}) \right], \quad (20)$$

where $h_m^i \equiv t_m^i - t_{m-1}^i$ is the mth time step for the ith

circuit node, and $I_t(\{x_j\}) \in \mathbb{R}$ is the interpolation operator required because t_m^i is not necessarily equal to t_m^j. For example, if linear interpolation is used, then

$$I_t(\{x\}) = \psi x(t_m) + (1 - \psi)x(t_{m-1}), \tag{21}$$

where m is such that $t \in [t_{m-1}, t_m]$, and $\psi = (t - t_{m-1})/(t_m - t_{m-1})$. In general, we assume that the interpolation operator is a linear function of the sequence values. Note that this is distinct from linear interpolation; any polynomial interpolation scheme is a linear function of the sequence values.

A general approach to solving for the sequences which satisfy the multirate discretization equations in (20) is to use a discretized waveform relaxation (WR) approach [19]. Roughly, on each WR iteration, the behavior of each node voltage is computed *for all time* using the best available information about the other node voltages. Then a second WR iteration is performed, with the waveform computation at each node using the updated information about the other nodes. The WR iterations are repeated until convergence is achieved. In particular, Algorithm 1, given below, is an idealized WR algorithm based on the backward-Euler discretization.

Algorithm 1 (Backward-Euler Discretized WR Algorithm)
For $k = 1$ *to* ∞
For $i = 1$ *to* n
For $m = 1$ *to* ∞
Solve:

$$x_i^k(t_m^i) = x_i^k(t_{m-1}^i) - h_m^i\left[a_{ii}x_i^k(t_m^i) + \sum_{j \neq i} a_{ij} I_{t_m^i}(\{x_j^{k-1}\})\right] \tag{22}$$

for $x_i^k(t_m^i)$.

Here k is the waveform iteration index and $\{x_i^0\}$ is an initial guess sequence whose initial value matches the test problem initial condition. The WR algorithm can be altered to use the CINNAMON or the MOTIS discretization scheme by changing (22) to

$$x_i^k(t_m^i) = x_i^k(t_{m-1}^i) - \frac{1 - e^{-h_m^i a_{ii}}}{a_{ii}} \cdot \left[a_{ii}x_i^k(t_{m-1}^i) + \sum_{j \neq i} a_{ij} I_{t_{m-1}^i}(\{x_j^{k-1}\})\right] \tag{23}$$

or

$$x_i^k(t_m^i) = x_i^k(t_{m-1}^i) - h_m^i\left[a_{ii}x_i^k(t_m^i) + \sum_{j \neq i} a_{ij} I_{t_{m-1}^i}(\{x_j^{k-1}\})\right] \tag{24}$$

respectively.

Clearly, there are more efficient approaches for computing the time points associated with the CINNAMON and MOTIS algorithms. The WR algorithm is mentioned here because of a prejudice of one of the authors and because it will be used to prove multirate A-stability.

B. A-Stability Theorem

The main result of this subsection is that the methods above are multirate A-stable. The definitions of multirate A-stability [18] and an interpolation property precede the theorem statement below.

Definition 1: The methods in (20) are multirate A-stable for our test problem if for any set of positive time steps,

$$\lim_{m \to \infty} x_i(t_m^i) = 0 \tag{25}$$

for all $i \in \{1, \cdots, n\}$.

Definition 2: An interpolation operator is maximum reducing if

$$\max_t |I_t(\{x\})| \leq \max_m |x(t_m)|. \tag{26}$$

For example, piecewise constant and linear interpolations are maximum reducing, and quadratic interpolation is not.

Theorem 3: When applied to solving (2), where $\boldsymbol{A}$ is assumed to be strictly or connectedly diagonally dominant with positive diagonals, the MOTIS, CINNAMON, and backward-Euler style multirate algorithms are multirate A-stable for any interpolation operator which is maximum reducing.

The following two lemmas will be useful in the proof of Theorem 3.

Lemma 1: Consider two real sequences, $z[m]$ and $y[m]$, $m \in \{0, 1, \cdots \infty\}$, for which $y[0] = 0$. If

$$y[m + 1] = y[m] - \alpha y[m] + \beta z[m], \tag{27}$$

where $\alpha, \beta \in \mathbb{R}$ and $\alpha \in [0, 1]$, or if

$$y[m + 1] = y[m] - \alpha y[m + 1] + \beta z[m], \tag{28}$$

where $\alpha, \beta \in \mathbb{R}$ and $\alpha > 0$, then

$$\max_m |y[m]| \leq \frac{|\beta|}{\alpha} \max_m |z[m]|. \tag{29}$$

The following is a special case of the main result in [20],

Lemma 2: The multirate integration methods given in (20) are multirate A-stable when applied to (2), where $\boldsymbol{A}$ is strictly or connectedly diagonally dominant with positive diagonals, if the iterates produced by the associated multirate discretized WR algorithms contract in a maximum norm over the sequence for any set of positive time steps.

Restated, Lemma 2 implies that if for any set of positive time steps, the WR iterates defined by (22), (23), and (24) satisfy the inequality

$$\|\delta \mathbf{y}^k\| \leq \gamma \|\delta \mathbf{y}^{k-1}\| \tag{30}$$

where $\delta \mathbf{y}^k \in \mathbb{R}^n$ is defined by $\delta y_i^k \equiv \max_m |x_i^k(t_m^i) - x_i^{k-1}(t_m^i)|$, $\|\cdot\|$ is any norm on $\mathbb{R}^n$, and $\gamma < 1$, then the MOTIS, CINNAMON, and backward-Euler style multirate algorithms are multirate A-stable.

Proof of Theorem 3: Subtracting iteration k from $k-1$ in (22), (23), and (24) results in

$$\delta x^k(t_m^i) = \delta x_i^k(t_{m-1}^i) - h_m^i\left[a_{ii}\,\delta x_i^k(t_m^i) + \sum_{j\neq i} a_{ij} I_{t_m^i}^i(\{\delta x_j^{k-1}\})\right] \quad (31)$$

$$\delta x_i^k(t_m^i) = \delta x_i^k(t_{m-1}^i) - \frac{1-e^{-h_m^i a_{ii}}}{a_{ii}} \cdot \left[a_{ii}\,\delta x_i^k(t_{m-1}^i) + \sum_{j\neq i} a_{ij} I_{t_{m-1}^i}^i(\{\delta x_j^{k-1}\})\right] \quad (32)$$

$$\delta x_i^k(t_m^i) = \delta x_i^k(t_{m-1}^i) - h_m^i\left[a_{ii}\,\delta x_i^k(t_m^i) + \sum_{j\neq i} a_{ij} I_{t_{m-1}^i}^i(\{\delta x_j^{k-1}\})\right] \quad (33)$$

where $\delta x_i^k(t_m^i) \equiv x_i^k(t_m^i) - x_i^{k-1}(t_m^i)$. Note that the linearity of the interpolation operator is exploited.

Lemma 1 can be combined with the maximum reducing property of the interpolation operator to show that the δx^k's of (31), (32), and (33) all satisfy the following inequality:

$$\max_m |\delta x_i^k(t_m^i)| \leq \sum_{j\neq i} \frac{|a_{ij}|}{a_{ii}} \max_m |\delta x_j^{k-1}(t_m^j)|. \quad (34)$$

Since A is connectedly diagonally dominant, there exists a norm such that

$$\|\delta y^k\| \leq \gamma \|\delta y^{k-1}\| \quad (35)$$

for some $\gamma < 1$ [10], where $\delta y^k \in \mathbb{R}^n$ and

$$\delta y_i^k \equiv \max_m |\delta x_i^k(t_m^i)|.$$

By comparing (30) with (35), it is clear that Lemma 2 can then be applied to complete the proof. ■

C. Voltage-Increment Time Step Control

The two most commonly used time step selection criteria are local truncation error [1] and node voltage change [6], [8], [9]. In the latter method, the time step for each circuit node is selected to ensure that the node voltage changes by a fixed increment. The voltage-increment time step control scheme combined with either XPSim or CINNAMON style algorithms has been shown to be effective for simulating MOS digital circuits if parasitics are *not* included, as these circuits are usually not tightly coupled. In this section we show by example that the voltage-increment approach is ineffective for tightly coupled problems in that it does *not* allow the XPSim algorithm to use large time steps, and must be modified to be used with a CINNAMON style algorithm.

Specifically, we consider computing the solution to the circuit in Fig. 2 using the CINNAMON and XPSim multirate algorithms, as given in (20). In this example, the circuit voltages change from 1 to 0 V, and this suggests that reasonably accurate solutions should be obtainable if the time steps are selected so as to ensure a 0.1 V change on each node. For the CINNAMON algorithm, however, this approach must be modified. Recalling the discussion in Section III, the CINNAMON algorithm tends to get stuck for tightly coupled stiff circuits, and for this example the change of voltage in one step of the CINNAMON algorithm will *never* be as large as 0.1 V, no matter how large the time step. For comparison purposes, we used as the step size for CINNAMON the time step required to make a forward-Euler algorithm change by 0.1 V. Note that Theorem 3 guarantees that this forward-Euler based time step selection strategy cannot destabilize the CINNAMON algorithm.

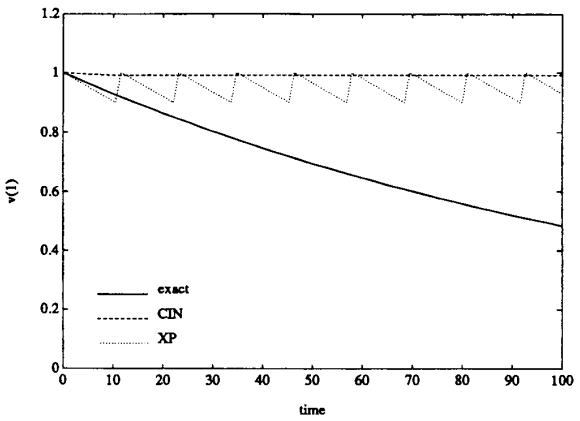


Fig. 9. $v_1(t)$ for $\Delta V = 0.1$.

The XPSim and CINNAMON multirate computed waveforms as well as the exact solution are plotted in Fig. 9. Here, piecewise-constant interpolation was used. As is clear from the figure, the voltage-increment time step control does not make it possible to take large time steps with XPSim. Also, the modified voltage-increment time step control algorithm for CINNAMON, mentioned above, did not ensure accurate results.

V. Conclusions

In this paper we have described some of the theoretical aspects of using exponential fitting for timing simulation and have focused on connectedly diagonally dominant problems, as this models the equations resulting from the way MOS transistors and interconnect parasitics are treated in timing simulation programs. We showed that for tightly coupled stiff problems, XPSim can be unstable and proved for our test problem that the MOTIS, CINNAMON, and backward-Euler integration algorithms are *multirate* A-stable. We also proved that in the limit of large time steps, CINNAMON, IPSim, and the well-known MOTIS semi-implicit algorithms produce *exactly the same results.*

The experimental results presented for tightly coupled stiff, but still connectedly diagonally dominant, problems demonstrate that backward-Euler is more accurate than any of the exponential-fitting methods, and can be *far*

more accurate for large time steps. In particular, for these problems the CINNAMON, IPSim, and MOTIS algorithms all tend to get "stuck" for large time steps. The XPSim method, on the other hand, can be unstable for large time steps because it frequently fits to a growing exponential. We also showed that ordering improves the accuracy of CINNAMON and IPSim, and can, but does not always, stabilize XPSim. Our work therefore indicates that when simulating digital circuits with interconnect parasitics, decomposing the problem into small blocks (e.g. partitioning a MOS digital circuit into logic gates) and using backward-Euler, or some other fully implicit method, will be more reliably accurate and more efficient than using exponential-fitting or semi-implicit methods.

We think this study makes one other practical point. If an event-driven time step control algorithm is being used, the fact that the CINNAMON, MOTIS, and IPSim style methods get "stuck" on tightly coupled stiff problems can fool the event-driven mechanism and produce very inaccurate results. For such problems XPSim does not get "stuck," but rather is unstable (a property shared by standard explicit multistep methods). These observations suggest that additional work on exponential fitting be focused on removing the stiffness for the XPSim style algorithms or improving the time step control for CINNAMON style algorithms [21].

Acknowledgment

The authors would like to thank Prof. R. Brayton and A. Ng at the University of California, Berkeley, and Prof. R. Rohrer at Carnegie-Mellon University for many valuable discussions. In addition the authors would like to thank the reviewers for many useful suggestions.

References

[1] L. W. Nagel, "SPICE2: A computer program to simulate semiconductor circuits," Tech. Rep. ERL-M520, Electronics Research Laboratory Report, University of California, Berkeley, May 1975.

[2] B. R. Chawla, H. K. Gummel, and P. Kozah, "MOTIS—An MOS timing simulator," *IEEE Trans. Circuits Syst.*, vol. CAS-22, pp. 901-909, Dec. 1975.

[3] R. A. Saleh, J. E. Kleckner, and A. R. Newton, "Iterated timing analysis and SPLICE1," in *Proc. Int. Conf. Computer-Aided Design* (Santa Clara, CA), Sept. 1983.

[4] B. D. Ackland and R. A. Clark, "Event-EMU: An event-driven timing simulator for MOS VLSI circuits," in *Proc. Int. Conf. Computer-Aided Design* (Santa Clara, CA), Nov. 1989.

[5] E. Sarkani and W. Liniger, "Exponential fitting of matricial multistep methods for ordinary differential equations," *Mathematics of Computation*, vol. 28, pp. 1035-1052, Oct. 1974.

[6] L. Vidigal, S. Nassif, and S. Director, "CINNAMON: Coupled INtegration and Nodal Analysis of MOs Networks," in *Proc. 23rd Design Automat. Conf.*, July 1986.

[7] R. Bauer, J. Fang, A. Ng, and R. Brayton, "XPSim: A MOS VLSI simulator," in *Proc. Int. Conf. Computer-Aided Design* (Santa Clara, CA), Nov. 1988.

[8] Y. H. Kim, S. H. Hwang, and A. R. Newton, "Electrical-logic simulation and its applications," *IEEE Trans. Computer-Aided Design*, vol. 8, pp. 8-22, Jan. 1988.

[9] P. Odryna and S. Nassif, "The ADEPT timing simulation program," *VLSI System Design*, Mar. 1986.

[10] R. S. Varga, *Matrix Iterative Analysis* (Automatic Computation Series). Englewood Cliffs, NJ: Prentice-Hall, 1962.

[11] I. W. Sandberg, "A nonnegativity-preservation property associated with certain systems of nonlinear differential equations," in *Proc. Int. Conf. Syst., Man, Cybern.*, 1974.

[12] A. Ng, private communications, 1989.

[13] G. de Micheli and A. Sangiovanni-Vincentelli, "Characterization of integration algorithms for the timing analysis of MOS VLSI circuits," *Int. J. Circuit Theory and Appl.*, vol. 10, pp. 299-309, Oct. 1982.

[14] H. C. Neto, L. M. Silveira, J. K. White, and L. M. Vidigal, "On exponential fitting for circuit simulation," in *Proc. Int. Symp. Circuits and Syst.* (New Orleans, LA), May 1990, pp. 514-518.

[15] L. M. Silveira, J. K. White, H. C. Neto, and L. M. Vidigal, "On exponential fitting for circuit simulation," Tech. Rep. VLSI Memo 90-593, Microsystems Technology Laboratories, Massachusetts Institute of Technology, Cambridge, MA, June 1990.

[16] C. F. Chen and P. Subramanyam, "The second generation MOTIS timing simulator—An efficient and accurate approach for general MOS circuits," in *Proc. Int. Symp. Circuits and Syst.* (Montreal, Canada), May 1984.

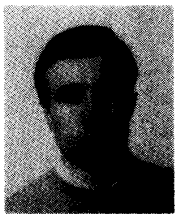
[17] H. Neto and L. Vidigal, "A multirate algorithm for fast circuit simulation," in *Proc. European Conf. Circuit Theory and Design* (Brighton, England), Sept. 1989.

[18] S. Skelboe and P. U. Andersen, "Stability properties of backward-Euler multirate formulas," *SIAM J. Sci. Stat. Comput.*, vol. 10, Sept. 1989.

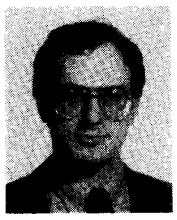
[19] E. Lelarasmee, A. Ruehli, and A. Sangiovanni-Vincentelli, "The waveform relaxation method for the time domain analysis of large scale integrated circuits," *IEEE Trans. Computer-Aided Design*, vol. CAD-1, July 1982.

[20] J. White and F. Odeh, "A connection between the convergence properties of waveform relaxation and the A-stability of multirate integration methods," in *Proc. NASECODE VII Conf.* (Copper Mountain, CO), Apr. 1991.

[21] H. C. Neto, "Multirate algorithms with exponentially fitted explicit integration for electrical circuit simulation" (in Portuguese), Ph.D. thesis, Department of Electrical and Computer Engineering, Instituto Superior Técnico, Universidade Técnica de Lisboa, Nov. 1991.



Luís Miguel Silveira (S'85) was born in Lisboa, Portugal, in 1963. He received engineer's (summa cum laude) and master's degrees in electrical and computer engineering in 1986 and 1989 from the Instituto Superior Técnico at the Technical University of Lisbon, and the M.S. and E.E. degrees in 1990 and 1991 from the Massachusetts Institute of Technology, Cambridge, MA. He is currently working toward the Ph.D. degree in electrical engineering and computer science at MIT. His research interests are in various aspects of computer-aided design of integrated circuits with emphasis on numerical simulation methods. Mr. Silveira is a member of Sigma Xi.

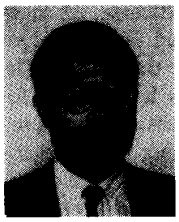


Jacob K. White (A'88) received the B.S. degree in electrical engineering and computer science from the Massachusetts Institute of Technology, Cambridge, and the S.M. and Ph.D. degrees in electrical engineering and computer science from the University of California, Berkeley.

He worked at the IBM T. J. Watson Research Center, Yorktown Heights, NY, from 1985 to 1987, was the Analog Devices Career Development Assistant Professor at the Massachusetts Institute of Technology from 1987 to 1989, and was a Presidential Young Investigator for the year 1988. He is currently an Associate Professor at MIT and his research focuses on theoretical and practical aspects of serial and parallel numerical methods for problems in circuit design and fabrication.

Horácio Neto received the electronics engineer degree in 1980 from the Instituto Superior Técnico at the Technical University of Lisbon, Portugal. He is currently completing his Ph.D. work on electrical simulation of MOS circuits.

Since 1980 he has been a Teaching Assistant in the Electronic Engineering Department of the Technical University of Lisbon, where he has taught courses on analog and digital electronics. Since 1980 he has also been a research engineer at INESC in Lisbon, where he is currently project manager on simulation and verification algorithms. His research interests include all aspects of the computer-aided design of integrated circuits, with emphasis on synthesis, simulation, and design verification.



Luís Vidigal was born in C. Rainha, Portugal, on August 4, 1948. He received the degree in electrical engineering from the Instituto Superior Tecnico, the M.Sc. degree from the University of Florida, and the Ph.D. degree from Carnegie Mellon University in 1980.

Since 1981 he has been teaching electronics at the Instituto Superior Tecnico, where he is a Full Professor in the Department of Electrical Engineering. He is also director of INESC—Instituto de Engenharia de Sistemas e Computadores, a private research organization owned by the university and Portuguese telecommunication operators.