On Exponential Fitting For Circuit Simulation

Horácio C. Neto [1] L. Miguel Silveira Jacob K. White Luís M. Vidigal [1]

Research Laboratory of Electronics
Dept. of Electrical Engineering and Computer Science
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

In this paper we present our results on the stability and accuracy properties of exponentially-fit integration algorithms, and demonstrate these properties on some test examples. We consider the multivariate test problem $\dot{\mathbf{x}} = -\mathbf{A}\mathbf{x}$ where $\mathbf{A} \in \Re^{n \times n}$ and is assumed to be irreducibly diagonally-dominant with positive diagonals, as this models the equations resulting from the way MOS circuits are treated in timing simulation programs. It is shown that for these problems, the CINNAMON exponentially-fit algorithm is A-stable, and an example is given where the algorithm in XPSim is unstable. A semi-implicit version of the XPSim algorithm is then described, and it is shown that this semi-implicit algorithm is A-stable. Examination of examples demonstrate that neither the stabilized XPSim algorithm nor the CINNAMON algorithm produces satisfactory results for very large timesteps. The effect of ordering on the accuracy and stability of the integration methods is also examined, and it is shown that ordering always enhances accuracy, though not significantly for large timesteps, and that the XPSim algorithm can be made more stable with a carefully chosen ordering.

1 Introduction

Designers of MOS digital circuits often use transistor-level simulation programs that are very fast but have limited accuracy when compared to circuit simulation programs like 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, and then by exploiting the assumption that each node has a capacitor to ground, can use simplified multistep integration algorithms [2, 3].

Recently, exponentially-fit integration methods [4] have been reinvestigated in an attempt to improve the performance and accuracy of timing simulation, as in the programs CINNAMON and XPSim [5, 6]. For the purposes of this paper, we define the first-order exponentially-fit integration algorithm as

$$x(t_{n+1}) = x(t_n) + (x(\infty) - x(t_n))(1 - e^{-\frac{h}{\tau}}) \qquad (1)$$

where the $x(\infty)$ and τ depend on the precise 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 are appealing in that when applied to numerically integrating the scalar linear differential equation $\dot{x} + dx = b$, $d, b \in \Re$, with an appropriate choice of τ and $x(\infty)$, the exact solution is produced, **no matter how large the timestep**. It is conjectured that this accuracy for scalar problems has the practical consequence that exponentially-fit integration methods retain reasonable accuracy on general problems for much larger timesteps than standard multistep methods. The theoretical justification for this large timestep behavior is limited however.

In this paper we present our results on the properties of exponentially-fit integration algorithms, and demonstrate these properties on some test examples. In the next section we show that consistency enforces a relation between $x(\infty)$ and τ, and that the methods used in both the programs CINNAMON and XPSim can be derived from Eqn. (1) using different values of $x(\infty)$ and τ. In Section 3, we describe the large timestep stability of the two exponentially-fit methods applied to a matrix test problem

$$\dot{\mathbf{x}} = -\mathbf{A}\mathbf{x} \qquad \mathbf{x}(0) = \mathbf{x}_0 \neq \mathbf{0} \qquad (2)$$

where $\mathbf{A} \in \Re^{n \times n}$. In our case, $\mathbf{A}$ is assumed to be irreducibly diagonally-dominant [7] with positive diagonals, as this models the equations resulting from the way MOS circuits are treated in timing simulation programs. It is shown that for these problems the CINNAMON exponentially-fit algorithm is A-stable, and an example is given where the algorithm in XPSim is unstable. A semi-implicit version of the XPsim algorithm is then described, and it is shown that this semi-implicit algorithm is A-stable. The example is used to also demonstrate that neither method produces satisfactory results for very large timesteps. In section 4, the effect of ordering on the accuracy and stability of the integration methods is examined. It is shown that ordering always enhances accuracy and that the XPSim algorithm can be stabilized with a carefully chosen ordering. Conclusions and acknowledgements are given in Section 5.

2 Explicit Exponential Fitting

Not all values of the $x(\infty)$ and τ parameters introduced in Eqn. (1) produce consistent integration methods, where by consistency we mean that the error introduced in one timestep, h, is $O(h^2)$. Using a Taylor series expansion of the exact solution about $x(t_n)$, we get that the exact solution for time t_{n+1} is given by

$$x^E(t_{n+1}) = x(t_n) + h\,\dot{x}(t_n) + O(h^2)$$

[1] INESC - Instituto de Engenharia de Sistemas e Computadores, Lisboa, Portugal.

whereas the approximate solution computed from (1) assuming no error on the previous timepoint yields

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

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

$$\frac{x(\infty) - x(t_n)}{\tau} = \overset{\bullet}{x}(t_n) \tag{3}$$

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

$$x(t_{n+1}) = x(t_n) + \tau(1 - e^{-\frac{h}{\tau}})\,\overset{\bullet}{x}(t_n) \tag{4}$$

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

Also note that when $h \to 0$

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

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 $[0, T]$

$$\lim_{h \to 0} \max_{0<n<N} \|x(t_n) - x^E(t_n)\| = 0 \tag{6}$$

where $t_N = T$.

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

$$\mathbf{x}(t_{n+1}) = \mathbf{x}(t_n) + \mathbf{D}^{-1}(\mathbf{I} - e^{-h\mathbf{D}})\,\overset{\bullet}{\mathbf{x}}(t_n) \tag{7}$$

where, $\mathbf{D}$ is a $N \times N$ diagonal matrix, with $d_i = \frac{1}{\tau_i}$.

Both of the approaches followed in the circuit simulators CINNAMON and XPSim can be reduced to the above formulation. It is in the choice of the fitting matrix $\mathbf{D}$ that the algorithms differ significantly. We will illustrate the difference between the two approaches for the test problem in Eqn. (2). In CINNAMON, a simple approach is used: the τ is selected from the diagonal term of the matrix as

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

and the $\mathbf{x}(\infty)$ is set according to the consistency relationship (3) to be:

$$x_i(\infty) = x_i(t_n) + \tau_i\,\overset{\bullet}{x}_i(t_n) \tag{9}$$

There is a simple circuit interpretation of the CINNAMON algorithm. Each node in the circuit is updated to a new timepoint by computing the exact solution to that node, given all the other nodes are treated as fixed voltage sources at the previous timestep.

In the case of digital logic circuitry, the steady-state value is often known, typically being equal to the power supply voltage or ground. This is exploited in the XPSim algorithm, which selects for $\mathbf{x}(\infty)$ *the correct* steady-state value. For our problem, $\mathbf{x}(\infty) = \mathbf{0}$, and by consistency

$$\tau_i = -\frac{x_i(t_n)}{\overset{\bullet}{x}_i(t_n)} \tag{10}$$

Clearly, the XPSim algorithm can not be used when the solution passes through zero. In fact, in that case XPSim is not even consistent, as the method is positive invariant. That is, the XPSim algorithm can only produce a positive $x(t_{n+1})$ from a positive $x(t_n)$. In our case 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 Eqn. (2), where $\mathbf{A}$ is strictly or irreducibly diagonally dominant, the exact solution, and the solution computed by XPSim, will be positive for all time [8].

3 Large Timestep Properties

Because exponential-fitting methods are tuned to scalar problems, they obviously are going to perform well when $\mathbf{A}$ of Eqn. (2) is strongly diagonally dominant, but they degrade surprisingly quickly when $\mathbf{A}$ is only weakly diagonally dominant. Consider the example in Figure 1, a tightly coupled two-node circuit where the initial condition at each node is 1 volt.

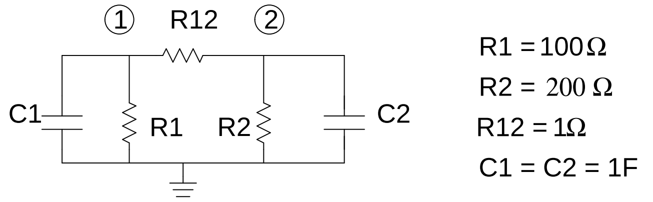

Figure 1: Test Circuit

A comparison of the computed results simulating this circuit using the backward-Euler(BE), the XPSim(XP), and the CINNAMON(CIN) algorithms (another algorithm, (IP) also in this figure, is described below) with a 0.1 second timestep is plotted in Fig. 2. As is clear from the picture, all algorithms produce roughly the the same accurate results.

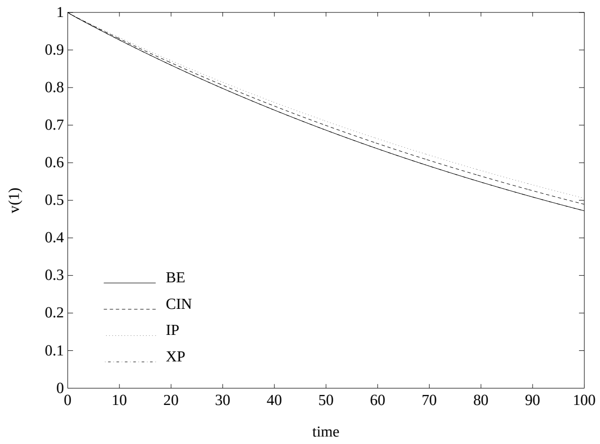

Figure 2: $v_1(t)$ for $h = 0.1$

In Fig. 3, the results from simulating the same example, but using a 1.1 second timestep, are plotted. For this case, disappointing results are achieved for the CINNAMON (CIN) and XPSim (XP) methods when compared to 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 too slowly.

If the timestep is set to 10 seconds, as in Fig. 4, 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 (the

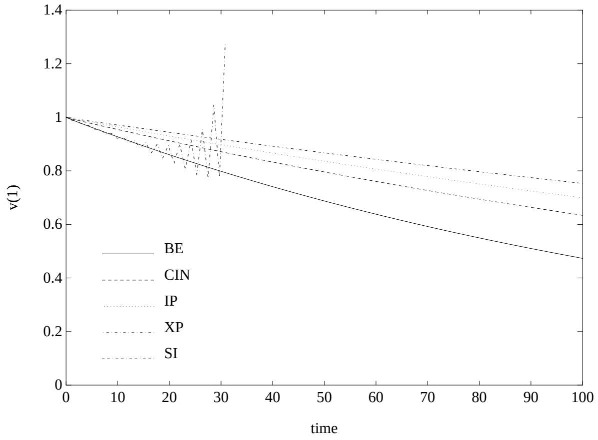

Figure 3: $v_1(t)$ for $h = 1.1$

algorithms (SI), (IP), also in these figures, are described below). This property of the CINNAMON algorithm, that of getting "stuck", is particularly insidious as the slow changing node may be misinterpreted as having achieved equilibrium. This will confuse an event-driven algorithm, and may lead to significant errors.

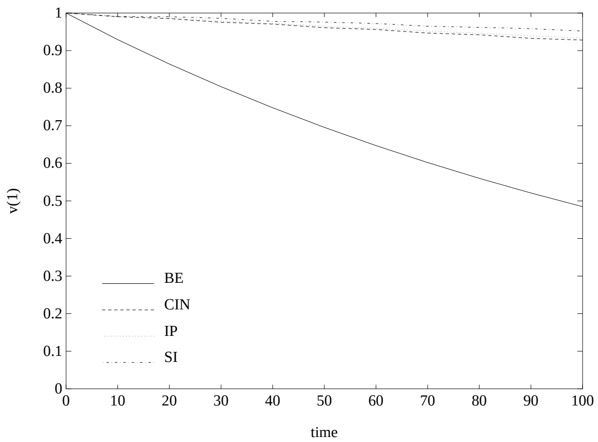

Figure 4: $v_1(t)$ for $h = 10$

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 is

$$x_i(t_{n+1}) = x_i(t_n) exp(-h \frac{a_{ii} x_i(t_{n+1}) + \sum_{j \neq i} a_{ij} x_j(t_n)}{x_i(t_n)}) \quad (11)$$

which we denote as IPSim. Note 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 algorithm used in the MOTIS [2] program, which for our test problem yields an update equation

$$x_i(t_{n+1}) = x_i(t_n) + h \left[a_{ii} x_i(t_{n+1}) + \sum_{j \neq i} a_{ij} x_j(t_n) \right]. \quad (12)$$

It is known [9] that the Gauss-Jacobi semi-implicit method is A-stable for $\mathbf{A}$ in (2) irreducibly diagonally dominant, and this result also holds for CINNAMON and IPSim, which we prove below.

Theorem 1 *When applied to solving Eqn. (2), where* $\mathbf{A}$ *is assumed to be strictly or irreducibly diagonally dominant, CINNAMON is A-stable.*

Proof:

The CINNAMON algorithm for updating x can be written in the form $\mathbf{x}(t_{n+1}) = \mathbf{M}\mathbf{x}(t_n)$ where $\mathbf{M}$ is given by

$$\mathbf{M} = \mathbf{I} - \mathbf{D}^{-1}(\mathbf{I} - e^{-h\mathbf{D}})\mathbf{A}. \quad (13)$$

Note that the individual entries of $\mathbf{M}$ are given by

$$m_{ii} = e^{-h/a_{ii}} \quad (14)$$

$$m_{ij} = (1 - e^{-h/a_{ii}}) \frac{a_{ij}}{a_{ii}} \quad i \neq j \quad (15)$$

It follows then that the CINNAMON algorithm is A-stable for our test problem if and only if the eigenvalues of $\mathbf{M}$ are inside the unit circle for all $h > 0$. That this is true can be shown by examining the maximum magnitude row sum of $\mathbf{M}$, as it is a bound on the magnitude of the eigenvalues of $\mathbf{M}$. Let $f_i(h)$ be defined as

$$f_i(h) \equiv \sum_{j=1}^{n} |m_{ij}| = (1 - e^{-h/a_{ii}}) \sum_{j \neq i}^{n} |\frac{a_{ij}}{a_{ii}}| + e^{-h/a_{ii}}. \quad (16)$$

Clearly $f_i(0) = 1$, $f_i(h) > 0$, and

$$\frac{df_i(h)}{dh} = -\frac{1}{a_{ii}} e^{-h/a_{ii}} (1 - \sum_{j \neq i}^{n} |\frac{a_{ij}}{a_{ii}}|) \quad (17)$$

is negative or equal to zero if $a_{ii} > 0$ and $\mathbf{A}$ is irreducibly diagonally dominant. Then $\forall h > 0$, $f_i(h) \leq 1$, where the equality holds for i such that $\sum_{j \neq i}^{n} |\frac{a_{ij}}{a_{ii}}| = 1$. Then $\sum_{j=1}^{n} |m_{ij}| \leq 1$, which bounds the eigenvalues of $\mathbf{M}$ to be inside or on the unit circle. However, the irreducibility property of $\mathbf{A}$ guarantees that any row i of $\mathbf{M}$ for which $\sum_{j=1}^{n} |m_{ij}| = 1$ is path connected to a row for which $\sum_{j=1}^{n} |m_{ij}| < 1$. This insures that $\mathbf{M}$ has no eigenvalues on the unit circle [7, 9], and proves the theorem. ∎

Theorem 2 *When applied to solving Eqn. (2), where* $\mathbf{A}$ *is assumed to be strictly or irreducibly diagonally dominant, and the initial condition is a positive vector, IPSim is A-stable.*

Proof:

For simplicity, we consider only the case of $\mathbf{A}$ strictly diagonally dominant, but the proof given here can be extended easily to the irreducibly diagonally dominant case. In general, the IPSim update equation for x_i can be written as

$$x_i(t_{n+1}) = x_i(t_n) exp(-h \frac{a_{ii} x_i(t_{n+1}) + \sum_{j \neq i} a_{ij} x_j(t_n)}{x_i(t_n)}). \quad (18)$$

We next show that the IPSim algorithm is A-stable, assuming that the initial condition is a positive vector, by proving that

$$max_{i \in \{1,\dots,n\}} |x_i(t_{n+1})| < max_{i \in \{1,\dots,n\}} |x_i(t_n)|. \quad (19)$$

Suppose (19) is not true, that there is some j for which $x_j(t_{n+1}) \geq max_{i \in \{1,\dots,n\}} |x_i(t_n)|$. Then,

$$a_{jj} x_j(t_{n+1}) + \sum_{k \neq j} a_{kj} x_k(t_n) \quad > 0 \quad (20)$$

as $\mathbf{A}$ is assumed to be strictly diagonally dominant and $a_{jj} > 0$. If the left-hand side of Eqn. (20) is positive, then by Eqn. (18)

$|x_j(t_{n+1})|$ must be less than $|x_j(t_n)|$ which forms the contradiction. ∎

The results using the IPSim(IP) and MOTIS(SI) algorithms to simulate the circuit in Fig. 1 with 1.01 and 10 second timesteps are plotted in Fig. 3 and Fig. 4 respectively. As the plots show, the IPSim algorithm is stable, but produces results not significantly more accurate than the MOTIS or CINNAMON algorithms. We make this statement more rigorous in the following theorem whose proof ends this section.

Theorem 3 *If* **A** *has positive diagonal entries, then in the limit as the timesteps becomes large, the CINNAMON, IPSim, and MOTIS algorithms produce identical results.*

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

$$x_i(t_{n+1}) = x_i(t_n) + h\left[a_{ii}x_i(t_{n+1}) + \sum_{j\neq i} a_{ij}x_j(t_n)\right], \quad (21)$$

$$x_i(t_{n+1}) = x_i(t_n) - \frac{1-e^{-ha_{ii}}}{a_{ii}}\left[\sum_{j=1}^{n} a_{ij}x_j(t_n)\right], \quad (22)$$

$$x_i(t_{n+1}) = x_i(t_n)exp(-h\frac{a_{ii}x_i(t_{n+1}) + \sum_{j\neq i} a_{ij}x_j(t_n)}{x_i(t_n)}). \quad (23)$$

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

$$x_i(t_{n+1}) = -\sum_{j\neq i}\frac{a_{ij}}{a_{ii}}x_j(t_n). \quad (24)$$

The result in Eqn. (24) 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 (23) cannot go to ∞. Therefore, the term from the numerator of the exponential's argument in Eqn. (23)

$$a_{ii}x_i(t_{n+1}) + \sum_{j\neq i} a_{ij}x_j(t_n) \quad (25)$$

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

4 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 [3]. Specifically, when calculating $x_i(t_{n+1})$ the use of the already computed values of $x_j(t_{n+1})\ \forall j < i$ will improve the accuracy of the solution. This has previously been exploited in several implementations of the semi-implicit integration algorithm and also in the CINNAMON circuit simulator [10].

Consider as an example the ordered XPSim algorithm, the update equation for which is

$$x_i(t_{n+1}) = x_i(t_n)\ exp\ \left[h\frac{\sum_{j<i} a_{ij}x_j(t_{n+1}) + \sum_{j\geq i} a_{ij}x_j(t_n)}{x_i(t_n)}\right] \quad (26)$$

where subscript index indicates the ordering.

In Figure 5 we present the plots obtained for the test circuit, under the same conditions of Figure 3, i.e., using a 1.1 second timestep. The waveforms shown correspond to backward-Euler (BE), CINNAMON (CINo), IPSim (IPo), XPSim (XPo), and the Gauss-Seidel version of the semi-implicit algorithm (SIo). For all but backward-Euler, the waveforms were computed using a random fixed ordering of the equations. From the figure, and comparing to the results shown in Figure 3, one can see that ordering improved the accuracy of all methods, most notably that of XPSim which, in this problem, is stabilized with this ordering scheme. Also from the plot we note that with ordering, the solution produced with CINNAMON becomes quite accurate and also that the solution produced with IPSim is no better than that obtained with the semi-implicit algorithm, albeit at the expense of more computation.

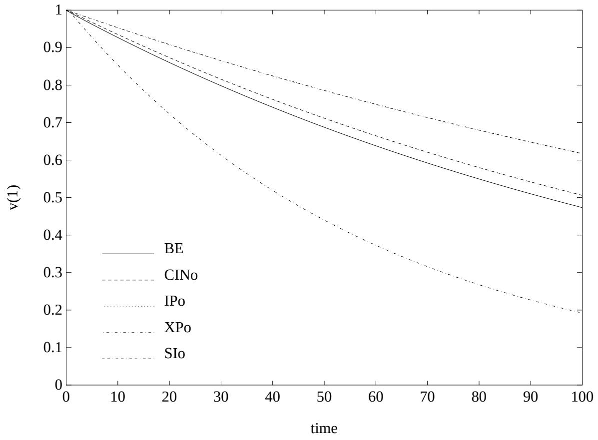

Figure 5: $v_1(t)$ for $h = 1.1$

From these results it is clear that the ordering scheme plays an important role, and an obvious question is as to whether some ordering schemes are naturally better than others. Also, the question of whether ordering does in fact stabilize the XPSim algorithm becomes relevant. In Figure 6 we show the plots obtained on the test circuit, using a large timestep ($h = 10$) by applying a *most-changed* ordering scheme to both XPSim and CINNAMON and comparing it to backward-Euler which is within 10% of the exact solution. The *most-changed* ordering scheme is obtained by calculating at each point in time which of the nodes will vary the most, and updating that node first. This scheme seems to stabilize XPSim as far as the experiments we did, but we note that for a large circuit it becomes computationally very expensive to compute the ordering. Also, from the plots we note that the waveform produced by ordered-CINNAMON decays more slowly than the exact solution, although the ordering helps it not getting completely "stuck", while the one produced by ordered-XPSim is much faster than the exact solution.

Clearly from this plot we note that for large timesteps the accuracy of these methods is poor, but assuming that some ordering scheme can be applied that stabilizes XPSim or avoids CINNAMON getting "stuck", a timestep control mechanism can be applied to produce the required accuracy on the computed solution. Of the methods mentioned, CINNAMON presents the lowest computational complexity together with the semi-implicit integration algorithm

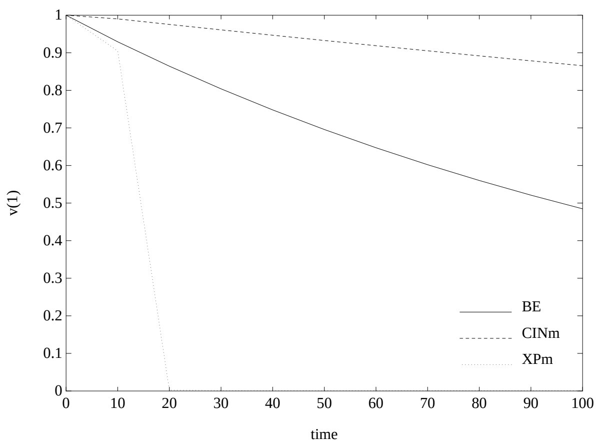

Figure 6: $v_1(t)$ for $h = 10$

5 Conclusions and Acknowledgments

In this paper we described some of the theoretical and practical aspects of using exponential-fitting for MOS digital circuit timing analysis. None of the methods are more accurate than backward-Euler for large timesteps and nondiagonal problems, although the methods may prove to be computationally less expensive. Of particular concern is the property that the CINNAMON exponential-fitting algorithms tend to get "stuck" for large timesteps, where as the XPSIM algorithms tend to be unstable. Which is preferable will depend on whether efficiency or reliability is the more important concern.

The authors would like to thank Prof. B. Brayton and A. Ng at the University of California, Berkeley for many valuable discussions. This work was supported by a research initiation grant from A. T. & T., the National Science Foundation, the Defense Advanced Research Projects Agency contract N00014-87-K-825, and the Portuguese INVOTAN committee.

References

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

[2] B. R. Chawla, H. K. Gummel, P. Kozah, "MOTIS - an MOS Timing Simulator", *IEEE Trans. on Circuits and Systems*, Vol. CAS-22, December 1975, pp.901-909.

[3] R. A. Saleh, J. E. Kleckner, A. R. Newton, "Iterated Timing Analysis and SPLICE1", *Proc. Int. Conf. on Computer-Aided Design*, Santa Clara, California, September 1983.

[4] Sarkani, E., and Liniger, W., "Exponential Fitting of Matricial Multistep Methods for Ordinary Differential Equations", *Mathematics of Computation*, October 1974

[5] Vidigal, L., Nassif, S., and Director, S., "CINNAMON: Coupled INtegration and Nodal Analysis of MOs Networks", *23rd Design Automation Conference*, July 1986.

[6] R. Bauer, J. Fang, A. Ng, R. Brayton, "XPSim: A MOS VLSI Simulator", *International Conference on Computer-Aided Design*, Santa Clara, November 1988.

[7] R. S. Varga, *Matrix Iterative Analysis*, Prentice-Hall Inc., Englewood Cliffs, New Jersey, 1962.

[8] Sandberg, I. W. "A nonnegativity-preservation property associated with certain systems of nonlinear differential equations," *Proc. Int. Conf. on Systems, Man, and Cybernetics*, 1974.

[9] J. White and A. Sangiovanni-Vincentelli, "Relaxation Techniques for the Simulation of VLSI Circuits", *Kluwer Academic Publishers*, 1986.

[10] Horácio Neto, Luís Vidigal, "A Multirate Algorithm for Fast Circuit Simulation", *European Conference in Circuit Theory and Design*, Brighton, September 1989.