Comparison of Reduced-Order Interconnect Macromodels for Time-Domain Simulation

Timo Palenius and Janne Roos

***Abstract*—A typical integrated-circuit model consists of nonlinear transistor models and large linear *RLC* networks describing the interconnects. During the last decade, various model-reduction algorithms have been developed for replacing each *RLC* network with an approximately equivalent, but much smaller, model. Since these reduced-order models are described in the frequency domain, they have to be linked to the transient analysis of the whole nonlinear circuit, which can be done by replacing these models with appropriate macromodels. In the interconnect literature, the actual macromodel realization, which has a great impact on the transient-simulation CPU time, is often overlooked. This paper presents a comprehensive comparison of nine reduced-order interconnect macromodels for time-domain simulation: the macromodels are reviewed, presented in a unified manner, and compared both theoretically and numerically. Since we have implemented all the nine macromodels into the APLAC circuit simulation and design tool, we are able to present a fair and meaningful CPU–time comparison.**

***Index Terms*—Interconnect simulation, macromodeling, model-order reduction, transient analysis.**

I. Introduction

As OPERATION frequencies and integration densities of digital very large scale integration (VLSI) circuits increase while device sizes shrink, there is a growing need to correctly model the interconnects between transistors or, e.g., those in package wiring. A typical integrated-circuit model consists of nonlinear transistor models and linear *RLC* networks describing the interconnects. Since the size of these *RLC* networks can be huge, various model-reduction algorithms [1]–[11] have been developed for replacing them with reduced-order models. This paper deals with the reduction of the important class of lumped *RLC* interconnect models; consequently, model-reduction algorithms that are tailored for, say, *RC* circuits only (e.g., [1]) or algorithms that can handle (dispersive multiconductor) transmission lines (e.g., [2] and [3]) are not considered. Any model-reduction algorithm should preserve the passivity of the original interconnect model in order to produce stable systems when connected to the rest of the circuitry [4]. Thus, algorithms like asymptotic waveform evaluation (AWE) [5], Padé via Lanczos (PVL) [6], or complex frequency hopping (CFH) [7], which do not preserve passivity, are not considered in this paper. Finally, the result of the reduction should fit naturally into the modified nodal analysis (MNA), which is routinely used in circuit simulators to formulate circuit equations. Therefore, the recently proposed algorithm efficient nodal order reduction (ENOR) [8], which produces z-parameters (instead of y-parameters) is not considered here.

In this paper, the passive reduced-order interconnect macromodeling algorithm (PRIMA) [4] will be used for reduction as its algorithmic steps provide good starting points for several macromodel realizations. Note, however, that the macromodeling discussion here is by no means limited to PRIMA; other passive multiinput–multioutput (MIMO) algorithms like the one proposed in [9], block rational Arnoldi [10], and SyMPVL [11] could also be used.

With PRIMA(-like) algorithm(s), the result of reduction for each *RLC* network is a reduced-order admittance matrix, where each y-parameter is typically given in terms of a set of dominant poles and the corresponding residues. Once this frequency-domain reduced-order model has been obtained, it has to be linked to the transient analysis of the whole nonlinear circuit since the simulation of digital circuits is usually performed in the time domain. The desired link can be created by replacing the reduced-order model with an appropriate macromodel. There are basically two ways of generating these macromodels [12, Ch. 7], which are as follows.

1) *Synthesis*: an equivalent-circuit macromodel, or a SPICE-netlist, is synthesized using basic circuit elements. Any time-domain circuit simulator can then be used.
2) *Recursive convolution*: a time-varying macromodel is generated. For many simulators, this method requires a modification of a simulator source code.

In the literature, several macromodels have been proposed [3], [4], [12]–[26]. In this paper, we will discuss most of these macromodels; the equivalent-circuit macromodels [13]–[15] and the time-varying macromodel [16] are *not* discussed further due to the following reasons. In [13] and [14], the macromodel was realized using z-parameters, while in [15] and [16], a set of rational functions, i.e., not poles and residues, were realized. (Naturally, if really needed, both z-parameters and rational functions can be more or less easily converted into a form suitable for the macromodeling methods discussed in this paper.)

In the literature, various model-reduction algorithms have been presented, while the actual realization of the reduced-order model, which has a great impact on the transient-simulation

Manuscript received December 15, 2003; revised June 1, 2004. This work was supported in part by the National Technology Agency of Finland under Grant 2586/31/02, by the Nokia Corporation, and by the APLAC Solutions Corporation.

T. Palenius was with the Circuit Theory Laboratory, Department of Electrical and Communications Engineering, Helsinki University of Technology, Espoo FIN-02015 HUT, Finland. He is now with the APLAC Solutions Corporation, Espoo FIN-02600, Finland (e-mail: timo.palenius@aplac.com).

J. Roos is with the Circuit Theory Laboratory, Department of Electrical and Communications Engineering, Helsinki University of Technology, Espoo FIN-02015 HUT, Finland (e-mail: janne@ct.hut.fi).

Digital Object Identifier 10.1109/TMTT.2004.834562

CPU time, is often overlooked. Even when the focus has been in macromodeling, the macromodels have rarely been compared to each other systematically. In fact, to the authors' best knowledge, the only systematic macromodel comparisons are given in [4], [17], and [18]; these comparisons are discussed further in Section V-D.

In this paper, we will present a comprehensive comparison of nine reduced-order interconnect macromodels for time-domain simulation. Five equivalent-circuit and four time-varying macromodels are reviewed, presented in a unified manner, related to PRIMA, and compared both theoretically and numerically. Since we have implemented all the nine macromodels into the APLAC circuit simulation and design tool [27], we are able to present a fair and meaningful CPU–time comparison.

In Section II, some useful background information is given. The five equivalent-circuit macromodels and the four time-varying macromodels are reviewed in Sections III and IV, respectively. In Section V, we present a thorough comparison of the nine macromodels along with two simulation examples. Finally, in Section VI, we present conclusions.

II. Background

Here, we present a brief overview of PRIMA, describe the steps needed to handle complex eigenvalues, and define the state-variable formulation used in the derivation of some of the macromodels.

A. PRIMA

The MNA [28] equations (with the rows corresponding to the current variables negated [4]) for an N-port can be written as

$$\begin{cases} \mathbf{C}\dfrac{\mathrm{d}\mathbf{x}(t)}{\mathrm{d}t} &= -\mathbf{G}\mathbf{x}(t)+\mathbf{B}\mathbf{u}(t) \\ \mathbf{i}(t) &= \mathbf{L}^{\mathrm{T}}\mathbf{x}(t) \end{cases} \tag{1}$$

where $\mathbf{C}$ and $\mathbf{G}$ are susceptance and conductance matrices, respectively, and $\mathbf{B}=\mathbf{L}$ is a selector matrix consisting of 1's, -1's, and 0's. Vector $\mathbf{x}(t)$ contains the nodal voltages (and some branch currents), and $\mathbf{u}(t)$ and $\mathbf{i}(t)$ are the port voltage and current vectors, respectively. Here, we assume that the dimension of the $\mathbf{C}$ and $\mathbf{G}$ matrices, n, is a large number and we are only interested in the behavior of the N-port as seen from the port nodes. Consequently, we seek a reduced-order model with dimension $q \ll n$ that approximates the behavior of the N-port sufficiently well in the frequency band of interest.

PRIMA transforms (1) into

$$\begin{cases} \tilde{\mathbf{C}}\dfrac{\mathrm{d}\tilde{\mathbf{x}}(t)}{\mathrm{d}t} &= -\tilde{\mathbf{G}}\tilde{\mathbf{x}}(t)+\tilde{\mathbf{B}}\mathbf{u}(t) \\ \mathbf{i}(t) &= \tilde{\mathbf{L}}^{\mathrm{T}}\tilde{\mathbf{x}}(t) \end{cases} \tag{2}$$

where the reduced-order matrices are obtained from $\tilde{\mathbf{C}}=\mathbf{X}^{\mathrm{T}}\mathbf{C}\mathbf{X}$, $\tilde{\mathbf{G}}=\mathbf{X}^{\mathrm{T}}\mathbf{G}\mathbf{X}$, $\tilde{\mathbf{B}}=\mathbf{X}^{\mathrm{T}}\mathbf{B}$, and $\tilde{\mathbf{L}}=\mathbf{X}^{\mathrm{T}}\mathbf{L}$. Here, the matrix $\mathbf{X}$ is a $n \times q$ congruence-transformation matrix obtained after q/N (rounded to an appropriate integer) iterations of the block Arnoldi algorithm [4].

Next, the first equation of (2) is premultiplied with $\tilde{\mathbf{G}}^{-1}$. Assuming that a basis of eigenvectors exists for the matrix $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}$, it can be written as $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}=\mathbf{S}\mathbf{\Lambda}\mathbf{S}^{-1}$, where $\mathbf{\Lambda}$ is a diagonal matrix containing the eigenvalues of $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}$ as its diagonal elements and $\mathbf{S}$ has the corresponding q eigenvectors as its columns. After premultiplying with $\mathbf{S}^{-1}$, (2) can be written as

$$\begin{cases} \mathbf{S}^{-1}\mathbf{S}\Lambda\mathbf{S}^{-1}\dfrac{\mathrm{d}\tilde{\mathbf{x}}(t)}{\mathrm{d}t} &= -\mathbf{S}^{-1}\tilde{\mathbf{x}}(t)+\mathbf{S}^{-1}\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{B}}\mathbf{u}(t) \\ \mathbf{i}(t) &= \tilde{\mathbf{L}}^{\mathrm{T}}\mathbf{S}\mathbf{S}^{-1}\tilde{\mathbf{x}}(t) \end{cases} \tag{3}$$

or, if we assume a change of variables $\mathbf{S}^{-1}\tilde{\mathbf{x}} \rightarrow \tilde{\mathbf{x}}$ as

$$\begin{cases} \mathbf{\Lambda}\dfrac{\mathrm{d}\tilde{\mathbf{x}}(t)}{\mathrm{d}t} &= -\mathbf{I}\tilde{\mathbf{x}}(t)+\mathbf{H}\mathbf{u}(t) \\ \mathbf{i}(t) &= \mathbf{E}^{\mathrm{T}}\tilde{\mathbf{x}}(t) \end{cases} \tag{4}$$

where $\mathbf{H}=\mathbf{S}^{-1}\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{B}}$, $\mathbf{E}=\mathbf{S}^{\mathrm{T}}\tilde{\mathbf{L}}$, and $\mathbf{I}$ is the $q \times q$ unity matrix. Note that (4) has the same dimensions as (2), but the coefficient matrices $\mathbf{\Lambda}$ and $\mathbf{I}$ are now diagonal.

Let us now consider how the port current $\mathbf{I}(s)=\mathcal{L}\{\mathbf{i}(t)\}$ can be written in terms of poles and residues. Consider the mth row in the first equation of (4) as follows:

$$\Lambda_m\frac{\mathrm{d}\tilde{x}_m(t)}{\mathrm{d}t}=-\tilde{x}_m(t)+\sum_{j=1}^{N}H_{mj}u_j(t). \tag{5}$$

Laplace-transforming (5) (assuming zero initial conditions) and solving for $\tilde{X}_m(s)$ yields

$$\tilde{X}_m(s)=\sum_{j=1}^{N}\frac{H_{mj}}{s\Lambda_m+1}U_j(s). \tag{6}$$

The current of port i, or the ith element of $\mathbf{I}(s)$, can now be written as

$$\begin{aligned} I_i(s) &= \sum_{m=1}^{q}E_{im}^{\mathrm{T}}\tilde{X}_m(s) \\ &= \sum_{j=1}^{N}\sum_{m=1}^{q}\frac{E_{im}^{\mathrm{T}}H_{mj}}{s\Lambda_m+1}U_j(s) \\ &= \sum_{j=1}^{N}\sum_{m=1}^{q}\frac{k_{ijm}}{s-p_m}U_j(s) \\ &= \sum_{j=1}^{N}\tilde{Y}_{ij}(s)U_j(s) \end{aligned} \tag{7}$$

where $p_m=-\Lambda_m^{-1}$ is the mth pole of the admittance $\tilde{Y}_{ij}(s)$ and $k_{ijm}=E_{im}^{\mathrm{T}}H_{mj}\Lambda_m^{-1}$ is the corresponding residue.

B. Complex Eigenvalues

Some of the eigenvalues of the real matrix $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}$ may be complex numbers, in which case, they appear in complex-conjugate pairs. Let us assume that q_r of the eigenvalues are real and the rest appear in q_{c} conjugate pairs such that $q=q_r+2q_{\mathrm{c}}$. Consider one such pair, $\Lambda_m^{\mathrm{r}} \pm \mathrm{j}\Lambda_m^{\mathrm{i}}$. The corresponding eigenvectors and, therefore, also the corresponding rows of matrices $\mathbf{H}$ and $\mathbf{E}$ in (4), are complex conjugate. Let the corresponding elements of vector $\tilde{\mathbf{x}}$ be $\tilde{x}_m^{\mathrm{r}} \pm \mathrm{j}\tilde{x}_m^{\mathrm{i}}$. Inserting these into (5), and requiring

the real and imaginary parts of the equation to hold independently, yields (the same pair of equations is obtained twice)

$$\begin{cases} \Lambda_m^{\mathrm{r}} \dfrac{\mathrm{d}\tilde{x}_m^{\mathrm{r}}(t)}{\mathrm{d}t} &= -\tilde{x}_m^{\mathrm{r}}(t) + \Lambda_m^{\mathrm{i}} \dfrac{\mathrm{d}\tilde{x}_m^{\mathrm{i}}(t)}{\mathrm{d}t} + \sum\limits_{j=1}^{N} H_{mj}^{\mathrm{r}} u_j(t) \\ \Lambda_m^{\mathrm{r}} \dfrac{\mathrm{d}\tilde{x}_m^{\mathrm{i}}(t)}{\mathrm{d}t} &= -\Lambda_m^{\mathrm{i}} \dfrac{\mathrm{d}\tilde{x}_m^{\mathrm{r}}(t)}{\mathrm{d}t} - \tilde{x}_m^{\mathrm{i}}(t) + \sum\limits_{j=1}^{N} H_{mj}^{\mathrm{i}} u_j(t). \end{cases} \tag{8}$$

Repeating the steps of (6) and (7) yields the contribution of the particular eigenvalue pair into the port current $I_i(s)$ in terms of the poles and residues that are, again, complex-conjugate pairs.

C. State-Variable Formulation

Consider one (generic) term in the summation of (7)

$$I(s) = \frac{k}{s-p} U(s) \mathrel{\hat{=}} kX(s) \tag{9}$$

where $X(s)$ is a state variable. The state-space model for this pole/residue pair can be written in the time domain (assuming zero initial conditions) as

$$\begin{cases} \dfrac{\mathrm{d}x(t)}{\mathrm{d}t} &= px(t) + u(t) \\ i(t) &= kx(t). \end{cases} \tag{10}$$

Similarly, a state-space model can be obtained for a pair of complex poles/residues. Let the complex conjugate poles and the corresponding residues be $a \pm \mathrm{j}b$ and $c \pm \mathrm{j}d$, respectively, and the complex conjugate state variables $x^{\mathrm{r}} \pm \mathrm{j}x^{\mathrm{i}}$. As before, inserting these into (10) and requiring the real and imaginary parts of the equation to hold independently yields

$$\begin{cases} \dfrac{\mathrm{d}x^{\mathrm{r}}(t)}{\mathrm{d}t} &= ax^{\mathrm{r}}(t) - bx^{\mathrm{i}}(t) + u(t) \\ \dfrac{\mathrm{d}x^{\mathrm{i}}(t)}{\mathrm{d}t} &= bx^{\mathrm{r}}(t) + ax^{\mathrm{i}}(t) \\ i(t) &= 2cx^{\mathrm{r}}(t) - 2dx^{\mathrm{i}}(t). \end{cases} \tag{11}$$

Possibly the simplest state-space coordinate system is obtained by introducing the following change of variables: $2x^{\mathrm{r}} \to x^{\mathrm{r}}$ and $-2x^{\mathrm{i}} \to x^{\mathrm{i}}$. The final state-space model for a pair of complex poles can then be written as

$$\begin{cases} \dfrac{\mathrm{d}x^{\mathrm{r}}(t)}{\mathrm{d}t} &= ax^{\mathrm{r}}(t) + bx^{\mathrm{i}}(t) + 2u(t) \\ \dfrac{\mathrm{d}x^{\mathrm{i}}(t)}{\mathrm{d}t} &= -bx^{\mathrm{r}}(t) + ax^{\mathrm{i}}(t) \\ i(t) &= cx^{\mathrm{r}}(t) + dx^{\mathrm{i}}(t). \end{cases} \tag{12}$$

The above change of the coordinate system can also be obtained by applying an appropriate similarity transformation to the state-space model [13], [19].

In nonlinear circuit simulation, the transient analysis is always preceded by a dc analysis. To ensure that the port voltages at time $t = 0$ coincide with the dc voltages, proper initial values for the state variables must be used. These are obtained by setting the time derivatives to zero in (10) and (12) as follows:

$$x(0) = -\frac{U_{\mathrm{DC}}}{p},\ x^{\mathrm{r}}(0) = -\frac{2aU_{\mathrm{DC}}}{a^2+b^2},\ x^{\mathrm{i}}(0) = -\frac{2bU_{\mathrm{DC}}}{a^2+b^2} \tag{13}$$

where U_{DC} is the dc port voltage.

III. Equivalent-Circuit Realizations

The macromodels presented here are realized with equivalent circuits consisting only of linear, constant value (i.e., time-independent) voltage-controlled current sources (static VCCSs) or voltage-controlled charge sources (dynamic VCCSs), which contain linear resistors and capacitors, respectively, as special cases. Therefore, these macromodels can be realized as a SPICE netlist without touching the simulator's internal source code. They are easy to implement, and the circuit simulator will automatically handle the calculation of the local truncation error (LTE) and the new transient analysis time step. The drawback is that these macromodels will necessarily generate additional nodes into the circuit. This tends to slow down the transient (as well as dc, ac, harmonic balance, etc.) analysis, as the size of the matrix equation that has to be solved grows.

For clarity, we will present the entire equivalent circuits for the methods presented in Sections III-A and B. For the rest of the methods, we will adopt a convention that only one VCCS at the ith port and the equivalent-circuit realizations for one real eigenvalue (pole, state variable) and one complex-conjugate eigenvalue (pole, state variable) pair will be shown in the figures. The generic summation indices i and j in the figures refer to the summations over the source and controlling ports, respectively (as in $\tilde{Y}_{ij}$), and m refers to the summation over the eigenvalues (poles, state variables) in the reduced-order model.

A. Direct Stamping I

In [20], Odabasioglu *et al.* suggested that (2) can be directly stamped into the MNA matrix of the whole circuit as

$$\begin{bmatrix} \text{Stamps for} & \mathbf{0} & \mathbf{0} \\ f(\mathbf{x}_{\mathrm{NL}}, \mathbf{u}) & \mathbf{I} & \mathbf{0} \\ \mathbf{0} \quad \mathbf{0} & \mathbf{I} & -\tilde{\mathbf{L}}^{\mathrm{T}} \\ \mathbf{0} \quad -\tilde{\mathbf{B}} & \mathbf{0} & \tilde{\mathbf{G}} + \tilde{\mathbf{C}}\dfrac{\mathrm{d}}{\mathrm{d}t} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathrm{NL}} \\ \mathbf{u} \\ \mathbf{i} \\ \tilde{\mathbf{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{j}_{\mathrm{NL}} \\ \mathbf{j} \\ \mathbf{0} \\ \mathbf{0} \end{bmatrix} \tag{14}$$

where $\mathbf{x}_{\mathrm{NL}}$ denotes the MNA variables of the nonlinear parts of the circuit and $\mathbf{u}$ and $\mathbf{i}$ are the port voltages and currents, respectively. Vectors $\mathbf{j}_{\mathrm{NL}}$ and $\mathbf{j}$ denote the sources connected to the nonlinear and port nodes.

A matrix stamp can be realized using any circuit simulator by finding an equivalent circuit that produces the same stamp. It is easy to see, using elementary circuit analysis, that the equivalent circuit presented in Fig. 1 is governed by (14). Note that the $N+q$ additional MNA variables, the elements of $\mathbf{i}$ and $\tilde{\mathbf{x}}$, correspond to the unknown nodal voltages of the $N+q$ additional nodes introduced by this equivalent circuit.

In [4], a slightly modified version of (14) was given: the last row of (14) was premultiplied with $\tilde{\mathbf{G}}^{-1}$. This may seem, at first glance, like a trivial modification, but it might have, depending on the implementation, some impact on the simulation speed. Namely, the elements of matrices $\tilde{\mathbf{C}}$, $\tilde{\mathbf{G}}$, $\tilde{\mathbf{B}}$, and $\tilde{\mathbf{L}}$ are all, in general, nonzero, whereas $\mathbf{I}$ is a diagonal matrix and the reduced-order matrix $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}$ has a special structure: as a built-in property of PRIMA, $\tilde{\mathbf{G}}^{-1}\tilde{\mathbf{C}}$ is a block Hessenberg matrix [4], implying that a large part of its elements are typically zero. If the equivalent circuit is realized such that only nonzero VCCSs are created, some speed-up can be expected. The number of additional nodes is still $N+q$.

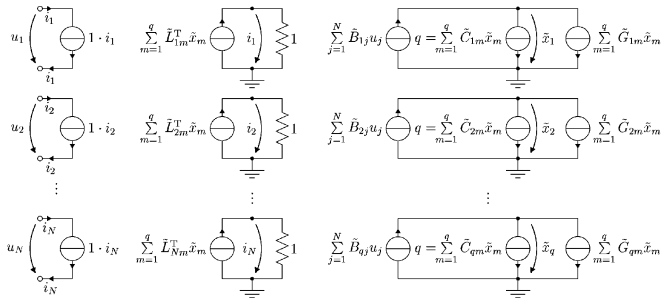

Fig. 1. Direct stamping I.

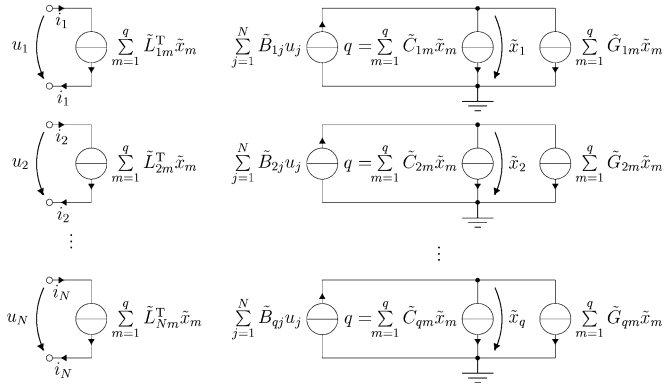

Fig. 2. Direct stamping II.

B. Direct Stamping II

As can be seen from Fig. 1, the port currents $\mathbf{i}$ in (14) are not needed in the realization. Eliminating $\mathbf{i} = \tilde{\mathbf{L}}^{\mathrm{T}}\tilde{\mathbf{x}}$ from (14) results in [21]

$$\begin{bmatrix} \text{Stamps for} & \mathbf{0} \\ f(\mathbf{x}_{\mathrm{NL}}, \mathbf{u}) & \tilde{\mathbf{L}}^{\mathrm{T}} \\ \mathbf{0} \quad -\tilde{\mathbf{B}} & \tilde{\mathbf{G}} + \tilde{\mathbf{C}}\dfrac{\mathrm{d}}{\mathrm{d}t} \end{bmatrix} \begin{bmatrix} \mathbf{x}_{\mathrm{NL}} \\ \mathbf{u} \\ \tilde{\mathbf{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{j}_{\mathrm{NL}} \\ \mathbf{j} \\ \mathbf{0} \end{bmatrix}. \quad (15)$$

The corresponding equivalent circuit is shown in Fig. 2. Now, the number of additional nodes is only q. Naturally, the last row of (15) could be premultiplied with $\tilde{\mathbf{G}}^{-1}$, as before.

C. Matsumoto's Method

In [18], Matsumoto *et al.* proposed an equivalent circuit, which is a realization of (5) and (8). The equivalent circuit, which produces q additional nodes, is shown in Fig. 3. The nodal equations for, e.g., the circuit in Fig. 3(b), can be written as

$$\sum_{j=1}^{N} H_{mj} U_j = (s\Lambda_m + 1)\,\tilde{X}_m \quad (16)$$

which yields (5) when $s\tilde{X}_m$ is replaced with $\mathrm{d}\tilde{x}_m/\mathrm{d}t$.

The equivalent circuit shown in Fig. 3(c) contains two dynamic VCCSs taking their controls $\tilde{x}^{\mathrm{r}}_m$ and $\tilde{x}^{\mathrm{i}}_m$ from across each other. In [18], a slightly different equivalent circuit was used, where, instead of two dynamic VCCSs, there were three capacitors connected in a π configuration. These two realizations are completely equivalent: both produce exactly the same matrix stamp, and they can be obtained from one another with

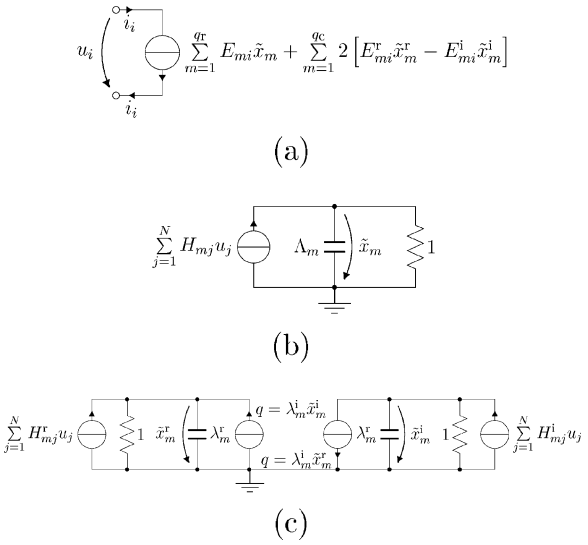

Fig. 3. Matsumoto's method. (a) A port VCCS. (b) Realization of a real eigenvalue. (c) Realization of a complex eigenvalue pair.

elementary circuit transformations. We have chosen the one in Fig. 3 because it bears closer resemblance to other methods considered in this paper, and because it creates one less VCCS for each complex-pole pair.

D. Transfer-Function Realization

The driving-point or transfer admittance given by (7) can be realized with an equivalent circuit that exploits dynamic VCCSs to synthesize the s terms in the denominator [21]. Complex poles $a+\mathrm{j}b$ and their corresponding residues $c+\mathrm{j}d$ can be combined with their conjugate pairs and the expression for the port current can be expanded as

$$I_i(s) = \sum_{j=1}^{N}\sum_{m=1}^{q_{\mathrm{c}}} \frac{2c_{ijm}s - 2(a_m c_{ijm} + b_m d_{ijm})}{(s-a_m)^2 + b_m^2} U_j(s). \quad (17)$$

The equivalent circuit, which creates $2N(q_r + 2Nq_{\mathrm{c}})$ additional nodes, is shown in Fig. 4.

E. Differential-Equation Macromodel

The state-space model of (10) and (12) can be realized with a relatively simple equivalent circuit [12], [22], [23], as shown in Fig. 5. The number of additional nodes is Nq.

IV. Time-Varying Macromodels

The macromodels presented here consist of linear time-varying VCCSs and independent current sources at the N-ports, as shown in Fig. 6. The values of the sources at each time point are calculated using internal state variables. Note that implementation of these methods usually requires additions to the simulator's source code. The advantage of these methods is that no additional nodes are created, but the values of the time-dependent (trans)conductance and the equivalent port-current source have to be updated at each time point. The conductance g can be identified as the coefficient of the (unknown) present-time voltage in the port current. All

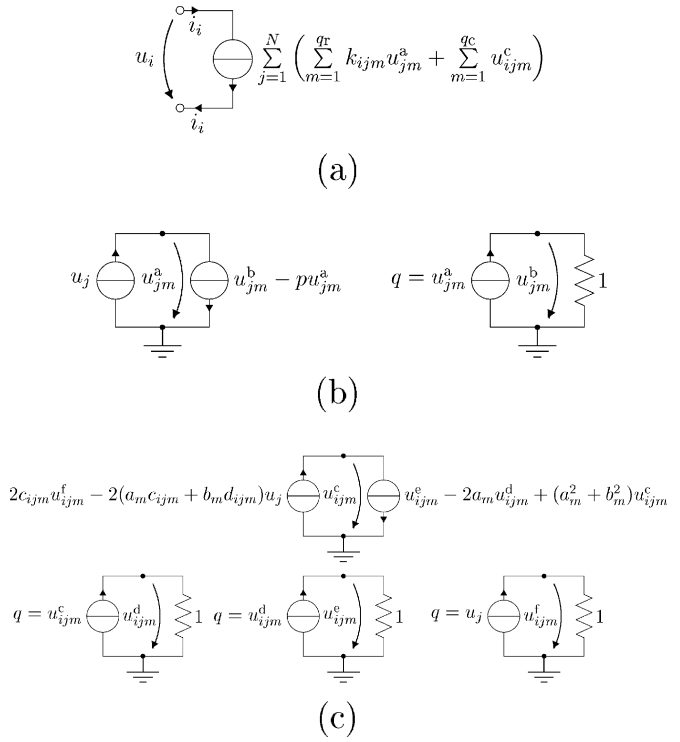

Fig. 4. Transfer-function realization. (a) A port VCCS. (b) Realization of a real pole. (c) Realization of a complex pole pair.

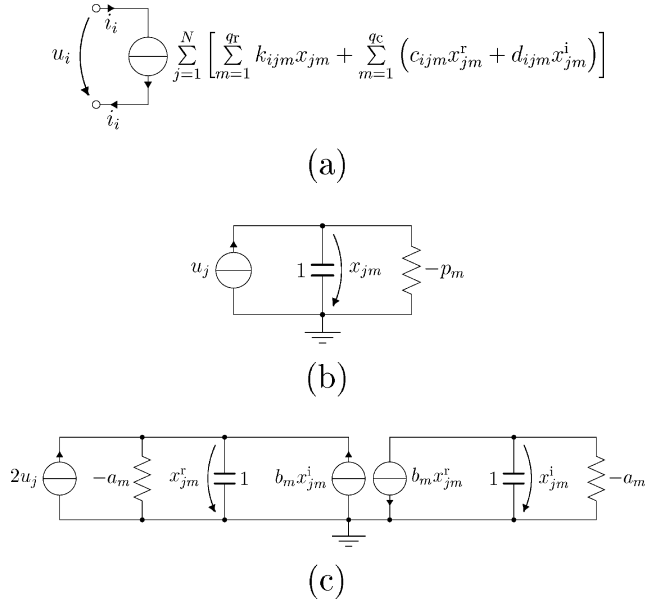

Fig. 5. Differential-equation macromodel. (a) A port VCCS. (b) Realization of a real state variable. (c) Realization of a complex state-variable pair.

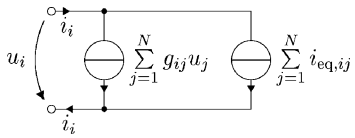

Fig. 6. Port VCCS and an independent current source.

the other terms in the port current form the equivalent port current-source $i_{\rm eq}$.

- Solve the DC voltages and set `t=0.0;`

`while (t<tend) {`

- Calculate new time step Δt_k
- Calculate g and $i_{\rm eq}$ for each port
- Solve the nodal voltages
- Update the state variables
- Calculate LTE and accept or reject Δt_k

`if (tstep==ACCEPTED)`

- Store x^{k+1}, u^{k+1} and set `t+=tstep;`

`}`

Fig. 7. Transient analysis of a reduced-order time-varying macromodel.

Here, we consider discrete time-domain samples of the port voltages and currents and the state variables. For ease of notation, we denote the discrete samples at different time points with superscripts. For example, the present and previous time port voltages $u(t_{k+1})$ and $u(t_k)$ are denoted by u^{k+1} and u^k, respectively. We also define the time steps $\Delta t_k = t_{k+1} - t_k$ and $\Delta t_{k-1} = t_k - t_{k-1}$. Note that the state-updating equations should simplify to the initial values given by (13) when $\Delta t_k \to \infty$ to avoid any discontinuities at $t = 0$.

The steps needed in the transient analysis of a nonlinear circuit containing a reduced-order time-varying macromodel are presented in Fig. 7. Note that Newton–Raphson iteration is needed at each time point when the overall circuit is nonlinear.

A. Kubota's Method

In [24], Kubota *et al.* approximated the time derivatives in (2) with the backward Euler (BE) rule

$$\frac{{\rm d}\tilde{x}^{k+1}}{{\rm d}t} = \frac{\tilde{x}^{k+1} - \tilde{x}^k}{\Delta t_k} \tag{18}$$

and obtained the following state-updating equations:

$$\begin{cases} \tilde{\mathbf{x}}^{k+1} &= \left(\tilde{\mathbf{G}} + \dfrac{\tilde{\mathbf{C}}}{\Delta t_k}\right)^{-1} \left(\tilde{\mathbf{B}}\mathbf{u}^{k+1} + \dfrac{\tilde{\mathbf{C}}}{\Delta t_k}\tilde{\mathbf{x}}^k\right) \\ \mathbf{i}^{k+1} &= \tilde{\mathbf{L}}^{\rm T}\tilde{\mathbf{x}}^{k+1}. \end{cases} \tag{19}$$

No additional nodes are created and the number of internal variables is only q, but one matrix inversion and several matrix multiplications are required to obtain the conductance and equivalent current.

Naturally, this method can be extended to any numerical integration scheme. The general form of a numerical integration algorithm for solving initial-value problems can be written as [29]

$$x^{k+1} = \sum_{r=0}^{\rho} a_r x^{k-{\rm r}} + \Delta t_k \sum_{r=-1}^{\rho} b_r \frac{{\rm d}x^{k-{\rm r}}}{{\rm d}t}. \tag{20}$$

The three methods generally used in circuit simulation, i.e., BE, trapezoidal rule (TR), and Gear–Shichman (GS) [30] methods, can be described with only four nonzero coefficients a_0, a_1, b_{-1}, and b_0. The coefficients for the three methods are collected in Table I.

TABLE I
COEFFICIENTS FOR THE INTEGRATION METHODS

Coeff.	BE	TR	GS
a_0	1	1	$\frac{(\Delta t_k+\Delta t_{k-1})^2}{\Delta t_{k-1}(2\Delta t_k+\Delta t_{k-1})}$
a_1	0	0	$-\frac{{\Delta t_k}^2}{\Delta t_{k-1}(2\Delta t_k+\Delta t_{k-1})}$
b_{-1}	1	$\frac{1}{2}$	$\frac{\Delta t_k+\Delta t_{k-1}}{2\Delta t_k+\Delta t_{k-1}}$
b_0	0	$\frac{1}{2}$	0

The present-time derivative can be solved from (20)

$$\frac{\mathrm{d}x^{k+1}}{\mathrm{d}t}=\frac{x^{k+1}-a_0x^k-a_1x^{k-1}-b_0\Delta t_k\dfrac{\mathrm{d}x^k}{\mathrm{d}t}}{b_{-1}\Delta t_k}. \tag{21}$$

Inserting this into (2) and using

$$\tilde{\mathbf{C}}\frac{\mathrm{d}\tilde{\mathbf{x}}^k}{\mathrm{d}t}=-\tilde{\mathbf{G}}\tilde{\mathbf{x}}^k+\tilde{\mathbf{B}}\mathbf{u}^k \tag{22}$$

for the previous-time derivative yields an updating equation for $\tilde{\mathbf{x}}^{k+1}$ for each integration method. Note that BE integration must be used at $t=0$ to make the solution consistent with the dc solution.

B. Time-Domain Differential-Equation Macromodel

The numerical integration formulas (20) can also be applied to the state-space model presented in Section II-C, thus avoiding the matrix inversion and multiplications [25]. Applying (21) to the left-hand side of (10) and solving for the present-time state variable yields an updating equation for x^{k+1} in the case of a real pole

$$x^{k+1}=\frac{(a_0h+b'p)\,x^k+a_1hx^{k-1}+u^{k+1}+b'u^k}{h-p} \tag{23}$$

where $b'=b_0/b_{-1}$ and $h=1/(b_{-1}\Delta t_k)$. Similarly, the case of complex poles is handled by applying (21) to (12); after some algebraic manipulation, the updating equations can be written as

$$\begin{cases} x^{\mathrm{r},k+1} &= \dfrac{c_1x^{\mathrm{r},k}+c_2x^{\mathrm{i},k}+a_1hx^{\mathrm{r},k-1}+c_3x^{\mathrm{i},k-1}+c_4}{h-a+\dfrac{b^2}{h-a}} \\ x^{\mathrm{i},k+1} &= \dfrac{-c_2x^{\mathrm{r},k}+c_1x^{\mathrm{i},k}-c_3x^{\mathrm{r},k-1}+a_1hx^{\mathrm{i},k-1}+c_5}{h-a+\dfrac{b^2}{h-a}} \end{cases} \tag{24}$$

where

$$\begin{aligned} c_1&=a_0h+b'\Big(a-\big(b^2/(h-a)\big)\Big)\\ c_2&=b\Big(b'+\big((b'a+a_0h)/(h-a)\big)\Big)\\ c_3&=a_1hb/(h-a)\\ c_4&=2\left(u^{k+1}+b'u^k\right)\\ c_5&=-\big(2b/(h-a)\big)\left(u^{k+1}+b'u^k\right). \end{aligned}$$

Inserting the coefficients a_0, a_1, b_{-1}, and b_0 from Table I yields the final state-updating equations for each integration method. As before, BE integration must be used at $t=0$.

C. Bracken's Method

In [26], Bracken *et al.* proposed a method that assumes the port voltages to be piecewise linear (PWL) functions of time. Hence, the port voltages can be composed of delayed ramps with different slopes at each time point. The latest change of slope ς is given as

$$\varsigma=\frac{u^{k+1}-u^k}{\Delta t_k}-\frac{u^k-u^{k-1}}{\Delta t_{k-1}}. \tag{25}$$

Consider one (generic) real pole/residue pair in (7). The time-domain port current corresponding to an input ramp $U(s)=\mathcal{L}\{\varsigma t\}=\varsigma/s^2$ is obtained with the inverse Laplace transform

$$i(t)=\varsigma k\left(-\frac{1}{p^2}-\frac{t-t_k}{p}+\frac{\mathrm{e}^{p(t-t_k)}}{p^2}\right)\epsilon(t-t_k) \tag{26}$$

where ϵ is the heaviside step function. Defining auxiliary variables d, e, and f for the coefficients of the constant term, the linear time-dependent term, and the $\exp(p\Delta t_k)$ term, respectively, the port current at time t_{k+1} can be calculated as

$$\begin{aligned} i^{k+1}=\varsigma k\left(-\frac{1}{p^2}-\frac{\Delta t_k}{p}+\frac{\mathrm{e}^{p\Delta t_k}}{p^2}\right)\\ +d^k+e^kt_{k+1}+f^k\mathrm{e}^{p(\Delta t_k+\Delta t_{k-1})} \end{aligned} \tag{27}$$

where the coefficients of u^{k+1} constitute the conductance and the rest of the terms form the equivalent current source. At each time point, after the new port voltages are obtained using Newton–Raphson iteration, the new change of slope ς is calculated, and the auxiliary variables are updated as follows:

$$d^{k+1}=d^k+\varsigma k\left(-\frac{1}{p^2}+\frac{t_{k+1}}{p}\right) \tag{28}$$

$$e^{k+1}=e^k-\varsigma\frac{k}{p} \tag{29}$$

$$f^{k+1}=f^k\mathrm{e}^{\Delta t_{k-1}}+\varsigma\frac{k}{p^2}. \tag{30}$$

The case of complex-conjugate poles is handled similarly, but this time, four auxiliary variables are needed for the constant, linear, $\sin(b\Delta t_k)\exp(a\Delta t_k)$, and $\cos(b\Delta t_k)\exp(a\Delta t_k)$ terms.

D. Frequency to Time Domain (FTD)

In [19], Liu *et al.* proposed a method, i.e., FTD, that uses the exact solution of differential equation (10)

$$x^{k+1}=\mathrm{e}^{p\Delta t_k}x^k+\int_0^{\Delta t_k}\mathrm{e}^{p\tau}u(t_{k+1}-\tau)\mathrm{d}\tau. \tag{31}$$

As before, the port voltages are assumed to change linearly between the time points t_k and t_{k+1}

$$u(t)=u^k+\frac{u^{k+1}-u^k}{\Delta t_k}(t-t_k). \tag{32}$$

Now, the integral can be evaluated in a closed form and the following state-updating equation is obtained in the case of a real pole

$$x^{k+1} = \mathrm{e}^{p\Delta t_k} x^k + \left[\frac{\mathrm{e}^{p\Delta t_k}}{p} - \frac{1}{p^2\Delta t_k}\left(\mathrm{e}^{p\Delta t_k} - 1\right)\right] u^k + \left[\frac{1}{p^2\Delta t_k}\left(\mathrm{e}^{p\Delta t_k} - 1\right) - \frac{1}{p}\right] u^{k+1}. \quad (33)$$

The port current is obtained by multiplying x^{k+1} by k, as in (10). The derivation of the state-updating equations in the case of a pair of complex poles—fairly complicated expressions involving exponential and trigonometric functions—can be found in [19].

V. Comparison of the Macromodels

Here, we consider the relations between the nine macromodels. The macromodels are then compared to each other using both theoretical considerations and transient-simulation examples. To enable a fair CPU–time comparison, all nine macromodels were implemented as built-in C-coded APLAC models. The transient analysis of the reduced-order models is also compared to that of the original unreduced circuit.

A. Relations Between the Macromodels

The PRIMA steps, and the various post-processing steps, are shown in Fig. 8. The macromodels corresponding to each step are illustrated in this figure. The time-varying macromodels are shown in grey.

Two of the time-varying macromodels have direct links to two equivalent-circuit realizations. Namely, replacing the charge sources in direct stamping I (III-A) with their companion models [29] and solving for the port currents, yields Kubota's method (IV-A). Similarly, time-domain differential-equation macromodel (IV-B) can be obtained from differential-equation macromodel (III-E). Also note that Bracken's method (IV-C) and FTD (IV-D) produce exactly the same port current, but the internal states are different.

B. Theoretical Considerations

For a fixed time step, the port currents for all the macromodels, except IV-C and IV-D are equivalent, in the limits of numerical accuracy. The five equivalent-circuit realizations create different numbers of additional nodes and internal VCCSs. The four time-varying macromodels differ in terms of the number of internal states and the LTE calculation. The number of nodes, VCCSs, and internal states are collected in Table II for the nine macromodels.

It must be remembered that matrix inversion and several matrix multiplications are required for IV-A to update the q states. For IV-A and IV-B, two or three (depending on the integration method) past values of the states must be stored in memory for the LTE calculation. Also, the previous-time port voltages (for IV-C, also the ones before those), must be stored for the time-varying macromodels.

The well-known expressions for the LTE (see, e.g., [29] and [30]) for the three integration methods considered in this paper

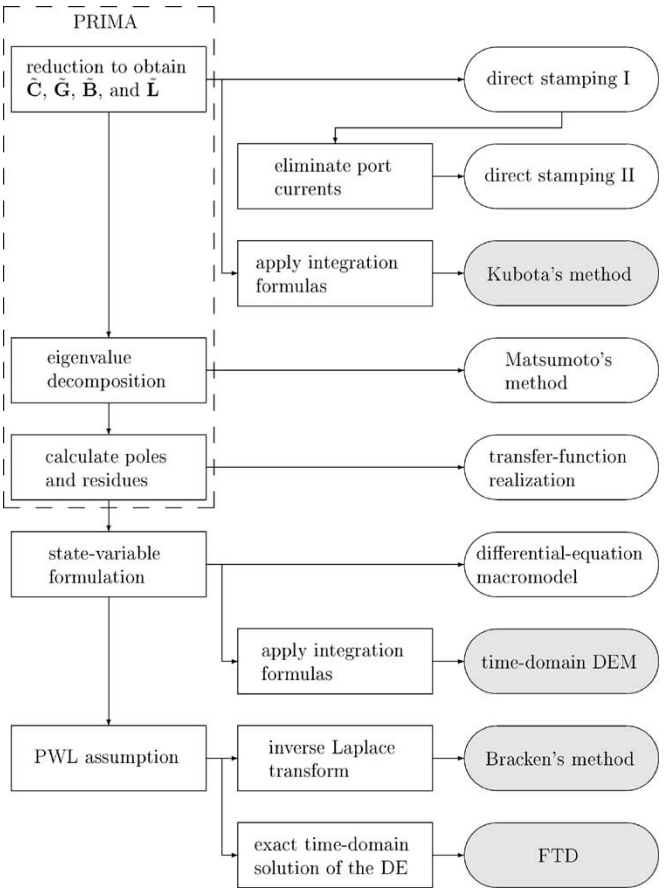

Fig. 8. Illustration of the PRIMA steps, the various post-processing steps, and the corresponding macromodels. The time-varying macromodels are shown in grey.

have been gathered in Table III. In this table, $x^{(n)}$ denotes the nth time derivative of $x(t)$, evaluated somewhere in the open interval $t_k < \xi < t_{k+1}$. The nth derivative can be approximated with the finite divided difference [31]

$$\frac{x^{(n)}(\xi)}{n!} \approx x\left[t_{k+1}, t_k, \ldots, t_{k+1-n}\right] \quad (34)$$

which implies that at least n past values of x must be kept stored in memory at each time point.

For the equivalent-circuit realizations, the expressions of Table III are applied internally by the circuit simulator to the charges of each dynamic VCCS. The largest LTE of the whole nonlinear circuit is then found and it can be used as a criterion for accepting/rejecting the current time step and as a parameter in calculating the new one [29].

For time-varying macromodels, the LTE can be calculated using the same expressions of Table III. It is not, however, obvious, which quantities these formulas should be applied to. One natural choice is to find a quantity that produces the same LTE as some of the equivalent-circuit realizations. As noted before, IV-A can be derived from III-A and IV-B from III-E so the expressions for charge can be determined easily by inspecting Figs. 1 and 5. Note that, for both equivalent circuits, the values of the elements connected to the additional nodes can be multiplied with a constant, without any change in the port quantities. Therefore, we can scale the charge, and thereby, the LTE, to

TABLE II
COMPARISON BETWEEN THE NINE METHODS

	method	add. nodes	VCCS's	states
III-A	Direct stamping I	$N+q$	$2(q^2+N(q+1))$	–
III-B	Direct stamping II	q	$2(q^2+Nq)$	–
III-C	Matsumoto's method	q	$2q_{\rm r}(N+1)+2q_{\rm c}(2N+3)$	–
III-D	Transfer-function realization	$2N(q_{\rm r}+2Nq_{\rm c})$	$Nq_{\rm r}(N+5)+12N^2q_{\rm c}$	–
III-E	Differential-equation macromodel	Nq	$Nq_{\rm r}(N+3)+Nq_{\rm c}(2N+7)$	–
IV-A	Kubota's method	–	N^2	q
IV-B	Time-domain diff.-eq. macromodel	–	N^2	Nq
IV-C	Bracken's method	–	N^2	$3N^2q$
IV-D	Frequency to time domain	–	N^2	Nq

TABLE III
LTE FOR THE INTEGRATION METHODS

BE	$-\frac{{\Delta t_k}^2}{2}x^{(2)}$
TR	$-\frac{{\Delta t_k}^3}{12}x^{(3)}$
GS	$\frac{{\Delta t_k}^2(\Delta t_k+\Delta t_{k-1})^2}{6(2\Delta t_k+\Delta t_{k-1})}x^{(3)}$

any amplitude. This does not, however, make the error calculation ambiguous since relative error is almost always used (the exception would be extremely small charges, where the relative error becomes unreasonably large).

IV-C and IV-D do not apply numerical integration, which is the source of error in the other macromodels. Instead, the error in IV-C and IV-D arises from the PWL assumption. With nonlinear drivers and loads, the port currents and, therefore, also the port voltages, are never PWL. In fact, it is customary when modeling nonlinear devices to force the derivatives of the terminal currents to be continuous in order to help the convergence of the Newton–Raphson iteration. Thus, IV-C and IV-D produce an error, which can be assumed small for a sufficiently small time step. The problem lies in estimating what is a "sufficiently small" time step.

C. Simulation Examples

The simulations were done with the APLAC circuit simulator on an HP9000/D270 workstation. All the CPU times are averages of three successive runs. The transient simulation times presented include everything, except the PRIMA reduction: parsing the netlist, reading the poles and residues from a file (produced by PRIMA), "building" the equivalent circuits, forming the MNA equations, the dc analysis for solving the initial state, and the actual transient analysis. TR integration was used in all simulations, except at $t=0$. BE was then used for two time points.

1) Three-Port: A nonlinear circuit containing a linear interconnect block [3] and three inverters is shown in Fig. 9. The seven transmission lines were replaced with 50 *RLC* sections each, and the linear block was treated as a three-port and reduced with PRIMA. The CPU time of the reduction using six different orders ($q=10,20,\ldots,60$) varied between 9.05–11.78 s. The dimension of the original $\mathbf{G}$ and $\mathbf{C}$ matrices was $n=731$ with 2191 and 718 nonzero elements, respectively. The excitation $u_{\rm in}$ was a voltage pulse with 1-ns rise and fall times, as shown in Fig. 10. The voltages $u_{\rm out}$ for the unreduced circuit and IV-B macromodel with $q=30$ are also shown in this figure (the curves corresponding to the other reduced-order macromodels have been omitted since the curves are indistinguishable). The simulation time step was fixed to 20 ps.

The simulation CPU times for each macromodel as a function of the order of reduction q are presented in Fig. 11. The CPU times are normalized to the CPU time of the transient simulation of the unreduced circuit (52.75 s). The curves are labeled according to the section in which the particular macromodel was introduced.

2) Ten-Port: A circuit containing five coupled transmission lines with capacitance, inductance, and resistance matrices [32]

$$\mathbf{C}=\begin{bmatrix} 2.227 & -0.522 & -0.036 & -0.016 & -0.010 \\ -0.522 & 2.432 & -0.514 & -0.032 & -0.016 \\ -0.036 & -0.514 & 1.327 & -0.514 & -0.036 \\ -0.016 & -0.032 & -0.514 & 2.432 & -0.522 \\ -0.010 & -0.016 & -0.036 & -0.522 & 2.227 \end{bmatrix}\frac{\rm pF}{\rm cm}$$

$$\mathbf{L}=\begin{bmatrix} 7.470 & 5.220 & 4.074 & 3.357 & 2.839 \\ 5.220 & 7.237 & 5.115 & 4.028 & 3.357 \\ 4.074 & 5.115 & 7.214 & 5.115 & 4.074 \\ 3.357 & 4.028 & 5.115 & 7.257 & 5.220 \\ 2.839 & 3.357 & 4.074 & 5.220 & 7.470 \end{bmatrix}\frac{\rm nH}{\rm cm}$$

$$\mathbf{R}=68.966\,\mathbf{I}\,\frac{\Omega}{\rm cm}$$

and ten inverters is shown in Fig. 12. Each line was again replaced with 50 *RLC* sections, and the capacitive and inductive coupling between the lines was taken into account with 51 mutual capacitors and 50 inductive mutual couplings for each pair of lines. The system of coupled lines was treated as a ten-port and reduced with PRIMA using $q=10,20,\ldots,60$. The CPU time for the reduction varied between 5.20–6.56 s. The dimension of the original $\mathbf{G}$ and $\mathbf{C}$ matrices was $n=515$ with 1535 and 2525 nonzero elements, respectively. The excitation $u_{\rm in}$ was the same pulse as in the previous example and time step was again fixed to 20 ps. The voltages $u_{\rm out}$ at the end of the victim line for the unreduced circuit and the IV-B macromodel with $q=30$ are shown in Fig. 13.

The simulation CPU times for each macromodel are presented in Fig. 14. The CPU times are normalized to the CPU time of the simulation of the unreduced circuit (134.93 s).

D. Analysis of the Results

Based on Table II and on the simulation CPU times presented in Section V-C, the following conclusions can be drawn.

1) III-A and III-B create a huge number of components, when q is large.

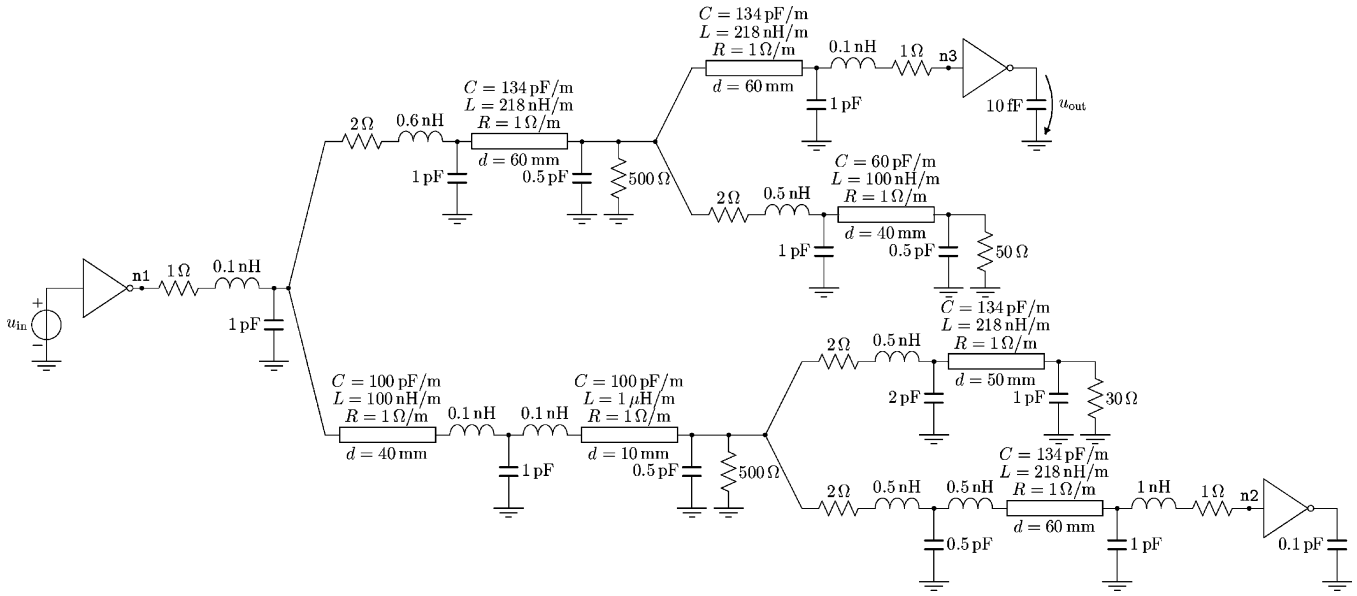

Fig. 9. Nonlinear circuit with a large linear interconnect block.

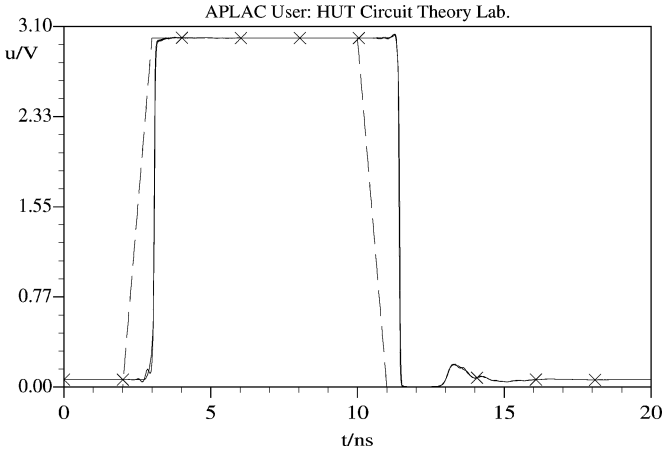

Fig. 10. Input voltage (– – –), and APLAC (—) and reduced-order model (—×—) output voltages.

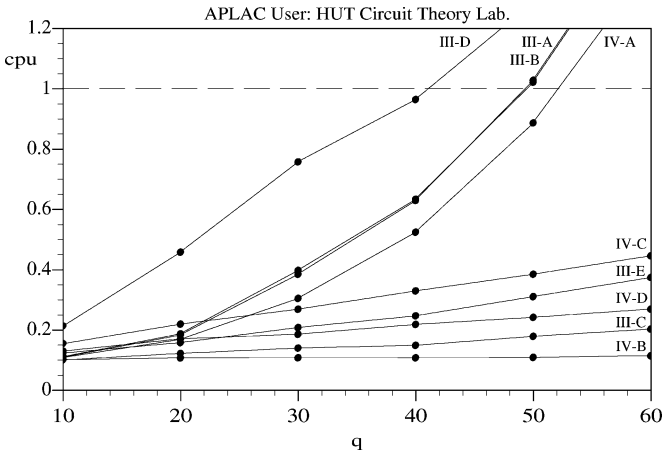

Fig. 11. Comparison of CPU times for the three-port.

2) III-C is fast since the number of components and nodes are both reasonable.

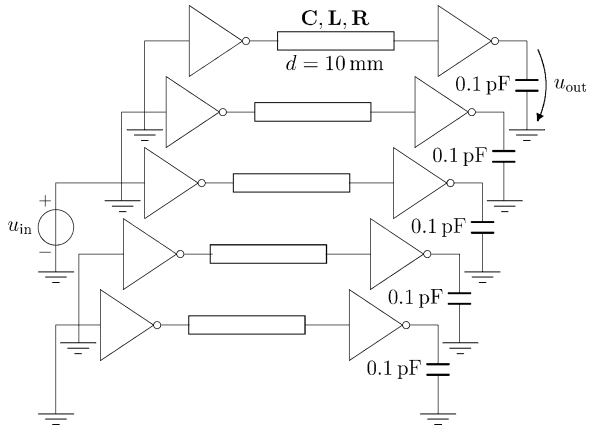

Fig. 12. Circuit with ten inverters and five coupled transmission lines.

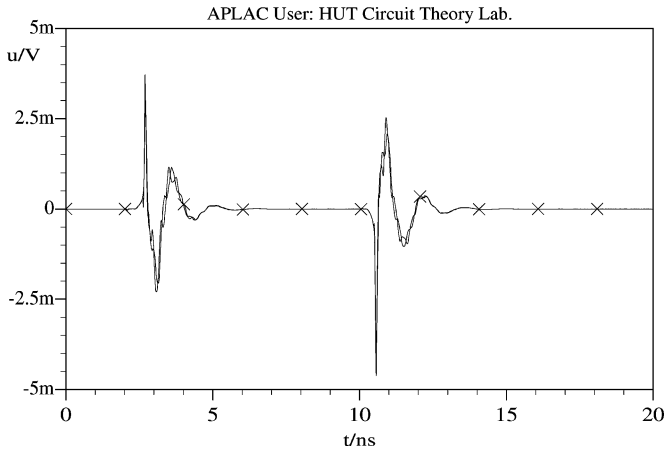

Fig. 13. APLAC (—) and reduced-order model (—×—) output voltages.

3) For III-D, the number of nodes and VCCSs created is enormous, especially when there are complex poles.
4) III-E creates a lot of additional nodes when both N and q are large.
5) IV-A is slow because of the internal matrix operations.

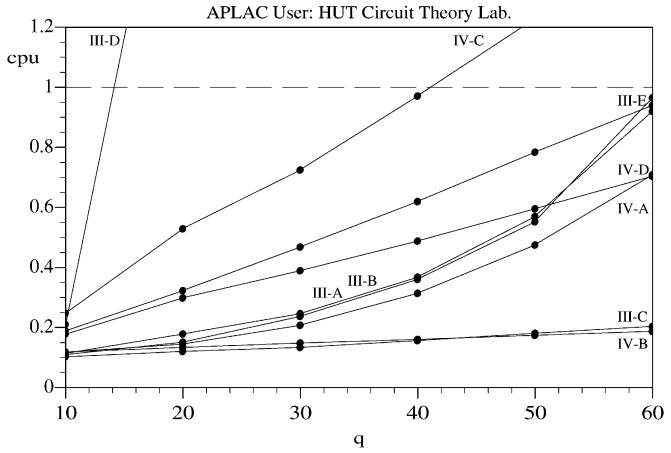

Fig. 14. Comparison of CPU times for the ten-port.

6) IV-B is fast because the state-updating equations are relatively simple.
7) The complicated state-updating equations of IV-C and IV-D make them slow when q and N are large.

The CPU time for direct-stamping methods, III-A and III-B, follow a square-type law, while the CPU time for IV-B–D grows linearly as a function of q. In [4], similar results were presented for a four-port, except that the advantage of using a time-varying macromodel seemed much larger than in our simulations. The methods compared in [4] were most probably similar to III-A and IV-C.

Diagonalizing the matrices makes III-C superior to the direct-stamping methods in terms of CPU time, which was also concluded in [18] (obviously, the authors of [4] were also aware of this). Although III-E has a simpler structure and the poles and residues have at least some physical meaning (they can be, e.g., stored in a file after the model reduction for reduced-order modeling library purposes), the savings in CPU time make III-C by far the best of the equivalent-circuit macromodels considered here.

IV-D is much faster and much simpler to implement than IV-C, which was also concluded in [17]. The state-updating equations, however, involve exponential and trigonometric functions, which makes IV-D slower than IV-B. Due to the speed and the easily obtained LTE estimate, we conclude that IV-B is the best time-domain macromodel considered in this paper.

VI. Conclusion

A comparison between nine different time-domain macromodels for the transient analysis of reduced-order circuits has been given. The methods have been presented in a unified manner, and they have been compared in terms of the internal nodes and components generated, as well as transient-analysis CPU time. The best equivalent-circuit and time-varying macromodels were the Matsumoto's method and time-domain differential-equation macromodel, respectively.

References

[1] K. J. Kerns and A. T. Yang, "Stable and efficient reduction of large, multiport *RC* networks by pole analysis via congruence transformations," *IEEE Trans. Computer-Aided Design*, vol. 16, pp. 734–744, July 1997.
[2] F.-Y. Chang, "Transient simulation of frequency-dependent nonuniform coupled lossy transmission lines," *IEEE Trans. Comp., Packag., Manufact. Technol. B*, vol. 17, pp. 3–14, Feb. 1994.
[3] R. Achar, P. K. Gunupudi, M. Nakhla, and E. Chiprout, "Passive interconnect reduction algorithm for distributed/measured networks," *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 287–301, Apr. 2000.
[4] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 645–654, Aug. 1998.
[5] L. T. Pillage and R. A. Rohrer, "Asymptotic waveform evaluation for timing analysis," *IEEE Trans. Computer-Aided Design*, vol. 9, pp. 352–366, Apr. 1990.
[6] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 639–649, May 1995.
[7] E. Chiprout and M. S. Nakhla, "Analysis of interconnect networks using complex frequency hopping (CFH)," *IEEE Trans. Computer-Aided Design*, vol. 14, pp. 186–200, Feb. 1995.
[8] B. N. Sheehan, "ENOR: Model order reduction of *RLC* circuits using nodal equations for efficient factorization," in *Proc. DAC'99*, New Orleans, LA, 1999, pp. 17–21.
[9] K. J. Kerns and A. T. Yang, "Preservation of passivity during *RLC* network reduction via split congruence transformations," *IEEE Trans. Computer-Aided Design*, vol. 17, pp. 582–591, July 1998.
[10] I. M. Elfadel and D. D. Ling, "A block rational Arnoldi algorithm for multipoint passive model-order reduction of multiport *RLC* networks," in *Proc. Int. Computer-Aided Design Conf.*, San Jose, CA, 1997, pp. 66–71.
[11] R. W. Freund and P. Feldmann, "The SyMPVL algorithm and its applications to interconnect simulation," in *Proc. SISPAD'97*, Boston, MA, 1997, pp. 113–116.
[12] M. Celik, L. Pileggi, and A. Odabasioglu, *IC Interconnect Analysis*. Norwell, MA: Kluwer, 2002.
[13] Z. Mu, "Applications of complex frequency hopping method in PCB signal integrity simulation," in *Proc. Int. Circuits and Systems Symp.*, vol. 6, Monterey, CA, 1998, pp. 78–81.
[14] M. Dinç and I. C. Göknar, "PSPICE subcircuits for passive reduced order interconnect models," in *Proc. ECCTD'01*, vol. 3, Espoo, Finland, 2001, pp. 33–36.
[15] D. H. Xie and M. Nakhla, "Delay and crosstalk simulation of high-speed VLSI interconnects with nonlinear terminations," *IEEE Trans. Computer-Aided Design*, vol. 12, pp. 1798–1811, Nov. 1993.
[16] S.-Y. Kim, N. Gopal, and L. T. Pillage, "Time-domain macromodels for VLSI interconnect analysis," *IEEE Trans. Computer-Aided Design*, vol. 13, pp. 1257–1270, Oct. 1994.
[17] Y. Liu, L. T. Pileggi, and A. J. Strojwas, "FTD: An exact frequency to time domain conversion for reduced order *RLC* interconnect models," in *Proc. DAC'98*, San Francisco, CA, 1998, pp. 15–19.
[18] Y. Matsumoto, Y. Tanji, and M. Tanaka, "Efficient SPICE-netlist representation of reduced-order interconnect model," in *Proc. ECCTD'01*, vol. 2, Espoo, Finland, 2001, pp. 145–148.
[19] Y. Liu, L. T. Pileggi, and A. J. Strojwas, "FTD: Frequency to time domain conversion for reduced-order interconnect simulation," *IEEE Trans. Circuits Syst. I*, vol. 48, pp. 500–506, Apr. 2001.
[20] A. Odabasioglu, M. Celik, and L. T. Pileggi, "PRIMA: Passive reduced-order interconnect macromodeling algorithm," in *Proc. Int. Computer-Aided Design Conf.*, San Jose, CA, 1997, pp. 58–65.
[21] T. Palenius and J. Roos, "Development and comparison of reduced-order interconnect macromodels for time-domain simulation," in *Proc. ICECS'02*, vol. 2, Dubrovnik, Croatia, 2002, pp. 757–760.
[22] R. Achar and M. S. Nakhla, "Simulation of high-speed interconnects," *Proc. IEEE*, vol. 89, pp. 693–728, May 2001.
[23] S. Aaltonen and J. Roos, "Simple reduced-order macromodels with PRIMA," in *Proc. ICECS'02*, vol. 1, Dubrovnik, Croatia, 2002, pp. 367–370.

[24] H. Kubota, A. Kamo, T. Watanabe, and H. Asai, "Noise analysis of power/ground planes on PCB by SPICE-like simulator with model order reduction technique," in *Proc. Int. Circuits and Systems Symp.*, vol. 5, Phoenix, AZ, 2002, pp. 649–652.

[25] T. Palenius and J. Roos, "An efficient reduced-order interconnect macromodel for time-domain simulation," in *Proc. Int. Circuits and Systems Symp.*, vol. 4, Bangkok, Thailand, 2003, pp. 628–631.

[26] J. E. Bracken, V. Raghavan, and R. A. Rohrer, "Interconnect simulation with asymptotic waveform evaluation (AWE)," *IEEE Trans. Circuits Syst. I*, vol. 39, pp. 869–878, Nov. 1992.

[27] *APLAC 7.92 Reference Manual and 7.92 User's Manual*, APLAC Solutions Corporation, Espoo, Finland, 2003. [Online]. Available: http://www.aplac.hut.fi/aplac.

[28] C.-W. Ho, A. E. Ruehli, and P. A. Brennan, "The modified nodal approach to network analysis," *IEEE Trans. Circuits Syst.*, vol. CAS-22, pp. 504–509, June 1975.

[29] L. O. Chua and P.-E. Lin, *Computed-Aided Analysis of Electronic Circuits: Algorithms and Computational Techniques*. Englewood Cliffs, NJ: Prentice-Hall, 1975.

[30] H. Shichman, "Integration system of a nonlinear network-analysis program," *IEEE Trans. Circuit Theory*, vol. CT-17, pp. 378–386, Aug. 1970.

[31] B. Carnahan, H. Luther, and J. Wilkes, *Applied Numerical Methods*. New York: Wiley, 1969.

[32] Y. Eo, S. Shin, W. R. Eisenstadt, and J. Shim, "Generalized traveling-wave-based waveform approximation technique for the efficient signal integrity verification of multicoupled transmission line system," *IEEE Trans. Computer-Aided Design*, vol. 21, pp. 1489–1497, Dec. 2002.

Timo Palenius was born in Espoo, Finland, in 1977. He received the M.Sc. (Tech.) degree in electrical engineering from the Helsinki University of Technology (HUT), Espoo, Finland, in 2002, and is currently working toward the Lic.Sc. (Tech.) degree in electrical engineering at HUT.

From 2000 to 2002, he was a Research Assistant, and from 2002 to 2004, a Research Scientist with the Circuit Theory Laboratory, Department of Electrical and Communications Engineering, HUT. In 2004, he joined the APLAC Solutions Corporation, Espoo, Finland. His research interests are interconnect simulation and transistor modeling. He has authored or coauthored one journal paper and two conference papers.

Janne Roos was born in Helsinki, Finland, in 1969. He received the M.Sc. (Tech.), Lic.Sc. (Tech.), and D.Sc. (Tech.) degrees in electrical engineering from the Helsinki University of Technology (HUT), Espoo, Finland, in 1994, 1996, and 1999, respectively.

He is currently a Senior Research Scientist with the Circuit Theory Laboratory, Department of Electrical and Communications Engineering, HUT. His research interests are numerical circuit simulation, especially PWL solution algorithms, interconnect simulation, and modeling of RF/microwave components using artificial neural networks. He has authored or coauthored two journal papers and 22 conference papers.