[1]

[1]Numerical Analysis Manuscript No. 97–05, Bell Laboratories, Murray Hill, New Jersey, March 1997. Available on WWW at {http://cm.bell-labs.com/cs/doc/97}.

Interconnect-Delay Computation and Signal-Integrity Verification Using the SyMPVL Algorithm

Peter Feldmann
Bell Laboratories
Lucent Technologies
Room 2B–209
700 Mountain Avenue
Murray Hill, NJ 07974–0636, USA
e-mail: `peterf@research.bell-labs.com`

Roland W. Freund
Bell Laboratories
Lucent Technologies
Room 2C–420
700 Mountain Avenue
Murray Hill, NJ 07974–0636, USA
e-mail: `freund@research.bell-labs.com`

Abstract

This paper describes a methodology for interconnect-delay estimation, modeling, and signal-integrity verification. The methodology relies on a recently introduced powerful algorithm, SyMPVL (Symmetric Matrix-Padé via Lanczos). SyMPVL is an efficient and numerically stable algorithm for the computation of matrix-Padé approximations for large, linear, passive network transfer functions. The approximated transfer function is used to compute signal waveforms with sufficient accuracy to estimate delay and verify against signal-integrity violations. It can also serve as a compact model of the interconnect network.

1 Introduction

Advances in VLSI technology lead to substantial decreases in the size and spacing of the wires that connect the functional blocks, the *interconnect*. Simultaneously, the technology also allows the increase in the number of layers of metal that can be used for interconnect. The smaller size and increased density of the interconnect networks generate new design and verification concerns for VLSI designers.

One consequence of reducing wire cross sections is the increase in the importance of wire resistance, which results in significant changes in the signal waveform shapes. Closely-packed interconnect wires on multiple layers lead to increased interference among otherwise unrelated signals. As a consequence, one signal can induce sufficiently strong signals in neighboring wires to cause the inversion of their logical values. In the presence of resistive effects combined with capacitive, and maybe even inductive coupling, the CAD tools that predict circuit performance and verify against signal-integrity violations can no longer rely on simplistic signal descriptions, such as delay and slope. Increasingly, these tools need to compute and use accurate signal waveforms.

Another problem generated by the new VLSI technologies is the explosion in the volume of information required to capture interconnect effects. At the chip level, interconnect is modeled by large, lumped, linear networks consisting mainly of capacitors, resistors, and sometimes inductors and magnetic coupling. These networks are typically generated by automatic parasitic-extraction programs that often produce vast quantities of data. Storing the data for entire chips, which may contain millions of interconnect networks, will overwhelm the largest computers. It becomes crucial to reduce interconnect data to a manageable size without compromising the capability to predict accurate signal waveforms.

In this paper, we describe a methodology for interconnect delay estimation, modeling, and signal-integrity verification. The methodology relies on a recently introduced powerful algorithm, SyMPVL (Symmetric Matrix-Padé via Lanczos) [9, 10]. SyMPVL is an efficient and numerically stable algorithm for the computation of matrix-Padé approximations for large, linear, passive network transfer functions. The approximated transfer function can be used to compute signal waveforms with sufficient accuracy to evaluate delays and verify against signal-integrity violations. It can also serve as a compact model of the interconnect network.

SyMPVL represents the specialization of the more general MPVL (Matrix-Padé via Lanczos) algorithm [6] to the symmetrically formulated problems that result from the modeling of passive interconnect. An important feature of SyMPVL when applied to RC networks, is the guaranteed stability and passivity of the resulting transfer-function approximations. Stability and passivity preservation is crucial if the approximation is to be used as a compact model of the interconnect network.

We restrict this presentation to RC circuits, which are the most typical for interconnect modeling. Even in the general case where inductances cannot be ignored, the problems result in a symmetric formulation and the SyMPVL algorithm is applicable. Some of the stability and passivity guarantees, however, do not hold in the general RLC case. Even in this case, however, the stability and passivity of the reduced model can be enforced by simple post-processing.

The paper is organized as follows. In Section II, we formulate the network transfer functions for RC circuits. In Section III, we review the symmetric matrix-Padé approximation of such transfer functions, and describe its computation with the SyMPVL algorithm. We also discuss the stability and passivity of the resulting approximation. In Section IV, we formulate typical interconnect modeling and signal-integrity verification problems that can be solved using approximated network transfer functions. Section V contains numerical results for a circuit example. Finally, in Section VI, we make some concluding remarks.

2 Formulation of network transfer functions

In this section, we derive the general form of the network transfer functions for RC circuits.

The connectivity of a circuit can be captured using a directional graph. The nodes of the graph correspond to the nodes of the circuit, and the edges of the graph correspond to each of the circuit elements. An arbitrary direction is assigned to graph edges, so one can distinguish between the source and destination nodes. The adjacency matrix, $\mathbf{A}$, of the directional graph describes the connectivity of a circuit. Each row of the matrix corresponds to a graph edge and, therefore, to a circuit element. Each column of the matrix corresponds to a graph or circuit node. The column corresponding to the datum (ground) node of the circuit is omitted in order to remove redundancy. By convention, a row of the connectivity matrix will contain $+1$ in the column corresponding to the source node, -1 in the column corresponding to the destination node, and 0 everywhere else. It is easy to see that Kirchhoff's laws, which depend only on connectivity, can be expressed using the adjacency matrix

$$\begin{aligned} &\text{KCL:} \qquad \mathbf{A}^{\mathrm{T}}\mathbf{i}_b = 0, \\ &\text{KVL:} \qquad \mathbf{A}\mathbf{v}_n = \mathbf{v}_b. \end{aligned} \tag{1}$$

Here, $\mathbf{i}_b$ and $\mathbf{v}_b$ are the vectors of branch currents and voltages, respectively, and $\mathbf{v}_n$ is the vector of the non-datum node voltages.

We are interested in analyzing RC circuits, and for simplicity, we assume that the circuit is excited by current sources only. In this case, the adjacency matrix and the branch current and voltage vectors can be partitioned according to circuit-element types as follows:

$$\mathbf{A} = \begin{bmatrix} \mathbf{A}_i \\ \mathbf{A}_g \\ \mathbf{A}_c \end{bmatrix}, \quad \mathbf{v}_b = \begin{bmatrix} \mathbf{v}_i \\ \mathbf{v}_g \\ \mathbf{v}_c \end{bmatrix}, \quad \mathbf{i}_b = \begin{bmatrix} \mathbf{i}_i \\ \mathbf{i}_g \\ \mathbf{i}_c \end{bmatrix}. \tag{2}$$

Here, the subscripts i, g, and c stand for branches containing current sources, resistors, and capacitors, respectively.

The set of circuit equations is completed by adding the so-called *branch constitutive relationships* (BCR's), which describe the physical behavior of the circuit elements. In the case of RC circuits, the BCR's are as follows:

$$\mathbf{i}_i = -\mathbf{I}_t(t), \quad \mathbf{i}_g = \mathcal{G}\mathbf{v}_G, \quad \mathbf{i}_c = \mathcal{C}\frac{d}{dt}\mathbf{v}_c. \tag{3}$$

Here, $\mathbf{I}_t(t)$ is the vector of current-source values, $\mathcal{G}$ and $\mathcal{C}$ are appropriately-sized diagonal matrices whose diagonal entries are the conductance and capacitance values of each element. It is clear that these values are positive for any physical circuit.

The modified nodal formulation (MNA) of the circuit equations is obtained by combining the Kirchhoff equations (1) with the BCRs (3), and eliminating as many current unknowns as possible. The resulting MNA equations are as follows:

$$\underbrace{\mathbf{A}_g^{\mathrm{T}}\mathcal{G}\mathbf{A}_g}_{\mathbf{G}}\mathbf{v}_n + \underbrace{\mathbf{A}_c^{\mathrm{T}}\mathcal{C}\mathbf{A}_c}_{\mathbf{C}}\frac{d}{dt}\mathbf{v}_n = \mathbf{A}_i^{\mathrm{T}}\mathbf{I}_t(t). \tag{4}$$

In the case of RC circuits, both matrices $\mathbf{G}$ and $\mathbf{C}$ are symmetric and non-negative definite. This property will be exploited in the sequel. In the general case, when resistors, capacitors, and inductors are simultaneously present, the resulting MNA equations (4) involve indefinite matrices $\mathbf{G}$ and $\mathbf{C}$.

The interconnect network is a linear circuit, and we are interested only in its external behavior at its ports. The external behavior of a linear circuit is described completely by its p-port network transfer functions, such as the Z, Y, or S-parameters. The network function for a p-port is a $p \times p$ matrix-valued rational function of s, the Laplace transform variable.

We now determine the Z-parameter matrix of an RC circuit. Applying the Laplace transform to the circuit equations (4) gives

$$\mathbf{G}\mathbf{V}_n(s) + \mathbf{C}s\mathbf{V}_n(s) = \mathbf{B}\cdot\mathbf{I}(s). \tag{5}$$

The expression for the voltages across the excitation current sources is then given by

$$\mathbf{V}_i(s) = \mathbf{A}_i\mathbf{V}_n = \underbrace{\mathbf{B}^{\mathrm{T}}(\mathbf{G}+s\mathbf{C})^{-1}\mathbf{B}}_{\mathbf{Z}(s)}\cdot\mathbf{I}(s). \tag{6}$$

The resulting p-port Z-parameter matrix is of the form

$$\mathbf{Z}(s) = \mathbf{B}^{\mathrm{T}}\left(\mathbf{G}+s\mathbf{C}\right)^{-1}\mathbf{B}, \tag{7}$$

where $\mathbf{G}$ and $\mathbf{C}$ are real symmetric $N \times N$ matrices, and $\mathbf{B}$ is a real $N \times p$ matrix. In the next section, we describe how matrix-valued functions of the form (7) can be efficiently approximated.

3 Lanczos-based Padé Approximation of Matrix Transfer Functions

Let $\mathbf{Z}$ be a matrix-valued function of the form (7) with $\mathbf{G},\ \mathbf{C} \in \mathbb{R}^{N\times N}$ being symmetric and $\mathbf{B} \in \mathbb{R}^{N\times p}$. Note that $\mathbf{Z}$ is a matrix-valued rational function:

$$\mathbf{Z} : \mathbb{C} \longmapsto (\mathbb{C} \cup \{\infty\})^{p\times p}. \tag{8}$$

Moreover, each pole s of $\mathbf{Z}$ is also an eigenvalue of the matrix pencil $\mathbf{G} + \lambda\mathbf{C}$, i.e., each pole s of $\mathbf{Z}$ satisfies the equation $\det(\mathbf{G} + s\mathbf{C}) = 0$.

3.1 Review of the Case $p = 1$

For the moment, assume that $p = 1$. In this case, $\mathbf{Z}$ is a scalar-valued rational function, and for each n, we can define a scalar-valued Padé approximant $\mathbf{Z}_n$ to $\mathbf{Z}$ as follows. A function of the form

$$\mathbf{Z}_n(s) = \frac{\phi_{n-1}(s)}{\psi_n(s)}, \tag{9}$$

where ϕ_{n-1} and ψ_n are polynomials of degree at most $n-1$ and n, respectively, is called an nth *Padé approximant* to $\mathbf{Z}$ if

$$\mathbf{Z}(s) = \mathbf{Z}_n(s) + \mathcal{O}(s^{q(n)}), \tag{10}$$

where $q(n)$ is as large as possible; see, e.g., [3]. The condition (10) means that the Taylor expansions of $\mathbf{Z}$ and $\mathbf{Z}_n$ about $s = 0$ agree in as many leading Taylor coefficients (the so-called *moments*) as possible. In general, $q(n) = 2n$.

The standard approach to computing $\mathbf{Z}_n$ is based on explicit moment generation. First, one computes the leading $q(n)$ Taylor coefficients of $\mathbf{Z}$, and from these, one then generates the coefficients of the polynomials ϕ_{n-1} and ψ_n in (9). This standard approach to computing $\mathbf{Z}_n$ is employed in the asymptotic waveform evaluation (AWE) technique [16, 17]. However, computing Padé approximants using explicit moment computations is inherently numerically unstable, and indeed, in practice, this approach can be used only for very moderate values of n, such as $n \leq 10$; see [6]. These numerical instabilities can be avoided by exploiting the Lanczos-Padé connection [13] and generating the Padé approximant $\mathbf{Z}_n$ via the Lanczos process [15]. The resulting algorithms for stably computing Padé approximants of transfer function are PVL [5, 6] for general circuits and its special variant SyPVL [9] for RLC circuits. In SyPVL, the Padé approximant $\mathbf{Z}_n$ is computed using the formula

$$\mathbf{Z}_n(s) = \boldsymbol{\rho}_n^{\mathrm{T}} \left(\mathbf{A}_n + s\mathbf{B}_n\right)^{-1} \boldsymbol{\rho}_n, \tag{11}$$

where the $n \times n$ matrices $\mathbf{A}_n$, $\mathbf{B}_n$ and the vector $\boldsymbol{\rho}_n$ are generated from a symmetric version of the Lanczos process.

3.2 The General Case $p \geq 1$

Now we return to the general case of p-ports with $p \geq 1$. One approach to obtaining approximations of $\mathbf{Z}$ is to compute scalar Padé approximants for each of the p^2 entries of $\mathbf{Z}$ by means of p^2 runs of PVL. However, a much more efficient approach is to use the concept of matrix-Padé approximation [3] that generates a matrix-valued approximation $\mathbf{Z}_n$ for all entries of $\mathbf{Z}$ in one run. Moreover, the reduced-order model generated by $\mathbf{Z}_n$ is much smaller than the reduced-order model obtained from p^2 individual PVL runs.

In analogy to (9), matrix-Padé approximants can be represented by means of a pair of numerator and denominator matrix polynomials whose coefficients can be obtained via explicit moment computations. However, this approach suffers from the same instability mentioned above for the case $p = 1$. Instead, we use an appropriate extension of the formula (11) to the general matrix case $p \geq 1$ in our definition of matrix-Padé approximants. More precisely, we say that a matrix-valued function

$$\mathbf{Z}_n \,:\, \mathbb{C} \longmapsto (\mathbb{C} \cup \{\infty\})^{p \times p}. \tag{12}$$

is an nth *matrix-Padé approximant* to $\mathbf{Z}$ if $\mathbf{Z}_n$ is of the form

$$\mathbf{Z}_n(s) = \boldsymbol{\rho}_n^{\mathrm{T}} \left(\mathbf{A}_n + s\mathbf{B}_n\right)^{-1} \boldsymbol{\rho}_n, \tag{13}$$

where $\boldsymbol{\rho}_n \in \mathbb{C}^{n \times p}$ and $\mathbf{A}_n$, $\mathbf{B}_n \in \mathbb{C}^{n \times n}$, and

$$\mathbf{Z}(s) = \mathbf{Z}_n(s) + \mathcal{O}(s^{q(n)}) \quad \text{with maximal } q(n). \tag{14}$$

In general, we have $q(n) \geq 2\lfloor n/p \rfloor$, with $q(n) > 2\lfloor n/p \rfloor$ if, and only if, so-called *deflation* occurs due to certain linear dependencies.

In the next section, we formulate a symmetric Lanczos-type algorithm that generates appropriate matrices $\mathbf{A}_n$, $\mathbf{B}_n$, and $\boldsymbol{\rho}_n$ such that the function $\mathbf{Z}_n$ defined in (13) is indeed an nth matrix-Padé approximant to $\mathbf{Z}$.

3.3 A Symmetric Lanczos-Type Algorithm

In order to compute matrix-Padé approximants to matrix-valued transfer functions (7), we need a Lanczos-type algorithm that can handle multiple starting vectors, namely the p columns of the matrix $\mathbf{B}$ in (7). Such a procedure was recently developed in [1] and further refined in [8]. We use a special symmetric variant of the algorithm described in [8] that exploits the symmetry of the matrices $\mathbf{G}$ and $\mathbf{C}$ in (7). We stress that Lanczos-type algorithms for multiple starting vectors are necessarily quite involved for two reasons. First, in the course of the algorithm, some of the generated vectors can become linearly dependent and thus need to be *deflated*. Second, if $\mathbf{G}$ is indefinite, so-called *look-ahead* techniques are required to avoid potential breakdowns due to division by quantities that cannot be excluded to be zero. On the other hand, if $\mathbf{G}$ is symmetric positive definite, then it is easy to see that look-ahead is not needed. The algorithm used here is the only existing Lanczos-type process that has both deflation and look-ahead built in.

Next, we give a precise statement of the symmetric Lanczos-type procedure used to compute matrix-Padé approximants to $\mathbf{Z}$. The algorithm is stated directly in terms of the matrices $\mathbf{G}$, $\mathbf{C}$, and $\mathbf{B}$, which describe the given function $\mathbf{Z}$ in (7). It is thus somewhat different from the version of the algorithm stated in [10] that assumed a suitable symmetric factorization of $\mathbf{G}$ amd is stated in terms of $\mathbf{G}$, $\mathbf{B}$, and the factors of $\mathbf{G}$.

The Lanczos-type procedure generates a sequence of *Lanczos vectors* $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$, which are constructed to be $\mathbf{G}$-orthogonal. More precisely, if no look-ahead steps are necessary, the Lanczos vectors satisfy

$$\mathbf{v}_i^{\mathrm{T}}\,\mathbf{G}\,\mathbf{v}_n = \begin{cases} \delta_n & \text{if } i = n, \\ 0 & \text{if } i \neq n, \end{cases} \qquad \text{for all} \quad i, n = 1, 2, \ldots . \tag{15}$$

If look-ahead steps do occur, then the Lanczos vectors are only cluster-wise $\mathbf{G}$-orthogonal, instead of the vector-wise $\mathbf{G}$-orthogonality (15).

We note that after n steps of the Lanczos-type algorithm below, the Lanczos vectors

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n \tag{16}$$

have been constructed. In addition, at the end of each nth step, there are $p_{\mathrm{c}} = p_{\mathrm{c}}(n)$ "auxiliary" vectors,

$$\hat{\mathbf{v}}_{n+1}, \hat{\mathbf{v}}_{n+2}, \ldots, \hat{\mathbf{v}}_{p_{\mathrm{c}}}, \tag{17}$$

available. These vectors will be turned into Lanczos vectors or deflated in successive iterations. Here, p_{c} denotes the current block size. Initially, $p_{\mathrm{c}} = p$, and then within the algorithm p_{c} is reduced by one every time a deflation occurs.

Algorithm 1 (Symmetric Lanczos-type method with deflation and look-ahead.)
INPUT:
Matrices $\mathbf{G} = \mathbf{G}^{\mathrm{T}},\ \mathbf{C} = \mathbf{C}^{\mathrm{T}} \in \mathbb{R}^{N \times N}$;
A block $\mathbf{B} = \begin{bmatrix} \mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_p \end{bmatrix} \in \mathbb{R}^{N \times p}$ of p starting vectors (the vectors $\mathbf{b}_i$);
The deflation tolerance `dtol`.

OUTPUT:
The $p_1 \times p$ matrix $\boldsymbol{\rho}$;
($p_1 = p-$(# of deflations during the first p iterations).)
The $n \times n$ matrices $\mathbf{T}_n$ and $\boldsymbol{\Delta}_n$.
(Here, n is the value of the iteration counter at termination.)

0) *For* $k = 1, 2, \ldots, m$, *set* $\hat{\mathbf{v}}_k = \mathbf{G}^{-1}\mathbf{b}_k$.
   Set $p_{\mathrm{c}} = p$.
   (p_{c} is the current block size.)
   Set $\mathcal{I}_{\mathbf{v}} = \emptyset$.
   ($\mathcal{I}_{\mathbf{v}}$ records indices of vectors that must be preserved due to inexact deflation. If $k \in \mathcal{I}_{\mathbf{v}}$,

then the newly constructed $\mathbf{v}$ vectors need to be $\mathbf{G}$-orthogonalized against all vectors $\mathbf{v}_i$ with $i \in \mathcal{C}_k$.)
Set $\gamma = 1$, $\mathcal{C}_\gamma = \emptyset$, *and* $\mathbf{V}^{(\gamma)} = \emptyset$.
(γ is the number of the currently constructed look-ahead cluster, and $\mathcal{C}_\gamma$ is the set of indices of the already constructed Lanczos vectors in the γth cluster $\mathbf{V}^{(\gamma)}$.)

For $n = 1, 2, \ldots,$ *do* (Build nth Lanczos vector $\mathbf{v}_n$.) :

1) (If necessary, deflate $\hat{\mathbf{v}}_n$.)

   1a) *Set* $\phi = n - p_{\mathrm{c}}$.

   1b) *If the current* γ*th cluster contains at least one vector, i.e.,* $\mathcal{C}_\gamma \neq \emptyset$*, then orthogonalize* $\hat{\mathbf{v}}_n$ *against the vectors* $\mathbf{v}_i$ *in that cluster*:
   For all $i \in \mathcal{C}_\gamma$ *(in ascending order), set*

$$\tau = \frac{\mathbf{v}_i^{\mathrm{T}} \hat{\mathbf{v}}_n}{\|\mathbf{v}_i\|^2}, \quad \hat{\mathbf{v}}_n = \hat{\mathbf{v}}_n - \mathbf{v}_i \tau, \quad t_{i,\phi} = t_{i,\phi} + \tau,$$

   and, if $\phi \leq 0$*, set* $\rho_{i,\phi+p} = t_{i,\phi}$.

   1c) (Deflate if $\|\hat{\mathbf{v}}_n\|$ is "small".)
   If $\|\hat{\mathbf{v}}_n\| >$ `dtol`*, then continue with step* 2).
   Otherwise, deflate $\hat{\mathbf{v}}_n$ *by doing the following* :

   (i) *If* $p_{\mathrm{c}} = 1$*, then stop.*
   (There are no more right Krylov vectors.)

   (ii) *If* $\phi > 0$ *and the deflated vector* $\hat{\mathbf{v}}_n$ *is nonzero, then set* $\mathcal{I}_{\mathbf{v}} = \mathcal{I}_{\mathbf{v}} \cup \{ \gamma(\phi) \}$, *and save the vectors* $\mathbf{v}_i$, $i \in \mathcal{C}_{\gamma(\phi)}$.

   (iii) *For* $i = n, n+1, \ldots, n + p_{\mathrm{c}} - 2$*, set* $\hat{\mathbf{v}}_i = \hat{\mathbf{v}}_{i+1}$.
   (The auxiliary vector $\hat{\mathbf{v}}_n$ is deflated. The indices of the remaining right auxiliary vectors are reduced by one.)
   Set $p_{\mathrm{c}} = p_{\mathrm{c}} - 1$.
   (The current right block size is reduced by one.)

   (iv) *Repeat all of step* 1).

2) (Normalize $\hat{\mathbf{v}}_n$ to obtain the nth Lanczos vector $\mathbf{v}_n$.)

   2a) *Set*

$$\phi = n - p_{\mathrm{c}} \quad \textit{and} \quad \mathbf{v}_n = \frac{\hat{\mathbf{v}}_n}{t_{n,\phi}},$$

   where $t_{n,\phi}$ *is a suitable scaling factors, e.g.,*

$$t_{n,\phi} = \|\hat{\mathbf{v}}_n\|_2 .$$

   If $\phi \leq 0$*, then set* $\rho_{n,\phi+p} = t_{n,\phi}$.

2b) (Update look-ahead clusters.)
Set $\mathbf{V}^{(\gamma)} = \begin{bmatrix} \mathbf{V}^{(\gamma)} & \mathbf{v}_n \end{bmatrix}$ *and* $\mathcal{C}_\gamma = \mathcal{C}_\gamma \cup \{n\}$.

2c) (If $\mathcal{C}_\gamma$ starts a new cluster, update $\gamma_{\mathbf{v}}$.)
If $\mathcal{C}_\gamma = \{n\}$, *set* $\gamma_{\mathbf{v}} = \gamma\left(\max\{1, \phi\}\right)$.

3) (Advance the block Krylov subspace and obtain new vector $\hat{\mathbf{v}}_{n+p_c}$.)

3a) *Obtain*

$$\mathbf{v} = \mathbf{G}^{-1}\mathbf{C}\,\mathbf{v}_n$$

by first computing $\mathbf{t} = \mathbf{C}\,\mathbf{v}_n$ *and then solving the linear system*

$$\mathbf{G}\,\mathbf{v} = \mathbf{t}$$

for $\mathbf{v}$.
If the current γ*th cluster contains at least one vector, i.e.,* $\mathcal{C}_\gamma \neq \emptyset$, *we may add arbitrary combinations of the vectors in the* γ*th cluster to* $\mathbf{v}$:
Set

$$\mathbf{v} = \mathbf{v} - \mathbf{V}^{(\gamma)}\left[t_{i,n}\right]_{i \in \mathcal{C}_\gamma},$$

with arbitrary $t_{i,n} \in \mathbb{R}$.

3b) ($\mathbf{G}$-orthogonalize $\mathbf{v}$ against previous look-ahead clusters.)
Define the temporary index set

$$\mathcal{I} = \{\,\gamma_{\mathbf{v}}, \gamma_{\mathbf{v}}+1, \ldots, \gamma-1\,\} \cup \bigcup_{\substack{k \in \mathcal{I}_{\mathbf{v}} \\ k < \gamma_{\mathbf{v}}}} \{k\}.$$

For all $k \in \mathcal{I}$ (*in ascending order*), *set*

$$\begin{aligned} \left[t_{j,n}\right]_{j \in \mathcal{C}_k} &= \left(\mathbf{\Delta}^{(k)}\right)^{-1}\left(\mathbf{V}^{(k)}\right)^{\mathrm{T}}\left(\mathbf{G}\,\mathbf{v}\right), \\ \mathbf{v} &= \mathbf{v} - \mathbf{V}^{(k)}\left[t_{j,n}\right]_{j \in \mathcal{C}_k}. \end{aligned}$$

Set $\hat{\mathbf{v}}_{n+p_c} = \mathbf{v}$.

4) (Compute $\mathbf{\Delta}^{(\gamma)}$, check for end of look-ahead cluster, and if necessary, perform "end-of-cluster" updates.)

4a) *Form*

$$\mathbf{\Delta}^{(\gamma)} = \left(\mathbf{V}^{(\gamma)}\right)^{\mathrm{T}}\left(\mathbf{G}\,\mathbf{V}^{(\gamma)}\right).$$

4b) *If the matrix* $\mathbf{\Delta}^{(\gamma)}$ *is singular* (*or in some sense "close" to singular*), *then set* $\gamma(n+1) = \gamma$ *and continue with step* 1).
Otherwise, the current cluster is complete and the following "end-of-cluster" updates are performed:

4c) ($\mathbf{G}$-orthogonalize the vectors $\hat{\mathbf{v}}_k$, $n+1 \leq k \leq n+p_c$, against $\mathbf{V}^{(\gamma)}$.)
For $k = n+1, n+2, \ldots, n+p_c$, *set*

$$\begin{aligned}
\phi &= k - p_c, \\
\tilde{\tau} &= \begin{cases} \left(\mathbf{G}\,\mathbf{V}^{(\gamma)}\right)^{\mathrm{T}} \hat{\mathbf{v}}_k, & \textit{if } \phi > 0 \textit{ and } \gamma(\phi) = \gamma, \\ \left(\left[\tilde{t}_{i,j}\right]_{i \in \mathcal{C}_{\gamma(k)}, j \in \mathcal{C}_\gamma}\right)^{\mathrm{T}} \left[\delta_{i,k}\right]_{i \in \mathcal{C}_{\gamma(k)}}, & \\ & \textit{otherwise}, \end{cases} \\
\tau &= \left(\mathbf{\Delta}^{(\gamma)}\right)^{-1} \tau, \\
\hat{\mathbf{v}}_k &= \hat{\mathbf{v}}_k - \mathbf{V}^{(\gamma)} \tau, \\
\left[t_{j,\phi}\right]_{j \in \mathcal{C}_\gamma} &= \left[t_{j,\phi}\right]_{j \in \mathcal{C}_\gamma} + \tau,
\end{aligned}$$

and, if $\phi \leq 0$, *set* $\left[\rho_{j,\phi+p}\right]_{j \in \mathcal{C}_\gamma} = \left[t_{j,\phi}\right]_{j \in \mathcal{C}_\gamma}$.

4d) *Set* $\gamma = \gamma + 1$, $\gamma(n+1) = \gamma$, $\mathcal{C}_\gamma = \emptyset$, *and* $\mathbf{V}^{(\gamma)} = \emptyset$.
(The current cluster is complete, and so the vector $\mathbf{v}_{n+1}$ constructed in the next iteration will start a new cluster.)

To get the nth matrix-Padé approximant $\mathbf{Z}_n$ to $\mathbf{Z}$, we need the following quantities from Algorithm 1:

$$\begin{aligned}
\boldsymbol{\rho} &= \left[\rho_{i,j}\right]_{1 \leq i \leq p_1,\, 1 \leq j \leq p}, \\
\mathbf{T}_n &= \left[t_{i,j}\right]_{1 \leq i,j \leq n}, \\
\mathbf{\Delta}_n &= \left[\delta_{i,j}\right]_{1 \leq i,j \leq n} \\
&= \operatorname{diag}\left(\mathbf{\Delta}^{(1)}, \mathbf{\Delta}^{(2)}, \ldots, \mathbf{\Delta}^{(\gamma)}\right).
\end{aligned} \tag{18}$$

Here, for $\boldsymbol{\rho}$, $\mathbf{\Delta}_n$, and $\mathbf{T}_n$ we use the convention that all the entries $\rho_{i,j}$, $\delta_{i,j}$, and $t_{i,j}$ that are not explicitly defined in Algorithm 1 are set to zero. Then

$$\mathbf{Z}_n(s) = \boldsymbol{\rho}_n^{\mathrm{T}} \left(\mathbf{\Delta}_n^{-1} + s \mathbf{T}_n \mathbf{\Delta}_n^{-1}\right)^{-1} \boldsymbol{\rho}_n, \quad \boldsymbol{\rho}_n = \begin{bmatrix} \boldsymbol{\rho} \\ \mathbf{0} \end{bmatrix}, \tag{19}$$

is the nth matrix-Padé approximant to $\mathbf{Z}$; see [8] for a rigorous proof of this statement.

3.4 Stability

For RLC circuits, Padé-based reduced-order models are in general not stable and not passive. However, if the order n is large enough so that the reduced-order models are sufficiently accurate, then the reduced-order models defined by $\mathbf{Z}_n$ are almost stable and passive, and can in fact be made stable and passive by a suitable "post-processing" of $\mathbf{Z}_n$. Such post-processing techniques will be described elsewhere.

However, in [10], we showed that the reduced-order models defined by $\mathbf{Z}_n$ are guaranteed to be stable and passive for the special cases of RC, RL, and LC circuits.

We now briefly review the case of RC circuits. In this case, the matrices $\mathbf{G}$ and $\mathbf{C}$ are symmetric positive semi-definite. As a result, the matrix $\mathbf{\Delta}_n$ generated by Algorithm 1 is the $n \times n$ identity matrix, i.e., $\mathbf{\Delta}_n = \mathbf{I}_n$ for all n. Therefore, equation (19) reduces to

$$\mathbf{Z}_n(s) = \boldsymbol{\rho}_n^{\mathrm{T}} \left(\mathbf{I}_n + s\mathbf{T}_n\right)^{-1} \boldsymbol{\rho}_n. \tag{20}$$

Furthermore, the matrix $\mathbf{T}_n$ in (20) is now guaranteed to be symmetric positive semi-definite.

By (20), all poles of $\mathbf{Z}_n$ are of the form

$$s = -\frac{1}{\lambda}, \quad \text{where } \lambda \text{ is an eigenvalue of } \mathbf{T}_n. \tag{21}$$

Since $\mathbf{T}_n$ is symmetric positive semi-definite, $\lambda \geq 0$, and thus all poles of $\mathbf{Z}_n$ are non-positive. Moreover, it can be shown that a possible pole $s = 0$ is simple. Altogether, this proves that the reduced-order models defined by $\mathbf{Z}_n$ are stable for RC circuits.

3.5 Passivity

We now show that the reduced-order models defined by $\mathbf{Z}_n$ are passive.

It is well known (see, e.g., [21, 2]) that the reduced-order model defined by the matrix-valued rational matrix $\mathbf{Z}_n$ given by (20) is passive if, and only if, the following three conditions are satisfied:

(i) $\mathbf{Z}_n(s)$ has no poles in $\{\, s \in \mathbb{C} \mid \operatorname{Re} s > 0 \,\}$ (the right half of the complex plane);

(ii) $\mathbf{Z}_n(\bar{s}) = \overline{\mathbf{Z}_n(s)}$ for all $s \in \mathbb{C}$;

(iii) For all $s \in \mathbb{C}$ with $\operatorname{Re} s > 0$,

$$\operatorname{Re}\left(\mathbf{x}^{\mathrm{H}} \mathbf{Z}_n(s)\, \mathbf{x}\right) \geq 0 \quad \text{for all} \quad \mathbf{x} \in \mathbb{C}^p. \tag{22}$$

It thus remains to verify (i)–(iii). Condition (i) is satisfied in view of the stability of $\mathbf{Z}_n$. Condition (ii) follows immediately from (20) and the fact that $\mathbf{T}_n$ and $\boldsymbol{\rho}_n$ are are real matrices. Finally, we verify condition (iii). Let s be any complex number with $\operatorname{Re} s > 0$. Since $\mathbf{T}_n$ is symmetric positive semi-definite, we have

$$\operatorname{Re}\left(\mathbf{y}^{\mathrm{H}} \left(\mathbf{I} + \bar{s}\,\mathbf{T}_n\right)\, \mathbf{y}\right) = \|\mathbf{y}\|_2^2 + (\operatorname{Re} s)\mathbf{y}^{\mathrm{H}}\mathbf{T}_n\mathbf{y} \geq 0 \tag{23}$$

for all $\mathbf{y} \in \mathbb{C}^n$. For any given $\mathbf{x} \in \mathbb{C}^p$, we set

$$\mathbf{y} = \left(\mathbf{I} + s\,\mathbf{T}_n\right) \boldsymbol{\rho}_n \mathbf{x}. \tag{24}$$

Then $\mathbf{y} \in \mathbb{C}^n$, and inserting (24) into (23) gives

$$\begin{aligned} 0 &\leq \operatorname{Re}\left(\mathbf{y}^{\mathrm{H}} \left(\mathbf{I} + \bar{s}\,\mathbf{T}_n\right) \mathbf{y}\right) \\ &= \operatorname{Re}\left(\mathbf{x}^{\mathrm{H}} \boldsymbol{\rho}_n \left(\mathbf{I} + s\mathbf{T}_n\right)^{-1} \boldsymbol{\rho}_n \mathbf{x}\right) \\ &= \operatorname{Re}\left(\mathbf{x}^{\mathrm{H}} \mathbf{Z}_n(s)\, \mathbf{x}\right). \end{aligned} \tag{25}$$

Hence the reduced-order model given by $\mathbf{Z}_n$ is passive.

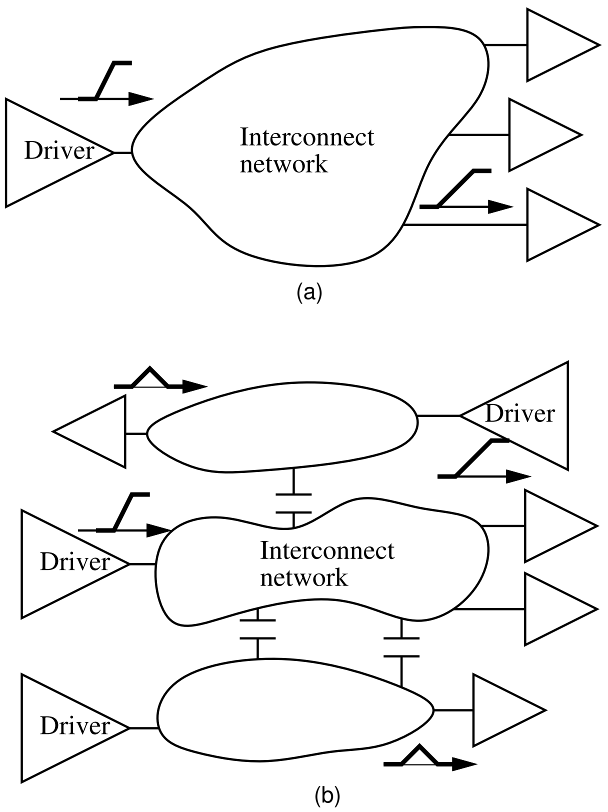

Figure 1: Unidirectional modeling

4 Interconnect analysis and modeling problems

4.1 Accurate waveform computations

The most typical case of a digital interconnect network is described in Figure 1(a). The output pin of a gate (the driver) is connected to the input pins of one or several other gates (the loads). In general, the driver and loads are nonlinear circuits, but often the loads can be approximated as linear capacitors. Under these assumptions the analysis of the driver and the loads can be decoupled. First, we compute the input impedance at the driver output pin, and the pin-to-pin transfer functions from the driver to each of the loads, as shown in Figure 2. Then, by combining the non-linear driver model with the input impedance, we compute the driver-side waveform. Applying the driver-to-load transfer functions to this waveform we compute the load-side voltages. The computation of transfer characteristics is also useful for checking against signal-integrity problems. The transfer function from an active driver port to the output ports of parasitically coupled interconnect networks (see Figure 1(b)) allows the computation of the waveforms due to cross-talk.

The expressions for the driving-point impedance $Z(s)$ and of the transfer functions, $H_1(s), H_2(s), \ldots,$ are computed using the SyMPVL algorithm. We excite the interconnect

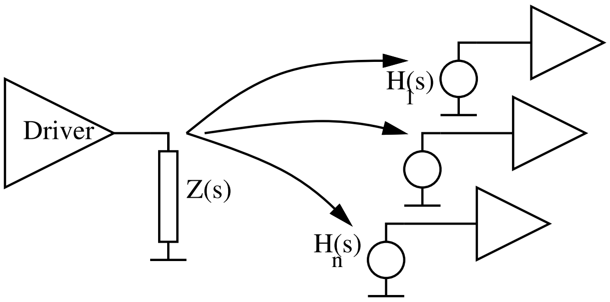

Figure 2: Cross-talk analysis

network by applying a current sources of unity value to the interconnect network at the driving points and observing the voltages across the current source, and at the load pins. The "observation" of the voltages can be done by means of *voltmeters*, i.e., zero-valued current sources attached to the nodes of interest. Therefore, we analyze an RC network excited by current sources. We formulate the impedance matrix as explained in Section II, and we compute a matrix-Padé approximation, $\mathbf{Z}_n(s)$ to it using SyMPVL. Assuming only one driver and using the Padé approximation of the Z-parameter matrix with this excitation, we observe that the values we are interested in represent the first column of the matrix-Padé approximation

$$\begin{bmatrix} Z(s) \\ H_1(s) \\ H_2(s) \\ \vdots \end{bmatrix} = \mathbf{Z}_n(s) \begin{bmatrix} 1 \\ 0 \\ 0 \\ \vdots \end{bmatrix} \tag{26}$$

Here the excitation matrix $\begin{bmatrix} 1 & 0 & \cdots \end{bmatrix}^{\mathrm{T}}$ has a 1 entry for the driver current source and 0s for the "observation" current sources. In the presence of multiple drivers, the transfer functions of interest will reside in the columns of the approximated Z-parameter matrix that correspond to each driver.

4.2 Interconnect modeling

Once wire resistance and coupling capacitance of interconnect networks can no longer be neglected, the amount of information produced by parasitic-extraction programs becomes enormous. Each interconnect path is essentially modeled by an RC netlist and there could be an order of millions of such paths on a modern chip. Since we are only interested in the port behavior of interconnect networks, the matrix network function, such as the Z-parameter set, contains all the relevant interconnect information. The matrix-Padé approximation can therefore be effectively used as a compact model for the interconnect. Instead of storing the netlist, which may contain a large number of RC elements, it is sufficient to store the products of the SyMPVL algorithm, namely the $\mathbf{T}_n$, $\mathbf{\Delta}_n$, and $\boldsymbol{\rho}_n$ matrices. These matrices can be used, as shown above, for delay calculation and signal-integrity verification using the expression for $\mathbf{Z}_n$ given in (19). The matrices can also be used to represent the interconnect network in a time-domain simulation of the functional blocks with a program such as SPICE. Starting from the approximate expression of the Z-parameter matrix, one can derive the underlying differential equations that correspond to the approximate system. We start from the reduced Z-parameter matrix expression

$$\mathbf{v}_i = \boldsymbol{\rho}_n^{\mathrm{T}} \left(\boldsymbol{\Delta}_n^{-1} + s\mathbf{T}_n \boldsymbol{\Delta}_n^{-1} \right) \boldsymbol{\rho}_n \mathbf{i}_i, \tag{27}$$

which we rewrite by introducing an n-vector of model state variables:

$$\begin{aligned} \left(\boldsymbol{\Delta}_n^{-1} + s\mathbf{T}_n \boldsymbol{\Delta}_n^{-1} \right) \mathbf{X} &= \boldsymbol{\rho}_n \mathbf{I}_i, \\ \mathbf{V}_i &= \boldsymbol{\rho}_n^{\mathrm{T}} \mathbf{X}. \end{aligned} \tag{28}$$

The corresponding reduced system of differential equations is given by

$$\begin{aligned} \boldsymbol{\Delta}_n^{-1} \mathbf{x} + \mathbf{T}_n \boldsymbol{\Delta}_n^{-1} \tfrac{d}{dt} \mathbf{x} &= \boldsymbol{\rho}_n \mathbf{i}_i, \\ \mathbf{v}_i &= \boldsymbol{\rho}_n^{\mathrm{T}} \mathbf{x}. \end{aligned} \tag{29}$$

Here, $\mathbf{i}_i$, and $\mathbf{v}_i$ are the variables that capture the port behavior of the interconnect and $\mathbf{x}$ are the local state variables of the interconnect. Note that the size of the vector $\mathbf{x}$ is only of size n, the size of the reduced-order model, which in most cases will be much smaller than the number of state variables of the interconnect circuit. The reduced system (29) can be used to replace the much larger system of differential equations (4) that fully describe the interconnect in a SPICE-type simulation.

5 Examples

We analyze an extracted layout composed of 83 signal paths. Some of these signal paths are capacitively coupled to each other. The network consists of about 1700 resistors and 2200 capacitors.

We analyze a cluster of 3 signal paths with 3 driver ports and 4 load ports, i.e., we compute a 7-by-7 matrix transfer function. In Figures 3–6, we display the response of applying a step voltage excitation at one of the driver ports, at the same port, and at 3 of the load ports, respectively. One of the load ports belongs to the same signal path as the driver, the second one belongs to a path that is capacitively coupled to the path of the driver, and the third one belongs to a path that is capacitively coupled directly only to the latter path, but not to the first one.

Various orders of approximation are employed. After 7 SyMPVL iterations the matrix-Padé approximation matches the 0-th and 1-st order matrix moments of the transfer function. We observe that the waveforms at the driving port and at the load port of the same signal path are already relatively well approximated. The waveforms in the coupling paths, on the other hand, are not. This result is to be expected. The first two transfer functions are

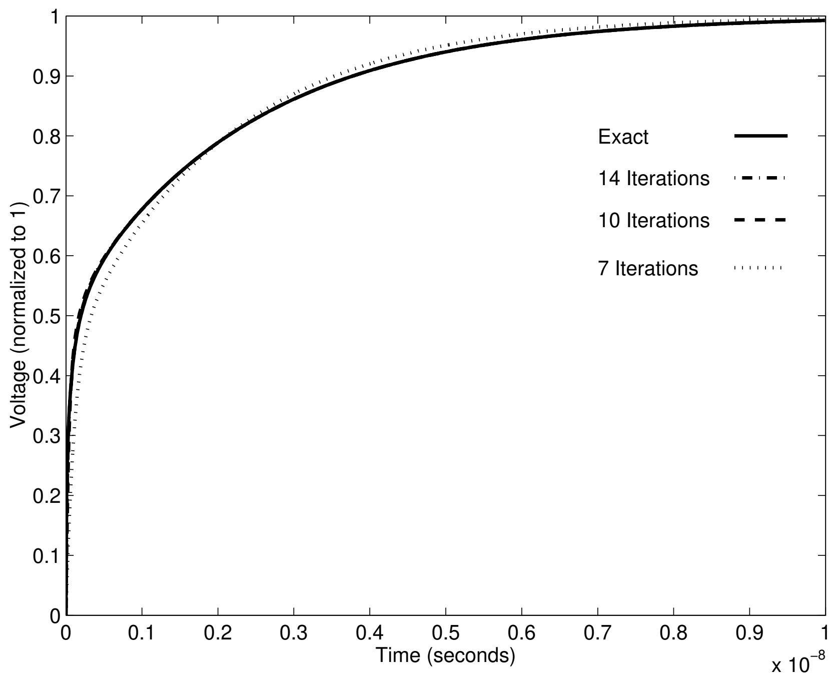

Figure 3: Waveform at driver output

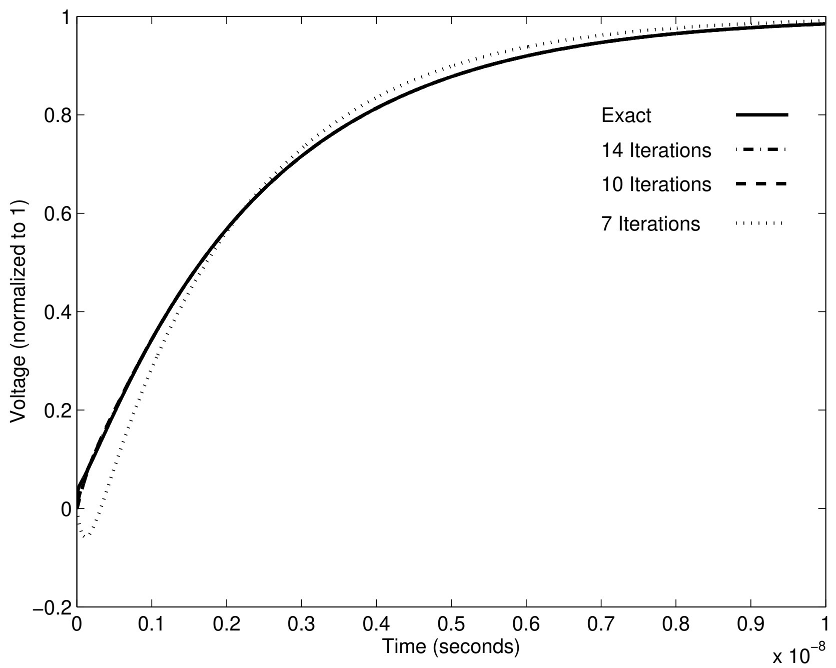

Figure 4: Waveform at load input in the same path

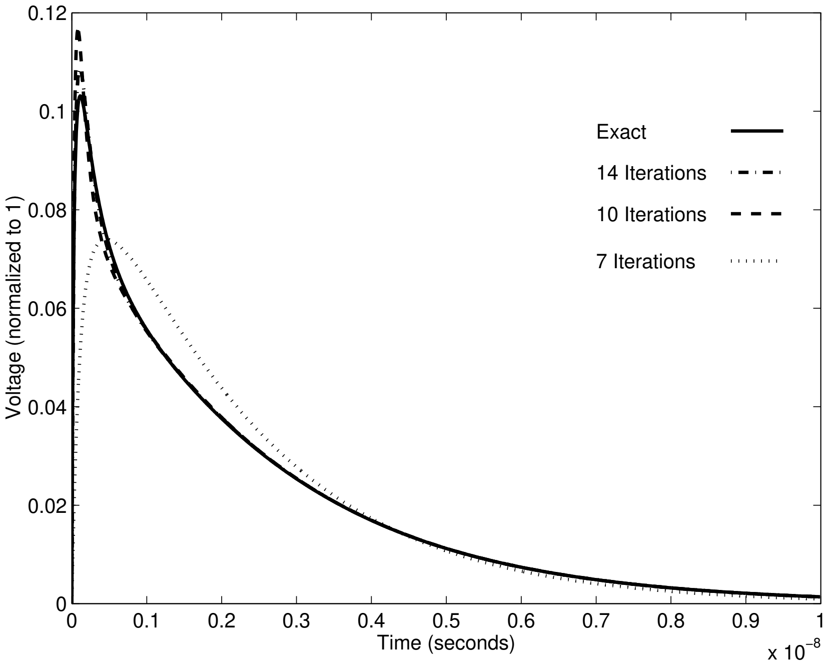

Figure 5: Waveform at load input in coupled path

of a low-pass form, while the latter two are high-pass. The matrix-Padé approximation is based on a Taylor expansion around the origin, therefore, its accuracy decreases as frequency increases. We observe that after 10 iterations the low-pass responses practically converged while the high-pass responses, while significantly closer, still exhibit about 10% error in the peak voltages. Note that SyMPVL, despite being based on a block-Lanczos algorithm, is not restricted to orders of approximation that are multiples of the block size (7 in this case). The accuracy of the approximations improves with each iteration, despite the fact that we can prove moment-matching properties only to iteration numbers that correspond to full blocks. After 14 iterations the first four matrix moments are exactly matched and the differences drop to less than 3%. Above 21 iterations, all the waveforms coincide with the exact analytically derived curves.

6 Concluding Remarks

We presented a methodology for interconnect analysis and modeling based on the novel SyMPVL algorithm. We showed that, with reasonable computational effort, SyMPVL can compute accurate signal waveforms in the interconnect. These waveforms can be used for precise delay calculation and for cross-talk analysis. The result of the SyMPVL algorithm is a matrix-Padé approximation of the interconnect block's network transfer function represented as three small matrices. The approximated transfer function is guaranteed to be stable and passive, and, therefore can be used as a reduced-order model of the interconnect network.

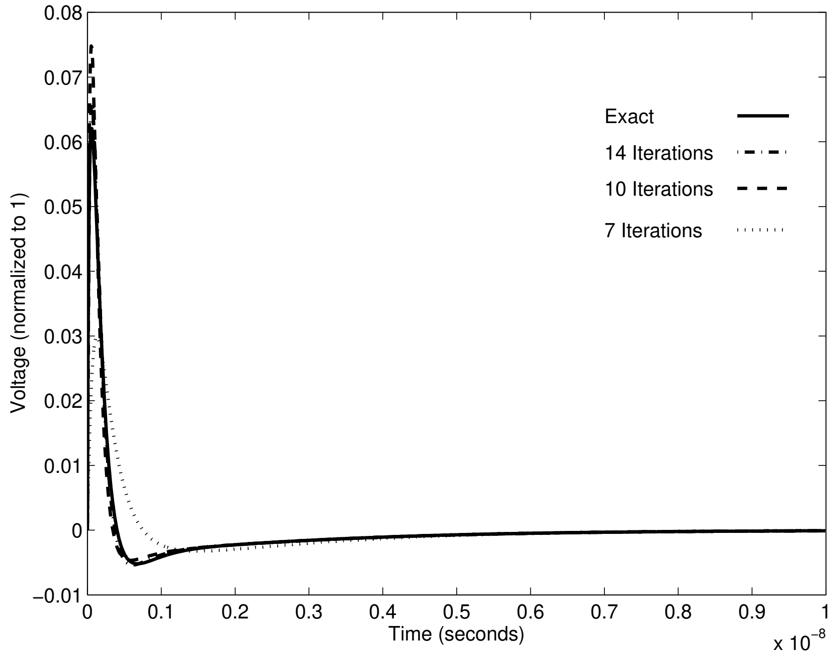

Figure 6: Waveform at load input in indirectly coupled path

The use of the reduced-order modeling of interconnect blocks leads to important saving in computing time and storage in higher-level analysis.

References

[1] J.I. Aliaga, D.L. Boley, R.W. Freund, and V. Hernández, "A Lanczos-type algorithm for multiple starting vectors," Numerical Analysis Manuscript No. 96–18, Bell Laboratories, Murray Hill, NJ, Sep. 1996.

[2] B.D.O. Anderson and S. Vongpanitlerd, *Network Analysis and Synthesis*, Englewood Cliffs, NJ: Prentice-Hall, 1973.

[3] G.A. Baker, Jr. and P. Graves-Morris, *Padé Approximants*, Second Edition, New York, NY: Cambridge University Press, 1996.

[4] J.E. Bracken, "Algorithms for passive modeling of linear interconnect networks," manuscript.

[5] P. Feldmann and R.W. Freund, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," in *Proc. Euro-DAC*, Sep. 1994.

[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] P. Feldmann and R.W. Freund, "Reduced-order modeling of large linear subcircuits via a block Lanczos algorithm," in *Proc. 32nd ACM/IEEE Design Automation Conf.*, June 1995.

[8] R.W. Freund, "An extension of the Lanczos-Padé connection to the matrix case," Numerical Analysis Manuscript, Bell Laboratories, Murray Hill, NJ, in preparation.

[9] R.W. Freund and P. Feldmann, "Reduced-order modeling of large passive linear circuits by means of the SyPVL algorithm," in *Tech. Dig. 1996 IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1996.

[10] R.W. Freund and P. Feldmann, "Reduced-order modeling of large linear passive multiterminal circuits," manuscript.

[11] R.W. Freund and N.M. Nachtigal, "Software for simplified Lanczos and QMR algorithms," *Appl. Numer. Math.*, vol. 19, pp. 319–341, 1995.

[12] G.H. Golub and C.F. Van Loan, *Matrix Computations*, Third Edition. Baltimore, MD: The Johns Hopkins University Press, 1996.

[13] W.B. Gragg, "Matrix interpretations and applications of the continued fraction algorithm," *Rocky Mountain J. Math.*, vol. 4, pp. 213–225, 1974.

[14] K.J. Kerns and A.T. Yang, "Stable and efficient reduction of large, multiport RC networks by pole analysis via congruence transformations," manuscript.

[15] C. Lanczos, "An iteration method for the solution of the eigenvalue problem of linear differential and integral operators," *J. Res. Nat. Bur. Standards*, vol. 45, pp. 255–282, 1950.

[16] 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.

[17] V. Raghavan, R.A. Rohrer, L.T. Pillage, J.Y. Lee, J.E. Bracken, and M.M. Alaybeyi, "AWE–inspired," in *Proc. IEEE Custom Integrated Circuits Conference*, May 1993.

[18] A. Ruhe, "Implementation aspects of band Lanczos algorithms for computation of eigenvalues of large sparse symmetric matrices," *Math. Comp.*, vol. 33, pp. 680–687, 1979.

[19] L.M. Silveira, M. Kamon, I. Elfadel, and J. White, "A coordinate-transformed Arnoldi algorithm for generating guaranteed stable reduced-order models of RLC circuits," in *Tech. Dig. 1996 IEEE/ACM International Conference on Computer-Aided Design*, Nov. 1996.

[20] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*, Second Edition, New York, N.Y.: Van Nostrand Reinhold, 1993.

[21] M.R. Wohlers, *Lumped and Distributed Passive Networks*, New York, N.Y.: Academic Press, 1969.