The SyMPVL Algorithm and Its Applications to Interconnect Simulation

Roland W. Freund and Peter Feldmann
Bell Laboratories, Lucent Technologies
700 Mountain Avenue
Murray Hill, NJ 07974–0636

***Abstract*—This paper describes SyMPVL, an algorithm for the computation of multi-port transfer functions of RLC circuits. SyMPVL employs a J-symmetric Lanczos-type algorithm for multiple starting vectors to reduce the original circuit matrices to a pair of banded symmetric matrices. These matrices, which are typically much smaller than the circuit matrices, determine a reduced-order model of the original multi-port. The transfer function of the model represents a matrix-Padé approximation of the multi-port matrix transfer function. Numerical results for SyMPVL applied to interconnect simulation problems are reported.**

I. Introduction

In recent years, reduced-order modeling techniques based on Padé approximation have been recognized to be powerful tools for the simulation of large linear or linearized electronic circuits. The first such technique was asymptotic waveform evaluation (AWE) [1], [2]. AWE generates a Padé approximation to the circuit's transfer function via explicit moment matching, which is an inherently numerically unstable procedure, and due to this instability, it is limited to Padé approximations of low order. The problems of AWE can be remedied by exploiting the well-known Lanczos-Padé connection [3] that allows to compute Padé approximations stably via the Lanczos process [4]. The use of the resulting PVL (**P**adé **V**ia a **L**anczos) algorithm was advocated in [5], [6]; see also [7] for further references.

Both AWE and PVL are techniques for the simulation of circuits with single inputs and outputs. In [8], we introduced the MPVL (**M**atrix **P**adé **V**ia a **L**anczos-type process) algorithm that extends PVL to the general case of multi-input multi-output circuits. The behavior of such circuits is characterized by a matrix-valued transfer function. MPVL generates a reduced-order model, the matrix transfer function of which represents the matrix-Padé approximation of the original circuit's matrix transfer function. MPVL is a general algorithm, in the sense that it can stably compute reduced-order models for any linear or linearized circuit.

Electronic circuits often contain large linear RLC subnetworks, especially in interconnect simulations. For example, such subnetworks may represent interconnect automatically extracted from layout as large RLC networks, models of IC packages, or models of wireless propagation channels. RLC circuits can be characterized in terms of square and symmetric matrix transfer functions. Direct application of the MPVL algorithm to RLC circuits does not take advantage of this special structure and cannot guarantee in general the preservation of stability and passivity of the approximation.

In this paper, we describe SyMPVL, a variant of MPVL that is tailored to the computation of the symmetric multi-port transfer function of an RLC circuit. SyMPVL employs a **J**-symmetric Lanczos-type algorithm for multiple starting vectors to reduce the original circuit matrices to a pair of banded symmetric matrices. These matrices, which are typically much smaller than the circuit matrices, determine a reduced-order model of the original multi-port. The transfer function of the model represents a matrix-Padé approximation of the multi-port matrix transfer function. Numerical results for SyMPVL applied to interconnect simulation problems are reported.

II. Formulation of Circuit Equations

In this section, we show how, for RLC circuits, a suitable formulation of the circuit equations result in symmetric multi-port transfer functions.

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 as follows:

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

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 RLC circuits, and for simplicity, we assume that the circuit is excited just by current sources. In this case, the adjacency matrix and the branch current and voltage vectors can be partitioned according to circuit-element types as follows:

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

Here, the subscripts i, g, c, and l stand for branches containing current sources, resistors, capacitors, and inductors, 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 RLC circuits, the BCR's are as follows:

$$\mathbf{i}_i = -\mathbf{I}_t(t),\ \mathbf{i}_g = \mathcal{G}\,\mathbf{v}_g,\ \mathbf{i}_c = \mathcal{C}\,\frac{d}{dt}\mathbf{v}_c,\ \mathbf{v}_l = \mathcal{L}\,\frac{d}{dt}\mathbf{i}_l. \quad (2)$$

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 matrix $\mathcal{L}$ is also diagonal in the absence of inductive coupling. Inductive coupling introduces off-diagonal terms in the inductance matrix, but $\mathcal{L}$ remains symmetric and positive definite.

The modified nodal formulation (MNA) of the circuit equations is obtained by combining the Kirchhoff equations (1) with the BCRs (2), and eliminating as many current unknowns as possible. For the case of RLC circuits, only inductor currents need to be left as unknowns. Setting

$$\begin{aligned} &\mathbf{G} = \begin{bmatrix} \mathbf{A}_g^{\mathrm{T}}\mathcal{G}\mathbf{A}_g & \mathbf{A}_l^{\mathrm{T}} \\ \mathbf{A}_l & \mathbf{0} \end{bmatrix}, \quad \mathbf{C} = \begin{bmatrix} \mathbf{A}_c^{\mathrm{T}}\mathcal{C}\mathbf{A}_c & \mathbf{0} \\ \mathbf{0} & -\mathcal{L} \end{bmatrix}, \\ &\mathbf{x} = \begin{bmatrix} \mathbf{v}_n \\ \mathbf{i}_l \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{A}_i^{\mathrm{T}} \\ \mathbf{0} \end{bmatrix}, \end{aligned} \quad (3)$$

the resulting MNA equations can be written compactly in matrix form as follows:

$$\mathbf{G}\,\mathbf{x} + \mathbf{C}\,\frac{d}{dt}\mathbf{x} = \mathbf{B}\,\mathbf{I}_t(t). \quad (4)$$

Note that the matrices $\mathbf{G}$ and $\mathbf{C}$ in (3) are symmetric. They are however indefinite in general.

We are interested in determining the network functions of the RLC block viewed as a p-terminal component. Since we allowed only current sources in our formulation, it is natural to determine the matrix $\mathbf{Z}(s)$ of $\mathbf{Z}$-parameters. By applying the Laplace transform to (4) and assuming zero initial conditions, we obtain

$$\begin{aligned} (\mathbf{G} + s\,\mathbf{C})\,\mathbf{X} &= \mathbf{B}\,\mathbf{I}_s(s), \\ \mathbf{V}_i &= \mathbf{B}^{\mathrm{T}}\,\mathbf{X}. \end{aligned} \quad (5)$$

Here, $\mathbf{X}$, $\mathbf{I}_s(s)$, and $\mathbf{V}_i$ represent the Laplace transforms of the unknown vector $\mathbf{x}$, the excitation current $\mathbf{I}_t(t)$, and the vector of voltages across the excitation sources, respectively. Eliminating $\mathbf{X}$ in (5) gives

$$\begin{aligned} \mathbf{V}_i = [\mathbf{A}_i \quad \mathbf{0}]\,\mathbf{X} &= \mathbf{Z}(s)\,\mathbf{I}_s(s), \\ \text{where} \quad \mathbf{Z}(s) &= \mathbf{B}^{\mathrm{T}}\,(\mathbf{G} + s\,\mathbf{C})^{-1}\,\mathbf{B}. \end{aligned} \quad (6)$$

Here, $\mathbf{G}$ and $\mathbf{C}$ are symmetric $N \times N$ matrices, and $\mathbf{B}$ is an $N \times p$ matrix. For general RLC circuits, $\mathbf{G}$ and $\mathbf{C}$ are indefinite. In the important special cases of RC, RL, and LC circuits, the formulation can be modified such that $\mathbf{G}$ and $\mathbf{C}$ become positive definite.

In the next section, we describe the SyMPVL algorithm for computing matrix-Padé approximants of the matrix-valued transfer function $\mathbf{Z}$ in (6).

III. The SyMPVL Algorithm

Note that $\mathbf{Z}$ is a rational matrix function whose order is N in general, where N is very large. The basic idea of matrix-Padé approximation [9], [10] is to replace $\mathbf{Z}$ by a rational matrix function of much lower order $n \ll N$ such that the Taylor expansions of $\mathbf{Z}$ and $\mathbf{Z}_n$ about some expansion point $s_0 \in \mathbb{C}$ agree in as many leading terms as possible; without loss of generality, we set $s_0 = 0$. More precisely, a matrix-valued function $\mathbf{Z}_n : \mathbb{C} \longmapsto (\mathbb{C} \cup \{\infty\})^{p \times p}$ is called an nth *matrix-Padé approximant* of $\mathbf{Z}$ if $\mathbf{Z}_n$ is of the form

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

where $\mathbf{A}_n$ and $\mathbf{B}_n$ are symmetric $n \times n$ matrices and $\boldsymbol{\rho}_n$ is an $n \times p$ matrix, and

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

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.

SyMPVL uses a $\mathbf{J}$-symmetric Lanczos-type algorithm for multiple starting vectors (namely the p columns of $\mathbf{B}$) to generate a sequence of nth matrix-Padé approximants $\mathbf{Z}_n$ to $\mathbf{Z}$. This algorithm is a special case of the more general procedure described in [11]. 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, so-called *look-ahead* techniques are required to avoid potential breakdowns due to division by quantities that cannot be excluded to be zero. For simplicity, in this paper we only state the algorithm without look-ahead.

First, we precompute a factorization of the symmetric matrix $\mathbf{G}$ of the form

$$\mathbf{G} = \mathbf{L}\,\mathbf{J}^{-1}\,\mathbf{L}^{\mathrm{T}}, \quad (7)$$

where $\mathbf{L}$ is a lower triangular matrix or a permutation of a lower triangular matrix, and $\mathbf{J}$ is a very simple matrix such that multiplications with $\mathbf{J}$ are basically free. For example, $\mathbf{J}$ could be a permutation matrix or a diagonal matrix with diagonal entries ± 1. Recall that $\mathbf{G}$ is positive definite for RC, RL, and LC circuits, and in these cases, we can choose $\mathbf{J} = \mathbf{I}$ in (7). Note that linear systems with coefficient matrices $\mathbf{L}$ and $\mathbf{L}^{\mathrm{T}}$ are "easy" to solve. Using (7), we rewrite $\mathbf{Z}$ from (6) as follows:

$$\mathbf{Z}(s) = \left(\mathbf{J}\,\mathbf{L}^{-1}\,\mathbf{B}\right)^{\mathrm{T}} \left(\mathbf{I} + s\,\mathbf{L}^{-1}\,\mathbf{C}\,\mathbf{L}^{-\mathrm{T}}\,\mathbf{J}\right)^{-1} \left(\mathbf{L}^{-1}\,\mathbf{B}\right).$$

The Lanczos-type algorithm is then applied to the matrix $\mathbf{A} = \mathbf{L}^{-1}\,\mathbf{C}\,\mathbf{L}^{-\mathrm{T}}\,\mathbf{J}$ and the right initial blocks $\mathbf{L}^{-1}\,\mathbf{B}$. Note that $\mathbf{A}$ satisfies $\mathbf{A}^{\mathrm{T}}\,\mathbf{J} = \mathbf{J}\,\mathbf{A}$, i.e., $\mathbf{A}$ is $\mathbf{J}$-symmetric.

The $\mathbf{J}$-symmetric Lanczos-type algorithm generates a sequence of vectors $\{\mathbf{v}_n\}_{n\geq 1}$ that are $\mathbf{J}$-orthogonal:

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

After n steps of Algorithm 1 below, the Lanczos vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_n$ have been constructed. In addition, at the end of each nth step, there are $p_c = p_c(n)$ "auxiliary" vectors, $\hat{\mathbf{v}}_{n+1}, \hat{\mathbf{v}}_{n+2}, \ldots, \hat{\mathbf{v}}_{p_c}$, available. These vectors will be turned into Lanczos vectors or deflated in successive iterations. Here, p_c denotes the current block size. Initially, $p_c = p$, and then within the algorithm p_c is reduced by one every time a deflation occurs.

Algorithm 1 ($\mathbf{J}$-symmetric Lanczos-type method with deflation, but without look-ahead.)

INPUT:

Matrices $\mathbf{G} = \mathbf{G}^{\mathrm{T}} = \mathbf{L}\,\mathbf{J}^{-1}\,\mathbf{L}^{\mathrm{T}}$, $\mathbf{C} = \mathbf{C}^{\mathrm{T}} \in \mathbb{R}^{N\times N}$;
A block $\mathbf{B} = \begin{bmatrix}\mathbf{b}_1 & \mathbf{b}_2 & \cdots & \mathbf{b}_p\end{bmatrix} \in \mathbb{R}^{N\times p}$;
The deflation tolerance `dtol`.

OUTPUT:

The $p_1 \times p$ matrix $\boldsymbol{\rho} = \left[\rho_{i,j}\right]_{1\leq i\leq p_1,\, 1\leq j\leq p}$, where
$p_1 = p-$(# of deflations during the first p iterations);
The $n \times n$ matrices $\mathbf{T}_n = \left[t_{i,j}\right]_{1\leq i,j\leq n}$ and
$\mathbf{\Delta}_n = \mathrm{diag}\,(\delta_1, \delta_2, \ldots, \delta_n)$, where n is the value of the iteration counter at termination.

0) *For $k = 1, 2, \ldots, p$, compute $\hat{\mathbf{v}}_k = \mathbf{L}^{-1}\mathbf{b}_k$ by solving the linear system $\mathbf{L}\,\hat{\mathbf{v}}_k = \mathbf{b}_k$.*
Set $p_c = p$. (p_c is the current block size.)
Set $\mathcal{I}_{\mathbf{v}} = \emptyset$. ($\mathcal{I}_{\mathbf{v}}$ records deflation. Any newly constructed $\mathbf{v}$ vector must be explicitly $\mathbf{J}$-orthogonalized against all vectors $\mathbf{v}_k$ with $k \in \mathcal{I}_{\mathbf{v}}$.)

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

1) (If necessary, deflate $\hat{\mathbf{v}}_n$.)
If $\|\hat{\mathbf{v}}_n\| >$ `dtol`, then continue with step 2). Otherwise, deflate $\hat{\mathbf{v}}_n$ by doing the following:

1a) *If $p_c = 1$, then stop.* (There are no more Krylov vectors.)

1b) *If $\phi = n - p_c > 0$, then set $\mathcal{I}_{\mathbf{v}} = \mathcal{I}_{\mathbf{v}} \cup \{\,\phi\,\}$, and save the vector $\mathbf{v}_\phi$.*

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

1d) *Repeat all of step* 1).

2) (Normalize $\hat{\mathbf{v}}_n$ to obtain $\mathbf{v}_n$.) *Set*

$$t_{n,n-p_c} = \|\hat{\mathbf{v}}_n\|_2 \quad \textit{and} \quad \mathbf{v}_n = \frac{\hat{\mathbf{v}}_n}{t_{n,n-p_c}}.$$

3) (Compute δ_n.) *Set*

$$\delta_n = \left(\mathbf{J}\,\mathbf{v}_n\right)^{\mathrm{T}}\,\mathbf{v}_n.$$

If $\delta_n = 0$, then stop. (The algorithm would require look-ahead in order to be able to continue.)

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

4a) *Obtain $\mathbf{v} = \mathbf{L}^{-1}\,\mathbf{C}\,\mathbf{L}^{-\mathrm{T}}\,(\mathbf{J}\,\mathbf{v}_n)$ by first solving the linear system $\mathbf{L}^{\mathrm{T}}\,\mathbf{t} = \mathbf{J}\,\mathbf{v}_n$ for $\mathbf{t}$ and then solving the linear system $\mathbf{L}\,\mathbf{v} = \mathbf{C}\,\mathbf{t}$ for $\mathbf{v}$.*

4b) ($\mathbf{J}$-orthogonalize $\mathbf{v}$ against previous vectors.)
Set $i_{\mathbf{v}} = \max\{\,1, n - p_c\,\}$ and define the temporary index set

$$\mathcal{I} = \{\,i_{\mathbf{v}}, i_{\mathbf{v}} + 1, \ldots, n-1\,\} \cup \bigcup_{\substack{i\in\mathcal{I}_{\mathbf{v}} \\ i<i_{\mathbf{v}}}} \{i\}.$$

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

$$\begin{aligned} t_{i,n} &= \begin{cases} t_{n,i}\,\dfrac{\delta_n}{\delta_i} & \textit{if } i = n - p_c, \\[2ex] \dfrac{\left(\mathbf{J}\,\mathbf{v}_i\right)^{\mathrm{T}}\,\mathbf{v}}{\delta_i} & \textit{otherwise}, \end{cases} \\ \mathbf{v} &= \mathbf{v} - \mathbf{v}_i\,t_{i,n}. \end{aligned}$$

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

5) ($\mathbf{J}$-orthogonalize the auxiliary vectors against $\mathbf{v}_n$.)
For $i = n - p_c + 1, \ldots, n$, set

$$\begin{aligned} t_{n,i} &= \begin{cases} \dfrac{\left(\mathbf{J}\,\mathbf{v}_n\right)^{\mathrm{T}}\,\hat{\mathbf{v}}_{p_c+i}}{\delta_n} & \textit{if } i \leq 0 \textit{ or } i = n, \\[2ex] t_{i,n}\,\dfrac{\delta_i}{\delta_n} & \textit{otherwise}, \end{cases} \\ \hat{\mathbf{v}}_{p_c+i} &= \hat{\mathbf{v}}_{p_c+i} - \mathbf{v}_n\,t_{n,i}. \end{aligned}$$

6) (In the initial iterations, set up $\boldsymbol{\rho}$.)
If $n \leq p_c$, set

$$\rho_{n,i} = t_{n,i-p} \quad \textit{for all} \quad n - p_c + p \leq i \leq p.$$

As it is shown in [10], the quantities $\boldsymbol{\rho}$, $\mathbf{\Delta}_n$, and $\mathbf{T}_n$ generated by n steps of Algorithm 1 define an nth matrix-Padé approximant $\mathbf{Z}_n$ of $\mathbf{Z}$ as follows:

$$\mathbf{Z}_n(s) = \rho_n^{\mathrm{T}}\left(\mathbf{\Delta}_n^{-1} + s\,\mathbf{T}_n\,\mathbf{\Delta}_n^{-1}\right)^{-1}\rho_n, \quad \rho_n = \begin{bmatrix}\rho \\ \mathbf{0}\end{bmatrix}. \tag{8}$$

Moreover, for the special cases of RC, RL, and LC circuits, it can be shown [12] that the reduced-order model given by (8) is stable and passive.

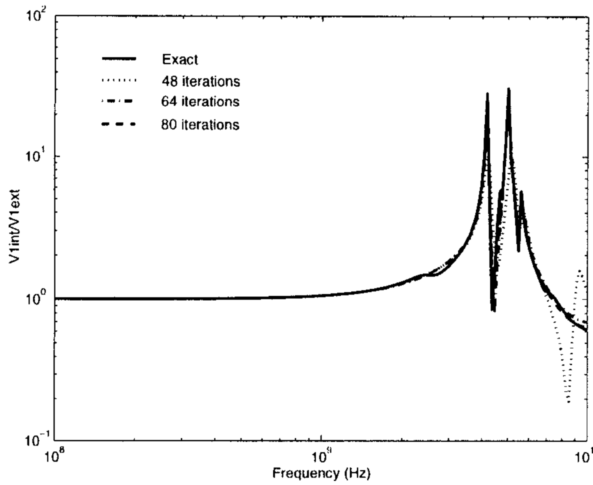

Fig. 1. Package: Pin no. 1 external to Pin no. 1 internal

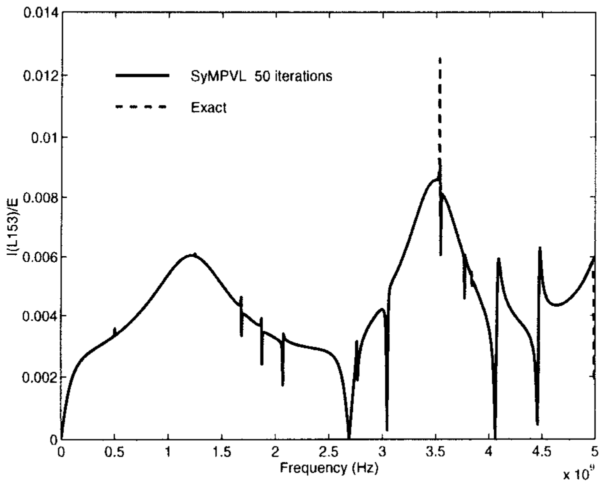

Fig. 2. The PEEC circuit transfer function

IV. Examples

We tested SyMPVL on a variety of interconnect-simulation problems. Here, we report results for three of them. The first example represents the approximation of a 16×16 transfer function of an IC package modeled as an RLC network. Figure 1 shows the transfer characteristic of one of the pins.

The second example is the circuit resulting from the PEEC modeling [13] of an electromagnetic problem. The circuit consists of only inductors, capacitors, and inductive couplings, and it is driven by a finite impedance source. The LC circuit's two-port transfer function was approximated with SyMPVL and the excitation was applied to the reduced-order model. Figure 2 shows the response of this circuit, the current flowing through one of the inductors.

The final example represents the simulation of cross-talk in a digital circuit. The interconnect is modeled by an RC network. In this example a 7×7 transfer function was approximated. Figure 3 shows the time-domain response of an interconnect wire terminal when a signal is switching in a neighboring wire.

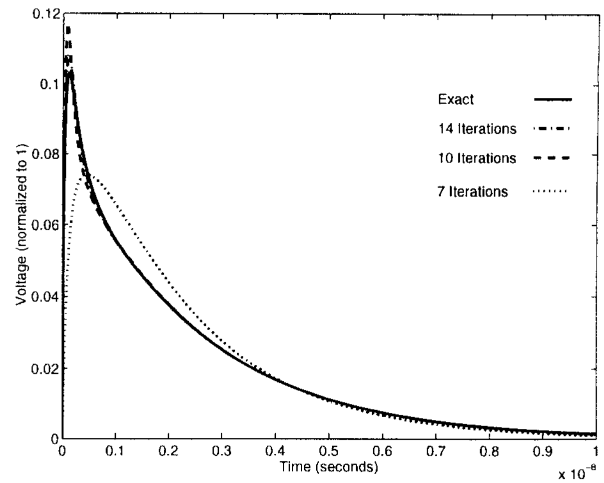

Fig. 3. Cross-talk waveform

References

[1] 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.

[2] 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.

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

[4] 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.

[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] R.W. Freund, "Circuit simulation techniques based on Lanczos-type algorithms," in *Systems and Control in the Twenty-First Century*, C.I. Byrnes, B.N. Datta, D.S. Gilliam, and C.F. Martin, Eds. Boston: Birkhäuser, 1997, pp. 171–184.

[8] 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.

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

[10] R.W. Freund, "An extension of the Lanczos-Padé connection to the matrix case," Numerical Analysis Manuscript, Bell Laboratories, Murray Hill, NJ, 1997.

[11] 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.

[12] R.W. Freund and P. Feldmann, "Reduced-order modeling of large linear passive multi-terminal circuits," Numerical Analysis Manuscript, Bell Laboratories, Murray Hill, NJ, 1997.

[13] A.E. Ruehli, "Equivalent circuit models for three-dimensional multiconductor systems," *IEEE Trans. Microwave Theory and Tech.*, vol. 22, pp. 216–221, Mar. 1974.