Reduced-Order Modeling of Large Linear Subcircuits via a Block Lanczos Algorithm

Peter Feldmann
AT&T Bell Laboratories
Room 2C–217
600 Mountain Avenue
Murray Hill, NJ 07974–0636

Roland W. Freund
AT&T Bell Laboratories
Room 2C–420
600 Mountain Avenue
Murray Hill, NJ 07974–0636

Abstract

In this paper, we describe a method for the efficient computation of accurate reduced-order models of large linear circuits. The method, called MPVL, represents a generalization of the previously published PVL algorithm, and employs a novel block Lanczos algorithm to compute matrix Padé approximations of matrix-valued network transfer functions. The reduced-order models, computed to the required level of accuracy, are used to speed up the analysis of circuits containing large linear blocks. The linear blocks are replaced by their reduced-order models, and the resulting, smaller, circuit can be analyzed with general purpose simulators, such as SPICE, with significant savings in simulation time and, practically, no loss of accuracy.

1 Introduction

It is frequently the case that an electronic circuit contains large linear subnetworks. Such a situation often arises when large interconnect networks are automatically extracted from layout, or when circuits contain models of distributed elements, such as transmission lines, ground planes, or three-dimensional structures. The direct use of time-domain nonlinear differential-equation integration, as implemented in SPICE-like simulators, would be inefficient or even prohibitive for such large problems.

A significantly more efficient way to analyze such circuits is to replace the linear subnetwork with a reduced-order model that approximates sufficiently well its external behavior. This way, the linear part of the problem is handled by an efficient method dedicated to the analysis of linear circuits, and the problem left for the nonlinear simulator to solve becomes substantially smaller. The overall approach results in important computational savings and can tackle problems of sizes that are out of the reach of conventional simulators.

Such a method was implemented in AWESpice [1] where the Asymptotic Waveform Evaluation (AWE) algorithm [2, 3] was used to analyze and approximate the linear subcircuit. In AWESpice, all the moments of the linear subcircuit are first generated through an efficient recursive procedure. Then, moment matching is applied to compute Padé approximations to all entries of the linear subcircuit admittance matrix. Finally, this approximated admittance matrix is used in a modified SPICE simulator to represent the large linear subnetwork. The moment-matching phase of the AWE algorithm, however, has inherent numerical limitations, discussed in [4], which restrict its applicability to linear subcircuits that can be modeled accurately using only a relatively small number of poles.

In this paper, we introduce a new algorithm called MPVL (**M**atrix **P**adé **V**ia a **L**anczos-type process), for the accurate and efficient computation of reduced-order models of large linear circuits. The MPVL algorithm represents a generalization of the recently introduced PVL (**P**adé **V**ia the **L**anczos process) algorithm [4, 5]. Similar to AWESpice, the new algorithm constructs a Padé approximation to the matrix-valued transfer function that characterizes the linear subcircuit. The MPVL algorithm, however, like its predecessor PVL, uses a numerically robust Lanczos-type process, instead of the numerically unstable moment generation, and therefore, it does not suffer from the numerical limitations of AWE. In particular, MPVL can compute Padé approximations of arbitrary linear network transfer-functions, up to any order, as required by the desired level of accuracy. Furthermore, the Padé approximation computed by MPVL is different from the one used by AWESpice. Instead of computing individual Padé approximations for each entry of the matrix-valued transfer-function, MPVL computes a single matrix-valued Padé approximation. The resulting reduced-order model can be easily and naturally integrated in the nonlinear simulation.

The remainder of the paper is organized as follows. In Section 2, we first formalize the replacement of linear blocks within nonlinear circuits with their reduced-order models. Secondly, we discuss reduced-order modeling of blocks having a single input and a single output. In Section 3, we state the novel block Lanczos algorithm, and we list some of its properties. In Section 4, we establishes the connection of the block Lanczos algorithm with the matrix Padé approximation that is used to model blocks having multiple inputs and outputs. In Section 5, we report the results of numerical experiments with the new algorithm for an illustrative example. In Section 6, we make some concluding remarks.

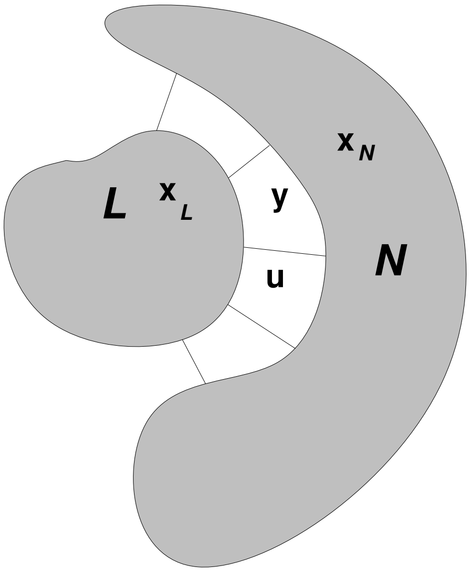

Figure 1: Circuit partitioning

2 Macromodels Generated from PVL

Using any circuit-equation formulation method such as modified nodal analysis, sparse tableau, etc. [6], a circuit can be described by the following system of first-order differential equations:

$$\frac{d}{dt}\hat{\mathbf{q}}(\mathbf{z},t)+\hat{\mathbf{f}}(\mathbf{z},t)=0. \tag{1}$$

Here, $\mathbf{z}=\mathbf{z}(t)$ is the vector of circuit variables at time t, the term $\hat{\mathbf{f}}(\mathbf{z},t)$ represents the contribution of nonreactive elements such as resistors, sources, etc., and the term $\frac{d}{dt}\hat{\mathbf{q}}(\mathbf{z},t)$ represents the contribution of reactive elements such as capacitors and inductors.

2.1 Circuits with Linear Subnetworks

From now on, we assume that a linear subnetwork $\mathcal{L}$ can be separated from the circuit, as shown in Figure 1. The circuit variables can then be partitioned as follows:

$$\mathbf{z}=\begin{bmatrix}\mathbf{x}_N\\ \mathbf{y}\\ \mathbf{x}_L\end{bmatrix}, \tag{2}$$

where $\mathbf{x}_L$ denotes the circuit variables exclusive to the linear subnetwork $\mathcal{L}$, $\mathbf{x}_N$ denotes the circuit variables exclusive to the remainder $\mathcal{N}$ of the circuit, and $\mathbf{y}$ represents the variables shared by the two blocks. Using the partitioning (2) and a suitable reordering of the equations, the circuit equations (1) can be rewritten in the following form:

$$\frac{d}{dt}\left(\begin{bmatrix} \mathbf{q}\left(\begin{bmatrix} \mathbf{x}_N \\ \mathbf{y} \end{bmatrix}, t\right) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{C} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{y} \\ \mathbf{x}_L \end{bmatrix}\right) + \begin{bmatrix} \mathbf{f}\left(\begin{bmatrix} \mathbf{x}_N \\ \mathbf{y} \end{bmatrix}, t\right) \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ \mathbf{G} \end{bmatrix} \cdot \begin{bmatrix} \mathbf{y} \\ \mathbf{x}_L \end{bmatrix} = 0. \tag{3}$$

In (3), the vector-valued functions $\mathbf{q}$ and $\mathbf{f}$ represent the contributions of resistive and reactive elements from the nonlinear partition of the circuit, and the matrices $\mathbf{C}$ and $\mathbf{G}$ represent, respectively, the contributions of resistive and reactive elements in the linear partition. Furthermore, in (3), without loss of generality, we assume that the vectors $\mathbf{q}$ and $\mathbf{f}$ have the same length, and that the matrices $\mathbf{C}$ and $\mathbf{G}$ are square and of the same size; this can always be achieved by padding $\mathbf{q}$, $\mathbf{f}$, $\mathbf{C}$, or $\mathbf{G}$ with additional zeros, if necessary. Note that there are three types of equations in (3). The leading equations involve only quantities from the nonlinear partition, while the trailing equations involve only quantities from the linear partition. The remaining equations connect quantities from both partitions, and in the sequel we denote by m the number of these connecting equations.

The system of differential equations (3) can be solved using nonlinear ordinary differential equation solvers, such as the ones used in SPICE-like simulators. When the cardinality of $\mathbf{x}_L$ is large, however, significant computational savings can be achieved by first replacing the linear subnetwork with a reduced-order model, which, within approximation error, maintains the same behavior at the interface. The rest of the paper describes the computation of such reduced-order models and their use to speed up the analysis of nonlinear circuits.

Formally, the equations referring to the linear subnetwork in system (3) can be separated through the introduction of a new m-dimensional vector, $\mathbf{u} = \mathbf{u}(t)$, of circuit variables, which represent interface signals. Indeed, (3) is equivalent to the following coupled system of equations:

$$\begin{aligned} \frac{d}{dt}\mathbf{q}\left(\begin{bmatrix} \mathbf{x}_N \\ \mathbf{y} \end{bmatrix}, t\right) + \mathbf{f}\left(\begin{bmatrix} \mathbf{x}_N \\ \mathbf{y} \end{bmatrix}, t\right) + \begin{bmatrix} 0 \\ \mathbf{I}_m \end{bmatrix}\mathbf{u} &= 0, \\ \mathbf{C}\frac{d}{dt}\begin{bmatrix} \mathbf{y} \\ \mathbf{x}_L \end{bmatrix} + \mathbf{G}\begin{bmatrix} \mathbf{y} \\ \mathbf{x}_L \end{bmatrix} &= \begin{bmatrix} \mathbf{I}_m \\ 0 \end{bmatrix}\mathbf{u}. \end{aligned} \tag{4}$$

Here, $\mathbf{I}_m$ denotes the $m \times m$ identity matrix. In the case of a nodal formulation, the variables $\mathbf{y}$ and $\mathbf{u}$ would represent, respectively, the voltages and the currents in the wires that connect the linear and the nonlinear partitions of the circuit. We now set

$$\mathbf{x} = \begin{bmatrix} \mathbf{y} \\ \mathbf{x}_L \end{bmatrix}, \quad \mathbf{B} = \begin{bmatrix} \mathbf{I}_m \\ 0 \end{bmatrix}, \quad \text{and} \quad \mathbf{L} = \begin{bmatrix} \mathbf{I}_p \\ 0 \end{bmatrix}, \tag{5}$$

where p denotes the length of the vector $\mathbf{y}$. Note that $\mathbf{L}^T$ is the matrix that selects the $\mathbf{y}$ subvector from $\mathbf{x}$, i.e., $\mathbf{y} = \mathbf{L}^T\mathbf{x}$. The linear subnetwork is then described by the following coupled system of equations:

$$\begin{aligned} \mathbf{C}\frac{d}{dt}\mathbf{x} + \mathbf{G}\mathbf{x} &= \mathbf{B}\mathbf{u}, \\ \mathbf{y} &= \mathbf{L}^T\mathbf{x}. \end{aligned} \tag{6}$$

Equations (6) describe an m-input p-output linear network, which can also be analyzed in terms of its $p \times m$ matrix of Laplace-domain transfer functions. The expression for the transfer functions is obtained as follows. We first apply the Laplace transform to the equations (6), and we get

$$\begin{aligned} s\mathbf{CX} + \mathbf{GX} &= \mathbf{BU}, \\ \mathbf{Y} &= \mathbf{L}^T\mathbf{X}. \end{aligned} \tag{7}$$

Here, $\mathbf{X}$, $\mathbf{U}$, and $\mathbf{Y}$ denote the Laplace transform of $\mathbf{x}$, $\mathbf{u}$, and $\mathbf{y}$, respectively. Next, we perform the substitutions (similar to the ones used in [4])

$$\mathbf{A} = -\left(\mathbf{G} + s_0\mathbf{C}\right)^{-1}\mathbf{C}, \quad \mathbf{R} = \left(\mathbf{G} + s_0\mathbf{C}\right)^{-1}\mathbf{B}, \tag{8}$$

where s_0 is a frequency shift, $s = s_0 + \sigma$, chosen such that the matrix $\mathbf{G} + s_0\mathbf{C}$ is nonsingular. From (7), we then obtain

$$\mathbf{Y} = \mathbf{L}^T\left(\mathbf{I} - \mathbf{A}\sigma\right)^{-1}\mathbf{RU}, \tag{9}$$

and the transfer-function matrix is given by

$$\mathbf{H}(s_0 + \sigma) = \mathbf{Y}(\sigma)(\mathbf{U}(\sigma))^{-1} = \mathbf{L}^T\left(\mathbf{I} - \mathbf{A}\sigma\right)^{-1}\mathbf{R}. \tag{10}$$

We are interested in approximating the matrix-valued transfer function $\mathbf{H}(s_0 + \sigma)$ with a reduced-order model that is still sufficiently accurate in the domain of interest.

2.2 Reduced-Order Modeling for Single-Input Single-Output Networks

We start by first discussing the scalar case when the linear subnetwork $\mathcal{L}$ has only one input and one output, i.e., $\mathcal{L}$ is a one-port. In the one-port case, the vectors $\mathbf{u}$ and $\mathbf{y}$ become scalar variables u and y, e.g., they are just the one-port current and voltage in a nodal representation, and the matrices $\mathbf{R}$ and $\mathbf{L}$ become column vectors. In [4], the PVL algorithm was introduced as a tool for computing the approximate poles, residues, zeros, and transfer function of a linear circuit described by equations (6). We now show that the same procedure can be employed to generate a reduced-order approximate model of the linear subnetwork.

We run the Lanczos process as described in [4] on the matrix $\mathbf{A}$ for k steps, using $\mathbf{L}$ and $\mathbf{R}$ as the left and right starting vectors. The Lanczos process produces two sequences of vectors, $\{\mathbf{v}_j\}_{j=1}^{k+1}$, and $\{\mathbf{w}_j\}_{j=1}^{k+1}$, and two tridiagonal matrices, $\mathbf{T}_k$ and $\tilde{\mathbf{T}}_k$ that have the following properties:

- The Lanczos vectors $\{\mathbf{v}_j\}_{j=1}^{k+1}$ and $\{\mathbf{w}_j\}_{j=1}^{k+1}$ are biorthogonal:

$$\mathbf{w}_i^T\mathbf{v}_j = \begin{cases} \delta_j, & \text{if } i = j, \\ 0, & \text{if } i \neq j, \end{cases} \quad \text{for all} \quad i, j = 1, 2, \ldots, k+1. \tag{11}$$

- The tridiagonal matrices

$$\mathbf{T}_k = \begin{bmatrix} \alpha_1 & \beta_2 & 0 & \cdots & 0 \\ \rho_2 & \alpha_2 & \beta_3 & \ddots & \vdots \\ 0 & \rho_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \beta_k \\ 0 & \cdots & 0 & \rho_k & \alpha_k \end{bmatrix}, \quad \tilde{\mathbf{T}}_k = \begin{bmatrix} \alpha_1 & \gamma_2 & 0 & \cdots & 0 \\ \eta_2 & \alpha_2 & \gamma_3 & \ddots & \vdots \\ 0 & \eta_3 & \ddots & \ddots & 0 \\ \vdots & \ddots & \ddots & \ddots & \gamma_k \\ 0 & \cdots & 0 & \eta_k & \alpha_k \end{bmatrix}, \tag{12}$$

  have the following relation to the original matrix $\mathbf{A}$:

$$\begin{aligned} \mathbf{A}\mathbf{V}_k &= \mathbf{V}_k\mathbf{T}_k + [\,0 \quad \cdots \quad 0 \quad \mathbf{v}_{k+1}\,]\,\rho_{k+1}, \\ \mathbf{A}^T\mathbf{W}_k &= \mathbf{W}_k\tilde{\mathbf{T}}_k + [\,0 \quad \cdots \quad 0 \quad \mathbf{w}_{k+1}\,]\,\eta_{k+1}, \end{aligned} \tag{13}$$

  where

$$\mathbf{V}_k = [\,\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_k\,] \quad \text{and} \quad \mathbf{W}_k = [\,\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_k\,] \tag{14}$$

  are matrices whose columns are just the Lanczos vectors.

- The matrices $\tilde{\mathbf{T}}_k$ and $\mathbf{T}_k$ satisfy

$$\tilde{\mathbf{T}}_k^T = \mathbf{D}_k\mathbf{T}_k\mathbf{D}_k^{-1}, \tag{15}$$

  where

$$\mathbf{D}_k = \mathbf{W}_k^T\mathbf{V}_k = \operatorname{diag}(\delta_1, \delta_2, \ldots, \delta_k). \tag{16}$$

One key feature of the Lanczos process is that it produces a very useful small-dimensional approximation, namely the matrix $\mathbf{T}_k$, to the (usually large-dimensional) matrix $\mathbf{A}$. The matrix $\mathbf{T}_k$ is—in some sense—the best $k \times k$ approximation to $\mathbf{A}$. We stress that this approximation is very good even if k is much smaller than the order N of the $N \times N$ matrix $\mathbf{A}$. In fact, the Lanczos algorithm is mostly applied to very large matrices, and typically, $k \ll N$.

The expression

$$\mathbf{H}_k(\sigma) = \mathbf{L}^T\mathbf{R}\;\mathbf{e}_1{}^T\,(\mathbf{I} - \mathbf{T}_k\sigma)^{-1}\,\mathbf{e}_1, \tag{17}$$

where $\mathbf{e}_1$ is the first unit vector, was shown in [4] to represent the kth order Padé approximation of the system transfer function, $\mathbf{H}(\sigma)$. The approximation is typically very good in a sufficiently large domain of the complex plane surrounding the expansion point s_0, to contain all frequencies of interest. The Padé approximation embodied by matrix $\mathbf{T}_k$ was used in [4] to compute the dominant poles, zeros, and residues of a linear system, and was differentiated in [5] in order to compute various sensitivities. We show that the same matrix $\mathbf{T}_k$ matrix contains the necessary information to compute the reduced-order model of the one-port.

By applying the substitutions (8) to the time-domain system of equations (6), we obtain

$$\begin{aligned} -\mathbf{A}\frac{d}{dt}\mathbf{x} + (\mathbf{I} + s_0\mathbf{A})\mathbf{x} &= \mathbf{R}\mathbf{u}, \\ \mathbf{y} &= \mathbf{L}^T\mathbf{x}. \end{aligned} \tag{18}$$

We set $\mathbf{x} = \mathbf{V}_k\mathbf{d}$, where $\mathbf{d}$ is a "short" column vector of size k, typically, $k \ll N$. This is in fact the only approximation that we are making: we constrain $\mathbf{x}$ to the k-dimensional subspace spanned by the right Lanczos vectors $\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k$. By premultiplying the first equation in (18) with the matrix $\mathbf{W}_k^T$, we obtain

$$\begin{aligned} -\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k\frac{d}{dt}\mathbf{d} + \mathbf{W}_k^T\mathbf{V}_k\mathbf{d} + s_0\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k\mathbf{d} &= \mathbf{W}_k^T\mathbf{R}u, \\ y &= \mathbf{L}^T\mathbf{V}_k\mathbf{d}. \end{aligned} \tag{19}$$

We observe that due to the Lanczos properties (11)–(16), we have $\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k = \mathbf{D}_k\mathbf{T}_k$ and $\mathbf{W}_k^T\mathbf{V}_k = \mathbf{D}_k$. Furthermore, note that $\mathbf{R} = \rho\mathbf{V}_k\mathbf{e}_1$, and $\mathbf{L} = \eta\mathbf{W}_k\mathbf{e}_1$ are—up to a scaling—just the starting vectors of the Lanczos process. Here, the scaling factors ρ, $\eta > 0$ are chosen such that $\|\mathbf{v}_1\|_2 = \|\mathbf{w}_1\|_2 = 1$. Using these properties, the system (19) becomes

$$\begin{aligned} -\mathbf{T}_k\frac{d}{dt}\mathbf{d} + (\mathbf{I} + s_0\mathbf{T}_k)\mathbf{d} &= \rho\ \mathbf{e}_1u, \\ y &= \eta\ \mathbf{e}_1^T\mathbf{D}_k\mathbf{d}. \end{aligned} \tag{20}$$

In other words, we can approximate the original system (4) with a smaller system in which the linear subnetwork is approximated by a reduced-order model whose transfer function is a kth order Padé approximation of the original system. The smaller system is as follows:

$$\frac{d}{dt}\mathbf{q}\left(\begin{bmatrix}\mathbf{x}_N \\ y\end{bmatrix}, t\right) + \mathbf{f}\left(\begin{bmatrix}\mathbf{x}_N \\ y\end{bmatrix}, t\right) + \begin{bmatrix}0 \\ 1\end{bmatrix} u = 0, \tag{21}$$

$$-\ \mathbf{T}_k\frac{d}{dt}\mathbf{d} + (\mathbf{I} + s_0\mathbf{T}_k)\mathbf{d} = \rho\ \mathbf{e}_1u, \tag{22}$$

$$y = \eta\ \delta_1 d_1. \tag{23}$$

Here, d_1 denotes the first component of $\mathbf{d} = [\,d_1 \quad d_2 \quad \cdots \quad d_k\,]^T$. In the following, we will also use the notation $\mathbf{d}_{2:k} = [\,d_2 \quad d_3 \quad \cdots \quad d_k\,]^T$. By (23), we then have

$$\mathbf{d} = \begin{bmatrix}\dfrac{y}{\eta\delta_1} \\ \mathbf{d}_{2:k}\end{bmatrix}. \tag{24}$$

Using (22), we can eliminate the interface variable u in (21), and reassemble a system of nonlinear equations. Together with (24), this leads to the system

$$\frac{d}{dt}\left(\begin{bmatrix}\mathbf{q}\left(\begin{bmatrix}\mathbf{x}_N \\ \mathbf{y}\end{bmatrix}, t\right) \\ 0\end{bmatrix} - \begin{bmatrix}0 \\ \mathbf{T}_k\end{bmatrix}\cdot\begin{bmatrix}\dfrac{y}{\rho\eta\delta_1} \\ \dfrac{\mathbf{d}_{2:k}}{\rho}\end{bmatrix}\right) + \begin{bmatrix}\mathbf{f}\left(\begin{bmatrix}\mathbf{x}_N \\ \mathbf{y}\end{bmatrix}, t\right) \\ 0\end{bmatrix} + \begin{bmatrix}0 \\ \mathbf{I} + s_0\mathbf{T}_k\end{bmatrix}\cdot\begin{bmatrix}\dfrac{y}{\rho\eta\delta_1} \\ \dfrac{\mathbf{d}_{2:k}}{\rho}\end{bmatrix} = 0, \tag{25}$$

which approximates the original system (3) and has a significantly reduced number of state variables.

An additional requirement for the reduced-order system (25) and the original system (3) to have similar solutions is compatibility in initial conditions, $\mathbf{x}_0 = \mathbf{V}_k \mathbf{d}_0$. Obviously, it is not possible to find values for $\mathbf{d}_0$ that correspond to arbitrary initial conditions, $\mathbf{x}_0$, in the original system. If, however, we restrict the initial conditions $\mathbf{x}_0$ to be zero except for the shared variable y, and, noting that

$$y_0 = \mathbf{l}^T \mathbf{x}_0 = \eta\ \mathbf{e}_1^T \mathbf{W}_k^T \mathbf{V}_k \mathbf{d}_0 = \eta\ \mathbf{e}_1^T \mathbf{D}_k \mathbf{d}_0 = \eta\ \delta_1 d_{01}, \tag{26}$$

the initial conditions $\mathbf{d}_0 = [\, y_0/(\eta\delta_1) \quad 0\,]^T$ are compatible with the initial conditions $\mathbf{x}_0 = [\, y_0 \quad 0\,]^T$.

In general, the interface between the nonlinear and the liner subnetworks is more complicated than a one-port. In this case the linear subnetwork is described by an $p \times m$ matrix of transfer functions. One way to construct a reduced-order model of such a linear system is to use the superposition property of linear networks and, as in AWESpice, obtain a Padè approximation separately for each pair of inputs and outputs. However, in this case, the computational cost of macromodeling and the number of macromodel state variables increase rapidly with the size of the interface. Moreover, by computing the individual transfer functions separately without sharing information, optimality is lost in some sense. In the following section, we introduce a superior method that is based on the computation of a matrix Padé approximation to the entire matrix-valued transfer function simultaneously. This method is based on a novel block Lanczos algorithm.

3 A Block Lanczos Algorithm

The block Lanczos algorithm we use is a generalization of the classical Lanczos algorithm [7]. We only state the block Lanczos algorithm in its simplest form; a more sophisticated version that incorporates deflation and look-ahead techniques is described in detail in [8].

The algorithm generates two sequences of vectors

$$\mathbf{v}_1, \mathbf{v}_2, \ldots, \mathbf{v}_k \quad \text{and} \quad \mathbf{w}_1, \mathbf{w}_2, \ldots, \mathbf{w}_k \tag{27}$$

that, for each $k = 1, 2, \ldots$, build bases for the spaces spanned by the first k vectors of the block Krylov sequences

$$\mathbf{R}, \mathbf{A}\mathbf{R}, \mathbf{A}^2\mathbf{R} \ldots \quad \text{and} \quad \mathbf{L}, \mathbf{A}^T\mathbf{L}, \left(\mathbf{A}^T\right)^2 \mathbf{L}, \ldots$$

respectively. The vectors are generated to be biorthogonal:

$$\mathbf{w}_i^T \mathbf{v}_j = \begin{cases} \delta_j, & \text{if } \ i = j, \\ 0, & \text{if } \ i \neq j, \end{cases} \quad \text{for all} \quad i, j = 1, 2, \ldots, k. \tag{28}$$

As before, we set

$$\mathbf{V}_k = [\,\mathbf{v}_1 \quad \mathbf{v}_2 \quad \cdots \quad \mathbf{v}_k\,] \quad \text{and} \quad \mathbf{W}_k = [\,\mathbf{w}_1 \quad \mathbf{w}_2 \quad \cdots \quad \mathbf{w}_k\,].$$

Here, the initial matrices $\mathbf{V}_m$ and $\mathbf{W}_p$ are obtained by biorthogonalizing the initial block $\mathbf{R}$ and $\mathbf{L}$ by a modified Gram-Schmidt-type process. In particular, we have

$$\mathbf{R} = \mathbf{V}_m \rho \quad \text{and} \quad \mathbf{L} = \mathbf{W}_p \eta, \tag{29}$$

where

$$\rho = [\,\rho_{ij}\,]_{i,j=1,2,\ldots,m} \quad \text{and} \quad \eta = [\,\eta_{ij}\,]_{i,j=1,2,\ldots,p}$$

are upper triangular matrices. The vectors (27) can be generated by $(m+p+1)$-term recurrences that can be compactly written in matrix formulation as follows:

$$\begin{aligned} \mathbf{A}\mathbf{V}_k &= \mathbf{V}_k \mathbf{T}_k + [\,0 \quad \cdots \quad 0 \quad \hat{\mathbf{v}}_{k+1} \quad \hat{\mathbf{v}}_{k+2} \quad \cdots \quad \hat{\mathbf{v}}_{k+m}\,], \\ \mathbf{A}^T \mathbf{W}_k &= \mathbf{W}_k \tilde{\mathbf{T}}_k + [\,0 \quad \cdots \quad 0 \quad \hat{\mathbf{w}}_{k+1} \quad \hat{\mathbf{w}}_{k+2} \quad \cdots \quad \hat{\mathbf{w}}_{k+p}\,]. \end{aligned} \tag{30}$$

The matrices

$$\mathbf{T}_k = [\,t_{ij}\,]_{i,j=1,2,\ldots,k} \quad \text{and} \quad \tilde{\mathbf{T}}_k = [\,\tilde{t}_{ij}\,]_{i,j=1,2,\ldots,k} \tag{31}$$

are banded matrices, where $\mathbf{T}_k$ has m subdiagonals and p superdiagonals, and $\tilde{\mathbf{T}}_k$ has p subdiagonals and m superdiagonals. The matrices (31) are—up to a diagonal scaling—the transpose of each other:

$$\tilde{\mathbf{T}}_k^T = \mathbf{D}_k \mathbf{T}_k \mathbf{D}_k^{-1}, \quad \text{where} \quad \mathbf{D}_k = \operatorname{diag}(\delta_1, \delta_2, \ldots, \delta_k). \tag{32}$$

The vectors

$$\hat{\mathbf{v}}_{k+1}, \hat{\mathbf{v}}_{k+2}, \cdots, \hat{\mathbf{v}}_{k+m} \quad \text{and} \quad \hat{\mathbf{w}}_{k+1}, \hat{\mathbf{w}}_{k+2}, \cdots, \hat{\mathbf{w}}_{k+p}$$

in (30) are auxiliary vectors. In the kth step of our block Lanczos Algorithm 1, we compute the new vectors $\mathbf{v}_k$ and $\mathbf{w}_k$ and a new pair of auxiliary vectors $\hat{\mathbf{v}}_{k+m}$ and $\hat{\mathbf{w}}_{k+m}$, and we update the remaining auxiliary vectors

$$\hat{\mathbf{v}}_{k+1}, \hat{\mathbf{v}}_{k+2}, \cdots, \hat{\mathbf{v}}_{k+m-1} \quad \text{and} \quad \hat{\mathbf{w}}_{k+1}, \hat{\mathbf{w}}_{k+2}, \cdots, \hat{\mathbf{w}}_{k+m-1}.$$

Algorithm 1 (A block Lanczos algorithm [8])

0) Set

$$\begin{aligned} \hat{\mathbf{v}}_i &= \mathbf{r}_i, \quad i = 1, 2, \ldots, m, \\ \hat{\mathbf{w}}_i &= \mathbf{l}_i, \quad i = 1, 2, \ldots, p. \end{aligned}$$

For $k = 1, 2, \ldots, k_{\max}$ do:

1) Compute $t_{k,k-m} = \|\hat{\mathbf{v}}_k\|_2$ and $\tilde{t}_{k,k-p} = \|\hat{\mathbf{w}}_k\|_2$.
If $t_{k,k-m} = 0$ or $\tilde{t}_{k,k-p} = 0$, then stop.

2) (*Computation of* $\mathbf{v}_k$ *and* $\mathbf{w}_k$.) Set

$$\mathbf{v}_k = \frac{\hat{\mathbf{v}}_k}{t_{k,k-m}}, \quad \mathbf{w}_k = \frac{\hat{\mathbf{w}}_k}{\tilde{t}_{k,k-p}}, \quad \delta_k = \mathbf{w}_k^T \mathbf{v}_k.$$

If $k \le m$, set $\rho_{kk} = t_{k,k-m}$.
If $k \le p$, set $\eta_{kk} = \tilde{t}_{k,k-p}$.

3) (*Computation of* $\hat{\mathbf{v}}_{k+m}$ *and* $\hat{\mathbf{w}}_{k+p}$.)
Set $i_m = \max\{1, k-m\}$ and $i_p = \max\{1, k-p\}$.
Set $\mathbf{v} = \mathbf{A}\mathbf{v}_k$ and $\mathbf{w} = \mathbf{A}^T\mathbf{w}_k$.
For $i = i_p, i_p + 1, \ldots, k$, set

$$t_{ik} = \frac{\mathbf{w}_i^T \mathbf{v}}{\delta_i} \quad \text{and} \quad \mathbf{v} = \mathbf{v} - \mathbf{v}_i t_{ik}.$$

For $i = i_m, i_m + 1, \ldots, k$, set

$$\tilde{t}_{ik} = \frac{\mathbf{v}_i^T \mathbf{w}}{\delta_i} \quad \text{and} \quad \mathbf{w} = \mathbf{w} - \mathbf{w}_i \tilde{t}_{ik}.$$

Set $\hat{\mathbf{v}}_{k+m} = \mathbf{v}$ and $\hat{\mathbf{w}}_{k+p} = \mathbf{w}$.

4) (*Update of* $\hat{\mathbf{v}}_{k+i}$, $1 \le i < m$.) For $i = 1, 2, \ldots, m-1$ do:

If $k + i \le m$, set

$$\rho_{k,k+i} = \frac{\mathbf{w}_k^T \hat{\mathbf{v}}_{k+i}}{\delta_k} \quad \text{and} \quad \hat{\mathbf{v}}_{k+i} = \hat{\mathbf{v}}_{k+i} - \mathbf{v}_k \rho_{k,k+i}.$$

If $k + i > m$, set

$$t_{k,k-m+i} = \frac{\tilde{t}_{k-m+i,k}\,\delta_{k-m+i}}{\delta_k} \quad \text{and} \quad \hat{\mathbf{v}}_{k+i} = \hat{\mathbf{v}}_{k+i} - \mathbf{v}_k t_{k,k-m+i}.$$

5) (*Update of* $\hat{\mathbf{w}}_{k+i}$, $1 \le i < p$.) For $i = 1, 2, \ldots, p-1$ do:

If $k + i \le p$, set

$$\eta_{k,k+i} = \frac{\mathbf{v}_k^T \hat{\mathbf{w}}_{k+i}}{\delta_k} \quad \text{and} \quad \hat{\mathbf{w}}_{k+i} = \hat{\mathbf{w}}_{k+i} - \mathbf{w}_k \eta_{k,k+i}.$$

If $k + i > p$, set

$$\tilde{t}_{k,k-p+i} = \frac{t_{k-p+i,k}\,\delta_{k-p+i}}{\delta_k} \quad \text{and} \quad \hat{\mathbf{w}}_{k+i} = \hat{\mathbf{w}}_{k+i} - \mathbf{w}_k \tilde{t}_{k,k-p+i}.$$

We remark that, in Algorithm 1, a breakdown, triggered by division by 0, will occur if one encounters $\delta_n = 0$ in steps 3-5. Furthermore, division by a nonzero yet small number $\delta_n \approx 0$ may result in numerical instabilities. However, we stress that these problems can be remedied by using a so-called look-ahead variant of this algorithm; we refer the reader to [8] for details.

4 Matrix Padé Approximation from the Block Lanczos Algorithm

We have shown that the Laplace-domain transfer function of the linear subnetwork is described by equation (10), where in general $\mathbf{H}(s)$ is a $p \times m$ matrix-valued function. We now describe how the block Lanczos algorithm from the previous section can be used to generate a reduced-order approximation to $\mathbf{H}(s)$

The transfer function can be expanded in an infinite Taylor series

$$\mathbf{H}(s_0 + \sigma) = \mathbf{L}^T (\mathbf{I} - \mathbf{A}\sigma)^{-1} \mathbf{R} = \sum_{j=0}^{\infty} \mathbf{L}^T \mathbf{A}^j \mathbf{R}\, \sigma^j, \tag{33}$$

where the coefficients of σ^j, $\mathbf{M}_j = \mathbf{L}^T \mathbf{A}^j \mathbf{R}$, $\quad j = 0, 1, \ldots$, represent the moments of the circuit response. Using the first relation in (30) and the fact that $\mathbf{T}_k$ is banded, one can show that

$$\mathbf{A}^j \mathbf{R} = \mathbf{V}_k \mathbf{T}_k^j \begin{bmatrix} \rho \\ 0 \end{bmatrix}, \quad j = 0, 1, \ldots, q' - 1, \tag{34}$$

where $q' = \lfloor k/m \rfloor$. Similarly, from the second relation in (30), one can deduce that

$$\left(\mathbf{A}^T\right)^j \mathbf{L} = \mathbf{W}_k \tilde{\mathbf{T}}_k^j \begin{bmatrix} \eta \\ 0 \end{bmatrix}, \quad j = 0, 1, \ldots, q'' - 1, \tag{35}$$

where $q'' = \lfloor k/p \rfloor$. On the other hand, by (33), each moment $\mathbf{M}_j$ can be written as follows:

$$\mathbf{M}_j = \mathbf{L}^T \mathbf{A}^j \mathbf{R} = \left(\mathbf{L}^T \mathbf{A}^{j'}\right) \left(\mathbf{A}^{j''} \mathbf{R}\right), \tag{36}$$

where $j = j' + j''$. If $j \leq q' + q'' - 2$, we can find j' and j'' with $0 \leq j' \leq q' - 1$ and $0 \leq j'' \leq q'' - 1$, and from (34)–(36) it follows that

$$\mathbf{M}_j = [\,\eta^T \quad 0\,] \left(\tilde{\mathbf{T}}_k^T\right)^{j'} \mathbf{W}_k^T \mathbf{V}_k \mathbf{T}_k^{j''} \begin{bmatrix} \rho \\ 0 \end{bmatrix} \quad j = 0, 1, \ldots, q' + q'' - 2. \tag{37}$$

Recall that $\mathbf{W}_k^T \mathbf{V}_k = \mathbf{D}_k$ and note that $\tilde{\mathbf{T}}_k^T = \mathbf{D}_k \mathbf{T}_k \mathbf{D}_k^{-1}$ and, therefore, $\left(\tilde{\mathbf{T}}_k^T\right)^{j'} = \mathbf{D}_k \mathbf{T}_k^{j'} \mathbf{D}_k^{-1}$. From (37), we thus get

$$\mathbf{M}_j = [\,\eta^T \quad 0\,]\, \mathbf{D}_k \mathbf{T}_k^j \begin{bmatrix} \rho \\ 0 \end{bmatrix}, \quad j = 0, 1, \ldots, q' + q'' - 2. \tag{38}$$

Furthermore, it can be shown that the relation (38) also holds for $j = q' + q'' - 1$, and we now set

$$q = q' + q'' = \lfloor k/m \rfloor + \lfloor k/p \rfloor. \tag{39}$$

Thus, by (38), the expression

$$\begin{aligned} [\,\eta^T \quad 0\,]\,\mathbf{D}_k\,(\mathbf{I} - \sigma\mathbf{T}_k)^{-1} \begin{bmatrix} \rho \\ 0 \end{bmatrix} &= \sum_{j=0}^{\infty} [\,\eta^T \quad 0\,]\,\mathbf{D}_k \mathbf{T}_k^j \begin{bmatrix} \rho \\ 0 \end{bmatrix} \sigma^k \\ &= \sum_{j=0}^{q-1} \mathbf{M}_j\,\sigma^j + \mathcal{O}(\sigma^q) \end{aligned} \tag{40}$$

has the same first q matrix Taylor coefficients as $\mathbf{H}(s_0 + \sigma)$, and consequently,

$$\mathbf{H}_k(s_0 + \sigma) = [\,\eta^T \quad 0\,]\,\mathbf{D}_k\,(\mathbf{I} - \sigma\mathbf{T}_k)^{-1} \begin{bmatrix} \rho \\ 0 \end{bmatrix} \tag{41}$$

is just a matrix Padé approximant of $\mathbf{H}(s_0 + \sigma)$.

Next, we determine the time-domain reduced-order model of the multiple-input multiple-output linear subnetwork. We perform the same substitution $\mathbf{x} = \mathbf{V}_k\mathbf{d}$ and premultiplication with $\mathbf{W}_k$ in (18) as in the single-input single-output case, thus restricting $\mathbf{x}$ to a k-dimensional subspace. The resulting expression,

$$\begin{aligned} -\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k\frac{d}{dt}\mathbf{d} + \mathbf{W}_k^T\mathbf{V}_k\mathbf{d} + s_0\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k\mathbf{d} &= \mathbf{W}_k^T\mathbf{R}\mathbf{u}, \\ \mathbf{y} &= \mathbf{L}^T\mathbf{V}_k\mathbf{d}, \end{aligned} \tag{42}$$

is similar to (19), only $\mathbf{y}$ and $\mathbf{u}$ are now vectors instead of scalars.

We observe that, from properties of the block Lanczos algorithm, $\mathbf{W}_k^T\mathbf{A}\mathbf{V}_k = \mathbf{D}\mathbf{T}_k$, by the biorthogonality of the Lanczos vectors, $\mathbf{W}_k^T\mathbf{V}_k = \mathbf{D}$, and from (29), we have

$$\mathbf{R} = \mathbf{V}_k \begin{bmatrix} \rho \\ 0 \end{bmatrix} \quad \text{and} \quad \mathbf{L} = \mathbf{W}_k \begin{bmatrix} \eta \\ 0 \end{bmatrix}. \tag{43}$$

By inserting these relations into expression (42), we obtain the time-domain reduced-order model of the linear subnetwork

$$\begin{aligned} -\mathbf{T}_k\frac{d}{dt}\mathbf{d} + (\mathbf{I} + s_0\mathbf{T}_k)\mathbf{d} &= \mathbf{D}_k \begin{bmatrix} \rho \\ 0 \end{bmatrix} \mathbf{u}, \\ \mathbf{y} &= [\,\eta^T \quad 0\,]\,\mathbf{D}_k\mathbf{d}. \end{aligned} \tag{44}$$

The transfer function corresponding to equations (44) represents a matrix Padé approximant of the original linear subnetwork transfer function.

By using a procedure similar to the one described for the single-input single-output case, the equations (44) can be used to replace a potentially much larger number of equations (6) that describe the linear part of the circuit.

5 An Illustrative Example

As an example we discuss the modeling of a low-noise amplifier designed for a radio-frequency application and implemented in an advanced BiCMOS process. The netlist, extracted from the actual layout with all the parasitics included, consists of 51 MOSFET devices, 26 bipolar transistors, 35 resistors, 6 inductors, and 381 capacitors. The size of the linearized circuit matrices is 414.

The amplifier is a two-port and therefore can be considered to have two independent inputs and outputs. The two-port is fully characterized by a 2×2 matrix-valued transfer-function that in fact contains only three independent entries. The MPVL algorithm was employed to compute a reduced-order model of the transfer-function matrix, and MPVL converged after 48 iterations. This corresponds to 2×24 matched matrix moments. Figures 2-4 plot the magnitudes of gain, normalized input impedance, and normalized output impedance of the amplifier. The graphs produced by MPVL in 48 iterations are indistinguishable from the true frequency response of the circuit as predicted by complex small-signal analysis (not shown on the graphs) for all three circuit characteristics.

Figure 2 shows that the amplifier gain obtained from the 48 iterations of the MPVL algorithm is indistinguishable from the results of the basic PVL algorithm which required only 40 iterations. This phenomenon is predictable. The MPVL algorithm approximates simultaneously the gain, the input impedance, and the output impedance of the amplifier, while PVL only considers the gain. PVL matches 2×40 moments of the gain function. MPVL, however, matches a total of 4×48 moments of the two-port even if only 48 are directly related to the gain. The additional information obtained from the remainder of the moments is sufficient for the algorithm to converge in 48 instead of the 80 iterations that would otherwise be necessary to match the same number of gain moments. Figure 2 also shows the gain predicted after 40 iterations of the MPVL algorithm when it has obviously not converged yet.

Figure 3 shows that, surprisingly, the input impedance computed with MPVL converges faster than the one computed with PVL. After 30 iterations MPVL predicts the correct input impedance, while PVL has not converged yet. Recall, however, that at this point MPVL matches a total of 4×30 system moments.

The output impedance of the amplifier, shown in Figure 4, requires 34 PVL iterations. As in the gain case, MPVL required several more iterations to converge.

This example also shows the advantages of the MPVL algorithm over using PVL repeatedly for each entry of the transfer-function matrix. One run of the MPVL algorithm for 48 iterations produced equal and better results than three runs of PVL with 40, 30+, 34 iterations respectively. Similarly, the number of state variables of the reduced-order model is only 48 compared with more than $40 + 30 + 34$ that would have been required otherwise. This advantage is only expected to grow when subnetworks with more inputs and outputs are considered.

Additional, more comprehensive, examples will be included in an extended version of the paper.

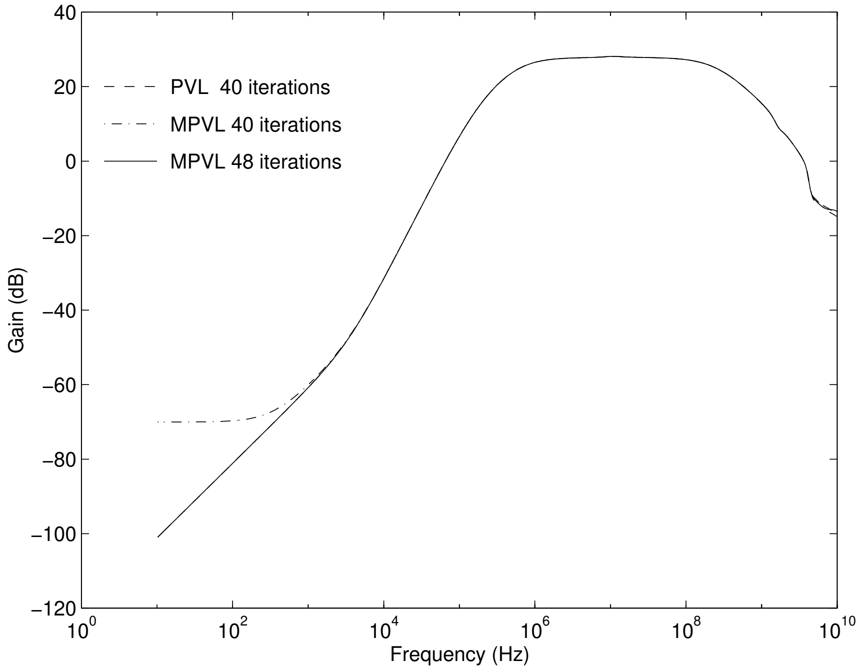

Figure 2: Low-noise amplifier gain

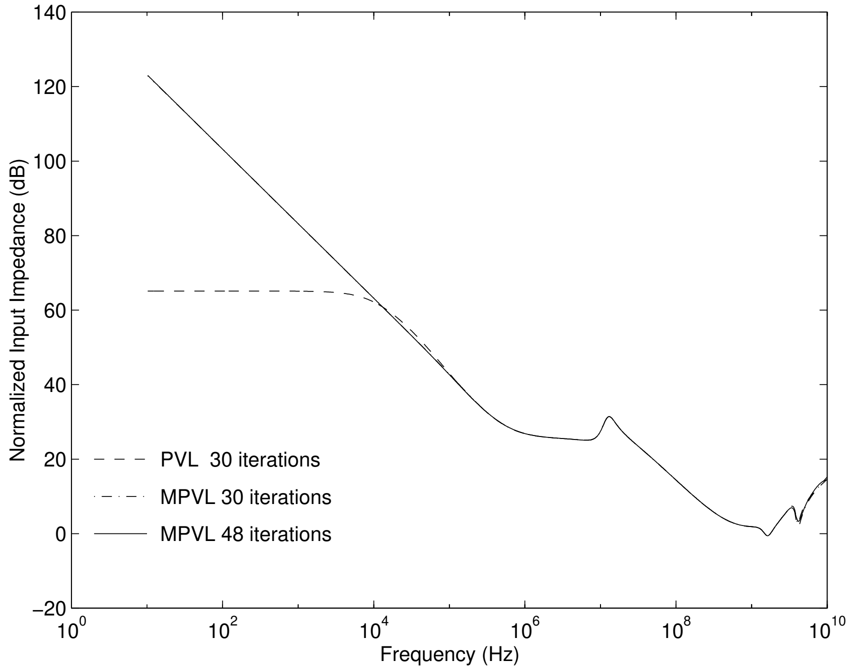

Figure 3: Low-noise amplifier input impedance

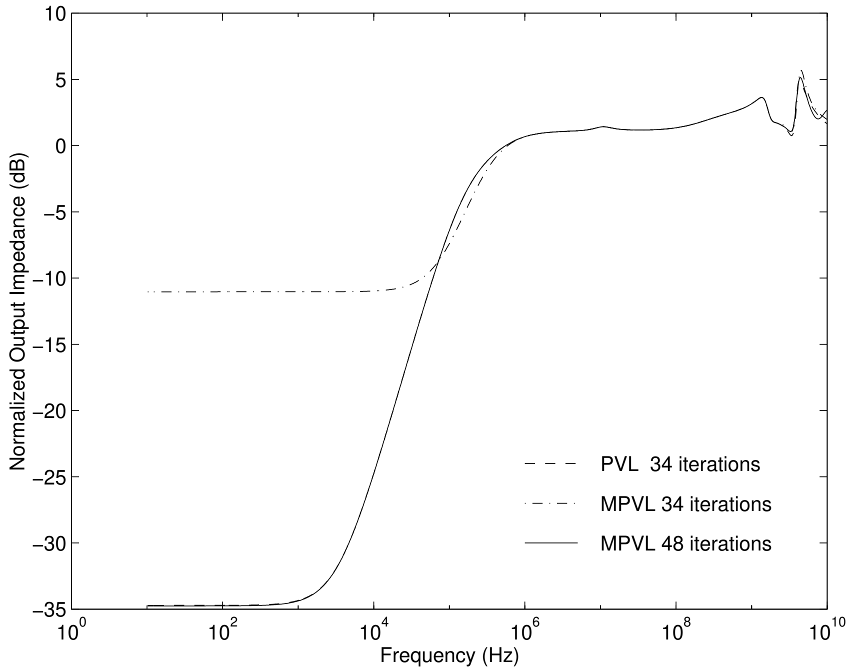

Figure 4: Low-noise amplifier output impedance

6 Conclusions

This paper introduces MPVL, a new algorithm for the computation of reduced-order approximate models of linear networks with multiple inputs and outputs. MPVL uses a block Lanczos process to compute a matrix Padé approximation of the linear network matrix-valued transfer function. The algorithm generalizes the recently published PVL algorithm that is handling one pair of inputs and outputs of the linear network at a time and computes a scalar Padé approximation of the corresponding transfer function. The reduced-order approximate model generated with MPVL can be naturally and easily used as replacement of a large linear subnetwork in a nonlinear circuit simulation, for substantial savings in total simulation time.

References

[1] V. Raghavan, J. E. Bracken, and R. A. Rohrer, "AWESpice: A general tool for the accurate and efficient simulation of interconnect problems," in *Proc. 29th ACM/IEEE Design Automation Conf.*, June 1992.

[2] 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.

[3] 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 Conf.*, May 1993.

[4] P. Feldmann and R. W. Freund, "Efficient linear circuit analysis by Padé approximation via the Lanczos process," in *Proc. EURO-DAC '94 with EURO-VHDL '94*, pp. 170–175, Sep. 1994.

[5] R. W. Freund. and P. Feldmann, "Efficient small-signal circuit analysis and sensitivity computations with the PVL algorithm," in *Tech. Dig. IEEE/ACM Int. Conf. Computer-Aided Design*, Nov. 1994.

[6] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*. New York, N.Y.: Van Nostrand Reinhold, 1983.

[7] 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.

[8] J. I. Aliaga, D. L. Boley, R. W. Freund, and V. Hernández, "A Lanczos-type algorithm for multiple starting vectors," Numerical Analysis Manuscript, AT&T Bell Laboratories, Murray Hill, NJ, 1994.