Efficient Small-Signal Circuit Analysis and Sensitivity Computations with the PVL Algorithm

R. W. Freund
AT&T Bell Laboratories
Murray Hill, NJ 07974-0636

P. Feldmann
AT&T Bell Laboratories
Murray Hill, NJ 07974-0636

Abstract

We describe the application of the PVL algorithm to the small-signal analysis of circuits, including sensitivity computations. The PVL algorithm is based on the efficient computation of the Padé approximation of the network transfer function via the Lanczos process. The numerical stability of the algorithm permits the accurate computation of the Padé approximation over any given frequency range. We extend the algorithm to compute sensitivities of network transfer functions, their poles, and zeros, with respect to arbitrary circuit parameters, with minimal additional computational cost, and we present numerical examples.

1 Introduction

The process of analyzing analog circuits with full accounting of parasitic elements, interconnect analysis at the board or chip level, and numerous other circuit simulation tasks often require the analysis of large linear networks. These networks can become extremely large, especially when circuits are automatically extracted from layout, or contain models of distributed elements, such as transmission lines, ground planes, antennas, and other three-dimensional structures. The use of time-domain differential-equation integration or complex phasor analysis, as implemented in SPICE-like simulators, would be inefficient or even prohibitive for such large problems.

Recently, in [1], we have introduced a new algorithm for the efficient analysis of large linear networks. This algorithm, called PVL (Padé Via Lanczos), computes in a numerically stable way the Padé approximation of a linear circuit response via the Lanczos process [2]. A key feature of the PVL algorithm is the fact that the accuracy of the approximation can be guaranteed for a specified frequency range. In addition, the efficiency of the algorithm is such that it allows computation of the circuit response over an entire frequency range with similar computational effort as classical phasor analysis would require for a single frequency point. Therefore, PVL can be used to replace complex phasor analysis altogether.

In Figure 1, we illustrate this point with an example taken from [1]. Here, we simulate a lumped-element equivalent circuit for a three-dimensional electromagnetic problem modeled via PEEC [3]. Using a single Padé approximant (of order 60), PVL accurately simulates the circuit response over a frequency range of 5 GHz. One would need to carry out a high-resolution complex phasor analysis involving a large number of frequency points to capture all the resonance frequencies. The response shown in Figure 1 was obtained with PVL at a cost that is smaller than phasor analysis for a single frequency point.

In addition to efficiency and accuracy, the PVL algorithm has other features that makes it useful in the analysis of large linear circuits. In this paper, we describe the application of the PVL algorithm to the small-signal analysis of circuits, including the computation of sensitivities of the frequency response or other measures related to it such as the location of its poles or zeros.

The paper is organized as follows. In Section 2, we
review the small-signal analysis of circuits. In Section 3, we sketch the PVL algorithm. In Section 4, we extend the PVL algorithm to compute sensitivities of the frequency response and related quantities. In Section 5, we present results of numerical experiments. In Section 6, we make some concluding remarks.

2 Small-Signal Circuit Equations

All of the established circuit-equation formulation methods, such as MNA, sparse tableau, etc. [4] lead to a system of differential equations of the form

$$f(z, t) + \frac{d}{dt} q(z, t) = 0.$$  \hspace{1cm} (1)

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

Most of the circuit simulations of interest can be performed starting from this formulation. The small-signal analysis of circuits is an important approximation technique that reduces the simulation of a nonlinear dynamic circuit to the analysis of a linear resistive circuit followed by the analysis of a linear dynamic circuit. We assume that all time-varying elements in the circuit are independent sources. This assumption allows us to simplify (1) to

$$f(z) + \frac{d}{dt} q(z) - e(t) = 0.$$  \hspace{1cm} (2)

Furthermore, we assume that the excitation term in (2), $e(t)$, is “small”, and thus all circuit variables $z$ will be “close” to equilibrium values, $z_0$, which represent the DC operating point of the circuit. The DC operating point can be computed by passivating all time-varying elements, i.e., setting $e(t) = 0$ and solving (2). With no time-varying elements, the second term in (2) vanishes, and the operating point $z_0$ is found as the solution of the system of nonlinear equations, $f(z_0) = 0$. By expanding the terms $f(z)$ and $q(z)$ in the circuit equations (2) into Taylor series about the operating point $z_0$, we obtain

$$f(z_0) + \left[ \frac{\partial f}{\partial z} \right]_{z_0} \delta z + \cdots$$

$$+ \left[ \frac{\partial q}{\partial z} \right]_{z_0} \delta z + \cdots - e(t) = 0.$$  \hspace{1cm} (3)

Since we assumed that all the deviations from the equilibrium values are “small”, all but the first-order terms can be neglected in (3). Using $f(z_0) = 0$ and $\frac{d}{dt} q(z_0) = 0$, and setting

$$G = \left. \frac{\partial f}{\partial z} \right|_{z_0} \quad \text{and} \quad C = \left. \frac{\partial q}{\partial z} \right|_{z_0},$$  \hspace{1cm} (4)

we obtain from (3) the linear dynamic system

$$G \delta z + C \frac{d}{dt} \delta z - e(t) = 0.$$  \hspace{1cm} (5)

that needs to be solved as part of the small-signal analysis. Since the system (5) is linear, the superposition law holds, and hence all sources of excitation can be treated separately. Therefore, without loss of generality, we can assume that there is only one source of excitation, $e(t) = b u(t)$. Then, the small-signal linear time-invariant dynamic system (5) is described by the following system of first-order differential equations:

$$C \frac{d}{dt} x = -G x + b u,$$  \hspace{1cm} (6)

$$y = 1^T x.$$  \hspace{1cm} (7)

Here, $x$ represents the small-signal circuit variables, $\delta z$, and $y$ is the output of interest.

Our goal is to determine the impulse response of the linear circuit with zero initial-conditions, which, in turn, can be used to compute the response to any excitation. We apply the Laplace transform to the system (6), assuming zero initial conditions. Then, from (6), we obtain

$$s C X = -G X + b U,$$  \hspace{1cm} (7)

$$Y = 1^T X,$$  \hspace{1cm} (7)

where $X$, $U$, and $Y$ denote the Laplace transform of $x$, $u$, and $y$, respectively. It follows from (7) that the Laplace-domain impulse response $H(s) = Y(s)/U(s)$ is given by

$$H(s) = 1^T (G + s_0 C)^{-1} b.$$  \hspace{1cm} (8)

The function $H(s)$ is called the frequency response or transfer function of the circuit. The frequency response $H(s)$ can be accurately and efficiently approximated by means of the PVL algorithm.

3 The PVL Algorithm

The PVL algorithm [1] uses the Lanczos process to compute a Padé approximation to the frequency response (8) in a numerically stable fashion. In this section, we review the PVL algorithm.

Let $s_0 \in \mathbb{C}$ be an arbitrary, but fixed expansion point such that the matrix $G + s_0 C$ is nonsingular.
Here, $G$ and $C$ are the matrices from the small-signal linear system (6). Our goal is to approximate the frequency response (8) in a region in the complex plane about the point $s_0$. Using the change of variables $s = s_0 + \sigma$ and setting

$$A = -(G + s_0 C)^{-1} C, \quad r = (G + s_0 C)^{-1} b,$$

we can rewrite (8) as follows:

$$H(s_0 + \sigma) = T^T (I - \sigma A)^{-1} r.$$  (10)

Here $I$ denotes the identity matrix.

### 3.1 Padé Approximation

By expanding $H(s_0 + \sigma)$ into a Taylor series about $\sigma = 0$, we obtain from (10) the representation

$$H(s_0 + \sigma) = \sum_{k=0}^{\infty} m_k \sigma^k,$$  (11)

where

$$m_k = T^T A^k r, \quad k = 0, 1, \ldots,$$  (12)

are the so-called moments of the frequency response.

Let $q \geq 1$ be an arbitrary integer. A rational function of the form

$$H_q(s_0 + \sigma) = \frac{b_0 + b_1 \sigma + \cdots + b_{q-1} \sigma^{q-1}}{1 + a_1 \sigma + a_2 \sigma^2 + \cdots + a_q \sigma^q},$$  (13)

with coefficients $a_1, a_2, \ldots, a_q, b_0, b_1, \ldots, b_{q-1} \in \mathbb{C}$, is called a $q$th Padé approximant (or Padé approximant of order $q$) to the frequency response $H(s_0 + \sigma)$ if the Taylor series of $H$ and $H_q$ about $\sigma = 0$ agree in at least the first $2q$ terms, i.e.,

$$H_q(s_0 + \sigma) = H(s_0 + \sigma) + O(\sigma^{2q+1})$$  (14)

The condition (14) just represents $2q$ equations for the $2q$ free parameters $a_1, a_2, \ldots, a_q, b_0, b_1, \ldots, b_{q-1}$ in (13), which suggests that a $q$th Padé approximant $H_q$ always exists. Indeed, it can be shown that—except for certain degenerate cases—the function $H_q$ exists and is unique; see, e.g., [5].

It turns out that Padé approximation is a very powerful tool for the analysis of large linear circuits. This was first demonstrated by Pillage and Rohrer [6] with their asymptotic waveform evaluation (AWE) approach that is based on the computation of the Padé approximant $H_q$; for a detailed description of AWE, we refer the reader to [7] and the references given there. In AWE, the Padé approximant $H_q$ is directly generated from the moments (12). Unfortunately, this procedure is inherently numerically unstable, as we illustrated in [1]. In fact, this is the reason why, in practice, AWE can be used only for fairly moderate values of $q$. In [1], we proposed a different algorithm, called PVL, for computing $H_q$. The PVL algorithm bypasses the moments (12), and instead, PVL exploits the connection [8] between Padé approximation and the Lanczos process [2] to generate the Padé approximant $H_q$ in a numerically stable manner.

### 3.2 The Lanczos Process

The classical Lanczos algorithm (applied to the matrix $A$ and the vectors $I, r$ from (10)) can be stated as follows.

**Algorithm 1 (Lanczos algorithm [2])**

1. Set $v = r, \quad w = 1, \quad v_0 = w_0 = 0, \quad \delta_0 = 1$.
2. For $n = 1, 2, \ldots, q$ do:
   1. Compute $r_n = \|v\|_2$ and $\eta_n = \|w\|_2$.
      If $r_n = 0$ or $\eta_n = 0$, then stop.
   2. Set
      $$v_n = \frac{v}{r_n}, \quad w_n = \frac{w}{\eta_n},$$
      $$\delta_n = w_n^T v_n, \quad \alpha_n = \frac{w_n^T A v_n}{\delta_n}, \quad \beta_n = \eta_n, \quad \gamma_n = \rho_n \frac{\delta_n}{\delta_{n-1}}.$$  (15)
      $$v = A v_n - v_n \alpha_n - v_{n-1} \beta_n,$$
      $$w = A w_n - w_n \alpha_n - w_{n-1} \gamma_n.$$

We remark that in Algorithm 1 a breakdown will occur if one encounters $\delta_n = 0$ or even $\delta_n \approx 0$ in (15). Therefore, our implementation of the PVL algorithm employs the look-ahead Lanczos algorithm described in [9] that remedies the breakdown problem.

The key property of Algorithm 1 is that it produces a very useful small-dimensional approximation, called $T_q$ in the sequel, to the (usually large-dimensional) matrix $A$. More precisely, $T_q$ is the $q \times q$ tridiagonal matrix

$$T_q = \begin{bmatrix}
\alpha_1 & \beta_2 & 0 & \cdots & 0 \\
\rho_2 & \alpha_2 & \beta_3 & \cdots & 0 \\
0 & \rho_3 & \cdots & \beta_4 & 0 \\
\vdots & \vdots & \ddots & \ddots & \ddots \\
0 & 0 & \cdots & \beta_q & \alpha_q
\end{bmatrix},$$  (16)

where $\rho_n, \alpha_n, \beta_n$ are the quantities generated by Algorithm 1. The matrix $T_q$ is—in some sense—the
best $q \times q$ approximation to $A$. We stress that this approximation is very good even if $q$ is much smaller than the order $N$ of the $N \times N$ matrix $A$. In fact, the Lanczos algorithm is mostly applied to very large matrices, and typically, $q \ll N$.

It turns out that the Lanczos tridiagonal matrix $T_q$ contains all the information that is needed to generate the $q$th Padé approximant $H_q$. The rational function $H_q$ is given by

$$H_q(s_0 + \sigma) = I^T r \cdot e_1^T (I - \sigma T_q)^{-1} e_1, \quad (17)$$

where $e_1 = [1 \ 0 \ \cdots \ 0]^T \in \mathbb{R}^q$ is the first unit vector in $\mathbb{R}^q$. Furthermore, the poles of $H_q$, i.e., the zeros of the denominator polynomial in $(13)$, are just the inverse eigenvalues of $T_q$. Indeed, by rewriting the expression $(17)$ of $H_q$ in terms of the eigendecomposition $T_q = S_q A_q S_q^{-1}$, we get

$$H_q(s_0 + \sigma) = I^T r \cdot e_1^T S_q (I - \sigma A_q)^{-1} S_q^{-1} e_1$$

$$= \sum_{j=1}^{q} \frac{I^T r \cdot \mu_j \nu_j}{1 - \sigma \lambda_j}. \quad (18)$$

Here, $A_q = \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_q)$ contains the eigenvalues of $T_q$, and $\mu_j$ and $\nu_j$ are the components of the vectors $\mu = S_q^T e_1$ and $\nu = S_q^{-1} e_1$. Note that, from $(18)$, we immediately obtain the so-called pole/residue representation of the Padé approximant:

$$H_q(s_0 + \sigma) = k_{\infty} + \sum_{j \neq \infty}^{q} \frac{-I^T r \cdot \mu_j \nu_j / \lambda_j}{\sigma - 1 / \lambda_j}. \quad (19)$$

The term $k_{\infty}$ in $(19)$ may result if one of the eigenvalues of $T_q$ is zero.

Finally, we remark that the zeros of the reduced-order model $H_q$, i.e., the zeros of the numerator polynomial in $(13)$, can also be computed easily from the Lanczos matrix $T_q$. In fact, it can be shown that

$$H_q(s_0 + \sigma) = I^T r \cdot \det \left( I - \sigma \tilde{T}_q \right) / \det \left( I - \sigma T_q \right), \quad (20)$$

where $\tilde{T}_q$ is the $(q - 1) \times (q - 1)$ matrix obtained from $T_q$ by deleting the first row and column in $(16)$. By $(20)$, the zeros of $H_q$ are just the inverses of the eigenvalues of $\tilde{T}_q$.

3.3 A Sketch of the PVL Algorithm

The computational procedure based on the Padé-Lanczos connection is the PVL algorithm. It can be sketched as follows.

**Algorithm 2** (Sketch of the PVL algorithm)

1. Run $q$ steps of the Lanczos process (Algorithm 1) to obtain the tridiagonal matrix $T_q$.
2. Compute an eigendecomposition

$$T_q = S_q \text{diag}(\lambda_1, \lambda_2, \ldots, \lambda_q) S_q^{-1} \quad (21)$$

of $T_q$, and set $\mu = S_q^T e_1$ and $\nu = S_q^{-1} e_1$.
3. Compute the poles and residues of $H_q$ by setting

$$p_j = \frac{1}{\lambda_j}, \quad k_j = \frac{I^T r \cdot \mu_j \nu_j}{\lambda_j} \quad (22)$$

for all $j = 1, 2, \ldots, q$ with $\lambda_j \neq 0$, and

$$k_{\infty} = \sum_{j \neq \infty}^{q} I^T r \cdot \mu_j \nu_j.$$

We remark that the PVL algorithm and AWE require roughly the same amount of computational work. As in AWE, the dominating cost is the computation of the LU factorization

$$G + s_0 C = LU, \quad (23)$$

which needs to be computed only once. Based on $(23)$, the vectors $A v_n$ and $A^T w_n$ required in step 2) of Algorithm 1 are obtained as follows. First, using forward-backward substitution, we solve

$$L U z = -C v_n \quad \text{and} \quad U^T L^T y = -w_n \quad (24)$$

for $z$ and $y$, and then, we set

$$A v_n = z \quad \text{and} \quad A^T w_n = C^T y. \quad (25)$$

Therefore, the PVL algorithm involves $2q$ forward-backward substitutions to generate the $q$th Padé approximant, which is the same as in AWE.

4 Computing Sensitivities with PVL

In [10], the AWE algorithm was extended to compute sensitivities of poles and zeros of the frequency response. In this section, we show how the PVL algorithm can be used to generate in a numerically stable manner sensitivities of the frequency response and related quantities. We note that the advantages of PVL over AWE are preserved in sensitivity computations, and our method is not restricted to low-order Padé approximants.

Our goal is the calculation of sensitivities of circuit quantities with respect to some parameter $p \in \mathbb{R}$. In the following, we always use $'$ to denote derivatives with respect to $p$. Moreover, we assume that the
matrices \( C, G \) and the vectors \( b, l \) in (6) are differentiable with respect to \( p \). This guarantees that the quantities \( A, I, r \), which define the frequency response \( H(s_0 + \sigma) \) in (10), are also differentiable, and we denote by
\[
A' = \frac{\partial A}{\partial p}, \quad I' = \frac{\partial I}{\partial p}, \quad \text{and} \quad r' = \frac{\partial r}{\partial p}
\]
their derivatives.

### 4.1 An Extended Lanczos Algorithm

Recall from (17) that the PVL algorithm computes the Padé approximation \( H_q(s_0 + \sigma) \) to the frequency response \( H(s_0 + \sigma) \) via the tridiagonal matrix \( T_q \) given in (16). It order to obtain sensitivities related to \( H_q \), we therefore need the derivative, \( T'_q \), of the matrix \( T_q \). First, we note that—since \( A, I, r \) are differentiable—all quantities generated by the Lanczos Algorithm 1 are also differentiable, and thus, by (16), we have
\[
T'_q = \begin{bmatrix}
\alpha'_0 & \beta'_0 & 0 & \cdots & 0 \\
\beta'_0 & \alpha'_0 & \beta'_1 & \cdots & 0 \\
0 & \beta'_1 & \alpha'_0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
0 & \cdots & 0 & \beta'_q & \alpha'_0
\end{bmatrix}
\]

Moreover, the entries \( \rho'_n, \alpha'_n, \beta'_n \) of \( T'_q \) can be computed by means of an “extended” version of the Lanczos algorithm that is obtained by differentiating all equations in Algorithm 1 and adding the differentiated relations to Algorithm 1. For example, by differentiating the first relation in (15), we obtain the update formula
\[
\delta'_n = w'_n v'_n + (w'_n)^T v_n,
\]
for \( \delta'_n \). Similarly, one derives formulas for computing the derivatives of all other Lanczos quantities. The resulting extended Lanczos process can be summarized as follows.

**Algorithm 3** (Extended Lanczos algorithm, including sensitivity computations)

1. Set \( v = r, w = I, v' = r', w' = I' \), \( v_0 = w_0 = v_0' = w_0' = 0, \delta_0 = 1, \text{and} \delta'_0 = 0 \).

For \( n = 1, 2, \ldots, q \) do:

1. Compute \( \rho_n = \|v\|_2 \) and \( \eta_n = \|w\|_2 \).

2. Set
\[
\begin{align*}
v_n &= \frac{v}{\rho_n}, \quad w_n &= \frac{w}{\eta_n}, \\
\rho'_n &= v_n^T v', \quad \eta'_n = w_n^T w', \\
v'_n &= \frac{v' - v_n \rho_n}{\rho_n}, \quad w'_n = \frac{w' - w_n \eta_n}{\eta_n}, \\
\delta_n &= w_n^T v_n, \quad \delta'_n = w_n^T v'_n + v_n^T w'_n, \\
\alpha_n &= \frac{w_n^T A v_n}{\delta_n}, \quad \beta_n = \frac{\eta_n \delta_n}{\delta_n - 1}, \quad \gamma_n = \frac{\delta_n}{\delta_n - 1}, \quad \gamma'_n = \frac{\delta'_n}{\delta_n - 1}, \\
v &= A v_n - v_n \alpha_n - v_n - \beta_n, \\
w &= A^T w_n - w_n \alpha_n - w_n - \gamma_n.
\end{align*}
\]

3. Compute the vectors
\[
\begin{align*}
\beta'_n &= \frac{\alpha'_n}{\delta_n - 1} + \beta_n \left( \frac{\beta_n}{\delta_n} - \frac{\beta'_n}{\delta_n - 1} \right), \\
\gamma'_n &= \frac{\alpha'_n}{\delta_n - 1} + \gamma_n \left( \frac{\gamma_n}{\delta_n} - \frac{\gamma'_n}{\delta_n - 1} \right), \\
t_1 &= (A v_n)' - v_n \alpha_n, \quad t_2 = (A^T w_n)' - w_n \alpha_n, \\
\alpha'_n &= \frac{w_n^T t_1 + (w_n')^T (A v_n - v_n \alpha_n)}{\delta_n}, \\
v' &= t_1 - v_n \alpha_n - v_n - \beta_n - v_n' \beta_n, \\
w' &= t_2 - w_n \alpha_n - w_n - \gamma_n - w_n' \gamma_n.
\end{align*}
\]

4.2 Sensitivities of \( H_q \)

For sensitivity computation in PVL, we run \( q \) steps of the extended Lanczos Algorithm 3, instead of Algorithm 1. As a result, we obtain both the Lanczos matrix \( T_q \) and its derivative \( T'_q \). Using these two matrices, we can compute all sensitivities of interest.

First, consider the sensitivity of the approximate frequency response \( H_q \). By differentiating the representation (17) of \( H_q \), we obtain
\[
H'_q(s_0 + \sigma) = \frac{\partial}{\partial p} H_q(s_0 + \sigma) = e_1^T (I - \sigma T_q)^{-1} \left[ \psi e_1 + \sigma P r \cdot T'_q (I - \sigma T_q)^{-1} e_1 \right],
\]
where \( \psi = I^T r' + r^T y \). Recalling Algorithm 2, in PVL, we also compute the eigen decomposition (21) of \( T_q \) and the vectors \( \mu \) and \( \nu \). Using (21), we obtain from (29) the following computationally more economic formula:
\[
H'_q(s_0 + \sigma) = \sum_{j=1}^q \frac{\mu_j}{1 - \sigma \lambda_j} \left[ \psi y_j + \sigma I^T r \sum_{k=1}^q \frac{P_k m_{jk}}{1 - \sigma \lambda_k} \right],
\]
where the coefficients \( m_{jk} \) are the entries of the matrix
\[
M = [m_{jk}]_{j,k=1,2,\ldots,q} = S_q^{-1} T_q S_q.
\]
Recall from (22) that the poles $p_j$ of $H_q$ are given as $p_j = 1/\lambda_j$, where $\lambda_j \neq 0, j = 1, 2, \ldots, q$, are the nonzero eigenvalues of $T_q$. Therefore, the sensitivity $p'_j$ of the $j$th pole $p_j$ is given by

$$p'_j = -\chi'_j/\lambda_j^2.$$  

(31)

Here, the sensitivity $\chi'_j$ of the $j$th eigenvalue $\lambda_j$ of $T_q$ is obtained using standard perturbation theory for eigenvalues; see, e.g., [11, 12]. If $\lambda_j$ is a simple eigenvalue of $T_q$, then we have $\chi'_j = y_j^T T'_q s_j$, where $s_j$ is the $j$th column of the eigenvector matrix $S_q$ in (21) and $y_j^T$ is the $j$th row of $S_q^{-1}$. The case of multiple eigenvalues can be treated similarly, see [12].

Finally, we note that sensitivities of the zeros of $H_q$ can be computed in an analogous manner, by exploiting the fact that, by (20), the zeros of $H_q$ are just the inverses of the eigenvalues of $T_q$.

### 4.3 Derivatives of Circuit Quantities

Algorithm 3 requires the computation of the derivatives of $A$, $I$, and $r$. Recall that $A$ and $r$ are defined in (9). Moreover, $b$ and $l$ are vectors that depend on the nature of excitation and of the output of interest, respectively. In view of (9), we need to compute the matrices $G'$, $C'$, and the vectors $b'$ and $l'$. Again, we denote by $p$ the parameter with respect to which we want to compute sensitivities. From (4), we get

$$G' = \frac{\partial}{\partial p} \left[ \frac{\partial f}{\partial z}(z_0(p), p) \right] = \frac{\partial^2 f}{\partial z^2} \frac{\partial z_0}{\partial p} + \frac{\partial^2 f}{\partial p \partial z}$$  

(32)

and

$$C' = \frac{\partial^2 q}{\partial z^2} \frac{\partial z_0}{\partial p} + \frac{\partial^2 q}{\partial p \partial z}.$$  

(33)

The vector $\partial z_0/\partial p$ represents the sensitivity of the operating point with respect to the parameter $p$ and can be obtained by solving the linear system

$$\frac{\partial f}{\partial z} \frac{\partial z_0}{\partial p} + \frac{\partial f}{\partial p} = 0,$$  

(34)

which results from the differentiation of the operating-point equation $f(z_0(p), p) = 0$.

In general, the computation of sensitivities requires the knowledge of second-order derivatives of the device modeling equations. Since in most cases second-order derivatives are not provided by models, they must be computed either through automatic differentiation [13, 14], or by finite differences.

In important special cases, second-order derivative information is easier to obtain. For example, when the parameter affects only the reactive part of the circuit equations the operating point is not dependent on the parameter $p$, and therefore the first term in (34) vanishes. Moreover, when the circuit is linear, then $G'$ and $C'$ can be easily obtained using stencils [4].

The vectors $b$ and $l$ depend in most cases solely on the topology of the circuit, and therefore their derivatives with respect to circuit parameters are zero. However, sometimes, the parameter $p$ can affect the excitation of a circuit and cause $b'$ to be nonzero, or an output function may result in a nonzero $l'$.

Finally, we show how to obtain the vectors $(Av_n)'$ and $(A^T w_n)'$ in step 2 of Algorithm 3. Recall that $u = Av_n$ is obtained by solving the linear system

$$(G + s_b C)u = -Cv_n.$$  

(35)

By differentiating (35), we see that $u' = (Av_n)'$ can be computed by solving the linear system

$$(G + s_b C)u' = -C'v_n - C'v_n - (G' + s_b C')u.$$  

(36)

A similar procedure is used to obtain $(A^T w_n)'$. Note that the coefficient matrix, $G + s_b C$, of systems (35) and (36) is the same, and therefore, no additional LU-factorization is required for sensitivity computations.

### 5 Examples

In this section, we present numerical results for two typical examples.

Our first example is the 741 operational amplifier, which was also analyzed in [10]. The magnitude and phase response of the operational amplifier obtained with PVL in 18 iterations are shown in Figures 2 and 3, respectively. They match with 5 digits of accuracy the response predicted by complex phasor analysis. Table 1 shows the poles that had converged after 18 PVL iterations, together with their sensitivities with respect to the compensation capacitor (30 pF).
Table 1: Poles and Sensitivities for the 741 Op-Amp

<table>
<thead>
<tr>
<th>Poles</th>
<th>Pole Sensitivities</th>
</tr>
</thead>
<tbody>
<tr>
<td>-8.6894e+01</td>
<td>8.66e+01</td>
</tr>
<tr>
<td>-6.4652e+05</td>
<td>1.433e+03</td>
</tr>
<tr>
<td>-2.0599e+07 ± 6.074e+06</td>
<td>-1.32e+05 ± 1.061e+05</td>
</tr>
<tr>
<td>-3.5081e+07 ± 2.763e+07</td>
<td>6.7e+04 ± 1.558e+05</td>
</tr>
<tr>
<td>-7.7302e+07</td>
<td>1.323e+05</td>
</tr>
<tr>
<td>-8.1750e+07</td>
<td>9.888e+03</td>
</tr>
<tr>
<td>-9.5812e+07 ± 7.595e+07</td>
<td>6.907e+05 ± 2.67e+07</td>
</tr>
</tbody>
</table>

Figures 4 and 5 show the normalized sensitivities of the magnitude and phase of the amplifier frequency response computed directly with the PVL algorithm. The responses match very well results obtained by perturbation over the entire frequency range. In contrast, the results published in [10] show discrepancies at higher frequencies. We remark that in this example the sensitivity parameter affects only the reactive part of the circuit equations, and therefore computation of second-order derivatives was not required.

Our second example is a simulation 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 circuit must operate over a range of temperatures and contains a sophisticated temperature stabilization scheme. Therefore, the sensitivity of the frequency response with respect to temperature is of considerable interest.

We study the frequency-domain characteristics of the amplifier gain—with and without temperature stabilization—and their sensitivities with respect to temperature. The two variants of the circuit were linearized around their DC operating points, and the resulting small-signal equivalent networks were analyzed, using the extended PVL algorithm.

A Padé approximant of order $q = 60$ models correctly the behavior of the amplifier from 10Hz to 10GHz. Figure 6 shows the magnitude of the circuit frequency-response, as produced by the algorithm, with the temperature stabilization circuitry turned successively on and off. Separately, we compared the Padé approximations to responses computed by complex phasor analysis and they were indistinguishable.

Figure 7 shows the normalized sensitivities (in dB) of the frequency-response magnitudes for the two circuit variants. The plots confirm that the temperature stabilization performs as expected, significantly reducing the sensitivity. Finally, Figure 8 shows only the normalized sensitivity of the temperature stabilized circuit at a more appropriate scale. Note that in this case the sensitivity must also account for the change in the operating point due to temperature. Since, in our present implementation, we cannot compute second-order derivatives, finite differences were used.
Figure 6: Low-Noise Amplifier magnitude responses

Figure 7: Low-Noise Amplifier magnitude sensitivities

6 Conclusions

This paper argues that, due to its robustness and efficiency, PVL should become the algorithm of choice for the small-signal analysis of electrical circuits. The advantages of the PVL algorithm are not limited to superior accuracy and efficiency, but include new capabilities difficult to implement with existing methods. In this paper, we extended the PVL algorithm to compute the sensitivity of network transfer functions and of its poles and zeros.

References
