An Efficient Transient Analysis Algorithm for Mildly Nonlinear Circuits

Fei Yuan, *Member, IEEE* and Ajoy Opal, *Member, IEEE*

***Abstract*—This paper presents a new and efficient transient analysis method for mildly nonlinear circuits. The method is based on Volterra series representation of nonlinear circuits. It characterizes nonlinear circuits using a set of linear circuits called Volterra circuits. The input of the first-order Volterra circuit is identical to that of the nonlinear circuit, whereas that of higher order Volterra circuits is obtained from the response of lower order Volterra circuits. Fourier series interpolation is employed to approximate the input of higher order Volterra circuits. These circuits are analyzed using the sampled-data simulation of linear circuits for computational efficiency and the response of nonlinear circuits is obtained at equally spaced intervals of time. The accuracy of the method is controlled by the order of Volterra and interpolating Fourier series. Various sources contributing to the error are analyzed. The method has been implemented in a computer program. Numerical results on example circuits demonstrate that the accuracy of the method is comparable to that of linear multistep predictor-corrector algorithms, but with greatly improved speed.**

***Index Terms*—Interpolating Fourier series, nonlinear circuits, transient analysis, Volterra series.**

I. Introduction

TRANSIENT, or time-domain, analysis is the most common analysis performed on nonlinear circuits. It is, however, also the most computationally intensive analysis. Among many time-domain analysis methods, linear multistep (LMS) formulas that are based on backward difference formulas (BDF) are the most robust and widely used algorithms for nonlinear circuits, especially stiff circuits [1], [2]. To achieve better computational efficiency and accuracy, explicit and implicit LMS formulas are usually integrated to form the so-called predictor-corrector (PC) algorithms in which an explicit LMS formula serves as the predictor and an implicit LMS formula serves as the corrector. PC algorithms are universal and effective in handling both mildly and harsh nonlinear circuits and are the main simulation engines of many commercial computer-aided design (CAD) tools [3]–[5]. Because of the need for Newton–Raphson iterations in every step of integration, these algorithms are computationally expensive. Many circuits encountered in telecommunication systems have a fixed dc operating point and the signals to be processed by these circuits are the ac inputs. When the amplitude of these inputs is small, the nonlinearities in these circuits can be characterized adequately using the truncated Taylor series expansion of these nonlinear characteristics at their dc operating point [6]. To analyze these circuits effectively, methods that minimize the cost of computation by taking the advantages of the mildly nonlinear characteristics of these circuits, and at the same time possess good computational accuracy, are highly desirable.

Manuscript received August 31, 2000; revised January 14, 2002. This paper was recommended by Associate Editor R. Saleh.

F. Yuan is with the Department of Electrical and Computer Engineering, Ryerson University, Toronto, ON M5B 2K3, Canada (e-mail: fyuan@ee.ryerson.ca).

A. Opal is with the Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, ON N2L 3G1, Canada (e-mail: aopal@vlsi.uwaterloo.ca).

Publisher Item Identifier S 0278-0070(02)04696-1.

In this paper, we give a new and efficient time-domain analysis method for mildly nonlinear circuits. The method extends the sampled data analysis technique for linear circuits given in [7] to nonlinear circuits by employing a Volterra functional series and interpolating Fourier series. It computes the response of nonlinear circuits at equally spaced intervals of time. We show that the accuracy of the method is comparable to that of PC algorithms but with greatly improved speed. The paper is organized as follows: In Section II, a brief review of the sampled-data simulation of linear circuits is given. In Section III, the Volterra circuit of nonlinear circuits is derived and an efficient sampled-data simulation technique for nonlinear circuits is developed. The method is based on interpolating Fourier series. We show that the use of simulation window avoids repetitive calculation of transition matrix and zero-state vector needed in analysis and significantly speeds up simulation. An in-depth examination of various factors affecting the speed and accuracy of the method is given in Section IV. In Section V, several nonlinear circuits are analyzed and the results are compared with those from PC analysis.

II. Sampled-Data Simulation of Linear Circuits

We begin our development with a brief review of the sampled-data analysis technique for linear circuits. Consider a linear circuit with an input $w(t)$. The circuit in the time domain is depicted by

$$\mathbf{G}\mathbf{v}(t) + \mathbf{C}\,\frac{d\mathbf{v}(t)}{dt} = \mathbf{g}w(t), \qquad \mathbf{v}(t)|_{t=0^-} = \mathbf{v}(0^-) \tag{1}$$

where $\mathbf{v}(t)$ is the network variable vector, $\mathbf{G}$ and $\mathbf{C}$ are the conductance and capacitance matrices, respectively, and $\mathbf{g}$ is a constant vector specifying the nodes to which $w(t)$ is connected. Using Laplace transform and its inverse, we obtain the time-domain response

$$\mathbf{v}(t) = \mathbf{M}(t)\mathbf{v}(0^-) + \mathbf{P}(t) \tag{2}$$

where

$$\begin{aligned}\mathbf{M}(t) &= \mathbf{N}(t)\mathbf{C}\\ \mathbf{N}(t) &= \mathcal{L}^{-1}\left[\mathbf{T}^{-1}(s)\right]\\ \mathbf{P}(t) &= \mathcal{L}^{-1}\left[\mathbf{T}^{-1}(s)\mathbf{g}W(s)\right]\\ \mathbf{T}(s) &= \mathbf{G}+s\mathbf{C}\end{aligned}$$

$W(s)$ is the Laplace transform of $w(t)$ and $\mathcal{L}^{-1}[.]$ is the inverse Laplace transform operator. Notice that $\mathbf{M}(t)$ is the transition matrix and is independent of the input, whereas $\mathbf{P}(t)$ is the zero-state vector and is input-dependent. Without loss of generality, let the input be $w(t) = e^{j\omega_o t}$. At $t = T$ where T is the time step, we have

$$\mathbf{v}(T) = \mathbf{M}(T)\mathbf{v}(0^-) + \mathbf{P}(T).$$

At $t \in [nT, nT+T)$, the origin of the time is shifted from $t = 0$ to $t = nT$. Subsequently, the input is given by $w(t) = e^{j\omega_o(t+nT)}$, and the initial condition becomes $\mathbf{v}(nT)$. Taking these into consideration, we arrive at

$$\mathbf{v}(nT+T) = \mathbf{M}(T)\mathbf{v}(nT) + \mathbf{P}(T)e^{jn\omega_o T}. \tag{3}$$

If ω_o and T are kept unchanged, $\mathbf{M}(T)$ and $\mathbf{P}(T)$ are constant and need to be computed only once. The computation required in each step is only one matrix-vector multiplication and one vector addition. The response of the circuit at equally spaced intervals of time can therefore be computed efficiently. If the circuit has multiple exponential inputs, the response can be obtained using superposition

$$\mathbf{v}(nT+T) = \mathbf{M}(T)\mathbf{v}(nT) + \sum_{k=1}^{K}\mathbf{P}_k(T)e^{jn\omega_{o,k}T} \tag{4}$$

where $\mathbf{P}_k(T)$ is the zero-state vector to the input $w_k(t) = e^{jw_{o,k}t}$ and K is the number of inputs.

It is worth noting that the use of complex exponential function as inputs is instrumental in obtaining the simple relation given by (4). One special case of interest is $\omega_o = 0$, corresponding to a step input. Sinusoidal inputs can also be handled with ease. For example, if the input is $w(t) = \sin(\omega_o t)$, the response is given by

$$\mathbf{v}(nT+T) = \mathbf{M}(T)\mathbf{v}(nT) + \mathcal{I}m\left[\mathbf{P}(T)e^{jn\omega_o T}\right] \tag{5}$$

where $\mathcal{I}m[.]$ denotes the imaginary part of a complex argument. Similarly, if $w(t) = \cos(\omega_o t)$, the response is obtained by replacing $\mathcal{I}m[.]$ with $\mathcal{R}e[.]$ in (5), where $\mathcal{R}e[.]$ denotes the real part.

III. Sampled-Data Simulation of Nonlinear Circuits

A. Volterra Circuits

Nonlinearities typically encountered in integrated circuits include the exponential I–V characteristics of diodes and BJTs, the square-law characteristics of the drain current of MOS transistors, pn-junction capacitances, and other types of nonlinearities often modeled using polynomials. Because these nonlinear characteristics are usually continuously differentiable, they can be represented using corresponding Taylor series expansion at their dc operating point. A nonlinear circuit in the time domain is therefore represented by the differential equation

$$\mathbf{G}\mathbf{v}(t) + \mathbf{C}\frac{d\mathbf{v}(t)}{dt} = \mathbf{g}w(t) + f[\mathbf{v}(t)], \quad \mathbf{v}(t)|_{t=0^-} = \mathbf{v}(0^-) \tag{6}$$

where $\mathbf{G}$ and $\mathbf{C}$ are, respectively, the conductance and capacitance matrices whose entries are made of the coefficients of linear elements and the first-order term of the Taylor series expansion of the nonlinear characteristics. The higher order terms are embedded in the nonlinear function $f[\mathbf{v}(t)]$.

In [8] and [9], it was shown that the response of a nonlinear system $y(t)$ can be represented by the Volterra functional series

$$y(t) = \sum_{m=1}^{\infty} y_m(t) \tag{7}$$

where

$$y_m(t) = \int_{-\infty}^{\infty}\cdots\int_{-\infty}^{\infty} h_m(t, \tau_1, \ldots, \tau_m)x(\tau_1)\cdots x(\tau_m)d\tau_1\cdots d\tau_m$$

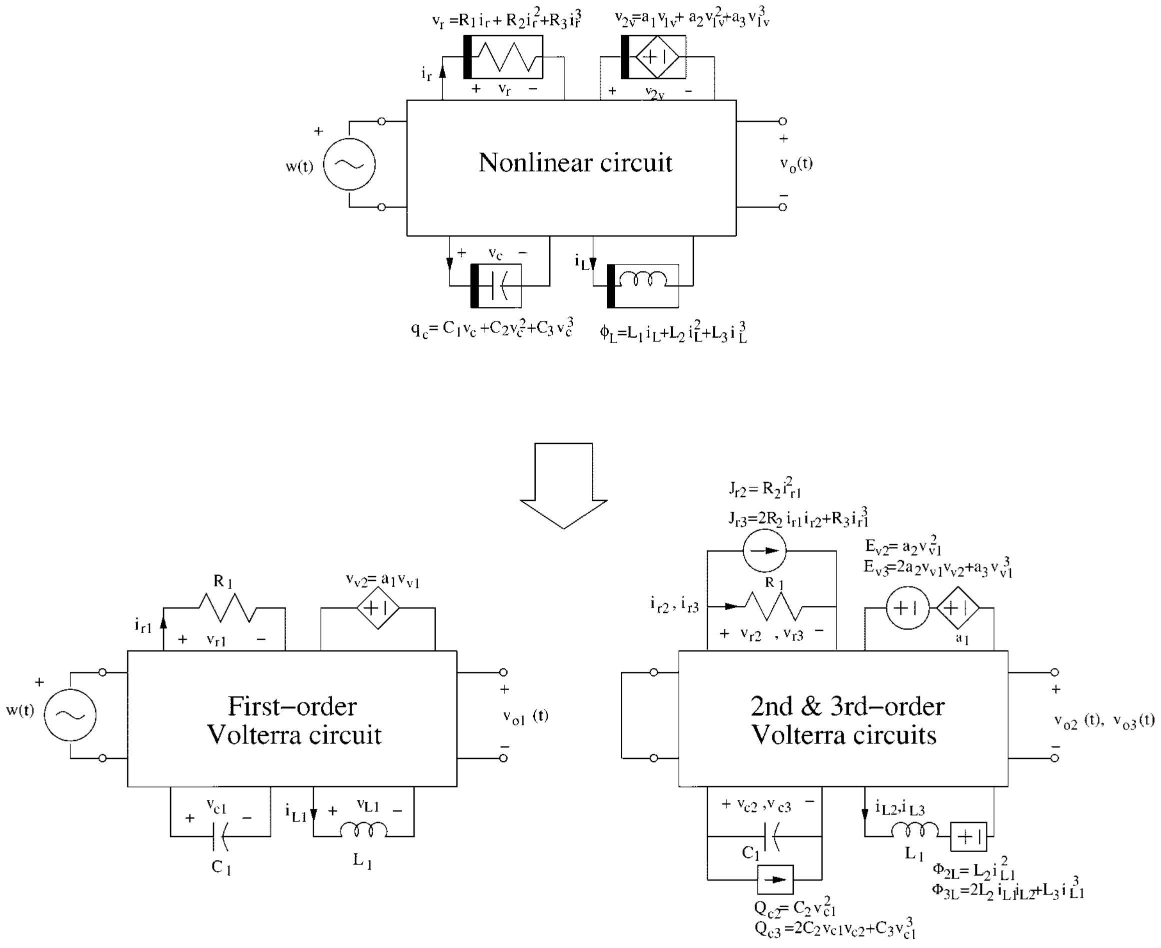
is the mth-order term of the Volterra series expansion of $y(t)$ and $h_m(t, \tau_1, \ldots, \tau_m)$ is the mth-order Volterra kernel. When the input is changed from $x(t)$ to $\epsilon x(t)$, where ϵ is a nonzero constant, the response becomes

$$y(t) = \sum_{m=1}^{\infty} y_m(t)\epsilon^m. \tag{8}$$

Equation (8) indicates that $y(t)$ is a polynomial in ϵ with the time-varying coefficients given by $y_m(t)$. Making use of this approach, representing $\mathbf{v}(t)$ in Volterra series expansion, substituting the results into (6), and equating the terms of the same order in ϵ result is the following set of differential equations:

$$\begin{aligned}\mathbf{G}\mathbf{v}_m(t) + \mathbf{C}\frac{d\mathbf{v}_m(t)}{dt} &= \mathbf{g}_m f_m[\mathbf{v}_1(t), \mathbf{v}_2(t), \ldots, \mathbf{v}_{m-1}(t)],\\ \mathbf{v}_m(t)|_{t=0^-} &= \mathbf{v}_m(0^-)\end{aligned} \tag{9}$$

where $\mathbf{v}_m(t)$ is the mth-order term of the Volterra series expansion of $\mathbf{v}(t)$, $f_m[\mathbf{v}_1(t), \ldots, \mathbf{v}_{m-1}(t)]$ is the input of the mth-order Volterra circuit, and $\mathbf{g}_m$ is a constant vector specifying the nodes to which the input is connected. Note $f_1(t) = w(t)$ and $\mathbf{g}_1 = \mathbf{g}$. The circuit depicted by (9) is termed the mth-order Volterra circuit. It is observed that these Volterra circuits are linear and are identical except their input in which the nonlinear characteristics are embedded. Also, the input of Volterra circuits is a function of the response of the lower order Volterra circuits only. A pictorial illustration of the Volterra circuit of nonlinear circuits containing nonlinear resistors, capacitors, inductors, and VCVT with up to the third-order nonlinear characteristics considered is shown in Fig. 1. Taking Laplace



Nonlinear circuit
First–order Volterra circuit
2nd & 3rd–order Volterra circuits
w(t)
v_o(t)

Fig. 1. Volterra circuits.

transform and its inverse, we obtain the time-domain response of (9)

$$\mathbf{v}_m(t) = \mathbf{M}(t)\mathbf{v}_m(0^-) + \mathcal{L}^{-1}\left[\mathbf{T}^{-1}(s)\mathbf{g}_m F_m(s)\right] \qquad (10)$$

where $F_m(s) = \mathcal{L}\{f_m[\mathbf{v}_1(t), \mathbf{v}_2(t), \ldots, \mathbf{v}_{m-1}(t)]\}$ and $\mathcal{L}[.]$ is the Laplace transform operator. Clearly seen from (10) is that to find $\mathbf{v}_m(t)$, $f_m(t)$ is needed.

B. Solution of Volterra Circuits

Let the input of the nonlinear circuit be $w(t) = \sin(\omega_o t)$. Because the input of the first-order Volterra circuit is identical to that of the nonlinear circuit, its response $\mathbf{v}_1(t)$ is obtained using the sampled-data analysis technique for linear circuits given earlier

$$\mathbf{v}_1(nT+T) = \mathbf{M}(T)\mathbf{v}_1(nT) + \mathcal{I}m\left[\mathbf{P}(T)e^{jn\omega_o T}\right]. \qquad (11)$$

To solve the second-order Volterra circuit, $f_2(t)$ is needed. To obtain $f_2(t)$, we use an interpolating function that interpolates $f_2(nT) = f_2[\mathbf{v}_1(nT)]$ to approximate $f_2(t)$. There are many interpolation techniques available. Polynomial-based interpolation, such as, Lagrange and Newton finite difference [10], are effective for low-order interpolation and become unstable once

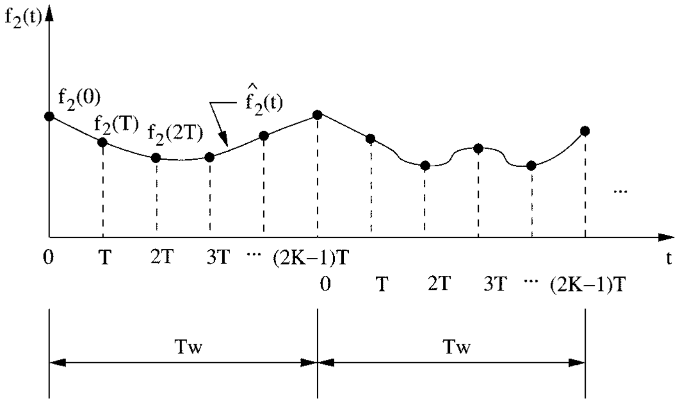


Fig. 2. Simulation window.

the order is high [11]. In addition, computation rises rapidly with the order of interpolation. Exponential interpolation [12] is effective only if the exponents are known and becomes expensive once the exponents are to be determined. Fourier series based interpolation is an efficient interpolation method [11], [14]. The order of the interpolation can be well above 1000 while still preserving the numerical stability. An attractive advantage of using high-order Fourier series interpolation is the accuracy of approximation. To minimize the cost of computation, a simulation window $\Delta = [0, (2K-1)T]$ shown in Fig. 2 is used. The sampled value of $f_2(t)$ in the window is first computed and a set of data $\{f_2(0), f_2(T), \ldots, f_2[(2K-1)]\}$ is obtained. $f_2(t)$ for $t \in \Delta$ is obtained from

$$f_2(t) = \hat{f}_2(t) + \Delta f_2(t) \tag{12}$$

where $\Delta f_2(t)$ is the error of interpolation and $\hat{f}(t)$ is obtained from interpolating Fourier series [12], [13]

$$\begin{aligned}\hat{f}(t) &= \mathcal{F}[f_2(nT)] \\ &= \frac{a_{2,0}}{2} + \sum_{k=1}^{K-1} [a_{2,k}\cos(k\omega_s t) + b_{2,k}\sin(k\omega_s t)] \\ &\quad + \frac{a_{2,K}}{2}\cos(K\omega_s t)\end{aligned} \tag{13}$$

where $\mathcal{F}[.]$ denotes interpolating Fourier series operator, $\omega_s = 2\pi/T_w$ where T_w is the width of the window, and the coefficients $a_{2,k}$ and $b_{2,k}$ are determined from the equations shown at the bottom of the page.

The second-order Volterra circuit for $t \in \Delta$ is depicted by

$$\begin{aligned}&\mathbf{G}\mathbf{v}_2(t) + \mathbf{C}\frac{d\mathbf{v}_2(t)}{dt} \\ &= \left\{\frac{a_{2,0}}{2} + \sum_{k=1}^{K-1}[a_{2,k}\cos(k\omega_s t) + b_{2,k}\sin(k\omega_s t)]\right. \\ &\qquad \left. + \frac{a_{2,K}}{2}\cos(K\omega_s t) + \Delta f_2(t)\right\}\mathbf{g}_2.\end{aligned} \tag{14}$$

Using the sampled-data analysis technique for linear circuits and superposition, and neglecting the interpolation error $\Delta f_2(t)$, we obtain the response of the circuit in the window

$$\begin{aligned}&\mathbf{v}_2(nT+T) \\ &= \mathbf{M}(T)\mathbf{v}_2(nT) + \mathcal{R}e\left\{\frac{a_{2,0}}{2}\mathbf{P}_o(T) + \frac{a_{2,K}}{2}\mathbf{P}_K(T)\right. \\ &\qquad \left. \cdot e^{jnK\omega_s T} + \sum_{k=1}^{K-1} a_{2,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right\} \\ &\quad + \mathcal{I}m\left\{\sum_{k=1}^{K-1} b_{2,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right\}\end{aligned} \tag{15}$$

where

$$\mathbf{P}_k(T) = \mathcal{L}^{-1}\left[\mathbf{T}^{-1}(s)\frac{\mathbf{g}_2}{s - jk\omega_s}\right]_{t=T}$$

is the zero-state vector of the circuit to the input $w_{2,k}(t) = e^{jk\omega_s t}$. Once the solution of the first- and second-order Volterra circuits is available, the sampled-data value of the input of the third-order Volterra circuit at $t = nT$, $t \in \Delta$ is obtained from $f_3(nT) = f_3[\mathbf{v}_1(nT), \mathbf{v}_2(nT)]$, and the solution of the circuit can be computed in a similar manner. The above process can be continued for higher order Volterra circuits with the number of samples per simulation window and the width of the window kept unchanged. It is evident from (7) that the response of the nonlinear circuit is obtained by summing up that of all Volterra circuits

$$\begin{aligned}\mathbf{v}(nT+T) &= \mathbf{M}(T)\mathbf{v}(nT) + \mathcal{I}m\left(\mathbf{P}(T)e^{jn\omega_o T}\right) \\ &\quad + \mathcal{R}e\left\{\frac{a_{2,0}}{2}\mathbf{P}_o(T) + \frac{a_{2,K}}{2}\mathbf{P}_K(T)e^{jnK\omega_s T}\right. \\ &\qquad \left. + \sum_{k=1}^{K-1} a_k\mathbf{P}_k(T)e^{jnk\omega_s T}\right\} \\ &\quad + \mathcal{I}m\left\{\sum_{k=1}^{K-1} b_k\mathbf{P}_k(T)e^{jnk\omega_s T}\right\}, \\ &\quad n = 1, 2, \ldots, 2K-1\end{aligned} \tag{16}$$

where

$$\mathbf{v}(nT) = \sum_{m=1}^{\infty} \mathbf{v}_m(nT),$$

$$a_k = \sum_{m=2}^{\infty} a_{m,k}, \quad \text{and} \quad b_k = \sum_{m=2}^{\infty} b_{m,k}.$$

IV. Discussion

In this section we examine important factors affecting the efficiency and accuracy of the method.

A. Efficiency

In the preceding derivation we have shown that if the number of samples per simulation window and the step size T are kept unchanged, $\mathbf{M}(T)$, $\mathbf{P}(T)$, and $\mathbf{P}_k(T)$, $k = 0, 1, 2, \ldots, K$

$$\begin{bmatrix} a_{2,0} \\ a_{2,1} \\ \cdots \\ a_{2,K}\end{bmatrix} = \frac{1}{K}\begin{bmatrix} 1 & 1 & \cdots & 1 \\ 1 & \cos(\omega_s T) & \cdots & \cos[(2K-1)\omega_s T)] \\ \cdots & \cdots & \cdots & \cdots \\ 1 & \cos(K\omega_s T) & \cdots & \cos[(2K-1)K\omega_s T)]\end{bmatrix}\begin{bmatrix} f_2(0) \\ f_2(T) \\ \cdots \\ f_2[(2K-1)T]\end{bmatrix}$$

$$\begin{bmatrix} b_{2,1} \\ b_{2,2} \\ \cdots \\ b_{2,K-1}\end{bmatrix} = \frac{1}{K}\begin{bmatrix} \sin(\omega_s T) & \sin(2\omega_s T) & \cdots & \sin[(2K-1)\omega_s T)] \\ \sin(2\omega_s T) & \sin(4\omega_s T) & \cdots & \sin[2(2K-1)\omega_s T)] \\ \cdots & \cdots & \cdots & \cdots \\ \sin[(K-1)\omega_s T] & \sin[2(K-1)\omega_s T] & \cdots & \sin[(K-1)(2K-1)\omega_s T]\end{bmatrix}\begin{bmatrix} f_2(T) \\ f_2(2T) \\ \cdots \\ f_2[(2K-1)T]\end{bmatrix}$$

will be constant. They can be computed in a preprocessing step prior to the start of simulation. The cost of computation of the preprocessing step is dominated by that for computing $\mathbf{P}_k(T)$, $k = 0, 1, 2, \ldots, K$. To minimize this cost, the relationship between $\mathbf{P}_1(T)$, the zero-state vector of the circuit to the input $e^{j\omega_s t}$ and $\mathbf{P}_k(T)$, the zero-state vector to the input $e^{jk\omega_s t}$, can be employed. It was shown in [15] that $\mathbf{P}_1(T)$ and $\mathbf{P}_k(T)$ are related to each other by

$$\begin{aligned}(\mathbf{G} + jk\omega_s\mathbf{C})\mathbf{P}_k(T) \\ &= (\mathbf{G} + j\omega_s\mathbf{C})\mathbf{P}_1(T) + \mathbf{g}\left(e^{jk\omega_s T} - e^{j\omega_s T}\right)\end{aligned}$$

from which $\mathbf{P}_k(T)$ can be obtained from $\mathbf{P}_1(T)$. For each $\mathbf{P}_k(T)$, one LU-factorization of $(\mathbf{G} + jk\omega_s\mathbf{C})$ is needed. LU factorization is less expensive as compared with computing $\mathbf{P}_k(T)$ from numerical integration. Once $\mathbf{M}(T)$, $\mathbf{P}(T)$, and $\mathbf{P}_k(T)$ are available, the response of nonlinear circuits is computed at equally spaced intervals of time and simulation proceeds in a window-by-window fashion. The response of the circuit in the nth step of the rth window is given by

$$\begin{aligned}\mathbf{v}(nT + T) \\ &= \mathbf{M}(T)\mathbf{v}(nT) + \mathcal{I}m\left\{\mathbf{P}(T)e^{jn\omega_o T}e^{j(r-1)(2K-1)\omega_o T}\right\} \\ &\quad + \mathcal{R}e\left\{\frac{a_{2,0}}{2}\mathbf{P}_o(T) + \frac{a_{2,K}}{2}\mathbf{P}_K(T)e^{jnK\omega_s T}\right. \\ &\qquad \left. + \sum_{k=1}^{K-1} a_{r,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right\} \\ &\quad + \mathcal{I}m\left\{\sum_{k=1}^{K-1} b_{r,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right\}. \end{aligned} \tag{17}$$

For different windows, because only $a_{r,k}$ and $b_{r,k}$ need to be computed and they can be computed with little additional cost, better efficiency is achieved.

B. Stability

The proposed method is an extension of the sampled-data analysis technique of linear circuits and inherits many of its properties thereby. The stability of the method can be examined from that of the interpolating Fourier series and the sampled-data simulation of linear circuits. Interpolating Fourier series has superior numerical stability over polynomial-based interpolation schemes, as demonstrated in [11]. It was shown in [7] that sampled-data analysis technique for linear circuits is an A-stable numerical integration algorithm. For a stable linear circuit it guarantees a stable numerical solution.

C. Input Waveforms

Although in the preceding development, a sinusoidal input signal was used, the input waveform is not restricted to be either sinusoidal or exponential and can be extended to include other functions. For linear circuits superposition applies and a given input waveform can be represented by a set of basis functions (possibly infinite). The response of the first-order Volterra circuit to each of these basis functions is first computed separately and the complete response of the circuit is then obtained

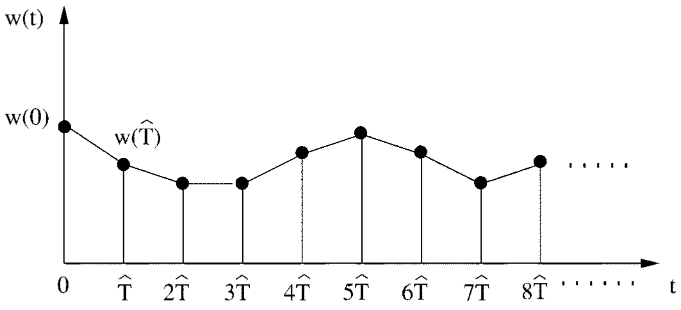


Fig. 3. Piecewise linear input.

by summing up its response to the basis functions. As an illustration, consider a nonlinear circuit with a piecewise-linear input shown in Fig. 3. The input in $[n\hat{T}, n\hat{T} + \hat{T}]$ is represented by

$$w(t) = w\left(n\hat{T}\right)u(t) + \frac{w\left(n\hat{T} + \hat{T}\right) - w\left(n\hat{T}\right)}{\hat{T}}t, \quad t \in \left[0, \hat{T}\right] \tag{18}$$

where $u(t)$ is a unit step input. Note that the time origin has been shifted from $t = 0$ to $t = n\hat{T}$. The zero-input response of the circuit is independent of the input and is determined solely from the transition matrix $\mathbf{M}(t)$. It can be shown that the response is obtained from [16]

$$\begin{aligned}\mathbf{v}(nT + T) = \mathbf{M}(T)\mathbf{v}(nT) + w\left(n\hat{T}\right)\mathbf{U}(T) \\ + \frac{w\left(n\hat{T} + \hat{T}\right) - w\left(n\hat{T}\right)}{\hat{T}}\mathbf{R}(T)\end{aligned} \tag{19}$$

where

$$\mathbf{U}(T) = \mathcal{L}^{-1}\left[\frac{\mathbf{T}^{-1}(s)}{s}\right]_{t=T}\mathbf{g} \tag{20}$$

$$\mathbf{R}(T) = \mathcal{L}^{-1}\left[\frac{\mathbf{T}^{-1}(s)}{s^2}\right]_{t=T}\mathbf{g} \tag{21}$$

are constant for fixed step size T. The two basis functions in this case are the unit step function

$$u(t) = \begin{cases} 1, & t \geq 0 \\ 0, & t < 0 \end{cases}$$

and the unit ramping function

$$r(t) = \begin{cases} t, & t \geq 0 \\ 0, & t < 0. \end{cases}$$

This approach can be generalized to use an infinite number of basis functions, such as Fourier series and wavelets, to represent arbitrary input waveforms. Once the solution of the first-order Volterra circuit is available, higher order Volterra circuits can be solved subsequently.

D. The Maximum Step Size

The upper bound of the step size is subject to the constraint set by Nyquist theorem. Specifically, because the highest frequency of the input signal is the frequency of the highest order term of the interpolating Fourier series given by $\omega_{\max} = K\omega_s$, the lower bound of the sampling frequency is therefore given by

$\omega_{\min} = 2\omega_{\max}$. Subsequently, the maximum step size is given by $T_{\max} = \pi/\omega_{\max}$. The actual step size is usually less than $T_{\max}$ and is determined from simulation accuracy and speed.

E. Accuracy

The accuracy of the method depends upon: 1) the order of Taylor series expansion in representation of nonlinear characteristics; 2) the order of Volterra series expansion in depicting nonlinear circuits; 3) the order of the interpolating Fourier series in approximation of the input of high-order Volterra circuits; 4) simulation window; and 5) error propagation. We examine each of them in detail.

1) The Order of Taylor Series Expansion: The order of Taylor series expansion of nonlinear elements depends upon the characteristics of the nonlinearities, the amplitude of the input signal, and the error of approximation. When the nth-order Taylor series expansion is employed to represent the output v of a nonlinear element, the truncation error is given by

$$\frac{v^{(n+1)}(\xi\Delta v)}{(n+1)!}(\Delta v)^{n+1}$$

where Δv is the displacement from the operating point and $0 < \xi < 1$. As an example, consider a forward biased diode characterized by $i_D = I_S[e^{v_D/V_T} - 1]$, where v_D and i_D are the voltage and current of the diode, respectively, I_S is the saturation current, and V_T is the thermal voltage. Let $v_D = v_d + V_D$ where V_D and v_d are the dc and ac components of v_D. Expanding e^{v_d/V_T} in Taylor series at the dc operating point gives

$$i_d = I_S e^{V_D/V_T}\left(\frac{v_d}{V_T} + \frac{v_d^2}{2V_T^2} + \frac{v_d^3}{6V_T^3} + \frac{v_d^4}{24V_T^4} + \frac{v_d^5}{120V_T^5} + \cdots\right)$$

where i_d is the ac component of i_D. If the fourth-order Taylor series expansion is used, the truncation error is estimated from $I_S e^{V_D/V_T}(v_{d,\max}^5/120V_T^5)$, where $v_{d,\max}$ is the peak value of v_d.

2) The Order of Volterra Series Expansion: Similar to Taylor series expansion in representation of nonlinear elements, the order of Volterra series expansion in depicting the nonlinear circuits depends upon the nonlinear characteristics of the circuits and the error of approximation. Consider a nonlinear resistor characterized by

$$v_r = R_1 i_r + R_2 i_r^2 + R_3 i_r^3 \tag{22}$$

where v_r and i_r are the voltage and current of the resistor, respectively, and R_1, R_2, R_3 are constants. Representing v_r and i_r in their Volterra series expansions to the order of three and substituting the results into (22) yields

$$\begin{cases} v_{r1} = R_1 i_{r1} \\ v_{r2} = R_1 i_{r2} + R_2 i_{r1}^2 \\ v_{r3} = R_1 i_{r3} + 2R_2 i_{r1} i_{r2} + R_3 i_{r1}^3. \end{cases} \tag{23}$$

Equation (23) reveals that the nonlinear resistor can be represented by three linear resistors characterized by $v_{rm} = R_1 i_{rm}$, $m = 1, 2, 3$, together with added voltage sources given by $E_2 = R_2 i_{r1}^2$ and $E_3 = 2R_2 i_{r1} i_{r2} + R_3 i_{r1}^3$ for the second and third Volterra circuits, respectively, as shown in Fig. 1. If the fourth-order Volterra series expansion is considered, we have

$$\begin{cases} v_{r1} = R_1 i_{r1} \\ v_{r2} = R_1 i_{r2} + R_2 i_{r1}^2 \\ v_{r3} = R_1 i_{r3} + 2R_2 i_{r1} i_{r2} + R_3 i_{r1}^3 \\ v_{r4} = R_1 i_{r4} + 2R_2 i_{r1} i_{r3} + R_2 i_{r2}^2 + 3R_3 i_{r1}^2 i_{r2}. \end{cases} \tag{24}$$

The difference is the last equation in (24) that takes into consideration the effect of fourth-order nonlinear characteristics. Equation (23) will be considered adequate if the difference between the response of the circuit with the third and fourth-order Volterra series expansions considered is negligible. A nonlinear circuit is said to be *mildly nonlinear* if it can be characterized by lower order Volterra series expansions, usually up to the fifth-order.

3) The Order of Interpolating Fourier Series: For a given simulation window, the higher the order of interpolating Fourier series, the more accurate the approximation. A downside of using high-order interpolation is the increased cost of computation because a large number of zero-state vectors are to be computed. The solution of the first-order Volterra series will be accurate if $\mathbf{M}(T)$ and $\mathbf{P}(T)$ are accurate. In this work, numerical Laplace inversion, an A-stable high-order numerical integration technique [2], is used to compute these quantities. In our implementation, the order of integration is 19. In addition, a multistep numerical Laplace inversion algorithm given in the Appendix is employed to minimize the error due to the large time interval. The error in computing $\mathbf{M}(T)$ and $\mathbf{P}(T)$ is therefore considered to be negligible. We thereby conclude that the error in solving the second-order Volterra circuit is dominated by the interpolation error $\Delta f_2(t)$. To estimate this error, the input of the second-order Volterra circuit is derived with the step sizes T and $T/2$, respectively. The order of interpolation is determined from the normalized mean square error (NMSE) [17]

$$\text{NMSE} = \frac{\sum_{k=0}^{2K-1}\left[v_{2,T}(nT) - v_{2,T/2}(nT)\right]^2}{\sum_{k=0}^{2K-1} v_{2,T/2}^2(nT)} \tag{25}$$

where $v_{2,T}(nT)$ and $v_{2,T/2}(nT)$ are the response of the circuit with the step sizes of T and $T/2$, respectively. In order to compute $v_{2,T}(nT)$ and $v_{2,T/2}(nT)$, $\mathbf{M}(T)$, $\mathbf{M}(T/2)$, $\mathbf{P}(T)$, $\mathbf{P}(T/2)$, $\mathbf{P}_k(T)$, and $\mathbf{P}_k(T/2)$ are needed. To minimize the computation, the relationship

$$\mathbf{M}(T) = \mathbf{M}^2\left(\frac{T}{2}\right) \tag{26}$$

and

$$\mathbf{P}(T) = \left[e^{j\omega_o(T/2)}\mathbf{I} + \mathbf{M}\left(\frac{T}{2}\right)\right]\mathbf{P}\left(\frac{T}{2}\right) \tag{27}$$

where $\mathbf{I}$ is an identity matrix of appropriate dimensions, can be employed (see the Appendix for details).

4) Simulation Window: When a simulation window is employed it is assumed that the truncated data series by the window is periodic with the period to be the width of the window. The rate of convergence of the interpolating Fourier series depends upon the boundary conditions of the data series. It was shown

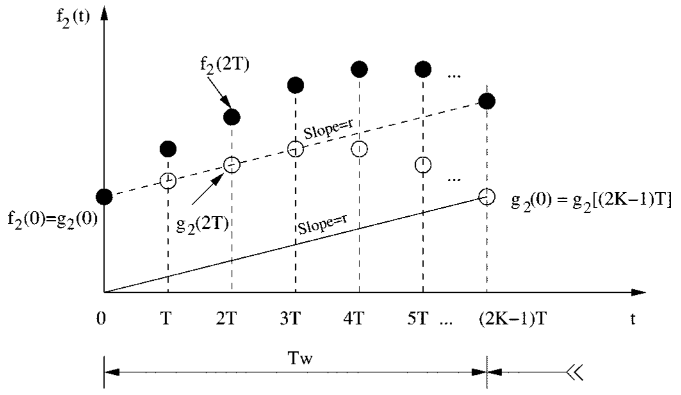


Fig. 4. Aperiodic sequence to periodic sequence transformation.

in [17] and [18] that a large error will arise if the function to be represented by interpolating Fourier series is discontinuous at the boundary. The error will be further increased if the derivatives of the function are also discontinuous [19]. To minimize the error due to the discontinuity of the function value at the boundary, the window should be chosen such that the input of Volterra circuits be periodic with respect to the simulation window. For circuits with only one sinusoidal input, the window size that is the same as the period of the sinusoidal input meets this requirement because the frequency components in the circuits are the frequency of the input and its harmonics, as revealed by (22). For circuits with nonsinusoidal inputs or multiple sinusoidal inputs, this approach becomes ineffective. Alternatively, a technique that reduces the effects of the discontinuity (Gibb effect) at the window edges can be employed. Specifically, consider the input of the second-order Volterra circuit $\{f_2(0),\, f_2(T),\, \ldots,\, f[(2K-1)T]\}$ shown in Fig. 4. We construct a new function $g_2(t)$ from

$$g_2(t) = f_2(t) + rt$$

where r is a constant, and impose $g_2(0) = g_2[(2K-1)T]$. The value of r is therefore given by

$$r = -\frac{f_2[(2K-1)T] - f_2(0)}{(2K-1)T}.$$

Because there is no discontinuity in $\{g_2(0),\, g_2(T),\, \ldots,\, g_2[(2K-1)]\}$, interpolating Fourier series can be employed to derive $g_2(t)$ without introducing a large error. Once $g_2(t)$ is available, $f_2(t)$ can be obtained from the inverse of the transform

$$f_2(t) = g_2(t) - rt.$$

Using this technique, (14) becomes

$$\begin{aligned} &\mathbf{G}\mathbf{v}_2(t) + \mathbf{C}\frac{d\mathbf{v}_2(t)}{dt} \\ &= \left\{ \frac{\hat{a}_{2,0}}{2} + \frac{\hat{a}_{2,K}}{2}\cos(K\omega_s t) \right. \\ &\quad \left. + \sum_{k=1}^{K-1}\left[\hat{a}_{2,k}\cos(k\omega_s t) + \hat{b}_{2,k}\sin(k\omega_s t)\right] - rt \right\} \mathbf{g}_2 \end{aligned} \tag{28}$$

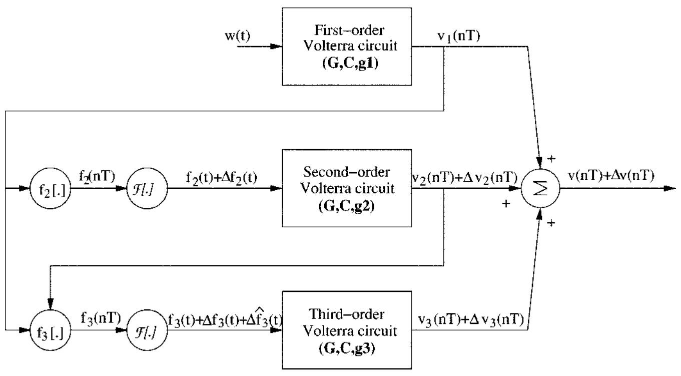


Fig. 5. Error analysis.

where $\hat{a}_{2,k}$ and $\hat{b}_{2,k}$ are the coefficients of the interpolating Fourier series that interpolates $\{g_2(0),\, g_2(T),\, \ldots,\, g_2[(2K-1)]\}$. The response of the circuit in the window is obtained from

$$\begin{aligned} \mathbf{v}_2(nT+T) =& \mathbf{M}(T)\mathbf{v}_2(nT) \\ &+ \mathcal{R}e\left\{ \frac{\hat{a}_{2,o}}{2}\mathbf{P}_o(T) + \frac{\hat{a}_{2,K}}{2}\mathbf{P}_K(T)e^{jnK\omega_s T} \right. \\ &\left. + \sum_{k=1}^{K-1}\left(\hat{a}_{2,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right) \right\} \\ &+ \mathcal{I}m\left\{ \sum_{k=1}^{K-1}\left(\hat{b}_{2,k}\mathbf{P}_k(T)e^{jnk\omega_s T}\right) \right] - r\mathbf{R}(T) \end{aligned} \tag{29}$$

where $\mathbf{R}(T)$ was given earlier. This approach can be further developed to minimize the error due to the discontinuity of the derivatives of the function.

5) Error Propagation: In this section we estimate the output error due to the error in interpolation. The error due to the truncation of Taylor and Volterra series is usually insignificant for mildly nonlinear circuits. To simplify the analysis, only up to the third-order Taylor and Volterra series expansions are considered. The approach can be generalized to a higher order Volterra circuits. Consider a nonlinear element model by (22). Let the interpolation error in deriving the input of the second-order Volterra circuit be $\Delta f_2(t)$, as shown in Fig. 5. The response of the second-order Volterra circuit is given by $\mathbf{v}_2(nT) + \Delta\mathbf{v}_2(nT)$, where

$$\Delta\mathbf{v}_2(nT) = \mathcal{L}^{-1}\left[\mathbf{T}^{-1}(s)\mathbf{g}_2\Delta F_2(s)\right]_{t=nT} \tag{30}$$

is the response to the error $\Delta f_2(t)$ and $\Delta F_2(s) = \mathcal{L}[\Delta f_2(t)]$. The sampled-data value of the input of the third-order Volterra circuit is computed from

$$\begin{aligned} f_3(nT) =& f_3[v_1(nT),\, v_2(nT) + \Delta v_2(nT)] \\ =& \left[2a_2 v_1(nT)v_2(nT) + a_3 v_1^3(nT)\right] \\ &+ 2a_2 v_1(nT)\Delta v_2(nT) \end{aligned} \tag{31}$$

from which the input is approximated using interpolation

$$\begin{aligned} f_3(t) =& \mathcal{F}\left[2a_2 v_1(nT)v_2(nT) + a_3 v_1^3(nT)\right] \\ &+ 2a_2\mathcal{F}[v_1(nT)\Delta v_2(nT)] + \Delta f_3(t) \\ =& \hat{f}_3(t) + \Delta\hat{f}_3(t) + \Delta f_3(t). \end{aligned} \tag{32}$$

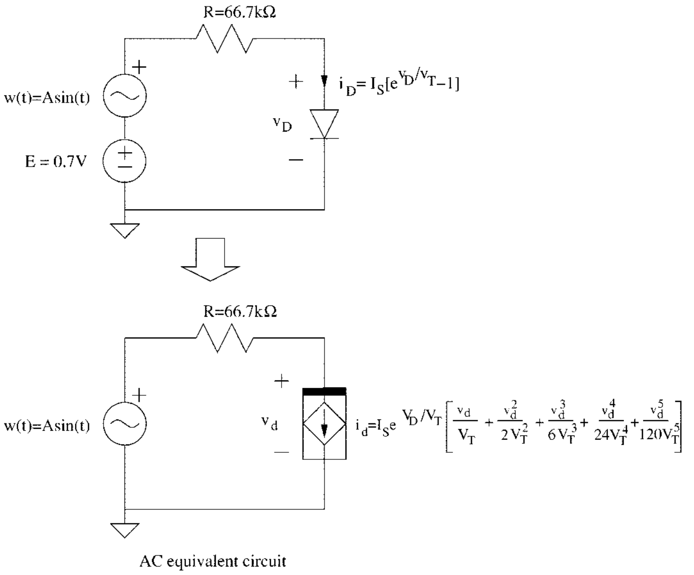

R=66.7kΩ
w(t)=Asin(t)
E = 0.7V
$i_D=I_S[e^{V_D/V_T}-1]$
v_D
AC equivalent circuit
v_d
$i_d=I_S e^{V_D/V_T}\left[\frac{v_d}{V_T}+\frac{v_d^2}{2V_T^2}+\frac{v_d^3}{6V_T^3}+\frac{v_d^4}{24V_T^4}+\frac{v_d^5}{120V_T^5}\right]$

Fig. 6. Diode circuit.

The response of the third-order Volterra circuit is given by $\mathbf{v}_3(nT)+\Delta\mathbf{v}_3(nT)$, where

$$\Delta\mathbf{v}_3(nT)=\mathcal{L}^{-1}\left\{\mathbf{T}^{-1}(s)\mathbf{g}_3\Delta\hat{F}_3(s)\right\}_{t=nT}+\mathcal{L}^{-1}\left\{\mathbf{T}^{-1}(s)\mathbf{g}_3\Delta F_3(s)\right\}_{t=nT} \quad (33)$$

and $\Delta\hat{F}_3(s)$ and $\Delta F_3(s)$ are the Laplace transform of $\Delta\hat{f}_3(t)$ and $\Delta f_3(t)$, respectively. The total error is obtained by summing up that of the second and third-order Volterra circuits

$$\Delta\mathbf{v}(nT)=\Delta\mathbf{v}_2(nT)+\Delta\mathbf{v}_3(nT).$$

V. Examples

The algorithm presented in the paper has been implemented in a computer program Sampled-Data Simulation of Nonlinear (SDSN) circuits. To assess the accuracy and efficiency of the method, a first-order PC algorithm implemented using forward Euler as the predictor and the backward Euler as the corrector and a first-order PC algorithm implemented using BDF have also been implemented in SDSN. Several nonlinear circuits are analyzed using both SDSN and PC, and the results are presented.

A. Diode Circuit

Consider a diode circuit shown in Fig. 6. The diode is characterized by $i_D(t)=I_S[e^{v_D(t)/V_t}-1]$, where $I_S=10^{-16}$ A, and $V_t=26$ mV at 300 K. The dc operating point is obtained by solving the circuit using Newton–Raphson iterations. Using the fifth-order Volterra series expansion, the ac behavior of the circuit is depicted by

$$w(t)=RI_S e^{V_D/V_T}\left[\frac{v_d(t)}{V_T}+\frac{v_d^2(t)}{2V_T^2}+\frac{v_d^3(t)}{6V_T^3}+\frac{v_d^4(t)}{24V_T^4}+\frac{v_d^5(t)}{120V_T^5}\right]+v_d(t). \quad (34)$$

In SDSN analysis, the width of the simulation window is set to one-tenth of the period of the signal, i.e., 0.1 s. The number of samples per simulation window in PC analysis is chosen to be 100. It was found that lowering the number of steps would result in a large error. The number of steps per simulation window in SDSN is 20. The termination criterion for Newton–Raphson analysis in PC analysis is set to 10^{-5}. The ac voltage across the diode is computed using both SDSN and PC, and the results are plotted in Fig. 7 for various input amplitude. The difference between SDSN and PC is plotted in Fig. 8. It is seen that the results from SDSN are in a good agreement with those from PC analysis. The maximum normalized difference is about 1%.

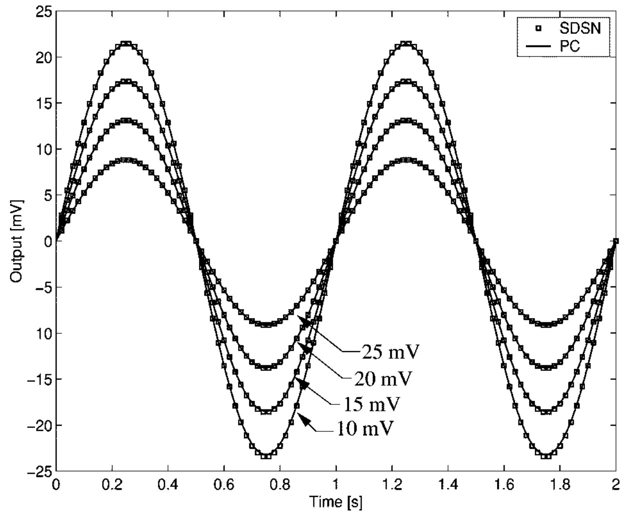


Fig. 7. Response of diode circuit.

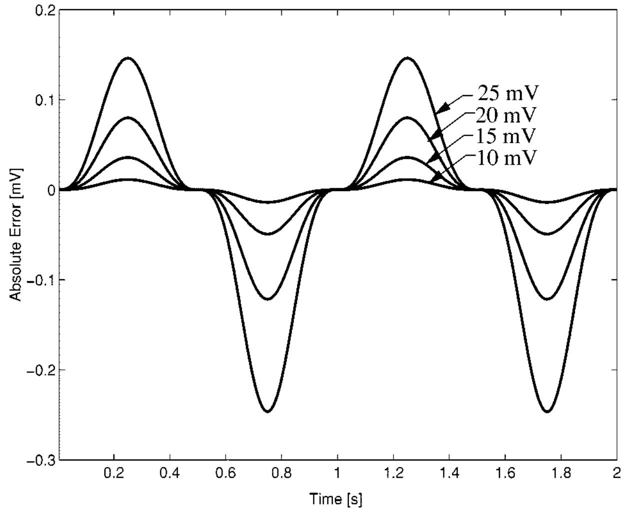


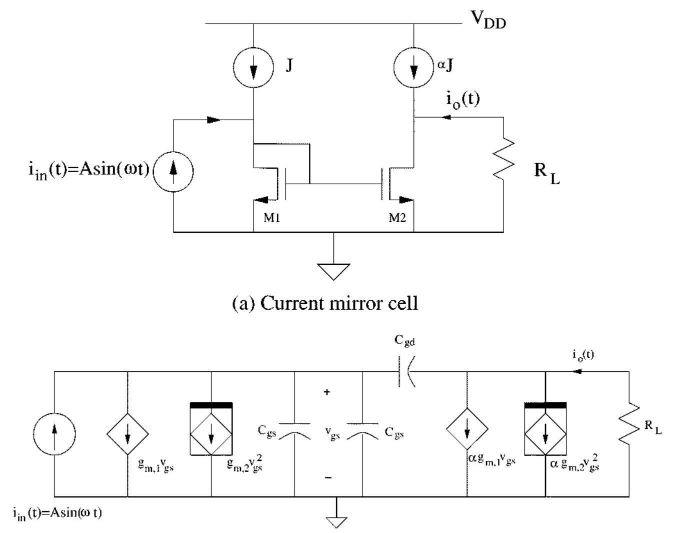
Fig. 8. Error between SDSN and PC.

B. Current Mirror

The second example investigated is a current mirror cell shown in Fig. 9. The cell is a building block for continuous-time current-mode circuits [21]. Both transistors are biased in saturation. Neglecting channel-length modulation and other second-order effects, we obtain the ac component of the channel current

$$i_d=g_{m1}v_{gs}+g_{m2}v_{gs}^2$$

where $g_{m1}=\mu_n C'_{ox}(W/L)(V_{GS}-V_T)$ is the linear transconductance, $g_{m2}=(1/2)\mu_n C'_{ox}(W/L)$ is the second-order non-



(b) AC equivalent circuit

Fig. 9. Current mirror.

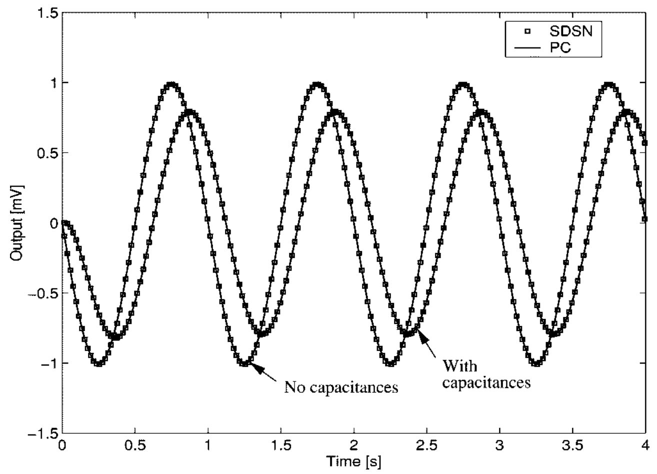


Fig. 10. Response of current mirror.

linear transconductance, μ_n is the surface mobility of free electrons, C'_{ox} is the gate capacitance per unit area, W and L are the width and length of the transistors, respectively, V_{GS} and V_T are the biasing and threshold voltages, respectively. The load resistance is 50 Ω, $C_{gs} = C_{gd} = 1$ pF. The mirror is biased such that $g_{m1} = 100$ mA/V and $g_{m2} = g_{m1}$. The ac circuit of the cell is shown in Fig. 9. The input is a sinusoidal current source with amplitude 1 mA and frequency 1 GHz. In our simulation, the current gain of the mirror α is set to unity. The output current is computed using both SDSN and PC. The results are shown in Fig. 10 with and without the gate-to-source and gate-to-drain capacitances considered. It is seen that when the capacitances are not considered, the mirror realizes $i_o \approx -\alpha i_{in}$. When these capacitances are considered, the output current is reduced. The results from SDSN is in a good agreement with those from PC analysis.

C. Low-Noise Amplifier

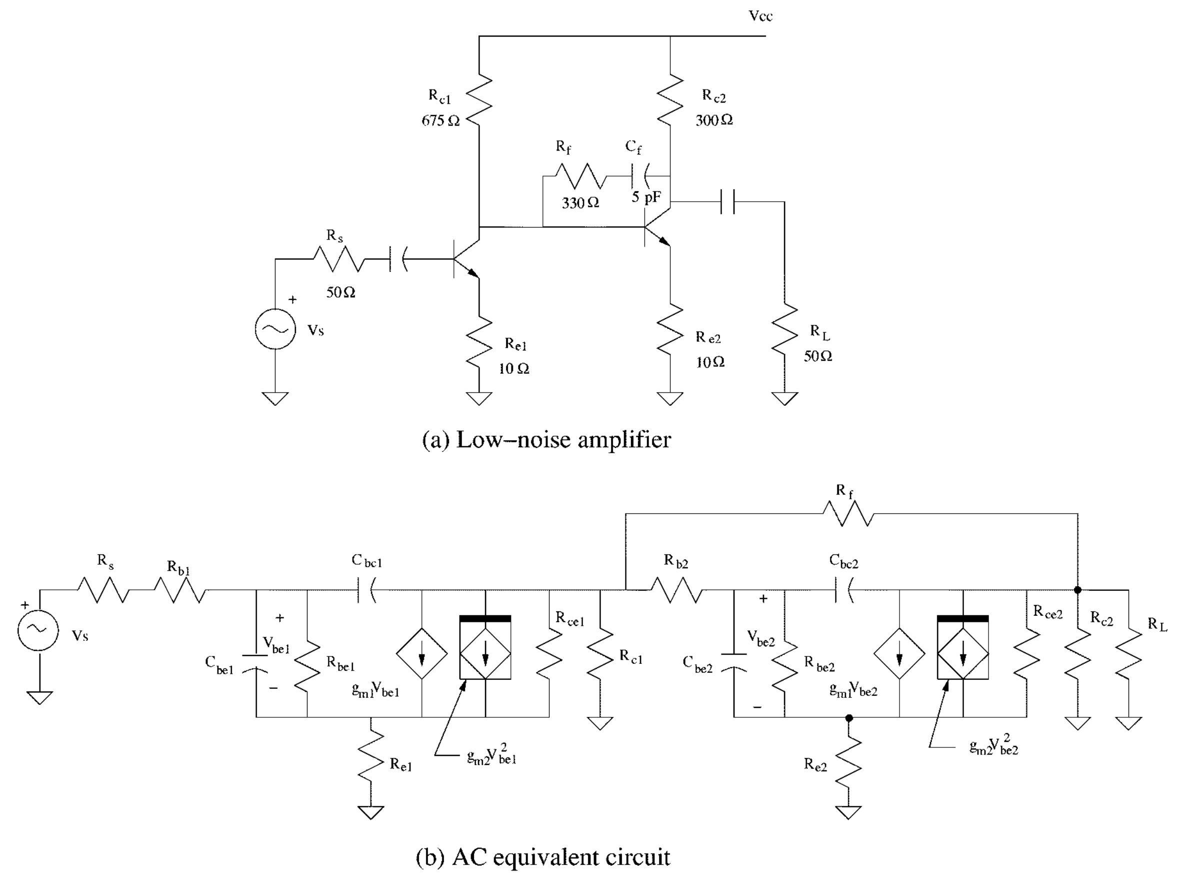
A low-noise amplifier shown in Fig. 11 is investigated. The amplifier consists of two stages. In addition to the emitter feedback in both stages, the shunt feedback is also employed to stabilize the gain. In our analysis, two BJTs are considered as the nonlinear elements with ac component of the collector current given by

$$i_c \approx g_{m,1}v_{be} + g_{m,2}v_{be}^2$$

where $g_{m,1} = I_C/V_T$ and $g_{m,2} = I_C/2V_T^2$, $I_C = \beta I_S e^{V_{BE}/V_T}$, β is the current gain, V_{BE} and v_{be} are the biasing and ac voltages, respectively. The biasing current is 5.5 mA, resulting in $g_{m,1} = 0.2115$ A/V and $g_{m,2} = 4.068$ A/V^2. The ac equivalent circuit of the amplifier is shown in the figure. The ac parameters of the BJTs are given as follows: $C_{be} = 6$ pF, $C_{bc} = 6$ fF, $R_{be} = 500\ \Omega$, $R_{ce} = 100\,\text{k}\Omega$, the base resistance is 11 Ω. The input is a sinusoidal voltage signal with frequency 1 GHz and amplitude 1 mV. The amplifier is analyzed using SDSN, together with the BDF-based PC algorithm implemented using Tableau formulation [2]. The results are plotted in Fig. 12. It is seen that both agree very well. The amplifier provides a voltage gain of about 16 dB.

D. Nonlinear Circuit With Capacitors

To further assess the method, the nonlinear circuit shown in Fig. 13 is investigated. The element values are $G_1 = 0.1$ S, $C_1 = 1.0$ F, and $C_2 = 0.1$ F. The nonlinear conductor is characterized by $i(t) = v(t) + 0.5v^2(t) + 0.25v^3(t) + 0.1v^4(t) + 0.1v^5(t)$. The voltage across C_2 is computed using both SDSN and PC. The number of steps per simulation window for SDSN and PC is 20 and 100, respectively. The results are shown in Fig. 14 for various input amplitudes. It is seen that the results from SDSN match those from PC analysis well. The difference between SDSN and PC is plotted in Fig. 15 with the maximum normalized difference near 1%. To show that the number of steps per simulation window in PC analysis should be sufficiently large. The circuit is solved with the window size set to 0.5 s and the number of steps per window for SDSN remains unchanged but that for PC is varied. The results are plotted in Fig. 16. The error due to a smaller number of steps per window in PC analysis is evident. The error due to the truncation of Volterra series is also examined by computing the difference between the response of the circuit with the order of Volterra series expansion to be 3 and 4, respectively, and the results are plotted in Fig. 17 with the input amplitude 0.5 V. It is observed that the normalized error due to the truncation of Volterra series is about 0.1%. The computational efficiency of SDSN is demonstrated in Fig. 18 where the CPU time of both SDSN and PC analysis plotted as a function of the number of simulation windows. CPU time is measured for both SDSN and PC programs coded in Matlab, an interactive mathematical language that runs in an interpretive mode [20]. The program was executed on a Sun Ultra 1 workstation with 450-MHz CPU and 256-MB RAM. It is seen the initial cost of computation of SDSN is higher than that of PC, mainly due to the cost of the preprocessing step where $\mathbf{M}(T)$, $\mathbf{P}(T)$, and $\mathbf{P}_k(T)$ are computed. CPU time of PC analysis arises rapidly with the number of simulation windows, whereas that of SDSN rises slowly, indicating that the computational cost of SDSN is nearly independent of the number of simulation windows. SDSN is significantly faster than PC. Also observed is that doubling the number of samples



Vcc
R_{c1} 675 Ω
R_{c2} 300 Ω
R_f 330 Ω
C_f 5 pF
R_s 50Ω
Vs
R_{e1} 10 Ω
R_{e2} 10Ω
R_L 50Ω
(a) Low–noise amplifier
R_s R_{b1} C_{bc1} R_{b2} C_{bc2} R_f
V_{be1} C_{be1} R_{be1} $g_{m1}V_{be1}$ R_{ce1} R_{c1} R_{e1} $g_{m2}V^2_{be1}$
V_{be2} C_{be2} R_{be2} $g_{m1}V_{be2}$ R_{ce2} R_{c2} R_L R_{e2} $g_{m2}V^2_{be2}$
(b) AC equivalent circuit

Fig. 11. Low-noise amplifier.

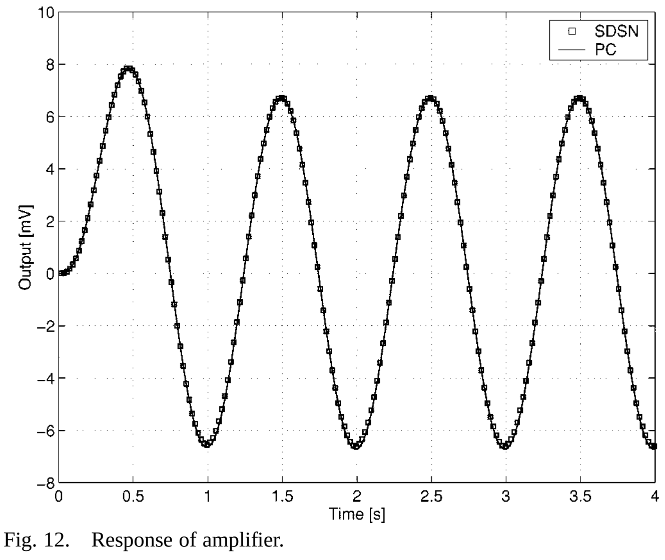


Fig. 12. Response of amplifier.

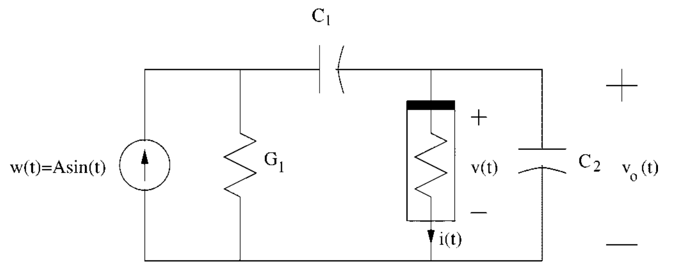


Fig. 13. Circuit with nonlinear resistor.

per simulation window only increases the computation of the preprocessing step. Its effect on the cost of computation for the coefficients of interpolating Fourier series is marginal.

VI. Conclusion

In this paper, we have introduced a new method for analysis of mildly nonlinear circuits at equally spaced intervals of time. The method is based on Volterra series representation of nonlinear systems and interpolating Fourier series approximation. It characterizes mildly nonlinear circuits using a set of Volterra circuits that are linear and topologically identical. These Volterra circuits are solved using sampled-data simulation. As compared to traditional SPICE-like methods, the proposed method does not require Newton–Raphson iterations or predictor-corrector integrations at each time step. Instead, it requires matrix/vector operations and FFT calculations for each time window, which is expected to require less computation in some, but not all cases. In analysis of a given circuit the proposed algorithm performs some (expensive) precomputation, i.e., the calculation of the $\mathbf{M}$ and $\mathbf{P}$ matrices/vectors in a preprocessing step only once, little computation is needed thereafter for each additional time point. The method will therefore be computationally efficient if we need the response of the circuit for many time points where the initial expensive calculation is amortized over many time points. SPICE-like simulators, however, recompute the solution of the circuit at every time instant, even if the circuit has gone through the same state at some previous time instants, i.e., SPICE-like simulators simply recompute the solution without making use

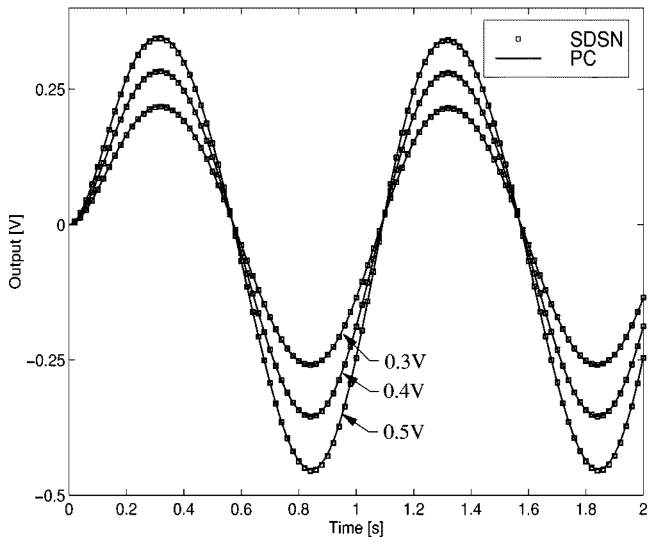


Fig. 14. Response.

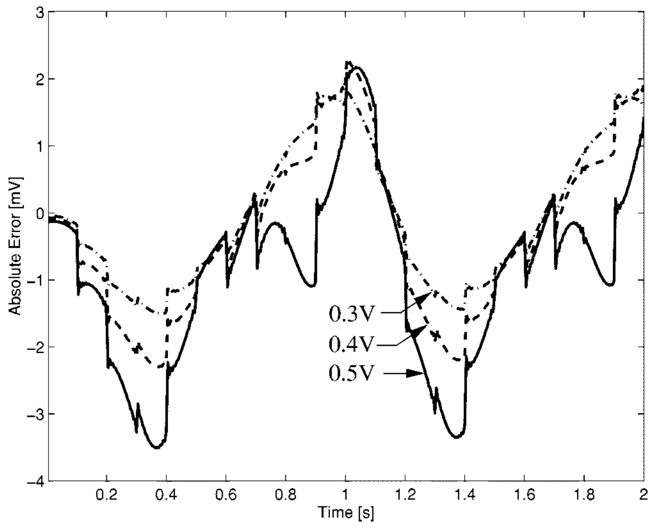


Fig. 15. Error between SDSN and PC.

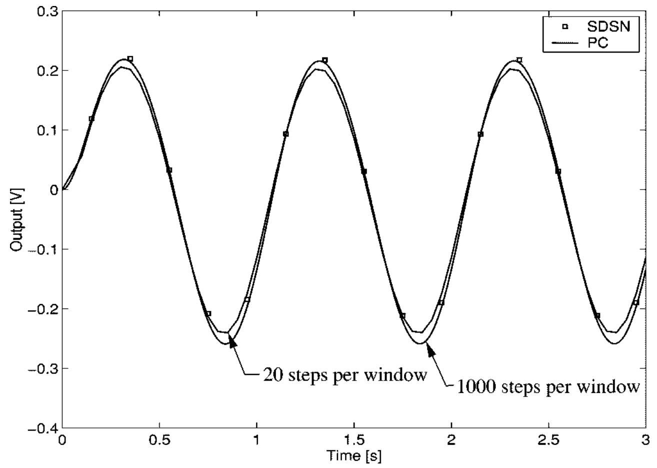


Fig. 16. Effect of step size in PC analysis.

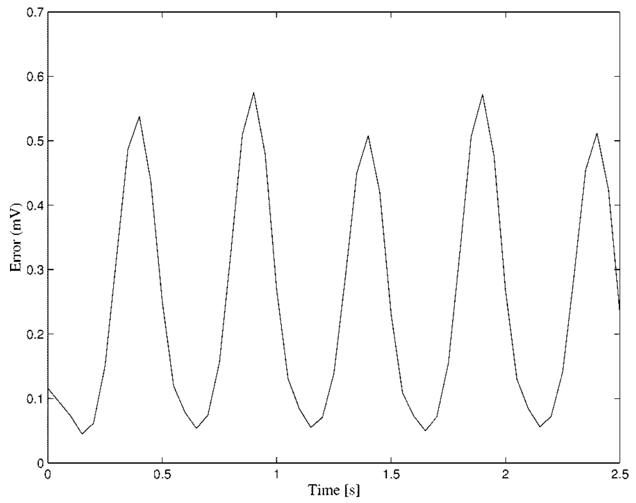


Fig. 17. Effect of the order of Volterra series expansion.

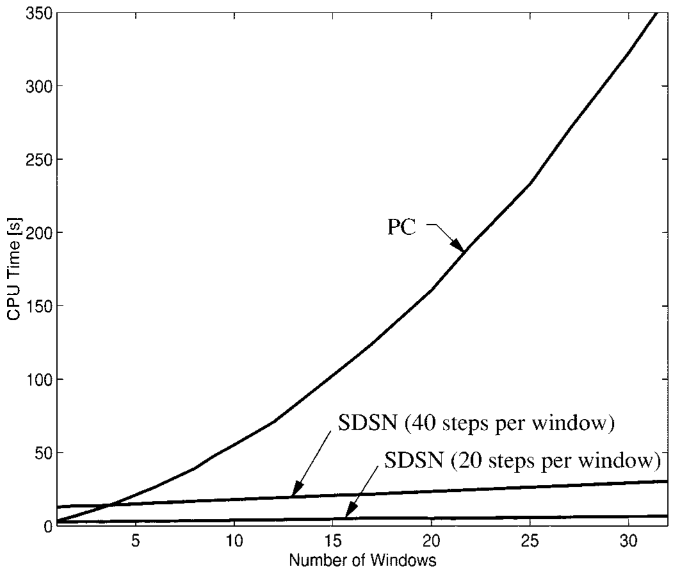


Fig. 18. CPU time.

of the past information. Thus, there are many tradeoffs in the traditional SPICE-like algorithms and the method presented in this paper.

The proposed method is most suitable for nonlinear circuits with a fixed dc operating point and small ac inputs. In these circuits, nonlinear elements can be characterized by using only a few terms of their Taylor series expansion at the dc operating point (mildly nonlinear). The method is applicable to circuits with either a single input or multiple inputs. Although the waveform of the inputs can be arbitrary, implementation will be significantly simplified if the inputs are sinusoidal.

The accuracy of the proposed method is determined by both the accuracy of modeling and that of simulation algorithm. For a given nonlinear circuit with known nonlinear characteristics, modeling accuracy can be predetermined prior to the start of simulation. This allows us to set the order of Taylor series expansion in characterization of the nonlinear elements and the order of Volterra series expansion in representation of the circuit. Both orders are usually set to be the same. The accuracy of simulation is mainly determined by the order of interpolating Fourier series used to approximate the input of high-order Volterra circuits and the size of the simulation window. The error generated during computing $\mathbf{M}$ and $\mathbf{P}$ is usually negligible when high-order numerical Laplace inversion algorithms are employed. An increase in the order of interpolating Fourier series lowers the interpolation error, subsequently better accuracy. The price paid, however, is an increase in the computational time of the preprocessing step where $\mathbf{P}_k$ vectors are computed. Due to the error caused by Gibb effect, the use of very low-order interpolating Fourier series should be avoided.

APPENDIX

It has been shown that $\mathbf{M}(T)$, $\mathbf{P}(T)$, $\mathbf{U}(T)$, and $\mathbf{R}(T)$ play a critical role in sampled-data analysis of nonlinear circuits. In this Appendix, we give an efficient algorithm to compute these quantities. Using numerical Laplace inversion, we obtain $\mathbf{N}(T)$ directly [2]

$$\mathbf{N}(T) = -\frac{1}{T}\sum_{i=1}^{J} K_i \left[\mathbf{G} + \frac{z_i}{T}\mathbf{C}\right]^{-1}$$

where z_i and K_i are complex numbers readily available [2] and J is the order of integration. Numerical Laplace inversion is based on the Padé approximation of e^{st} at the origin. Its accuracy deteriorates if the displacement from the origin is large. This drawback can be minimized by employing a multistep algorithm. In doing so, the interval T is divided into N subintervals of equal width $h = T/N$. At $t = h$

$$\mathbf{N}(h) = -\frac{1}{h}\sum_{i=1}^{J} K_i \left[\mathbf{G} + \frac{z_i}{h}\mathbf{C}\right]^{-1}. \tag{35}$$

In the second step, the origin is shifted from $t = 0$ to $t = h$. Consequently, the response of the circuit at $t = h$ becomes the initial condition of the second step. It can be shown that

$$\mathbf{N}(2h) = \mathbf{Q}(h)\mathbf{N}(h)$$

where $\mathbf{Q}(h) = \mathbf{N}(h)\mathbf{C}$. Continuing this process we obtain

$$\mathbf{N}(nh + h) = \mathbf{Q}(nh)\mathbf{N}(h) \tag{36}$$

where $\mathbf{Q}(nh) = \mathbf{Q}^n(h)$. Similarly, $\mathbf{P}(h)$ is obtained from

$$\mathbf{P}(h) = -\frac{1}{h}\sum_{i=1}^{J} K_i \left(\mathbf{G} + \frac{z_i}{h}\mathbf{C}\right)^{-1} \left[\frac{\mathbf{g}}{\frac{z_i}{h} - j\omega_o}\right]. \tag{37}$$

Employing the multistep approach, at $t = nh$ we have

$$\mathbf{P}(nh + h) = \mathbf{P}(h)e^{j\omega_o nh} + \mathbf{Q}(h)\mathbf{P}(nh).$$

In a similar manner, one can show that

$$\mathbf{U}(nh + h) = \mathbf{M}(h)\mathbf{U}(nh) + \mathbf{U}(h) \tag{38}$$

and

$$\mathbf{R}(nh + h) = \mathbf{M}(h)\mathbf{R}(nh) + \mathbf{R}(h) + nh\mathbf{U}(h). \tag{39}$$

Notice that numerical Laplace inversion is performed in the first step only. Once $\mathbf{N}(h)$, $\mathbf{P}(h)$, $\mathbf{U}(h)$, and $\mathbf{R}(h)$ are available, $\mathbf{N}(T)$, $\mathbf{P}(T)$, $\mathbf{U}(T)$, and $\mathbf{R}(T)$ can be computed efficiently.

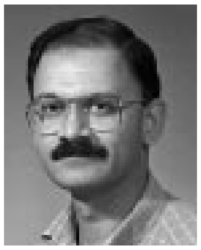
REFERENCES

[1] C. W. Gear, "Simultaneous numerical solution of differential-algebraic equations," *IEEE Trans. Circuit Theory*, vol. 18, pp. 89–95, Jan. 1971.
[2] J. Vlach and K. Singhal, *Computer Methods for Circuit Analysis and Design*, 2nd ed. New York: Van Nostrand Reinhold, 1994.
[3] *Hspice User's Manual*: Meta-Software, Inc., 1996.
[4] R. Kielkowski, *Inside Spice*. New York: McGraw-Hill, 1998.
[5] *Spectre User's Guide*: Cadence Design Systems, Inc., 1998.
[6] P. Wambacq and W. Sansen, *Distortion Analysis of Analog Integrated Circuits*. Boston, MA: Kluwer, 1998.
[7] A. Opal, "Sampled data simulation of linear and nonlinear circuits," *IEEE Trans. Computer-Aided Design Integrated Circuits Syst.*, vol. 15, pp. 295–307, Mar. 1996.
[8] M. Schetzen, *The Volterra and Wiener Theory of Nonlinear Systems*. New York: Wiley, 1981.
[9] W. Yu, S. Sen, and B. Leung, "Time varying Volterra series and its applications to the distortion analysis of a sampling mixer," in *Proc. 40th Midwest Symp. Circuits and Systems*, vol. 1, Sacramento, CA, Aug. 1997, pp. 245–248.
[10] R. L. Burden and J. D. Faires, *Numerical Analysis*, 6th ed. Pacific Grove, CA: Brooks/Cole, 1997.
[11] K. Singhal and J. Vlach, "Interpolation using the fast Fourier transform," *Proc. IEEE*, p. 1558, Dec. 1972.
[12] R. W. Hamming, *Numerical Methods for Scientists and Engineers*, 2nd ed. New York: Dover, 1986.
[13] H. R. Schwarz and J. Waldvogel, *Numerical Analysis—A Comprehensive Introduction*. Chichester, U.K.: Wiley, 1989.
[14] K. P. Prasad and P. Satyanarayana, "Fast interpolation algorithm using FFT," *Electron. Lett.*, vol. 22, pp. 185–187, Feb. 1986.
[15] K. Raahemifar and A. Opal, "Efficient transient solution of linear circuits to sinusoidal inputs," *IEEE Trans. Circuits Syst. I*, vol. 42, pp. 36–40, Jan. 1995.
[16] F. Yuan, "Statistical analysis of nonlinear current-mode circuits," in *Proc. Canadian Conf. Electrical and Computer Eng.*, Toronto, Canada, May 2001, paper no. 158.
[17] Z. W. Wang, J. J. Soltis, and W. C. Miller, "Improved approach to interpolation using the FFT," *Electron. Lett.*, vol. 28, pp. 2320–2322, Dec. 1992.
[18] F. J. Harris, "On the use of windows for harmonic analysis with the discrete Fourier transform," *Proc. IEEE*, vol. 66, no. 1, pp. 51–83, Jan. 1978.
[19] M. D. Macleod, "Fast interpolation by FFT with greatly increased accuracy," *Electron. Lett.*, vol. 29, pp. 1200–1201, June 1993.
[20] *Matlab: The Language of Technical Computing*: The MathWorks Inc., 1998. version 5.
[21] M. Ismail and T. Fiez, Eds., *Analog VLSI—Signal and Information Processing*. New York: McGraw-Hill, 1994, ch. 6.

Fei Yuan (S'94–M'99) received the B.E. degree from Shangdong University, Jinan, China, in 1985, and the M.A.Sc. and Ph.D. degrees from University of Waterloo, Waterloo, ON, Canada, in 1995 and 1999, respectively.

During 1985 through 1989, he was a Lecturer in the Department of Electrical Engineering, Changzhou Institute of Technology, Jiangsu, China. In 1989, he was a Visiting Professor at Humber College of Applied Arts and Technology, Toronto, Canada. During 1989 through 1994, he worked for Paton Controls Limited, Sarnia, Ontario, Canada as a Controls Engineer. Since July 1999, he has been with the Department of Electrical and Computer Engineering, Ryerson University, Toronto, ON, Canada, where he is an Assistant Professor. His current research interests include design and simulation of mixed analog–digital circuits for optical data communications.

Dr. Yuan received the award Excellence of Teaching from Changzhou Institute of Technology in 1988, the award Science and Technology Innovations from the Changzhou municipal government, Jiangsu, China, in 1988, and the Postgraduate Scholarship Award from the Natural Sciences and Engineering Research Council of Canada for 1997 and 1998.



Ajoy Opal (S'86–M'88) received the B.Tech. degree from the Indian Institute of Technology, New Delhi, India, in 1981, and the MA.Sc. and Ph.D. degrees from University of Waterloo, Waterloo, ON, Canada, in 1984 and 1987, respectively.

During 1989 through 1992, he worked for Bell-Northern Research in the area of analog circuit simulation. Since 1992, he has been an Associate Professor with the Department of Electrical and Computer Engineering, University of Waterloo. He works in the area of simulation of analog and mixed digital–analog circuits, including switched capacitors, switched currents, oversampled sigma–delta modulators, as well as circuit theory and filter design.