CMST 7(2) 91-109 (2001)

DOI:10.12921/cmst.2001.07.02.91-109

COMPUTATIONAL METHODS IN SCIENCE AND TECHNOLOGY 7(2), 91-109 (2001)

RECURSIVE CONVOLUTION ALGORITHMS FOR TIME-DOMAIN SIMULATION OF ELECTRONIC CIRCUITS

GRZEGORZ BLAKIEWICZ[1] AND WŁODZIMIERZ JANKE[2]

[1]*Techn. University of Gdańsk, Dept. of Electronics, Telecommunication and Informatics Narutowicza 11, 80-952 Gdańsk, Poland*

[2]*Techn University of Koszalin, Dept of Electronics Partyzantów 17, 75-411 Koszalin, Poland*

(Received 10 June, 2001)

Abstract. A family of semi-analytical recursive algorithms of convolution calculations as a convenient tool for electronic circuit simulation is described. The formulas defining these algorithms are presented and their numerical performance - accuracy, numerical complexity and stability are analyzed. The main purpose of this paper is to compare the recursive convolution algorithms with the known algorithms of differential equation integration in application to time-domain circuit simulation. In addition, simple examples of simulation are presented. The main advantage of the proposed approach results from the excellent stability performance of recursive convolution algorithms.

Keywords: time-domain circuit simulation, numerical algorithms, recursive convolution

I. INTRODUCTION

The standard time-domain description of electronic circuits is in terms of nonlinear differential equations and the simulation procedure contains two principal steps. In the first step, all time derivatives in differential equations are replaced by finite difference approximations according to selected numerical integration formulas. In the second step, the resulting set of non-linear algebraic equations is solved using Newton-Raphson or other algorithms. These steps are to be used repeatedly in the consecutive time points. The application of numerical integration formulas is equivalent to the replacement of each dynamic element or subcircuit by the time-discretized companion model corresponding to a selected integration scheme. It is known that the effective use of higher-order algorithms for numerical integration of differential equations is limited because of the stability problems, especially for "stiff' equation systems. As an alternative approach, a dynamic linear subcircuit may be represented by convolution relation and the companion model for each subcircuit may be derived from it.

The main purpose of the paper is to discuss the applicability of a family of recursive convolution algorithms to time-domain simulation of electronic circuits in comparison with traditional algorithms of an ordinary differential equation integration (ODEI). Electronic circuits under considerations are assumed to be one-dimensional, continuous-time, lumped, stationary, causal, and dynamic. The circuits are, in general, non-linear but it is assumed that they can be decomposed into linear dynamic subcircuits and non-linear resistive (inertialess) subcircuits. It can be easily proved that all lumped electronic circuits consisted of linear and non-linear resistors, capacitors, and inductors as well as transistors and other semiconductor devices fulfil this assumption (see Appendix A).

II. DESCRIPTION OF LINEAR SUBCIRCUITS

A linear, dynamic subcircuit is usually described in the *s*-domain by its transmittance matrix **H**(*s*)

$$\mathbf{Y}(s) = \mathbf{H}(s)\mathbf{X}(s) \tag{1}$$

where **Y**(*s*) and **X**(*s*) are the *s*-domain representations ofthe output and input signal vectors **y**(*t*) and **x**(*t*) respectively. Usually, the subcircuits to be represented by time-discretized macromodels are simple single-input single-output (SISO) blocks. Therefore in subsequent considerations *Y*(*s*), *X*(*s*), *H*(*s*),*y*(*t*) and *x*(*t*) are scalar functions.
It is assumed that the transmittance *H*(*s*) is a rational function:

$$H(s) = \frac{a_0 + a_1 s + \ldots + a_m s^m}{b_0 + b_1 s + \ldots + b_n s^n} = K_0 + \sum_{i=1}^{I_1} \sum_{j=1}^{\vartheta_i} \frac{K_{i,j}}{(s - \sigma_i)^j} + \sum_{i=I_1+1}^{I} \left[\frac{K_i}{s - \alpha_i} + \frac{K_i^*}{s - \alpha_i^*} \right] \tag{2}$$

where

$$n \geq m; \qquad \sum_{i=1}^{I_1} \vartheta_i + 2(I - I_1 + 1) = n. \tag{3}$$

The quantities α_i are complex numbers and K_0, σ_i, are real ones. K_i are real for i - 1,...,I$_1$, and complex for $i = I_1 + 1$, . . . , I. The transmittance in (2) represents a subcircuit with single or multiple, real or complex conjugate poles.

A linear electronic subcircuit composed of lumped devices can be directly represented by the transmittance in (2) [1]. In other cases, for example for a set of interconnects on a printed circuit board, the transmittance given by (2) is accepted as so called Pade approximation and several methods exist for finding such an approximation for the subsystem transmittance [2-7].

In the major part of further considerations, the transmittance *H*(*s*) with single poles is assumed (v_i = 1 for i = 1,2,... I_i). The multiple pole case is considered in the Appendix B.

The time-domain representation ofthe output *y*(*t*), according to the above assumptions, is a sum of *I* components

$$y(t) = \sum_{i=1}^{I} y_i(t) \tag{4}$$

where

$$y_0(t) = K_0 x(t) \tag{5}$$

$$\frac{\mathrm{d}y_i}{\mathrm{d}t} = \sigma_i y_i(t) + K_i x(t), \qquad \text{for} \qquad i = 1,2,\ldots I_1 \tag{6}$$

$$\begin{cases} y_i(t) = w_i(t) + w_i^*(t) \\ \dfrac{\mathrm{d}w_i(t)}{\mathrm{d}t} = \alpha_i w_i(t) + K_i x(t), \qquad \text{for} \quad i = I_1 + 1, \dots, I \\ \dfrac{\mathrm{d}w_i^*(t)}{\mathrm{d}t} = \alpha_i^* w_i^*(t) + K_i^* x(t) \end{cases} . \tag{7}$$

The initial state of the subcircuit is $y_i(0) = y_{i,0}$ for i = 1, 2, 3, ..., $I_1 w_i^*(0) = w_{i,0}^*$, and $w_i(0) = w_{i,0}$ for $i = I_1 + 1, \dots I$.

The alternative time - domain description of a causal subcircuit is a convolution integral

$$y(t) = \int_{-\infty}^{t} h(t - v) \cdot x(v) \cdot \mathrm{d}v \tag{8}$$

where the impulse response $h(t)$ of the subcircuit is the inverse Laplace transform of the transmittance $H(s)$

$$h(t) = \mathcal{L}^{-1}\{H(s)\} = K_0 \delta(t) + \sum_{i=1}^{I_1} K_i \mathrm{e}^{\alpha_i t} + \sum_{i=I_1+1}^{I} (K_i \mathrm{e}^{\alpha_i t} + K_i^* \mathrm{e}^{\alpha_i^* t}) \tag{9}$$

The components $y_i(t)$ of the output in (4) are expressed as the convolutions of $x(t)$ with the respective terms of the impulse response $h(t)$

$$y_i(t) = K_i \mathrm{e}^{\alpha_i t} \int_0^t x(\tau) \mathrm{e}^{-\alpha_i \tau} \mathrm{d}\tau + y_{i,0} \mathrm{e}^{\alpha_i t}, \qquad \text{for} \quad i = 1, 2, \dots, I_1 \tag{10}$$

$$\begin{cases} y_i(t) = w_i(t) + w_i^*(t) \\ w_i(t) = K_i \mathrm{e}^{\alpha_i t} \displaystyle\int_0^t x(\tau) \mathrm{e}^{-\alpha_i \tau} \mathrm{d}\tau + w_{i,0} \mathrm{e}^{\alpha_i t}, \qquad \text{for} \quad i = I_1 + 1, \dots, I \\ w_i^*(t) = K_i^* \mathrm{e}^{\alpha_i^* t} \displaystyle\int_0^t x(\tau) \mathrm{e}^{-\alpha_i^* \tau} \mathrm{d}\tau + w_{i,0}^* \mathrm{e}^{\alpha_i^* t} \end{cases} \tag{11}$$

The component $y_0(t)$ is given by (5) and the initial conditions are the same as above.

For the purpose of a numerical circuit simulation in the time domain, the equations (6), (7) or (10), (11) should be time discretized that results in the difference equations

$$\hat{u}_i[n] = \sum_{k=1}^{R_1} c_{k,i} \hat{u}_i[n-k] + K_i \sum_{l=0}^{R_2} d_{l,i} x[n-l], \tag{12}$$
$$n = 1, 2, 3, \dots, \quad i = 1, 2, \dots, L$$

where $\hat{u}_i[n]$ denotes an approximated (numerical) value of the output component sample $y_i[n]$ or $w_i[n]$, $w_i^*[n]$. The symbol $u_i[n]$ represents the accurate value taken at the n-th point of time.

In this paper, two groups of algorithms corresponding to Eqn. (12) are reviewed, the first based on the known ODEI formulas [1, 8-10] and the second - based on the recursive convolution formula (RCF) [11, 13, 14], with special attention paid to "semi-analytical" form of the recursive convolution algorithms (SARA) [12, 15-17],

III. ODEI ALGORITHMS

The known algorithms of numerical ODEI are briefly reviewed in this section from the point of view of applications to electronic circuit simulation in the time domain. The accuracy, numerical complexity, and stability of the algorithms are considered here. The applications of ODEI algorithms to the time-domain circuit simulation have been extensively discussed in the literature [1, 8, 9, 18],

Popular families of numerical integration algorithms are Adams-Bashforth, Adams-Moulton, and Gear algorithms [1,8], Each algorithm of a given order is characterised by specific formulas for coefficients $c_{k,i}$, $d_{l,i}$ in general equation (12). In the expressions for the coefficients presented below, the symbol ζ_i is used for the time normalised coefficient σ_i

$$\zeta_i = \begin{cases} -\sigma_i\Delta & \text{for} \quad i = 1, \dots I_1 \\ -\alpha_i\Delta & \text{for} \quad i = I_1 + 1 \dots I \end{cases} \tag{13}$$

where Δ is the length of a time step in numerical calculations assumed to be constant if not otherwise specified. In the rest of the section, the subscripts "i" corresponding to the number of the component in (4)-(7), (9)-(12) are omitted as the formulas for all components have the same general form given by (12).

Expressions for coefficients in (12) for ODEI algorithms of low order are given in Tables I—III. The Adams-Moulton algorithm (Table II) with $R_1 = R_2 = 1$ is usually known as the trapezoidal formula. The Gear algorithm (Table III) for R_1 - 1 (also known as implicit Euler algorithm) is the same as the respective Adams-Moulton algorithm.

Table I. The coefficients of Adams-Bashforth algorithms for ODEI ($R_1 = R_2$) [1, 8]

Order R	Coefficients: c_k, d_l
1	$c_1 = 1 - \zeta, \quad d_0 = \Delta$
2	$c_1 = 1 - \frac{2}{3}\zeta, \quad c_2 = \frac{\zeta}{2}, \quad d_0 = \frac{3}{2}\Delta, \quad d_1 = \frac{\Delta}{2}$
3	$c_{12} = 1 - \frac{23}{12}\zeta, \quad c_2 = \frac{16}{12}\zeta, \quad c_3 = 1 - \frac{5}{12}\zeta, \quad d_0 = \frac{23}{12}\Delta, \quad d_1 = \frac{16}{12}\Delta, \quad d_2 = \frac{5}{12}\Delta$

Table II. The coefficients ofAdams-Moulton algorithms for ODEI [1, 8]

Order R	R_1, R_2	Coefficients: c_k, d_l
1	$R_1 = 1, R_2 = 0$	$c_1 = \frac{1}{1+\zeta}$, $d_0 = \frac{K\Delta}{1+\zeta}$
2	$R_1 = 1, R_2 = 1$	$c_1 = \frac{2-\zeta}{2+\zeta}$, $d_0 = d_1 = \frac{K\Delta}{2+\zeta}$
3	$R_1 = 2, R_2 = 2$	$c_1 = \frac{12-18\zeta}{12+5\zeta}$, $c_2 = \frac{\zeta}{12+5\zeta}$, $d_0 = \frac{5K\Delta}{12+5\zeta}$, $d_1 = \frac{8K\Delta}{12+5\zeta}$, $d_2 = \frac{K\Delta}{12+5\zeta}$

Table III. The coefficients ofthe Gear algorithms for ODEI ($R_2 = 0$) [1, 8]

Order R	Coefficients: c_k, d_l
2	$c_1 = \frac{4}{3+2\zeta}$, $c_1 = \frac{1}{3+2\zeta}$, $d_0 = \frac{2K\Delta}{3+2\zeta}$
3	$c_1 = \frac{18}{11+6\zeta}$, $c_2 = \frac{-9}{11+6\zeta}$, $c_3 = \frac{2}{11+6\zeta}$, $d_0 = \frac{6K\Delta}{11+6\zeta}$
4	$c_1 = \frac{48}{25+12\zeta}$, $c_2 = \frac{-36}{25+12\zeta}$, $c_3 = \frac{16}{25+12\zeta}$, $c_4 = \frac{-3}{25+12\zeta}$, $d_0 = \frac{12K\Delta}{25+12\zeta}$

The numerical complexity of the algorithm is understood as a number of multiplications required for obtaining a single component ofthe output and it depends on the algorithm order R. Expressions for complexity of ODEI algorithms are given in Table IV, where $N = T/\Delta$ is the total number ofthe time steps over the whole time period T ofthe analysis.

In error estimation, only the truncation error [8, 9] (related to the truncation ofthe Taylor series) is taken into consideration. The maximum local truncation error ε_T of ODEI algorithms is

$$|\varepsilon_T|_{\max}^{\mathrm{ODEI}} = C_R \Delta^{R+1} \sup_{t\in(0,T)} |y^{(R+1)}(t)| \tag{14}$$

where R is the algorithm order. Coefficients C_R for specific algorithms are given in Table V.

The numerical stability of the algorithms depends on the value of ζ (see Eqn. 13) and is usually specified by the determination of stability regions in a complex ζ-plane. Stability regions for the presented algorithms are represented as dashed areas in Figs. 1-3, [1,8],

Table IV. The complexity of ODEI algorithms

Family of algorithms	Complexity
Adams-Basforth	$2RN$
Adams-Moulton	$(2R-1)N$, for $R>1$
Gear	$(R+1)N$

Table V. The values of the coefficients C_R in the formula (14) [1]

Order R	Adams-Bashforth	Adams-Moulton	Gear
1	1/2	1/2	1/2
2	5/12	1/12	2/9
3	3/8	1/24	3/22
4	251/720	19/720	12/125

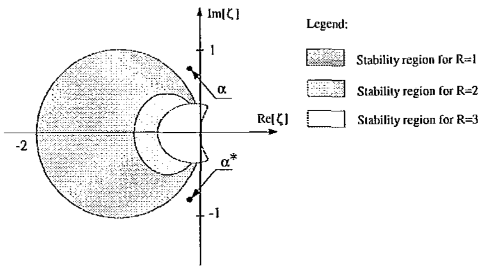

Im[ζ]
1
α
Re[ζ]
-2
α*
-1
Legend:
▨ Stability region for R=1
▨ Stability region for R=2
☐ Stability region for R=3

Fig. 1. Stability regions (dashed) of the Adams-Bashforth algorithms

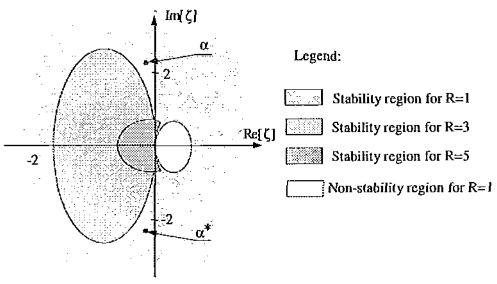

Im[ζ]
α
2
Re[ζ]
-2
-2
α*
Legend:
▨ Stability region for R=1
▨ Stability region for R=3
▨ Stability region for R=5
☐ Non-stability region for R=1

Fig. 2. Stability regions (dashed) of the Adams-Moulton algorithms

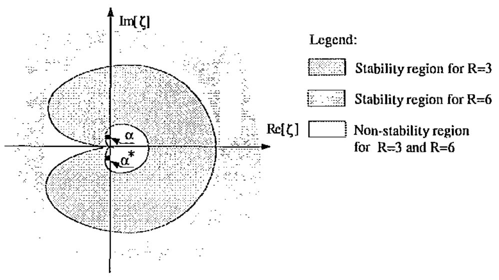

Im[ζ]
Re[ζ]
α
α*
Legend:
▨ Stability region for R=3
▨ Stability region for R=6
☐ Non-stability region for R=3 and R=6

Fig. 3. Stability regions (dashed) of the Gear algorithms

Adams-Bashforth and higher order Adams-Moulton algorithms cannot be effectively used for simulation of systems described by "stiff' differential equations because of the limitation of their stability regions [1, 8], The stability of Gear algorithms is better, but for higher order Gear algorithms, the left halfplane contains some restricted areas of instability (a larger area for higher order algorithms). The instability of Adams-Bashforth, Adams-Moulton (for $R> 2$) and Gear (for $R > 2$) algorithms is illustrated in Figs. 1-3, where the complex conjugate pole α, α^* pair" laying in the left half-plane represents a stable circuit, but the corresponding values $\zeta = \alpha\Delta$ and $\zeta^* = \alpha^*\Delta$ belong to the region of the algorithms instability. In addition, 2-nd order Gear algorithm, stable for constant time step length can become unstable for a variable step [8], (this remark applies to all considered algorithms of the order higher than 2).

In general, the potential accuracy of the higher order algorithms for ODEI in application to the time domain simulation of electronic circuits, and the possibility of the dynamic accommodation of time step length can be utilised only in a limited way [1, 8, 17, 18], because of the stability problems.

IV. RECURSIVE CONVOLUTION ALGORITHMS

A. General Remarks

The applicability of the difference formula (12) to the convolution-based circuit simulation is not obvious because the scheme of direct numerical calculations of the integral (8) differs from Eqn. (12). For example, the rectangular rule for the calculation of integral gives

$$y[n] = \Delta \sum_{m=1}^{n} x[m]\,h[n-m], \qquad \text{for } n = 1,2,\ldots,N. \tag{15}$$

In general, by applying R-th order numerical integration formula to Eqn. (8), one obtains

$$y[nR] = \sum_{k=R}^{n} \sum_{r=0}^{R} \rho_r h[n-kR+r]\,x[kR-r], \qquad \text{for } n = 1,\ldots N,\ R \geq 1,\ n \geq R. \tag{16}$$

The direct algorithms (15) and (16) are practically useless in circuit simulation because of its high numerical complexity (the number of multiplications is proportional to the square of the total number of samples N).

The recursive convolution algorithms, which have a much better numerical performance are applicable to a special form of function $h(t)$ expressed by (9) and can be used for simulation of all electronic circuits. In order to explain the idea of the recursive convolution formula (RCF) [11-17], the integral (10) is rearranged to obtain

$$y_i(t_n) = \mathrm{e}^{\sigma_i\Delta_n} y_i(t_n - \Delta_n) + K_i \mathrm{e}^{\sigma_i t_n} \int_{t_n-\Delta_n}^{t_n} x(\tau)\mathrm{e}^{-\sigma_i\tau}\mathrm{d}\tau. \tag{17}$$

The quantity Δ_n is the distance between the time points t_{n-1}, and t_n.

The application of a selected formula to the calculation of the integral in (17) leads to particular recursive convolution algorithms

$$y_i[n] = \Phi_{i,n} y_i[n-1] + K_i \sum_{l=0}^{R_2} q_{l,i} x[n-l] \tag{18}$$

where

$$\Phi_{i,n} = \exp(\sigma_i \Delta_n) . \tag{19}$$

The recursive algorithm (18) based on (17), known as recursive convolution formula (RCF) [11, 13, 14], may be treated as a special case of the difference equation (12) with $\Phi_{i,n} = c_{1,i}$ and is referred to as a single step recursive convolution algorithm. The quantity Δ_n is a parameter in formula (17) and is used as a time step length in the algorithm (18). Eqns. (17)-(19) are valid for real or complex values, but only the real pole case is considered here.

Two approaches to determining the coefficients $q_{l,i}$ in the algorithm (18) are mentioned in the literature. In the first, the integral in (17) is replaced by the sum of discrete terms according to commonly used integration schemes [11, 13, 14]. In the second, the $x(\tau)$ function under the integral in (17) is approximated by a polynomial:

$$x(\tau) = \sum_{r=0}^{R_2} A_r \tau^r \tag{20}$$

and next, the integration is performed analytically [12, 15-17]. The first group of algorithms is referred to as "common recursive algorithms" (CRA) and the second - as "semi-analytical recursive algorithms" (SARA). The comparison of the low order CRA and SARA algorithms [15] shows that SARA algorithms assure better accuracy than CRA at the same numerical effort, therefore only SARA algorithms are discussed in the rest of the paper.

B. Single Step SARA Algorithms

In this subsection, the single step SARA algorithms ($R_1 = 1$) with a uniform time discretization, i. e. with constant time step Δ for Δ_n in (17)-(19), are considered. The formulas for coefficients $q_{l,i}$ in the algorithms (18) of consecutive order are obtained after expressing A_r quantities in Eqn. (20) in terms of some last samples of $x(t)$ and next, by deriving analytically the integral in (17). The resulting expressions for coefficients q_l (the subscript "i" corresponding to the component number in the output (4) is omitted) are given in Table VI.

Calculation of a single output component sequence $\{y_i[n]\}$ associated with a single term in the circuit impulse response (9) is equivalent to solving the single, first order differential equation (6) or (7).

The numerical complexity of SARA algorithms is a function of the algorithm order R and the number of samples N

$$\mathrm{COM(SARA)} = (R+1)N. \tag{21}$$

In general, the order of single-step SARA algorithm given by (18) is equal to $R_2 + 1$.

The estimation of the local truncation error ε_T of SARA algorithms for a single term of the impulse response is given by the formula [15]

$$|\varepsilon_T|_{\max}^{\text{SARA}} = K|E_R(\zeta)|\Delta^{R+1} \sup_{t\in(0,T)} |x^{(R)}(t)| \,. \tag{22}$$

Contrary to the error estimate expressed by (14) for ODEI algorithms, the maximum truncation error of SARA depends on the respective derivatives of the input $x(.)$ rather than output $y(.)$. The terms E_R depend on the quantity ζ according to formulas given in Table VII and plots shown in Fig. 4 for real values of ζ.

Table VI. The coefficients of single step SARA algorithms, Eqn. (18)

Order $R = R_2 + 1$	q_i
1	$q_0 = \frac{\Delta}{\zeta}(1 - \Phi)$
2	$q_0 = \frac{\Delta}{\zeta^2}(-1 + \zeta + \Phi)$; $q_0 = \frac{\Delta}{\zeta^2}(1 - (1 + \zeta)\Phi)$
3	$q_0 = \frac{\Delta}{2\zeta^3}[2 - 3\zeta + 2\zeta^2 - (2 - \zeta)\Phi]$; $q_1 = \frac{\Delta}{\zeta^3}[-2(1 - \zeta) + (2 - \zeta^2)\Phi]$; $q_2 = \frac{\Delta}{2\zeta^3}[2 - \zeta - (2 + \zeta)\Phi]$
4	$q_0 = \frac{\Delta}{6\zeta^4}\left[-6\zeta + 7\zeta^2 - 6(1 - \zeta)^3 + (6 - 6\zeta + 2\zeta^2)\Phi\right]$; $q_1 = \frac{\Delta}{2\zeta^4}\left[6 - 10\zeta + 6\zeta^2 - [3 + (3 + 2\zeta)(1 - \zeta^2)]\Phi\right]$; $q_2 = \frac{\Delta}{2\zeta^4}\left[-6 + 8\zeta - 3\zeta^2 + (6 - 2\zeta - 2\zeta^2)\Phi\right]$; $q_3 = \frac{\Delta}{2\zeta^4}\left[6 - 6\zeta + 2\zeta^2 - (6 - \zeta^2)\Phi\right]$

The stability considerations of SARA algorithms are very simple. From (18), it is seen that the output $y[n]$ is bounded because for $x(t) = 0$ one obtains

$$\left|\frac{y_i[n]}{y_i[n-1]}\right| \doteq |\Phi_i| = |\exp(\sigma_i\Delta)| < 1, \qquad \text{for} \quad \sigma_i < 0. \tag{23}$$

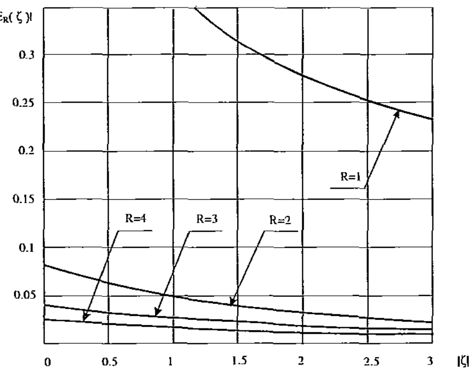


Fig. 4. Dependence of term E_R in Eqn. (22) on normalized quantity ζ

Table VII. Quantities E_R in Eqn. (22) defining maximum truncation error of single step SARA algorithms

R	$\lvert E_R(\zeta)\rvert$	$\lim\limits_{\zeta\to 0}\lvert E_R(\zeta)\rvert$
1	$\lvert\Phi - 1 + \zeta\rvert / \zeta^2$	1/2
2	$\lvert[2 - \zeta - (2+\zeta)\Phi]/(2\zeta^3)\rvert$	1/12
3	$\lvert[2(-3+3\zeta-\zeta^2)+(6-\zeta^2)\Phi]/(6\zeta^5)\rvert$	1/24
4	$\lvert[12-18\zeta+11\zeta^2-3\zeta^3-(12-6\zeta-\zeta^2+\zeta^3)\Phi]/(12\zeta^5)\rvert$	19/720

The same observation is valid for complex exponent instead α_i of σ_i, if Re(α_i) < 0. It is evident that single step SARA algorithms are A-stable. They are stable for a stable system regardless of the value of a time step length and the algorithm order. This feature of algorithms is especially valuable in applications to simulation of stiff systems (with a high spread of time constant values).

C. Multi-Step SARA Algorithms

Multi-step semi-analytical recursive convolution algorithms (MSARA) [17] may be thought of as a generalisation of the single-step algorithms and are derived by fitting the coefficients c_k,

d_l in (12) to achieve the exact values of the output $y[n]$ assuming the input $x(t)$ to be a polynomial of the order $R - 1$

$$x(t) = p_{R-1}(t) = \sum_{r=0}^{R-1} A_r t^r. \tag{24}$$

The order R of a multi-step algorithm depends on the numbers R_1 and R_2 in (12)

$$R = R_1 + R_2 \tag{25}$$

and various combinations of R_1 and R_2 are possible. The single-step algorithm is a special case with $R_1 = 1$.

Table VIII. The coefficients of MSARA algorithms ($R_2 = 0$)

$R = R_1 + 1$	C_k	d_l
2	$c_1 = [-1 + (1 + 2\zeta)\,\Phi^2]]/\beta^2$ $c_2 = \Phi[1 - (1 + \zeta)\,\Phi]/\beta^2$	$d_0 = -\Delta(1 - \Phi)^2/\beta_2$
3	$c_1 = [4 - 9\Phi + (5 + 6\zeta)\,\Phi^3]/\beta_3$ $c_2 = [-1 + 9\Phi^2 - (8 + 6\zeta)\,\Phi^3]/\beta_3$ $c_3 = \Phi[1 - 4\Phi + (3 + 2\zeta)\,\Phi^2]/\beta_3$	$d_0 = 2\Delta(1 - \Phi)^3/\beta_3$
4	$c_1 = 2[9 - 32\Phi + 36\Phi^2 - (13 + 12\zeta)\,\Phi]/\beta_4$ $c_2 = 3[-3 + 8\Phi - 24\Phi^3 + (19 + 12\zeta)\,\Phi^4]/\beta_4$ $c_3 = 2[1 - 12\Phi^2 + 32\Phi^3 - (21 + 12\zeta)\,\Phi^4]/\beta_4$ $c_4 = \Phi[-2 + 9\Phi - 18\Phi^2 + (11 + 6\zeta)\,\Phi^3]/\beta_4$	$d_0 = 6\Delta(1 - \Phi)^4/\beta_4$

$$\beta_2 = -1 - \sigma + (1 + 2\sigma)\,\Phi$$

$$\beta_3 = 3 + 2\sigma - (8 + 6\sigma)\,\Phi + (5 + 6\sigma)\,\Phi^2$$

$$\beta_4 = 17 - 66\Phi + 93\Phi^2 - 50\Phi^3 + (1 - \sigma)(-6 + 24\Phi - 36\Phi^2 + 24\Phi^3)$$

Formulas for coefficients c_k, d_l in (12) for the assumed order R are derived in the following steps:

a) express the coefficients A_r in a polynomial $p_{R-1}(t)$, given by (20), in terms of the R last samples of $x(t)$ spaced by time step Δ;

b) substitute the polynomial $p_{R-1}(\tau)$ for $x(\tau)$ in (10) (or (11) for the complex conjugate case) and evaluate analytically the integral to achieve $y(t)$ or $w(t)$, $w^*(t)$;

c) substitute the appropriate samples of $u[n-k]$ for $y(t)|_{t=(n-k)\Delta}$ or $w(t)|_{t=(n-k)\Delta}$; for $k = 1, 2, \ldots R_1$ and the samples $x[n-l] = p_{R-1}(t)|_{t-(n-l)\Delta}$ for $1 = 1, 2, \ldots R_2$ for $x[n-1]$ in Eqn. (12);

d) solve the obtained linear system of equations ($n = 1, 2, \ldots, R$) for coefficients c_k, d_l.

The examples of resulting formulas for coefficients c_k d_l. of the multi-step algorithms of the order 2,..., 4 are presented in Table VIII. The case $R = 1$ ($R_1 = 1$, $R_2 = 0$) corresponds to single-step, first order algorithm (see Table VI).

Table IX. Quantities $F_R(\xi)$ in Eqn. (26) defining maximum truncation error of MSARA algorithms ($R_2 = 0$)

R	$F_R(\zeta)$	$\lim\limits_{\zeta \to 0} \lvert F_R(\zeta) \rvert$
2	$\dfrac{4\Phi - (3+2\zeta)\Phi^2 - 1}{2\zeta[1+\zeta-(1+2\zeta)\Phi]}$	2/9
3	$\dfrac{(11+6\zeta)\Phi^3 - 18\Phi^2 + 9\Phi - 2}{3\zeta[(5+6\zeta)\Phi^2 - (8+6\zeta)\Phi^2 + 3 + 2\zeta]}$	3/22
4	$\dfrac{-(25+12\zeta)\Phi^4 + 48\Phi^3 - 36\Phi^2 + 16\Phi - 3}{2\zeta[-(26+24\zeta)\Phi^3 + (57+36\zeta)\Phi^2 - (42+24\zeta)\Phi + 11 + 6\zeta]}$	12/125

The numerical complexity of MSARA algorithms is the same as for the single-step algorithms (see Eqn. 21). The local truncation error is estimated for an input $x(t)$ assumed to be a polynomial of R-th order (higher by one than the order of the polynomial for which the algorithm is derived). The resulting expression for the maximum local truncation error for a single l-th term of the output is:

$$|\varepsilon_T|_{\max}^{\mathrm{SARA}} = K \cdot \Delta^{R+1} \cdot F_R(\zeta) \cdot \sup_{t \in (0,T)} |x^{(R)}(t)| \,. \tag{26}$$

The formulas for function $F_R(\xi)$ and its asymptotic value for $\xi \to 0$ are given in Table IX. The function $|F_R(\xi)|$ for real values of ξ is plotted in Fig. 5 for several values of R.

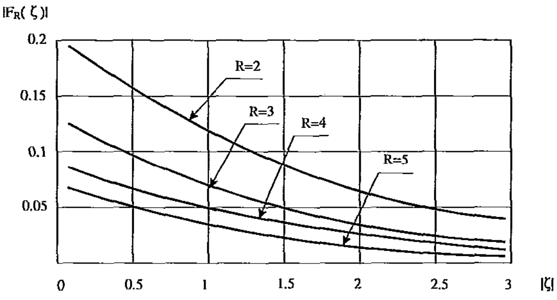


Fig. 5. Dependence of F_R on the value ζ

An analysis of MSARA algorithms stability is more involved than of the single step ones. The general stability properties have not been investigated but the calculations based on several variants of MSARA give stable results.

D. Comparison of ODEI and SARA Algorithms

The comparison of the main properties of the single-step SARA and ODEI algorithms, considered as two approaches to numerical simulation of electronic circuits in the time domain shows that both groups of algorithms have similar accuracy. It can be confirmed by comparing (14) with (22), the data given in Tables V, VII and plots in Figs. 4 and 5. It can be noticed that for $\zeta \to 0$ and when the condition

$$\sup_{t\in(0,T)} |y^{(R+1)}(t)| \cong K \sup_{t\in(0,T)} |x^{(R)}(t)|$$

is fulfilled (very common situation in numerical simulation of electronic circuits), values of error estimates $|\varepsilon_T|_{\max}^{\mathrm{ODEI}}$ and $|\varepsilon_T|_{\max}^{\mathrm{SARA}}$ are very close.

The main advantage of the single-step recursive convolution algorithms is their numerical stability, independent of the algorithm order and the length of a time step. The multi-step SARA algorithms are less efficient than the single-step ones. It is due to its lower accuracy. This is seen by comparing (14), (27) and the asymptotic values of $F_R(\zeta)$. An efficiency of the single step SARA algorithms can be additionally evaluated by examples of numerical calculations given in the next section.

V. EXAMPLES

The examples presented in this section explain the idea of SARA algorithm application to circuit simulation and show the advantages of such algorithms over ODEI algorithms.

A linear circuit shown in Fig. 6 has been simulated with 2-nd order algorithms. In the first step, the circuit transmittances H_1 (s) and H_2(s) are calculated:

$$H_{\mathrm{I}}(s) = \frac{V_{\mathrm{I}}(s)}{V_e(s)}, \qquad H_2(s) = \frac{V_{\mathrm{II}}(s)}{V_e(s)} \tag{27}$$

$$V_{\mathrm{I}}(s) = \mathcal{L}\{v_{\mathrm{I}}(t)\},\ V_{\mathrm{II}}(s) = \mathcal{L}\{v_{\mathrm{II}}(t)\},\ V_e(s) = \mathcal{L}\{e(t)\}.$$

For example, the first transmittance is of the form:

$$H_1(s) = \frac{R_2C_2s + 1}{R_1C_1R_2C_2s^2 + (R_1C_1 + R_1C_2 + R_2C_2)s + 1}. \tag{28}$$

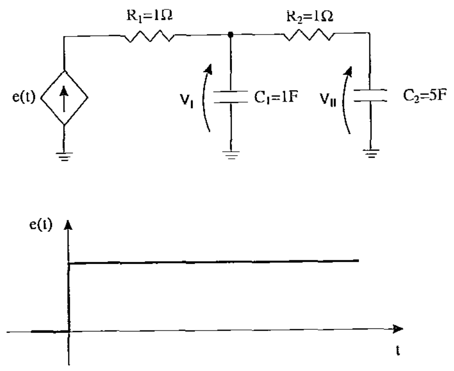


Fig. 6. Exemplary linear circuit

The transmittance (28) after substitutions of values $R_1 = 1$, $R_2 = 1$, $C_1 = 1$, $C_2 = 5$ is decomposed into simple terms

$$H_1(s) = \frac{1 + 5s}{1 + 11s + 5s^2} = \frac{0.9478}{s + 2.105} + \frac{0.0522}{s + 0.095} = \frac{K_1}{s - \sigma_1} + \frac{K_2}{s - \sigma_2}. \tag{29}$$

To calculate variable $v_l(t)$, a convolution should be evaluated

$$v_I(t) = h(.) * e(.), \qquad \text{where} \qquad h(t) = K_1 e^{\sigma_1 t} + K_2 e^{\sigma_2 t}. \tag{30}$$

The second order SARA algorithm is applied to the time discretization of the convolution (30)

$$\begin{aligned} v_1[n] &= v_{1,1}[n] + v_{1,2}[n], \qquad v_1[n] = v_1(t)|_{t=nT/N} \\ v_{1,1}[n] &= \Phi_{1,1}v_{1,1}[n-1] + q_{0,1}e[n] + q_{1,1}e[n-1] \\ v_{1,2}[n] &= \Phi_{1,2}v_{1,2}[n-1] + q_{0,2}e[n] + q_{1,2}e[n-1] \end{aligned} \tag{31}$$

The coefficients of (31) are calculated according to Table VI. The results of numerical simulation of the circuit, obtained with the above SARA algorithm and with ODEI algorithm of the same order are compared in Fig. 7 with the exact waveforms derived analytically. The results of calculations with SARA algorithms (Fig. 7a) and results obtained analytically are indistinguishable. In Fig. 7b one can observe some differences between numerical results and exact curves.

The transient simulation of certain electric non-linear circuits may cause troubles and numerical results of simulation may differ from the exact ones. Among the sources of these errors are the instability of integration methods and low precision of results at each time step. The circuit chosen for presentation of such difficulties is a stable, highly selective amplifier (Fig. 8) excited by a harmonic voltage. The simulation results presented in Fig. 9, obtained with ODEI algorithms: 3-rd order Gear (1) and 4-th order Gear (2), are compared with analytical solution (3). Both numerical results (1) and (2) are erroneous. According to the first solution, the circuit is highly damped, whereas the second solution shows that the circuit is unstable. For the assumed conditions of calculations ($\Delta = 1s$), the 3-rd order Gear algorithm is stable, but its accuracy is too low to assure correct results. On the other hand, the 4-th order Gear algorithm may potentially assure higher accuracy, but it is unstable.

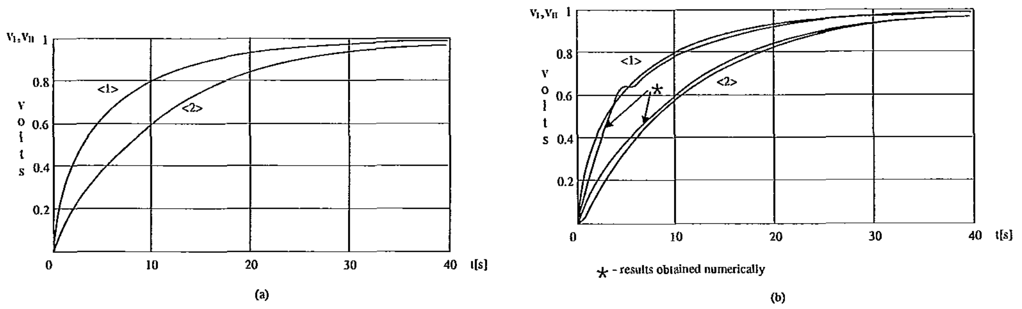


Fig. 7. Results of numerical simulations of circuit given in Fig. 6: a) 2-nd order SARA algorithms, b) trapezoidal algorithm, <1> - $v_I(t)$, <2> - $v_{II}(t)$

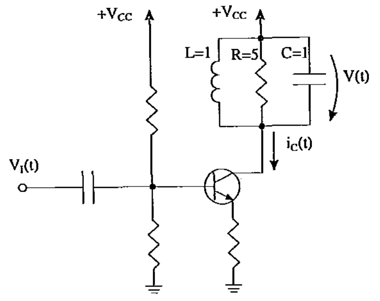
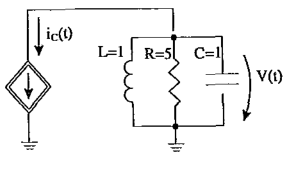
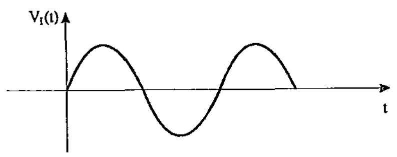


Fig. 8. Highly selective amplifier





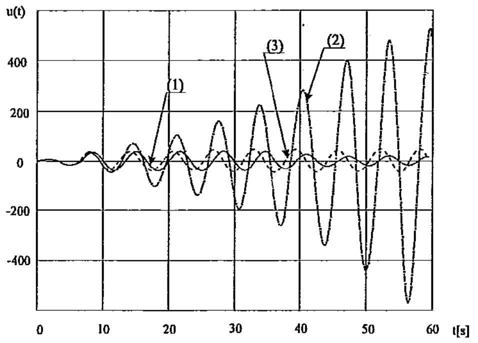


Fig. 9. Results of numerical simulation of the circuit shown in Fig. 8 obtained with ODEI algorithms

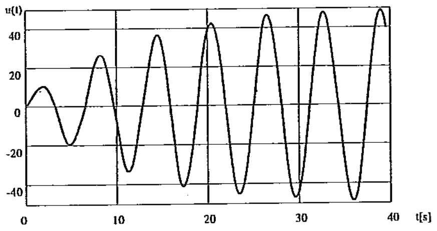


Fig. 10. Results of numerical simulation of circuit shown in Fig. 8 obtained with SARA algorithms

The circuit has been simulated with SARA algorithms taking the same time step and the same algorithm orders. The results of simulations are given in Fig. 10 (take notice of different scales of abscissas in Figs. 9 and 10). The numerical solutions and the exact one are hardly distinguishable. It is due to a higher accuracy of SARA algorithms than of Gear algorithms.

VI. CONCLUSIONS

The convolution relation is a description of dynamic linear subcircuits of electronic circuit, alternative to ordinary differential equation. Consequently, the algorithms for time-discretization of convolution may replace the algorithms for numerical integration of differential equations in application to the time-domain circuit simulation. In the paper, a family of semi-analytical recursive algorithms (SARA) for convolution discretization has been described and compared with widely known algorithms for ordinary differential equation integration (ODEI), namely: Adams-Moulton, Adams-Bashforth and Gear algorithms, from point of view of the applicability to time-domain circuit simulation. The numerical complexity, local truncation errors and stability performance of the algorithms have been considered. It is shown that the algorithms of respective order belonging to both families (SARA and ODEI) have similar accuracy and numerical complexity. A subtle difference concerning algorithm accuracy is in that the estimation of the local truncation error of SARA algorithms is expressed by a time derivative of the input rather than the output, as in the case of ODEI algorithms.

The principal advantage of the single-step semi-analytical recursive convolution algorithms is their *A*-stability: they are numerically stable (for a stable subcircuit) regardless of the algorithm order and the length of the discretization step. As a consequence, the proposed SARA algorithms are highly effective in simulation of circuits described by "stiff' systems - with widely separated time constants.

The applicability of the semi-analytical recursive convolution algorithms to a circuit simulation is confirmed by examples presented in Section V. The *A*-stability of SARA algorithms makes it possible to accommodate the algorithm order and time-step length to the demands of specific applications. As a consequence, one can easily obtain a satisfactory accuracy of simulation at low numerical effort that in some cases is impossible with ODEI algorithms.

Apart from constant-step SARA's presented in the paper, it is possible to elaborate the variable-step algorithms. The family of periodically-nonuniform step algorithms [16] may serve as an example. For the effective use of recursive convolution algorithms in the time-domain cir cuit simulation, the additional procedur es for cir cuit partitioning into linear dynamic subcircuits and resistive non-linear subcircuits should be developed. New, variable-step SARA algorithms should be elaborated including the procedure of the time-step length accommodation to the desired accuracy.

APPENDIX A

Decomposition of elementary non-linear subcircuits to linear inertial and non-linear resistive subcircuits

The decomposition of an electronic circuit into linear dynamic and non-linear resistive subcircuits can not be done directly if the circuit contains non-linear capacitors or inductors (or non-linear capacitances inside the models of semiconductor devices). Non-linear capacitor (or inductor) can be replaced by proper equivalent circuit as shown in Fig. A. 1.

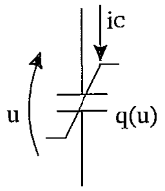


Fig. A. 1. Nonlinear capacitance

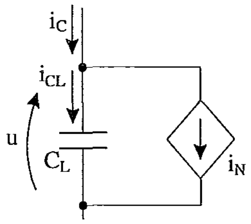


Fig. A.2. An equivalent circuit of nonlinear capacitance

A non-linear capacitance is represented, in general, by non-linear dependence $q = f(u)$ of the electric charge on the applied voltage (Fig. A. 1). The current $i_C(t)$ flowing through this device is:

$$i_C(t) = \frac{\mathrm{d}q(t)}{\mathrm{d}t} = \frac{\mathrm{d}q}{\mathrm{d}u}\,\frac{\mathrm{d}u(t)}{\mathrm{d}t} = C_d(u)\,\frac{\mathrm{d}u(t)}{\mathrm{d}t} \tag{A.1}$$

where $C_d(u)$ is a voltage-dependent differential capacitance. The Eqn. (A.l) can be rearranged as follows:

$$i_C(t) = \frac{\mathrm{d}q}{\mathrm{d}u}\frac{\mathrm{d}u(t)}{\mathrm{d}t} + C_L\frac{\mathrm{d}u(t)}{\mathrm{d}t} - C_L\frac{\mathrm{d}u(t)}{\mathrm{d}t} = i_{CL}(t) + i_N(t) \tag{A.2}$$

where:

$$i_{CL}(t) = C_L\frac{\mathrm{d}u(t)}{\mathrm{d}t} \tag{A3}$$

$$i_N(t) = i_{CL}(t)\left(\frac{1}{C_L}\frac{\mathrm{d}q}{\mathrm{d}u} - 1\right) \tag{A4}$$

Here, the C_L denotes an arbitrarily chosen value of an artificial, linear capacitance and $i_N(t)$ represents the non-linear controlled current source. The equivalent circuit of non-linear capacitor is shown in Fig. A.2. Based on the presented decomposition, all electronic circuits containing lumped devices (linear or non-linear) can be represented by linear dynamic subcircuits and resistive non-linear subcircuits.

APPENDIX B

Convolution calculation for a multiple pole

In Sec. IV only a single pole case in the system transmittance (2) is considered. Equations for SARA and MSARA algorithms coefficients given in Tables VI, VIII can be indirectly used for systems with multiple poles. The multiple pole in the transmittance corresponds to the multiple convolution in the time domain, therefore the output $y(t)$ is

$$y(t) = x(.) * h_e(.) * h_e(.) * h_e(.) * \ldots * h_e(.) \tag{Bl}$$

where the elementary impulse response $h_e(t)$ corresponds to a single-pole transmittance. From Eqn. (B.1) it is seen that $y(t)$ can be calculated by applying recursively an algorithm for a single pole case. This method of calculations is conceptually simple but numerically complex and ill conditioned.

As an alternative approach, a new algorithms can be derived. For a pole of υ-th multiplicity a single component $h_i(t)$ in the system impulse response is

$$h_i(t) = (K_{i,0} + K_{i,1}t + K_{i,2}t^2 + \ldots + K_{i,\vartheta_i-1}t^{\vartheta_i-1})\mathrm{e}^{\sigma_i t}, \qquad i = 1,2,\ldots I_1. \tag{B2}$$

In this case, the recursive convolution formula (17) takes the form:

$$\begin{aligned} y_i(t_n) &= \mathrm{e}^{\sigma_i\Delta_n}y_i(t_n-\Delta_n) + \int_{t_n-\Delta_n}^{t_n} x(\tau)h_i(t_n-\tau)\,\mathrm{d}\tau = \mathrm{e}^{\sigma_i\Delta_n}y_i(t_n-\Delta_n) + \\ &+ \int_{t_n-\Delta_n}^{t_n}\left[\sum_{r=0}^{R_2}\tau^r\right]\left[K_{i,0} + K_{i,1}(t_n-\tau) + K_{i,2}(t_n-\tau)^2 + \ldots + K_{i,\vartheta_i-1}(t_n-\tau)^{\vartheta_i-1}\right]e^{\sigma_i(t-\tau)}\,\mathrm{d}\tau. \end{aligned} \tag{B.3}$$

After introducing an auxiliary polynomial:

$$p(\tau, t_n) = \left[\sum_{r=0}^{R_2} \tau^r\right]\left[K_{i,0} + K_{i,1}(t_n - \tau) + K_{i,2}(t_n - \tau)^2 + \ldots + K_{i,\vartheta_i - 1}(t_n - \tau)^{\vartheta_i - 1}\right] \tag{B.4}$$

the recursive formula is

$$y_i(t_n) = e^{\sigma_i \Delta_n} y_i(t_n - \Delta_n) + \int_{t_n - \Delta_n}^{t_n} p(\tau, t_n)\, \tilde{h}_i(t_n - \tau)\mathrm{d}\tau \tag{B.5}$$

where $\tilde{h}_i = e^{\sigma_i t}$. The next steps of SARA algorithm coefficient derivation are the same as described in Sec. IV.A-IV.C, with the substitution

$$x(\tau) := p(\tau, t_n) \tag{B.6}$$

Acknowledgement

This work has been supported by Grant No. 8 T11B 037 16 ofthe Polish State Committee for Scientific Research.

References

[1] L. O. Chua and P. M. Lin, *Computer-aided analysis ofelectronic circuits,* Prentice Hall (1975).

[2] L. T. Pillage and R. A. Rohrer, IEEE Trans, on Computer Aided Design, 9, 352 (1990).

[3] C. L. Ratzlaffand L. T. Pillage, RICE, IEEE Trans, on Computer Aided Design, **13,** 763 (1994).

[4] T. Pillage and R. A. Rohrer, IEEE Circuits & Devices Magazine, 12 (1994).

[5] F. Y. Chang, IEEE Trans, on Computers, *Packaging and Manufacturing Technology — Part B: Advanced Packaging,* 17, 3 (1994).

[6] M. Celik, O. Ocali, M. A. Tan, A. Atalar, IEEE Trans, on Circuits and Systems - *1: Fundamental Theory and Applications,* 42, 6 (1995).

[7] F. Y. Chang, IEEE Trans, on Circuit and Systems - *P. Fundamental Theory and Application,* 39, 180(1992).

[8] K. G. Nichols et al., *IEE Proceedings - Circuits Devices Systems.* **141(4),** 242 (1994).

[9] N. Mohan et al., Proc. IEEE, **82(8),** 1287 (1994).

[10] C. W. Gear, IEEE Trans. Circuit Theory, **18,** 89 (1971).

[11] S. Lin and E. S. Kuh, IEEE Trans, on Circuits and Systems - *P. Fundamental Theory and Applications,* 39, 879 (1992).

[12] T. V. Nguyen, IEEE Trans, on Computer-Aided Design oflntegrated Circuits and Systems, 13, 1301 (1994).

[13] W. Janke and J. Zarębski, ISCAS, 2377 (1990).

[14] A. Samlyen and A. Dabuleanu, IEEE Trans. Power App. Sys., **94,** 561 (1975).

[15] W. Janke and G. Blakiewicz, IEE Proc. - *Circuits Devices Sys.,* **142,** 125 (1995).

[16] G. Blakiewicz and W. Janke, Proc ofECCTD'95, 2, 851 (1995).

[17] G. Blakiewicz and W. Janke, Proc. ofThe European Conference on Circuit Theory and Design, 3, 1452 (1997).

[18] B. Lindberg, BIT **14,** 430 (1974).

[19] W. Janke and W. Pietrenko, Proc. ofThird IEEE Conference on Electronics, Circuits and Systems (ICECS'96), 880 (1996).