View metadata, citation and similar papers at core.ac.uk
brought to you by CORE
provided by Memorial University Research Repository

IEEE Int. Symposium on Industrial Electronics (ISIE 95), Athens, Greece, 10-14 July 1995, pp.445-450.
Copyright © 1995 IEEE (DOI 10.1109/ISIE.1995.497038).

Improving Performance of Parameter Extractors Through Symbolic Simulation

W.M. Zuberek

Department of Computer Science
Memorial University of Newfoundland
St. John's, Canada A1C-5S7
e-mail: wlodek@cs.mun.ca

A. Konczykowska

Centre National d'Etudes des Télécommunications
Laboratoire de Bagneux
92220 Bagneux, France
e-mail: aga@bagneux.cnet.fr

Abstract

In parameter extraction programs, the performance of repeated analyses of linear (or linearized) circuits can be significantly improved by representing the dependence of circuit responses on some parameters in a symbolic form. This symbolic form can then be evaluated very efficiently for different sets of parameter values. An intergrated numerical–symbolic parameter extraction program, called FIT–S, has been developed in which all linear circuit analyses can be performed using symbolic or numerical approach. A comparisons of execution times is presented for extraction of a submicron HEMT's parameters.

KEYWORDS: parameter extraction, symbolic simulation, performance improvements.

1. INTRODUCTION

Verifying designs through simulation has become an indispensable part of IC design process. However, increasing complexity of IC circuits continuously creates demand for new and more efficient methods for analysis of analog circuits. In some applications, symbolic circuit analysis can significantly improve the performance of computer–aided circuit analysis tools. Parameter extraction is an example of such applications.

Because existing device models use large sets of parameters, the determination of values of these parameters becomes a nontrivial task if device characteristics are to be represented accurately in the full range of operating conditions. Usually these model parameters cannot be determined by direct measurements because of device nonlinearities. Different approaches to parameter extraction have been developed which include general or specialized, and direct or iterative extraction methods. Specialized methods extract some subsets of model parameters, for example, model resistances, or capacitances, or DC parameters only, while general methods determine all parameters of the model. Direct extraction methods approximate model equations by linear functions and determine the values of parameters graphically or by solving linearized equations, while iterative methods fit the model responses to a set of measured characteristics by minimizing an objective function that quantitatively characterizes the fit. Sometimes a mixed approach is used in which some parameters are extracted using a direct method, and the remaining ones by an iterative procedure.

Iterative extraction of model parameters can be regarded as an optimization process [1, 2, 3, 4] which minimizes the (total) differences between a set of measurement data and the corresponding circuit responses by adjusting the values of model parameters (which are optimization variables). The result of this optimization determines such values of model parameters for which the circuit responses are 'as close as possible' to the measurement data (in the sense of the error function used).

Simulation–based parameter extraction [5] uses a circuit simulator rather than an explicit set of model equations to obtain the circuit responses during the extraction process. A simple advantage of such an approach is that the extractor can use many capabilities of the circuit simulator, so all packaging and mounting parasitics can easily be taken into account during extraction, and also the extraction can use many types of measurement data, including noise, distortion, etc. However, iterative extraction process often requires numerous analyses of a circuit with the same topology, especially when numerous parameters are extracted from large sets of measurement data. If a significant part of the measurement data corresponds to linear analyses of the circuit (as is usually the case for microwave applications), symbolic simulation can be uses to eliminate repetitive solutions of circuit equations. For linear analyses, the dependence of circuit responses on some variables can be derived as a symbolic function, and if the analyzed circuit is rather simple (which is typical for parameter extraction), the symbolic functions are reasonably complex and usually exact.

A practical implementation of a parameter extractor with numerical and symbolic simulation was obtained by designing an interface from an existing symbolic simulator (SYBILIN [6]) to a simulation–based parameter extractor (the FIT program). This integrated numerical–symbolic simulation is used in a simulation–based data–driven parameter extraction program called FIT–S [7].

The paper briefly describes the basic characteristics of numerical and symbolic simulations, discusses the specifics of symbolic simulation in the context of parameter extraction, and compares the performance of the proposed integrated numerical–symbolic approach with a numerical simulation for a typical example of parameter extraction.

2. NUMERICAL AND SYMBOLIC SIMULATION

Characteristic features of popular 'third–generation' (numerical) circuit simulators [8] include modified nodal analysis and Newton–Raphson iteration to solve the system of simultaneous nonlinear algebraic equations which describe the balance of currents at the nodes of the network in terms of node voltages (and some branch currents) X [9, 10]:

$$F(X) = 0$$

If the solution is denoted by X^*, the Newton–Raphson iteration solves the original system of nonlinear equations through a sequence of linear approximations to the nonlinear function $F(X)$ at points $X^{(j)}$, $j = 1, 2, ...$

$$F(X^{(j)}) + G(X^{(j)})(X^* - X^{(j)}) \approx 0$$

where $G(X^{(j)})$ is the Jacobian of F with respect to X evaluated at $X^{(j)}$. The $(j+1)$–st approximation to the solution X^* is obtained by solving a system of simultaneous linear equations with respect to the correction $\Delta^{(j)}$

$$G(X^{(j)})\Delta^{(j)} = -F(X^{(j)})$$

with $X^{(j+1)} = X^{(j)} + \Delta^{(j)}$. The iteration terminates when $\Delta^{(j)}$ is sufficiently small.

This basic scheme is used in the DC operating point, DC transfer curve, and even time–domain analysis; in the last case, the dependence upon time is eliminated by approximating the differential equations by difference equations [10, 11]. Only frequency–domain (small–signal) analyses are significantly different because they require (for each frequency) a solution of a system of simultaneous linear equations in the complex domain; this is often done by separating the real and imaginary parts of coefficients and variables, and solving a twice as large system of linear equations in the real domain.

The principle of symbolic simulation [12, 13] is to derive analytic (or symbolic) network functions using (some of) circuit parameters as variables in the derived functions. Circuit responses are obtained very efficiently by evaluations of these symbolic functions for different values of circuit parameters (i.e., variables).

For linear, lumped and stationary circuits, the transfer functions $\mathcal{H}(s)$ of two–port networks are in the form of rational functions of the complex frequency s:

$$H(s) \;=\; \frac{\mathcal{F}_j(s)}{\mathcal{F}_k(s)}$$

in which the numerator $\mathcal{F}_j(x)$ and the denominator $\mathcal{F}_k(x)$ are characteristic polynomials of the two–port:

$$\mathcal{F}_i(s) \;=\; \sum_{\ell=0}^{n_i} s^\ell \mathcal{P}_{i\ell}(x_1, ..., x_m)$$

and the coefficients $\mathcal{P}(x_1, ..., x_m)$ are (nested or expanded) polynomial functions in symbolic elements $x_1, ..., x_m$. In the fully expanded form, the polynomial coefficients are in the 'sum–of–product' form:

$$\mathcal{P}(x_1, ..., x_m) \;=\; \sum_{i=1}^{p} C_i \prod_{j=1}^{r} x_{ij}$$

where C_i are real numbers, x_{ij} are circuit parameters, and p and r depend upon the topology of the circuit.

Circuit representations used to derive the symbolic functions are usually different than the ones used for numerical simulation. The algorithm used in the symbolic analyzer integrated with FIT uses the Coates flowgraph representation, in which variables corresponding to graph nodes are the same as those used in the modified nodal analysis. The integration of these two simulators was thus relatively straightforward.

An important aspect of integrated symbolic–numerical analysis is the representation of symbolic functions. FIT represents the (symbolic) characteristic polynomials in a modified 'sum–of–product' form. By extracting common factors and rearranging the terms, the coefficients $\mathcal{P}(x_1, ..., x_m)$ can be represented equivalently as

$$\mathcal{F}_i \;=\; s^{k_i}\mathcal{T}_i \sum_{j=0}^{n_i} s^j \mathcal{R}_{ij}$$

where each $\mathcal{T}_i$ is a product of a constant C_i and (some) symbols x_{ik}, $k = 1, ..., m_i$

$$\mathcal{T}_i \;=\; C_i \prod_{k=1}^{m_i} x_{ik}$$

and each $\mathcal{R}_{ij}$, $j = 0, 1, ..., n_i$, is a sum of products

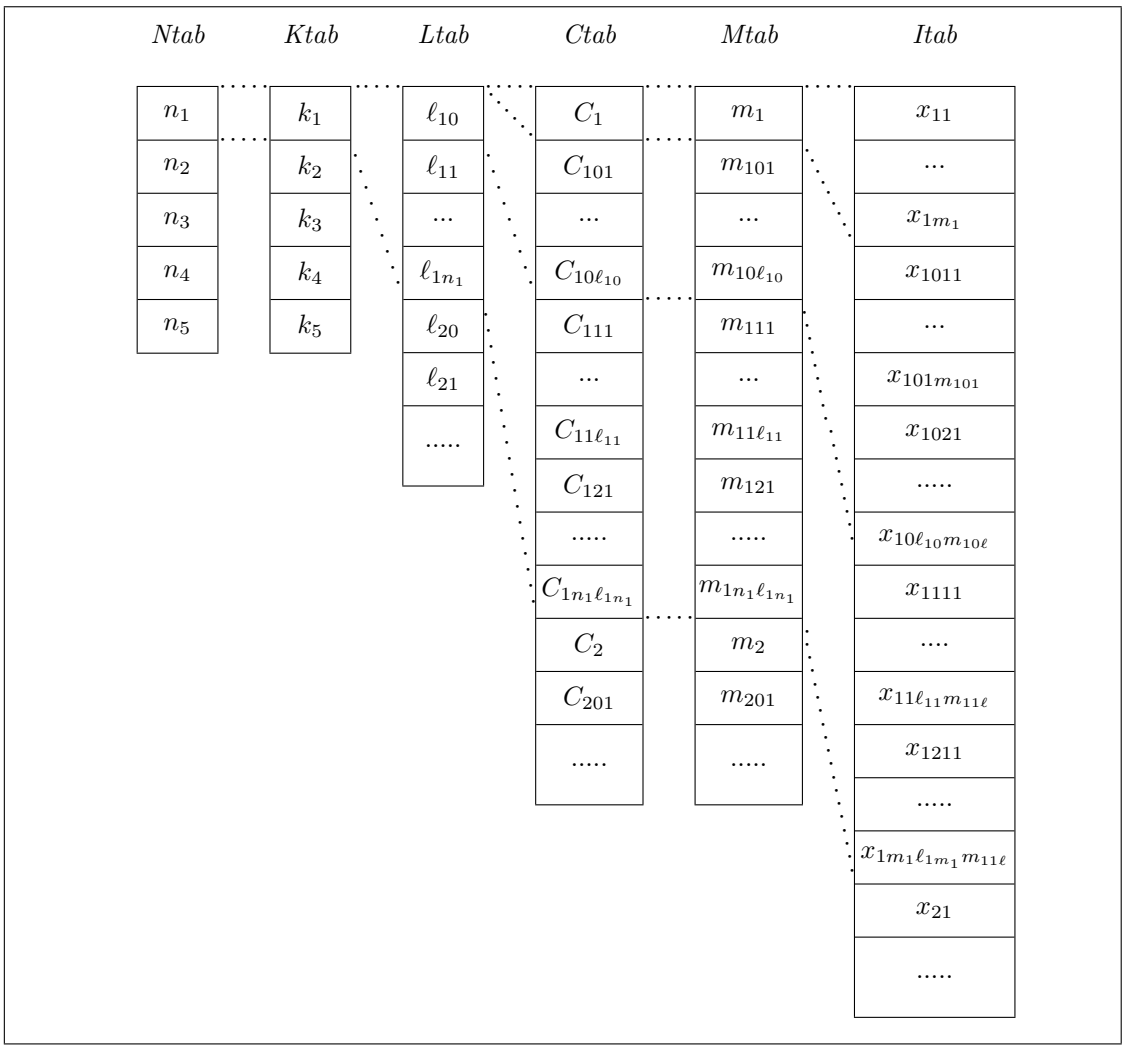

Fig.1. Representation of symbolic functions.

$$\mathcal{R}_{ij} \;=\; \sum_{k=1}^{\ell_{ij}} C_{ijk} \prod_{\ell=1}^{m_{ijk}} x_{ijk\ell}$$

FIT represents the products $\mathcal{T}_i$ and sums $\mathcal{R}_{ij}$ using a collection of arrays (actually, vectors) which store (real) coefficients and (integer) indices to other arrays as well as symbol identifiers (i.e., integer indices to a 'Symbol Table' which is a link between the numerical and symbolic simulators):

- *Ntab* – integer, the degrees of characteristic polynomials,
- *Ktab* – integer, the exponents of s associated with the products $\mathcal{T}_i$,
- *Ltab* – integer, the numbers of terms in $\mathcal{R}_{ij}$ sums,
- *Ctab* – real, the values of coefficients C_i and C_{ijk},
- *Mtab* – integer, the lengths of (i.e., the number of symbols in) products $\mathcal{T}_i$ and all terms of $\mathcal{R}_{ij}$,
- *Itab* – integer, the identifiers of symbols used in all products $\mathcal{T}_i$ and all terms of $\mathcal{R}_{ij}$.

The organization of these arrays for 5 symbolic functions, $i = 1, ..., 5$, is sketched in Fig.1. An outline of the evaluation procedure using these arrays is given in the next section.

3. PARAMETER EXTRACTION WITH SYMBOLIC SIMULATION

Analysis of the small–signal, linear behavior of a circuit is an important part of parameter extraction in general, but it is especially significant in the case of microwave

applications. Using symbolic rather than numerical simulation for linear analyses can save much of the 'computational effort' required by numerical simulation, and considerably speed up the extraction process. Moreover, it is often the case that the extraction of a set of parameters is decomposed into a sequence of 'partial extractions', performed on subsets of parameters (and relevant subsets of measurement data [7]). For such partial extractions, the sets of parameters usually contain only a few symbols, which means that the corresponding symbolic functions are also quite simple and their evaluations very efficient.

Since the frequency–domain analyses are performed for the circuit with the same topology, the generation of symbolic functions can be done only once. Furthermore, the circuit parameters (or symbols) used in symbolic analysis can be subdivided into *variable* symbols and *fixed* symbols. Variable symbols can change their values from one optimization step to another; these symbols include optimization variables, i.e., a subset of parameters updated in the optimization loop (all such symbols are called *direct*), and all those symbols whose values depend upon the operating point solution (these symbols are called *dependent*). All remaining symbols are 'fixed' in the sense that their values do not change during the optimization process. Consequently, all fixed symbols can be replaced in the symbolic functions by their numerical values during the generation of the symbolic functions, reducing the functions and simplifying all subsequent evaluations.

The values of *variable* symbols can be retrieved in two steps: (i) at the beginning of the optimization loop (for all *direct* symbols), and (ii) after each operating point solution (for all *dependent* symbols). The values of *variable* symbols are used for a transformation of the symbolic functions to their reduced form, performing evaluation of all products $\mathcal{T}_i$ and sums $\mathcal{R}_{ij}$:

$$\mathcal{F}_i^{(r)} \;=\; s^{k_i} A_i \sum_{j=0}^{n_i} s^j A_{ij}$$

where all A_i and A_{ij}, $j = 0, 1, ..., n_i$, are constants provided that no frequency–dependent elements are used. Only this very simple polynomial form needs to be evaluated for each frequency.

For the representation of symbolic functions described in Section 2, the outline of the evaluation of reduced functions is as follows (`Nf` is the number of characteristic polynomials, equal to 5 in Fig.1);

```
il := 0;
im := 0;
is := 0;
for i := 1 to Nf do
   A[i] := Product(im,is);
   for j := 0 to Ntab[i] do
      sum := 0.0;
      il := il + 1;
      for l := 1 to Ltab[il] do
         sum := sum + Product(im,is)
      endfor;
      A[i,j] := sum
   endfor
endfor;
```

where the real function `Product` increments the variables `im` and `is`, so the 'passing by reference' mechanism is assumed (and `ST` denotes the "Symbol Table", i.e., an array containing the values of all symbols as well as their attributes):

```
real function Product (int im, int is);
begin
   real val;
   int last;
   im := im + 1;
   val := Ctab[im];
   last := is + Mtab[im];
   while is < last do
      is := is + 1;
      val := val * ST[Itab[is]]
   endwhile;
   return val
end;                                  -.
```

4. EXAMPLE

A comparison of execution times for numerical and symbolic simulations is given for parameter extraction of a submicron (0.25 μ) HEMT device on InP substrate. A small-signal model (with its parameters) is shown in Fig.2.

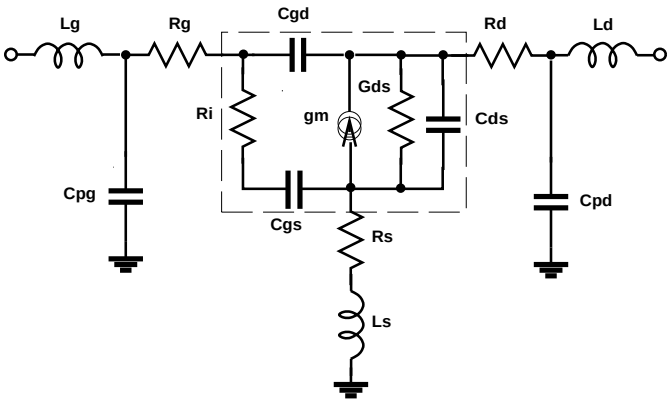

Fig.2. HEMT small-signal model.

For the model shown in Fig.2, the (five) reduced symbolic functions are polynomials of degrees 3, 6, 8, 6 and 7, while the exponents of the common factors are equal to -2, -2, -3, -3 and -3.

FIT–S performs all evaluations of the symbolic functions in the real domain, independently for the real and

imaginary parts because such an evaluation appears to be the most efficient one. However, the evaluation of the symbolic functions is only a rather small part of all computations involved in parameter extraction; the values of symbolic functions must be converted into S–parameters (or some other form which is used by the measurement data), they must be stored in a database of results, compared with the corresponding measurement values to update the value of the error function, etc. Therefore, a more realistic performance comparison of symbolic and numerical analysis is obtained by measuring the total execution times for a typical extraction process. Such comparison results, with corresponding values of the speedup, are shown in the following table in which the columns correspond to data groups with 10, 20, 50 and 100 frequencies (the execution times are in seconds, on a SPARCstation 2, for 100 iteration steps):

frequencies per data group	10	20	50	100
symbolic simulation	3.36	3.70	4.46	5.88
numerical simulation	8.69	12.4	23.7	42.3
speedup	2.6	3.4	5.3	7.2

The evaluation times of the polynomials account for a rather small part of the total execution time of symbolic analysis. The evaluation time of (reduced) characteristic functions (i.e., the evaluation of the polynomials as well as the exponential factors) and its contribution to the execution time of symbolic analysis is as follows (the execution times are in seconds, for a SPARCstation 2, and correspond to 100 iteration steps):

frequencies per data group	10	20	50	100
symbolic simulation	3.36	3.70	4.46	5.88
function evaluation	0.14	0.28	0.72	1.45
percent of execution time	4.2	7.6	16.1	24.7

If the conversion of symbolic function values to S–parameters is taken into account, the comparison is as follows:

frequencies per data group	10	20	50	100
symbolic simulation	3.36	3.70	4.46	5.88
function evaluations	0.26	0.60	1.36	2.78
percent of execution time	7.7	16.2	30.5	47.3

The difference between the total execution times and the evaluation times shown in the last table is practically independent of the number of frequencies; this difference corresponds to: (i) updating circuit parameters (in the numerical simulator), (ii) finding operating point solutions, (iii) retrieving the values of all variable symbols, (iv) evaluation of coefficients of reduced functions, and (v) storing the results and evaluation of the error function (only this part depends upon the number of frequencies, but its contribution is rather insignificant).

5. CONCLUDING REMARKS

An integration of symbolic approach with traditional numerical circuit analysis can reduce several times the simulation time. This reduction can be used for more sophisticated simulation strategies, which – in general – are more computationally demanding.

In the case of parameter extraction, the analyzed circuits are rather small (typically they contain less than 10 nodes and less than 15 elements), so the symbolic functions are relatively simple and no function approximations are really needed. Moreover, many symbols can usually be eliminated during the generation of symbolic function because their values do not change during the extraction (*fixed* symbols). More general applications of symbolic and integrated numerical–symbolic simulation must take into account that for larger circuits the symbolic functions become very complex, so additional function simplification is required [12].

The presented approach was developed with the assumption that no frequency–dependent elements are used in the small-signal analysis (the proposed 'reduced' does not support frequency–dependency). If this assumption is not true and frequency–dependent elements are to be taken into account, the approach must be modified by introducing slightly different reduction step, which creates 'reduced symbolic products' by eliminating all frequency–independent symbols. These (usually small) reduced symbolic products must be evaluated for each frequency (instead of simple polynomials). An outline of the modified reduction step, which creates (new, reduced) tables *LtabR*, *CtabR*, *MtabR* and *ItabR*) can be as follows:

```
il := 0;
im := 0;
is := 0;
imr := 0;
isr := 0;
for i := 1 to Nf do
   sbase := isr;
   imr := imr + 1;
   CtabR[imr] := Reduce(im,is,isr);
   MtarR[imr] := isr - sbase;
   for j := 0 to Ntab[i] do
      sum := 0.0;
      imbase := imr;
      il := il + 1;
      for k := 1 to Ltab[il] do
         isbase := isr;
         val := Reduce(im,is,isr);
         if isr = isbase then
            sum := sum + val
         else
            imr := imr + 1;
            CtabR[imr] := val;
            MtabR[imr] := isr - isbase
         endif
      endfor;
```

```
      if sum <> 0.0 then
         imr := imr + 1;
         CtabR[imr] := sum;
         MtabR[imr] := 0
      endif;
      LtabR[il] := imr - imbase;
   endfor
endfor;
```

where the procedure `Reduce` deals with one symbolic product, checking for frequency–dependent symbols, and returning the product of all frequency–independent symbols (while copying all frequency–dependent symbols to *ItabR*):

```
real function Reduce (int im, int is, int isr);
begin
   real val;
   int last;
   im := im + 1;
   val := Ctab[im];
   last := is + Mtab[im];
   while is < last do
      is := is + 1;
      if frequency_dependent(symbol(Itab[is])) then
         isr := isr + 1;
         ItabR[isr] := Itab[is]
      else val := val * ST[Itab[is]] endif
   endwhile;
   return val
end;
```

It should be observed that if there are no frequency–dependent symbols, the (completely) reduced form (all elements of *MtabR* are zeros and all elements of *LtabR* are equal to 1) is equivalent to a 'reduced polynomial' but is less efficient in both representation and evaluation time than a simple polynomial discussed earlier.

In FIT–S, symbolic analysis is used for (small–signal) frequency–domain analyses only. If ongoing research succeeds in developing symbolic methods that can be applied to non–LLS (linear, lumped and stationary) circuits, further reduction of the 'computational effort' required for simulation–based parameter extraction will be possible.

Acknowledgement

The Natural Sciences and Engineering Research Council of Canada partially supported this research through Research Grant A8222.

References

[1] G.L. Bilbro, M.B. Steer, R.J. Trew, C-R Chang, S.G. Skaggs, "Extraction of the parameters of equivalent circuits of microwave transistors using tree annealing"; IEEE Trans. on Microwave Theory and Techniques, vol.38, no.11, pp.1711-1718, 1990.

[2] P. Conway, C. Cahill, W.A. Lane, S.U. Lidholm, "Extraction of MOSFET parameters using the simplex direct search optimization method"; IEEE Trans. on Computer-Aided Design, vol.4, no.4, pp.694-698, 1985.

[3] K. Doganis, D.L. Scharfetter, "General optimization and extraction of IC device model parameters"; IEEE Trans. on Electron Devices, vol.30, no.9, pp.1219-1228, 1983.

[4] K. Garwacki, "Extraction of BJT model parameters using optimization method"; IEEE Trans. on Computer-Aided Design, vol.7, no.8, pp.850-854, 1988.

[5] W.M. Zuberek, A. Konczykowska, C. Algani, H. Wang, J. Dangla, "Simulation–based parameter extraction, its implementation and some applications"; IEE Proc. on Circuits, Devices and Systems, vol.141, no.2, pp.129-134, 1994.

[6] A. Konczykowska, V. Morin, J. Godin, M. Bon, "Symbolic analysis for CAD of microwave circuits"; Proc. Symp. on Computer Aided Design of Microwave Circuits, London, England, 1985.

[7] W.M. Zuberek, A. Konczykowska, "FIT–S, a simulation–based data–driven parameter extraction program"; Technical Report #9403, Department of Computer Science, Memorial University of Newfoundland, St. John's, Canada A1C 5S7, 1994 (available through anonymous ftp at "`ftp.cs.mun.ca`" file "`pub/techreports/tr_9405.ps.Z`").

[8] D.O. Pederson, "A historical review of circuit simulation"; IEEE Trans. on Circuits and Systems, vol.31, no.1, pp.103–111, 1984.

[9] E. Cohen, "Program reference for SPICE 2"; Memorandum UCB/ERL M592, University of California, Berkeley, CA 94720, 1976.

[10] W.J. McCalla, "Fundamentals of computer-aided circuit simulation"; Kluwer Academic Publ. 1988.

[11] J. Vlach, K. Singhal, "Computer methods for circuit analysis and design"; Van Nostrand Reinhold 1983.

[12] G. Gielen, W. Sansen, "Symbolic analysis for automated design of analog intergrated circuits"; Kluwer Academic Publ. 1991.

[13] P-M. Lin, "Symbolic network analysis"; Elsevier 1991.