23-rd Asilomar Conference on Signals, Systems and Computers, Pacific Grove, CA, 30 Oct.–1 Nov. 1989, pp.440-444.
Copyright © 1989 IEEE (DOI 10.1109/ACSSC.1989.1200829).

TABLE–DRIVEN CIRCUIT ELEMENTS IN SPICE–LIKE SIMULATION PROGRAMS

W.M. Zuberek and M.S. Zuberek

Department of Computer Science
Memorial University of Newfoundland
St. John's, Canada A1C-5S7

Abstract

Table-driven approximation of semiconductor device characteristics and table-driven behavioral descriptions are two particularly attractive examples of enhancements that can significantly speed up circuit simulation by eliminating many time-consuming calculations in which device voltages and currents are derived from properties and geometries of semiconductor materials. Several extensions to a SPICE-compatible simulation tool are described, that provide table-driven capabilities for all nonlinear SPICE circuit elements.

1. INTRODUCTION

Computer-aided circuit analysis or circuit simulation is an accepted method of verifying designs before the expensive and time-consuming fabrication phase. Even with designs of modest size, an accurate simulation provides invaluable insight into the performance of the product. However, existing trends in integrated circuit design towards increasing device densities and shrinking circuit geometries results in designs of increasing complexities. This – in turn – increases the simulation time and memory requirements. Several techniques have been proposed to counterbalance these increasing requirements [DNV,FRS], but using "higher-level abstraction" wherever possible remains the simplest and the most effective solution to this complexity problem.

The notion of "abstraction" in this context means a method to replace an object by a simplified one that only defines the interaction of the object with its environment, while deleting the internal organization details of the object [Niess]; for example, the whole comparator module can be replaced by a single "block" with input-output characteristics equivalent to those of the original design. The virtue of abstraction is data reduction, sometimes by one or more orders of magnitude. For very large systems, one level of abstraction may not suffice, it has to be applied a number of times in succession creating "hierarchical abstraction".

Table-driven approximation of semiconductor device characteristics and table-driven behavioral descriptions are two particularly attractive examples of "higher-level abstraction" that can significantly speed up circuit simulation by eliminating many time-consuming calculations in which device voltages and currents are derived from properties and geometries of semiconductor materials. An implementation of table-driven capabilities in a SPICE-like circuit simulators is outlined in this paper.

The SPICE-2 program [Coh,Vlad] developed at the University of California, Berkeley, has become one of the most popular "second-generation" circuit simulators. However, SPICE-2 is a "batch oriented" program with a "closed" structure [Coh] which is too "inflexible" in applications that require efficient analyses for numerous variants of the same circuit (e.g., interactive simulation or circuit optimization). In such cases, a more flexible structure of the circuit simulator is needed, in which different analyses (for the same circuit topology) can be performed selectively, and which allows to access the internal representation of circuit elements in order to modify element values. SPICE-PAC [Zub1] is a simulation package obtained by redesigning the SPICE-2G.6 simulation program.

SPICE-PAC, in a way similar to the SPICE program, provides a "predetermined" set of (parameterized) circuit elements and semiconductor devices. For example, SPICE allows circuits to contain linear and nonlinear elements (capacitors, inductors, dependent voltage and current sources [Vlad]), but the only nonlinear function which is supported by SPICE is a polynomial (in one - for capacitors and inductors - or more - for dependent sources - variables). Since in many cases this is too restrictive [BVS,TYIS], a number of extension have built into SPICE-PAC to allow "external" specification of nonlinear functions by user-supplied tables of data. The package offers a number of (internal) interpolating methods for multidimensional data, and also provides an interface to user-supplied (or library) "external" interpolation routines.

The paper is organized in three main sections. Section 2 presents table-driven elements implemented in the SPICE-PAC package. Section 3 discusses "internal" interpolation methods, i.e., the methods that are built into the package, while section 4 describes an interface to "external" interpolation routines. Applications of these capabilities are illustrated by simple examples of table-driven elements.

2. TABLE-DRIVEN ELEMENTS

The characteristics of nonlinear elements (capacitors, inductors, dependent sources) as well as "input-output" characteristics of modules (or "subcircuits") can be specified by (multidimensional) tables of numerical data with an associated interpolating method.

Table-driven circuit elements are described in a way similar to the original SPICE syntax (with some small differences for different classes of elements):

```
Name  node+ node-  itp(n) arg1 arg2 ... argn
                   DIM(d1,d2,...,dn)  x1,x2,...
```

or, if the data are shared by a number of elements, they can be described by a "TABLE" pseudoelement to avoid repetition of descriptions:

```
Name  node+ node-  itp(n) arg1 arg2 ... argn
                   USE(tname)
```

where `Name` is the unique name of the element with the first letter `N` determining the class the element (`R` for resistors, `C` for capacitors, etc.); `node+` and `node-` are element terminals; `itp` indicates the interpolation method:

`itp:`	value:	derivative:
`PWL` or `PWL1`	piecewise linear	approximated
`PWL2`	piecewise linear	piecewise linear
`PWQ` or `PWQ1`	piecewise quadratic	approximated
`PWQ2`	piecewise quadratic	piecewise quadratic
`PWC` or `PWC1`	piecewise cubic	approximated
`PWC2`	piecewise cubic	piecewise cubic

`n` is the number of arguments (i.e., the number of controlling voltages and/or currents); `arg1, arg2, ... argn` are the arguments in the SPICE sense, i.e., each `arg` is either a pair of nodes that determine the controlling voltage, or a voltage source name that indicates the controlling current flowing through this source; `DIM(d1,d2,...)` is an optional specification of the "structure" of parameters `x1,x2,...`, e.i., if these parameters describe a multidimensional array of data, the dimensions of this array are indicated in the `DIM` option as `n` consecutive values `d1,d2,...,dn`; `tname` is the name of a "TABLE" pseudoelement, and `x1,x2,...` are the consecutive numerical values describing the characteristics of `Name`.

General syntax of "TABLE" pseudoelements is as follows:

```
.TABLE  tname  ARG(n)  DIM(d1,d2,...)  x1,x2,x3,...
```

where `ARG(n)` s an optional specification of the number of controlling voltages and/or currents (with the default value equal to 1) and all other symbols are as before.

Example 1. A semiconductor diode characteristic is approximated by a table-driven voltage controlled current source with piecewise linear, quadratic and cubic interpolation methods. The differences between the original (reference) diode and the approximated elements are shown in a few selected points of the transfer characteristics:

```
****    SPICE-PAC.2G6c:89.05b (MUN)  DATE : 27 MAY 89
** table-driven dependent sources
VDD 1 0
* reference diode
RL 1 2 1K
DD 2 0 DMOD
.MODEL DMOD D
* diodes as voltage-controlled-current sources
R1 1 3 1K
G1 3 0 PWL(1) 3 0  USE(TAB)
R2 1 4 1K
G2 4 0 PWQ(1) 4 0  USE(TAB)
R3 1 5 1K
G3 5 0 PWC(1) 5 0  USE(TAB)
.TABLE TAB    (-10 -1E-11, 0 0, 0.05 1.09E-13,
+ 0.10 5.68E-13, 0.15 3.44E-12, 0.20 2.30E-11,
+ 0.25 1.58E-10, 0.30 1.09E-09, 0.35 7.53E-09,
+ 0.40 5.21E-08, 0.45 3.60E-07, 0.50 2.49E-06,
+ 0.55 1.72E-05, 0.60 1.19E-04, 0.65 8.21E-04,
+ 0.70 5.67E-03, 0.75 3.92E-02, 0.80 2.71E-01,
+ 0.85 1.87E+00, 0.90 1.29E+01)
* DC analysis
.DC VDD -2 4 1.0
.PRINT DC V(2),V(2,3),V(2,4),V(2,5)
.END

***** DC TRANSFER CURVE         TEMP :  27.00 DEG C

     VDD         V(2)      V(2,3)    V(2,4)    V(2,5)
 -2.00d+00   -2.00d+00  1.00d-11 -1.87d-09  2.25d-07
 -1.00d+00   -1.00d+00  1.00d-11 -1.05d-09  6.43d-08
  0.00d+00    2.74d-29  2.74d-29  2.74d-29 -1.85d-24
  1.00d+00    6.29d-01  1.07d-02  5.77d-03 -6.78d-03
  2.00d+00    6.63d-01  7.22d-03  4.28d-03 -1.40d-02
  3.00d+00    6.77d-01  1.13d-02  6.16d-03 -8.07d-03
  4.00d+00    6.86d-01  1.03d-02  5.27d-03 -4.51d-03
```

The few numerical results indicate that the differences are usually in range of millivolts, that the quadratic interpolation results are approximately twice "better" than the linear ones, and that the results of cubic interpolation may actually be worse than the results of linear interpolation. A more complete illustration of the differences between the reference voltage and the three interpolated values is shown in Fig.1; it clearly confirms previous observations.

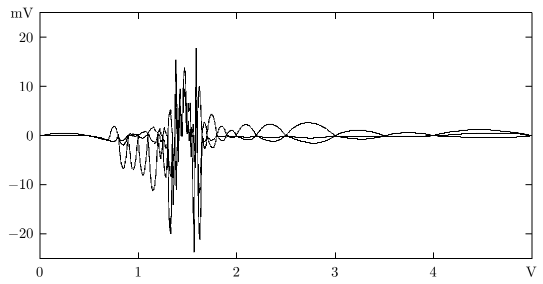

Fig.1. Errors of internal interpolation methods – example 1.

Example 2. In this example, the transfer characteristic of a MOSFET inverter is modeled by a table-driven voltage-controlled voltage source; the transfer characteristic with the selected data points is shown in Fig.2.

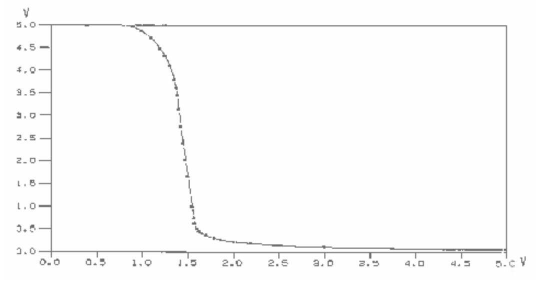

Fig.2. Inverter's transfer curve with selected data points.

Similarly as before, the results of linear, quadratic and cubic interpolation are compared with the simulation of the original inverter [GlaDo].

```
****    SPICE-PAC.2G6c:89.09a (MUN)  DATE : 14 OCT 89
.OPTIONS DEFL=2.25E-6  LIMPTS=501
VIN 1 0 0
VDD 9 0 5V
* MOS subcircuit (reference level)
M1 2 1 0 0 NENHS W=11.2U AD=61P PD=42U
.MODEL NENHS NMOS LEVEL=3 TOX=33E-9 LD=0.19E-6
+       XJ=0.27E-6 UO=650 RSH=0 VMAX=13E4 ETA=0.25
+       KAPPA=0.5 NSUB=5E14 VTO=0.946 CGSO=2.43E-10
+       CGDO=2.43E-10 CJSW=3.3E-10 CJ=69E-6 PB=0.7
+       THETA=0.1 MJ=0.5 MJSW=0.3 NFS=1E10
M2 9 2 2 0 NDEPS W=4.2U L=6.25U
.MODEL NDEPS NMOS LEVEL=3 TOX=33E-9 LD=0.19E-6
+       XJ=0.27E-6 UO=650 RSH=0 VMAX=13E4 ETA=0.25
+       KAPPA=0.5 NSUB=5E15 VTO=-2.078 CGSO=2.43E-10
+       CGDO=2.43E-10 CJSW=3.3E-10 CJ=69E-6 PB=0.7
+       THETA=0.04 MJ=0.5 MJSW=0.3 NFS=1E10
* table-driven voltage-controlled voltage sources
E1 3 0 PWL(1) 1 0  USE(Tdata)
R1 3 0 1K
E2 4 0 PWQ(1) 1 0  USE(Tdata)
R2 4 0 1K
E3 5 0 PWC(1) 1 0  USE(Tdata)
R3 5 0 1K
.TABLE Tdata (0.00 5.000, .500 5.000, .700 4.999,
+  .80 4.998, .900 4.958, 1.00 4.863, 1.10 4.708,
+ 1.20 4.473, 1.25 4.311, 1.30 4.100, 1.35 3.792,
+ 1.37 3.594, 1.40 3.129, 1.45 2.377, 1.50 1.661,
+ 1.55 .9898, 1.58 .6148, 1.60 .5070, 1.65 .4120,
+ 1.70 .3599, 1.80 .2961, 1.90 .2556, 2.00 .2266,
+ 2.20 .1871, 2.50 .1508, 3.00 .1165, 3.50 .0966,
+ 4.00 .0836, 5.00 .0674)
* DC analysis
.DC VIN 0 5 0.01
.PRINT DC V(2) V(2,3) V(2,4) V(2,5)
.END
```

The differences between the original (reference) output and the three interpolation methods are shown in Fig.3. It can be observed that the extrema of differences correspond to sharp changes of the transfer curve shown in Fig.2.

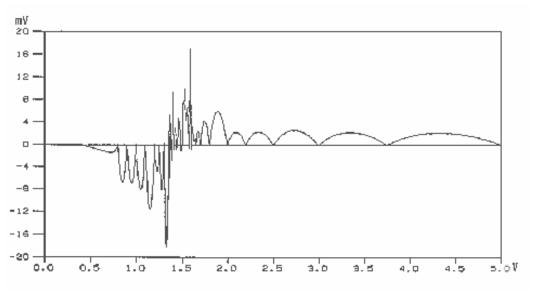

Fig.3. Errors of internal interpolation methods – example 2.

3. INTERNAL INTERPOLATION

Because of the Newton-Raphson method used in the solution of nonlinear circuit equations [Coh], both the values and of table-driven elements and the values of (partial) derivatives (with respect to controlling voltages and/or currents) are needed by the simulation algorithm.

Interpolation of one-dimensional data follows Neville's algorithm [PFTV], closely related to and sometimes confused with Aitken's algorithm, the latter now considered obsolete. Calculations can be arranged in a "tableau" with the argument x "bracketed" by the x_{i+j} values and the size of the tableau depending upon the degree of the interpolation polynomial:

$$\begin{array}{llllll}
x_i: & y_i = P_1 & & & \\
 & & P_{12} & & \\
x_{i+1} & y_{i+1} = P_2 & & P_{123} & \\
 & & P_{23} & & P_{1234} \\
x_{i+2} & y_{i+2} = P_3 & & P_{234} & \\
 & & P_{34} & & \\
x_{i+3} & y_{i+3} = P_4 & & &
\end{array}$$

Neville's algorithm is a recursive way of filling in the numbers in the tableau a column at a time, from left to right. It is based on the following recursive relationship between the polynomials P [PFTV]:

$$P_{i(i+1)\ldots(i+m)} = \frac{(x-x_{i+m})P_{i\ldots(i+m-1)}+(x_i-x)P_{(i+1)\ldots(i+m)}}{x_i-x_{i+m}}.$$

In multidimensional interpolation, the basic idea is to break a multidimensional problem into a succession of one-dimensional interpolations. To do a two-dimensional m-order interpolation in the X direction and n-order interpolation in the Y direction, first an $(m+1)\times(n+1)$ block of the tabulated function is located that contains the desired point (x, y), and then $(m+1)$ one-dimensional interpolations are performed in the direction Y (to "reduce" the second dimension), and then one one-dimensional interpolation in the X direction to find the required value. The same scheme can be used for more dimensions, although the number of one-dimensional interpolations grows rapidly with the number of dimensions and the order of interpolation.

All internal interpolation methods are simple local methods; more sophisticated algorithms can be used through the "external" interpolation mechanism.

4. EXTERNAL INTERPOLATION

For all nonlinear elements, interfaces have been implemented in SPICE-PAC to "external", user-defined routines that – if used – replace the "standard" SPICE-PAC's capabilities. In particular, for table-driven elements, these interfaces can be used to access interpolation methods which are not available in the package. Circuit elements that use externally specified evaluation methods (either library or used-defined) are called enhanced elements [Zub2].

Generally, the interfacing routines pass the values of controlling voltages and/or currents (and some other information) to the evaluation routines and return the values of interpolated functions and/or their derivatives, as required by the simulation package.

The syntax of enhanced circuit elements follows the original SPICE's syntax with the `FUN(idf)` section indicating use of external routines:

```
Name  node+ node-  FUN(idf) ARG(n) arg1 arg2 ... argn
                   DIM(d1,d2,...,dn)  x1,x2,...
```

`idf` is a numerical identifier of the evaluation function, `ARG(n)` indicates the number of controlling voltages and/or currents `arg1, arg2, ... argn` with the default value of 1, `DIM(d1,...,dn)` is an optional specification of dimensions for "structured" parameters, and `x1,x2,...` are numerical data, as before.

Several enhanced elements can share the same set of numerical data using the "TABLE" pseudoelement:

```
Name  node+ node-  FUN(idf) ARG(n) arg1 arg2 ... argn
                   DIM(d1,d2,...,dn)  USE(tname)
```

There is one interfacing routine for each class of enhanced elements; the `idf` parameter is used for further identification within each class of elements.

For (nonlinear) controlled sources the interfacing routine is called `SPUDSE`, and it must be defined in a way consistent with the following (FORTRAN) header:

```
 SUBROUTINE SPUDSE (IPS,IDF,VA,NA,VP,NP,VAL,SUM)
 DOUBLE PRECISION VA(NA),VP(NP(1))
 INTEGER NP(1)
```

where `IPS` is a unique internal identifier of the circuit element (i.e., `IPS` is a pointer to the element descriptor; there are routines which convert descriptor pointers into corresponding circuit element names and vice versa); `IDF` is the function identifier `idf` from the element description; `VA` is the vector of controlling voltages and/or currents; `NA` is the number of arguments `n`; `VP` is a vector of parameters `x1,x2,...`, and `NP` is an auxiliary vector with at least two elements; the first element always indicates the number of parameters; if the second is nonzero, the elements `NP(2),NP(3),...` indicate consecutive dimensions from the `DIM` option, i.e., `NP(2)=d1`, etc.; `VAL` returns the value of required attribute, and `SUM` returns the sum of products of arguments and derivatives, while the values of (partial) derivatives are returned in the vector `VA` (replacing the values of arguments). The following example shows how to use external interpolation for table-driven controlled sources.

Example 3. In order to use Akima spline for approximation of the transfer curve of the MOSFET inverter from example 2, the SPUDSE routine interfaces SPICE-PAC to the implementation of the Akima method that is available in the IMSL library [IMSL]. Akima splines are designed specifically to combat "wiggles" of approximation functions that often appear in spline functions. It is assumed that Akima splines are identified by the function identifier `idf=11`.

```
      SUBROUTINE SPUDSE (IPS,IDF,VA,NA,VP,NP,VAL,SUM)
C
C  DCSAKM is an IMSL routine that determines AKIMA
C  spline interpolants. DCSVAL evaluates the cubic
C  spline, and DCSDER evaluates its n-th derivative
C  ("n" is the first argument).
C
      IMPLICIT DOUBLE PRECISION (A-H,O-Z)
      DIMENSION VA(NA),VP(1),NP(1)
      DIMENSION XBRK(50),COEF(200)
      SAVE XBRK,COEF
      COMMON /SPPABC/ IOFILE
      DATA MK / 0 /
      IF (IDF.EQ.11) THEN
C ... Akima spline interpolation
        IF (NA.NE.1) GO TO 90
        IF (MK.EQ.0) THEN
          MK=NP(1)/2
          IF (MK.GT.50) GO TO 90
          CALL DCSAKM(MK,VP(1),VP(MK+1),XBRK,COEF)
        ENDIF
        ARG=VA(1)
        VAL=DCSVAL(ARG,MK-1,XBRK,COEF)
        DER=DCSDER(1,ARG,MK-1,XBRK,COEF)
        VA(1)=DER
        SUM=ARG*DER
      ELSE
C ..... other evaluation methods
      ENDIF
      RETURN
   90 WRITE(IOFILE,900) IPS,IDF,NA,NP(1)
  900 FORMAT(' ... SPUDSE : incorrect arguments :',4I4)
      STOP
      END
```

A comparison of Akima interpolation results with the original inverter's output is shown in Fig.4. The errors are also in the range of millivolts, and two characteristic extrema are clearly visible, as in Fig.3.

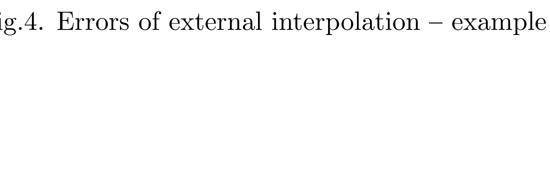

Fig.4. Errors of external interpolation – example 3.

5. CONCLUDING REMARKS

Table-driven simulation not only provides an opportunity for SPICE-PAC to simulate devices for which detailed models do not presently exist in circuit simulators [SNSPH], but also to represent characteristics of existing elements and devices more efficiently. It is known [BVS,DeCh,Rauh] that table-driven models can be used to eliminate most of the time-consuming evaluations of device models in which electrical properties are derived from technological and geometric parameters (it appears that in simulation of integrated circuits, evaluations of device models consume about 90% and sometimes even more of the total simulation time [DNV,Rauh]). Furthermore, table-driven simulation provides a simple solution to hierarchical simulation in which the characteristics of whole "blocks" (or subcircuits) can be represented by table-driven elements; DC characteristics can be assigned to (multidimensional) controlled voltage and current sources, while time-dependent properties can be represented by capacitors with nonlinear table-driven characteristics of charge and capacitance [FRS].

Table-driven circuit elements are especially convenient for experiment-oriented applications in which measurement data as well as predicted or estimated characteristics can be used directly in element specification.

Interfaces to external evaluations of element characteristics can be used not only for table-driven elements but also for more "demanding" applications that use a number of software tools integrated into a sigle system.

The extensions presented in this paper are implemented in SPICE-PAC versions 2G6c.89 and beyond.

Acknowledgement

The Natural Sciences and Engineering Research Council of Canada partially supported this research through Operating Grant A8222, and Northern Telecom Canada through Memorial University Interaction Program.

References

[BVS] J. Barby, J. Vlach, K. Singhal, "Optimized polynomial splines for FET models" Proc. 1984 Int. Symp. on Circuits and Systems, Montreal, Canada, pp.1159-1162.

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

[DeCh] A. Deng, L.O. Chua, "Canonical piecewise linear analysis and modelling of electronic circuits"; Proc. 1986 Int. Symp. on Circuits and Systems, San Francisco CA, pp.476-479.

[DNV] S. Director, S.R. Nassif, L.M. Vidigal, "A new way to speed up circuit simulation"; Electronics, vol.59, pp.71-74, August 1986.

[FRS] U. Feldmann, K-G. Rauh, K. Steger, "Circuit simulation on vector processors"; Proc. IEEE COMPEURO'87 Conf. on VLSI in Computers, Hamburg, West Germany, pp.246-249, 1987.

[GlaDo] L.A. Glasser, D.W. Dobberpuhl, "The design and analysis of VLSI circuits"; Addison-Wesley 1985.

[IMSL] IMSL - User's Manual, "FORTRAN subroutines for mathematical applications", vol.2, ch.3, "Interpolation and approximation", version 1.0, 1987.

[NPSS] A.R. Newton, D.O. Pederson, A.L. Sangiovanni-Vincentelli, C.H. Sequin, "Design aids for VLSI: the Berkeley perspective"; IEEE Trans. Circuits and Systems, vol.28, no.7, pp.666-680, 1981.

[Niess] C. Niessen, "Hierarchical design methodologies and tools for VLSI chips"; Proc. IEEE, vol.71, no.1, pp.66-75, 1983.

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

[PFTV] W.H. Press, B.P. Flannery, S.A. Teukolsky, W.T. Vetterling, "Numerical recipes - the art of scientific computing"; Cambridge University Press 1986.

[Rauh] K-G. Rauh, "A table model for circuit simulation"; Proc. ESSCIRC'86, pp.211-213, 1986.

[SNSPH] H. Statz, P. Newman, I.W. Smith, R.A. Pucel, H.A. Haus, "GaAs MESFET device and circuit simulation in SPICE"; IEEE Trans. on Electron Devices, vol.34, no.2, pp.160-169, 1987.

[TYIS] T. Takada, K. Yokoyama, M. Ida, T. Sudo, "A MESFET variable-capacitance model for GaAs integrated circuit simulation"; IEEE J. on Microwave Theory and Techniques, vol.30, no.5, pp.719-724, 1982.

[Vlad] A. Vladimirescu, K. Zhang, A.R. Newton, D.O. Pederson, A.L. Sangiovanni-Vincentelli, "SPICE Version 2G - User's Guide (10 Aug. 1981)"; Department of Electrical Engineering and Computer Sciences, University of California, Berkeley CA 94720, 1981.

[Zub1] W.M. Zuberek, "SPICE-PAC 2G6c - An Overview"; Technical Report #8903, Department of Computer Science, Memorial University of Newfoundland, St. John's, Newfoundland, Canada A1C 5S7, 1989.

[Zub2] W.M. Zuberek, M.S. Zuberek, "Enhanced dependent sources in the SPICE-PAC package of simulation subroutines"; Technical Report #8701, Department of Computer Science, Memorial University of Newfoundland, St. John's, Newfoundland, Canada A1C 5S7, 1987.