Courseware for Symbolic Analysis

ZDENĚK KOLKA
Department of Radio Electronics
Brno University of Technology
Purkyňova 118, 612 00 Brno
CZECH REPUBLIC

Abstract: - The paper deals with a program for symbolic, semisymbolic, and numerical analyses of linearized circuits and systems. It serves as a supporting tool of teaching electrical engineering and as a useful tool for designers. The program is able to analyze electronic circuits as well as signal-flow diagrams with behavioral blocks in the AC domain.

Key-Words: - Symbolic Analysis, Linearized Circuits, Courseware

1 Introduction

Symbolic analysis represents a supplement to numerical analysis in the field of linearized electronic circuits. An analytical expression for a network function can provide more information than multiple Spice simulations.

The program presented performs symbolic and semisymbolic analysis of any network function in the AC domain with two optional symbolic approximation methods.

The program was named SNAP (Symbolic Network Analysis Program). Its development started eight years ago. At that time the authors did not know there had been another program of the same name of Lin and Alderson.

2 Program Configuration

The program configuration is based on standard toolchain model used for Spice simulators consisting in separate modules for schematic entry and for analyser, fig. 1. The complete circuit description is passed to the symbolic analyzer (SNAP) in the form of a Spice-like netlist. Any schematic editor that can generate appropriate netlist format can be used with SNAP.

Because SNAP is primarily used as an educational tool, it has been integrated with the schematic editor supplied with PSpice™ simulator. Fig. 1 shows two possible workflows. In the first case the schematic editor library contains lineraized circuit devices and blocks. User has to supply all numerical values of small-signal parameters, if desired. This mode is suitable for classical symbolic analysis of elementary circuits and namely for educational purposes, fig. 2.

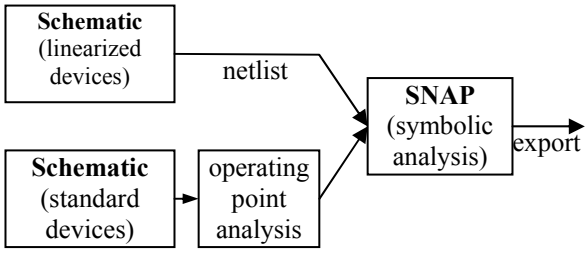


Fig. 1 Two possible configurations.

The second mode of operation is based on the utilization of standard analog devices (including nonlinear ones) from the (P)Spice library. Small signal parameters needed for pole/zero and approximate analyses are automatically extracted from the Spice operating-point (.OP) analysis.

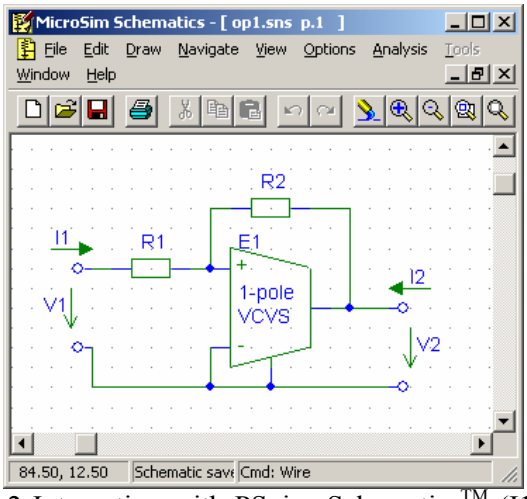


Fig. 2 Integration with PSpice Schematics™ (I1/V1 and I2/V2 are special symbols for input and output ports).

The SNAP netlist format offers several ways of device parameter specification, tab. 1.

Tab. 1 Device parameter specification

	syntax	example
1	*<symbol>*	R1
2	*<symbolic expression>*	R1+s*L1
3	*<numerical value>*	10k
4	*<numerical expression>*	{R1/2+1k}
5	*<symbol>* = *<value>*	R2=10k R2={R1/2+1k}

Symbolic expressions are constrained to those that can be transformed into a ratio of two polynomials in the Laplace operator. If numerical values of all parameters are available then it is possible to run all numeric-based analyses in addition to the symbolic analysis. The combined parameter definition (case 5) allows using the symbol for symbolic processing and the numerical value for numerical processing. It is always used in the of case automatic small-signal parameter extraction.

Fig. 2 shows an example of circuit diagram drawn using linearized devices. The circuit is always considered as a two-port. Input and output ports are defined by means of special devices. The type of network function for analysis is chosen in SNAP.

Fig. 3 shows the analysis flowchart. Graphical user interface is shown in fig. 8.

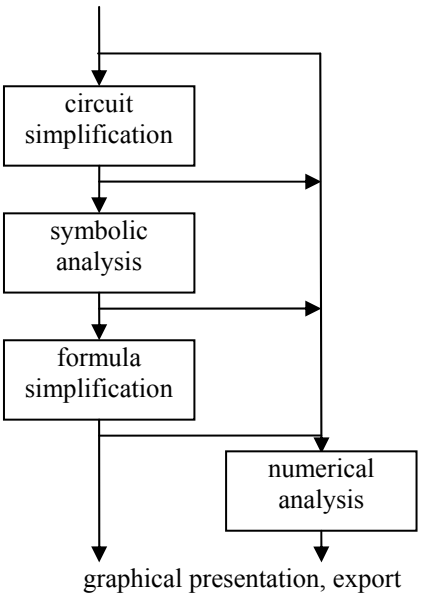
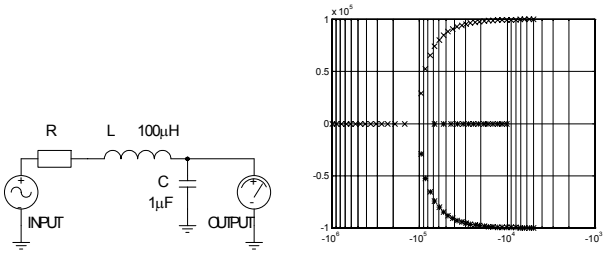


Fig. 3 SNAP flowchart.

If numerical values of all parameters are available then it is possible to display frequency characteristics and to perform pole/zero and symbolic approximation analyses. In addition to graphical presentation the results can be exported. Currently, export to Matlab and Maple is available.

Fig. 4 shows an example for Matlab. The network function saved as a compact Matlab function can be called directly in Matlab. The function allows computing its numerical value for a given s, to obtain coefficients of numerator and denominator, and to work with symbols for circuit variables. Fig. 4 shows a simple script for root-locus diagram using function *Ku_RLC()* generated by SNAP.




```
Ku_RLC('export'); global R;          %binding of variable R into Ku_RLC
N=40; Rstep=logdiv(1,100,N);         %logarithmic division of R
for i=1:N,                           %roots
  R=Rstep(i); Ro(1:2,i)=roots(Ku_RLC('denom'));
end;
                                     %graph
semilogx(real(Ro(1,:)),imag(Ro(1,:)),'wx',real(Ro(2,:)),imag(Ro(2,:)),'wx');
grid on;
```

Fig. 4 Example of root-locus diagram generation in Matlab.

3 Symbolic Analysis

The chapter describes symbolic algorithms implemented in SNAP. Numerical algorithms are well-known, see [1] for a comprehensive survey.

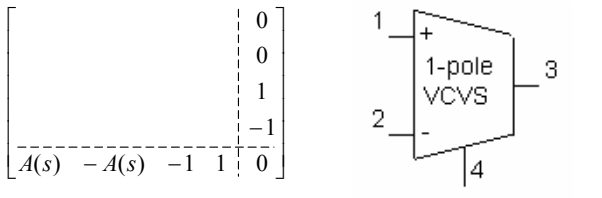
3.1 Exact Symbolic Analysis

Circuit equations are formulated by means of modified nodal analysis (MNA) [1] in the form

$$\mathbf{H}\,\mathbf{x} = \mathbf{0}\ , \qquad (1)$$

where **x** is a vector of nodal voltages and additional currents for elements without admittance description. Each network function can be expressed by means of Cramer rule as a ratio of two algebraic cofactors of the hybrid matrix **H**. No signal sources are necessary.

Parameters of each network element appear in some entries of the hybrid matrix. They are usually represented as a pattern called a *stamp*. The exceptional feature of SNAP is that stamps of all elements are stored in its library. The user can easily edit the library to add models of new devices. Fig. 5 shows the library section for one-pole VCVS.




```
[Ef1]
nodes  = 4       ;number of nodes
params = 2       ;number of parameters
names  = A0, f1  ;default names
add    = 1       ;new variables
                 ;stamp entries
mat1= -1  1  0  2 : A0/(1+s/(2*pi*f1))
mat2= -1  3  0  4 :-1
mat3=  3 -1  4  0 : 1
```

Fig. 5 One-pole VCVS model in SNAP library.

The stamp element mat_i= *a b c d : x* appears with positive sign on positions (*a,b*) and (*c,d*), and with negative sign on (*a,d*) and (*c,b*). The coordinates are numbered according to local numbers of the device nodes. Number -1 is for the added row and column, and number 0 means "no entry" to the matrix.

The symbolic solution of (1) represents symbolic calculation of a determinant by means of Laplace expansion. It is known the number of symbolic terms grows exponentially with the circuit size [2]. Thus, the exact symbolic analysis is practically constrained to very small idealized circuits.

3.2 Approximate Symbolic Analysis

Symbolic analysis can be performed for larger circuits too but at the expense of approximation. The expression generated loses its universality [2]. Generally, the symbolic approximation is based on the knowledge of numerical contribution of an individual symbolic term to numerical value of the whole expression. The influence is usually checked on a set of user defined frequencies. The terms with low influence are discharged. Unlike the classical symbolic analysis at least approximate numerical values of all parameters of network devices are required. It somehow mimics simplifications done by a designer during "hand" analysis of a network but with exact control over approximation errors.

The process of simplification can be applied to network equations or to the symbolic expression already generated. The former approach is called Simplification Before Generation (SBG) and the latter one is called Simplification After Generation (SAG) [3]. Both methods are implemented in Snap.

Equation (1) can be simplified in order to get a less complex symbolic solution. The approximation algorithm tests contribution of each parameter if it could be set to zero or infinity, i.e. removed from (1) and consequently from the resulting symbolic solution [2]. If the element was, for example, a resistor, it is equivalent to removing the part or shorting its terminals. Generally, the operations with elements of **H** are equivalent to the omission of negligible currents or voltages and not only to the whole device removal [4].

Each network function can be expressed as a ratio of two algebraic cofactors

$$F = (-1)^{\alpha} \frac{\det(\mathbf{H}_1)}{\det(\mathbf{H}_2)} \ . \tag{2}$$

Matrices $\mathbf{H}_1$ and $\mathbf{H}_2$ are derived from **H** by means of adding and deleting some rows and columns. Details can be found in any textbook of classical circuit theory. Each determinant can be expanded by any element h_{ij} of matrix **H** as

$$\det(\mathbf{H}_k) = \Delta' + h_{ij}\Delta_{i':j'} \ , \tag{3}$$

where Δ' is the determinant without h_{ij}, and $\Delta_{i':j'}$ is a cofactor of $\mathbf{H}_k$ (h_{ij} has different coordinates in $\mathbf{H}_1$ or $\mathbf{H}_2$), $k = 1$ or 2. The network parameters are considered different for the simplification process despite the fact that they can be marked with the same symbol. Network function (2) can be written as

$$F = \frac{a_{ij}h_{ij} + b_{ij}}{c_{ij}h_{ij} + d_{ij}} \ . \tag{4}$$

Now, it is easy to determine how F is changed for $h_{ij} \to 0$ or $h_{ij} \to \infty$.

The approximation algorithm tests both operations for all elements, records the error, and then removes the least significant elements by performing the already tested operation. This is repeated until the user-specified maximum error is reached.

Let F_N be the original network function and F_A the approximated one. A reasonable error criterion can be defined as

$$E(f) = \frac{F_A(f)}{F_N(f)} \ , \tag{5}$$

giving $E = 1$ for $F_A = F_N$. In order to compare influences of all elements removal, the criterion must be further transformed into a real number by means of the weighted sum of magnitude and phase errors

$$e(f) = \frac{\left|20\log\left|E(f)\right|\right|}{\Delta M(f)} + \frac{\left|\arg(E(f))\right|}{\Delta P(f)} \ , \tag{6}$$

where $\Delta M(f)$ and $\Delta P(f)$ are frequency-dependent weights. The symmetrical criterion gives $e = 0$

for $F_A = F_N$.

After the matrix simplification the symbolic solution of (1) is generated, fig. 3. The expression can be further simplified by means of SAG algorithms. A sensitivity based method is implemented in Snap as the most powerful one [3].

Fig. 7 shows graphical interface of the approximate analysis module. The user can see the original and the approximated functions. Each design point can be configured with different maximum magnitude and phase error.

4 Example

Fig. 6 shows a simple two-transistor amplifier. Students designed all network parameters in order to meet DC and LF specs (not shown in the figure). The task is to determine which phenomenon determines the amplifier dominant pole by means of approximate symbolic analysis.

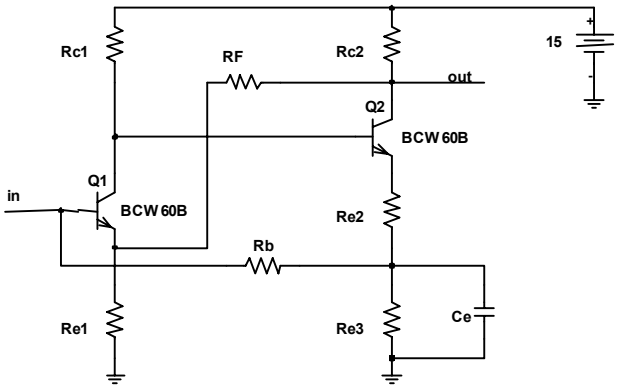


Fig. 6 Two-transistor amplifier.

The analysis starts with drawing the amplifier in PSpice Schematics using standard library devices. Then DC operating point is computed and the translation module extracts all small-signal parameters provided by PSpice.

The full expression for voltage transfer represents approximately 10 pages. Using two design points $f_1 = 1\text{kHz}$; $f_2 = 2\text{MHz}$ with the maximum allowed error of $\Delta M = 1\text{dB}$ and $\Delta P = 5°$ a simplified expression shown in fig. 8 was obtained. The first-order function shows clearly the dominant pole is determined by the C_μ capacitance of Q2 (multiplied by the Miller effect).

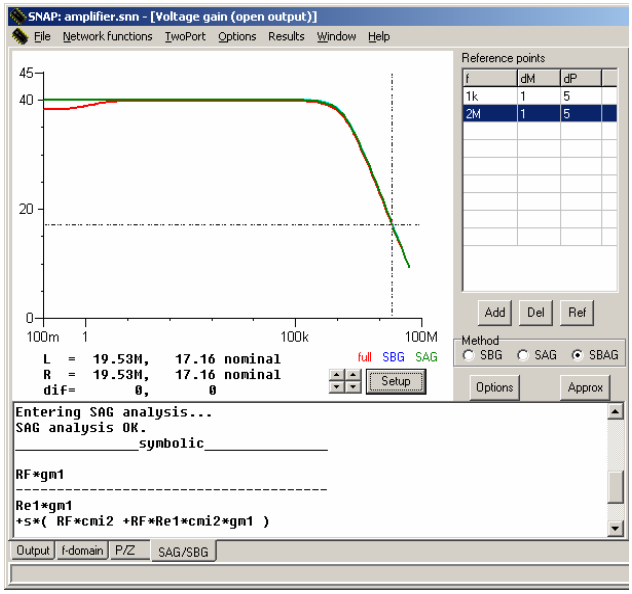


Fig. 7 Graphical user interface of Snap – module for approximate symbolic analysis.

6 Conclusions

The method of submatrix inversion recycling fully exploits sparsity of matrices for larger electronic circuits. It requires fewer operations than standard full-matrix approaches.

7 Acknowledgement

This research has been supported by the Grant Agency of the Czech Republic under the contract no. 102/05/0771.

References:

[1] VLACH, J.–SINGHAL, K.:, *Computer Methods for Circuit Analysis and Design*, New York: Van Nostrad Reinhold, 1994

[2] SOMMER, R. et al.: *Equation-Based Symbolic Approximation by Matrix Reduction with Qualitative Error Prediction*, Alta-Frequenza Rivista di Elettronica, Vol.5, No.6, 1993, pp.29-37

[3] DAEMS, W.-WAMBACQ, P.-SANSEN, W.: *Evaluation of Error-Control Stategies for the Linear Symbolic Analysis of Analog Integrated Circuits*, IEEE Trans. on Circ. and Syst.-I, vol. 46, no. 5, 1999, pp. 594-606

[4] KOLKA, Z.: *Symbolic Analysis of Transistor-level Models*, Proc. of the EDS 2002 conference, Brno, 2002, pp. 246 - 249