Passive Circuit Synthesis using Genetic Algorithms in MATLAB

VLADISLAV DUREV, ELISSAVETA GADJEVA
Department of Electronics
Technical University of Sofia
8 Kl. Ohridski blvd., 1000 Sofia
BULGARIA

Abstract: - An approach to passive circuit synthesis in MATLAB is proposed in the present paper. The Modified Nodal Analysis is used for the construction of the admittance circuit matrix. A circuit of passive low-pass filter is used for the demonstration of the method. The algorithm is easy to implement in MATLAB environment. A methodology for the verification of the developed approach is proposed together with structural and structural-parametric optimization of the circuit.

Keywords: - Genetic Algorithm (GA), Modified Nodal Analysis (MNA), Circuit Simulator, Circuit Synthesis

1 Introduction
The fast growth of the microelectronic technologies requires effective and flexible tools for modeling and simulation. The computer models of the electronic elements and circuits should be very precise and close to reality. This guarantees the successful implementation of the devices and reduces the costs for redesign and production. An important problem that calls for solution is the automatic creation of a circuit model for a given circuit characteristic, for example the frequency response. The genetic algorithms (GA) [1,2,3] are very applicable in circuit synthesis because this is a problem with a number of variables with different ranges, which can be varied to achieve a certain characteristic. The space of search of the genetic algorithm depends only on the range of variation of the input independent variables. In the present paper a general approach to passive circuit synthesis is developed and compared to the results, obtained via similar methods [1,3]. The presented optimization approach is applicable and with a good accuracy. It is based on computer models, developed using the possibilities of MATLAB and GA toolbox [4].

2 General Concept of the Genetic Algorithms
The GA is a global search evolutionary method, which takes its idea from the natural selection. The individuals, which are often represented as binary strings, are selected according to their fitness values. The fitness values estimate the individuals if they are best fitted or not according to a given purpose function. At the end, only the best solution for the individuals remains in the final population.

There is a variety of examples for multiple variable optimization in microelectronics. The geometric technology parameters for a given technology can be optimized to achieve given electrical characteristics [5]. The structure of the circuits can also be optimized as well as the values of the components. The main advantage of the GA is that they do not need information or other knowledge for the investigated problem. Only the definition of the purpose function is important for realizing the effective global search.

The basic genetic algorithm consists of the following stages [4]: Create population, Assign fitness, Selection, Recombination, Mutation, Evaluation, Reinsertion.

3 Modified Nodal Analysis
The Modified Nodal Analysis (MNA) [6] is based on the standard nodal analysis. The circuit elements without admittance description are defined by the corresponding component equations. As a result, the circuit matrix order increases, but MNA does not impose restrictions on the element types and is very applicable for circuit description with programming language. MNA fully describes the nature and structure of the circuit using several matrices. The MNA equations have to be formulated (represented in a computer program) automatically in a simple, comprehensive manner. Once formulated, the system of equations has to be solved. There are two main aspects to be considered when choosing
algorithms for this purpose: accuracy and speed. The MNA has been proved to accomplish these tasks. In this paper the MNA is applied to circuits with passive elements and independent sources.

4 Structural Synthesis of a Low-Pass Filter

A PSpice [7] model of a passive low-pass filter is presented in Fig. 1 and its frequency response is presented in Fig. 2 [1]. The resistor $R_s$ represents the internal resistance of the voltage source and $R_l$ is the load resistor. The input voltage source $V_{in}$ and the resistors $R_s$ and $R_l$ are fixed in the structure, i.e. the structure optimization will be performed only for the components $L_l$, $L_c$, $C_j$, $C_h$, $C_e$, and $C_k$. The circuit has five nodes including the ground node GND (lp1, lp2, lp3, lp4 and GND). The GA makes the difference between the initial frequency response and the optimized frequency response minimal. The independent input variables for the structural synthesis are the nodes of the circuit, excluding the GND node. The range of variation for every node is from 0 (GND) to 4 (number of nodes excluding the GND node). The component values for the structural synthesis are fixed and only the node numbers are optimized. The initial values of the model parameters for the circuit in Fig. 1 are:

$\begin{align*}
R_s &= 1 \\
R_l &= 1 \\
L_l &= 0.9769 \\
L_c &= 1.2647 \\
C_j &= 1.3708 \\
C_h &= 0.4087 \\
C_e &= 2.1530 \\
C_k &= 1.9517
\end{align*}$

The initial parameters of the algorithm are:

$\begin{align*}
NIND &= 200 \\
MAXGEN &= 200 \\
NVAR &= 18 \\
PRECI &= 200 \\
GGAP &= 0.7
\end{align*}$

where [4]:

![Fig. 1. Initial circuit of the low pass filter](image)

The frequency response in Fig. 2 is an input for the optimization program in MATLAB and it is stored in Microsoft Excel format .xls. All the output data from the program are stored in .xls format. The FieldD matrix is used to fix the ranges of the input variables [4].

```
% N1_1
FieldD = [PRECI                          … 
          0                                   …
          NODES_NUMBER-1 …
          1                                   …
          0                                   …
          1                                   …
          1                                   …];
```

NODES_NUMBER represents the number of the nodes in the circuit including the GND node. For the structural optimization the algorithm varies only the incident nodes for each of the components from 0 (GND) to (NODES_NUMBER – 1). As the node values of $V_{in}$, $R_s$ and $R_l$ are fixed, the other six components $L_l$, $L_c$, $C_j$, $C_h$, $C_e$, and $C_k$ are to be optimized in the circuit. Each of the components has two pins, which can be connected to the nodes of the circuit, so the number of the variables for the structural synthesis is twelve, i.e. NVAR = 12. The input matrix Input is created for the circuit which is with a PSpice netlist-like structure:

```
Input = [1 1 2 1; 
         2 round(N11(ix)) round(N12(ix)) 0.9769; 
         2 round(N21(ix)) round(N22(ix)) 1.2647; 
         3 round(N31(ix)) round(N32(ix)) 1.3708; 
         3 round(N41(ix)) round(N42(ix)) 0.4087; 
         3 round(N51(ix)) round(N52(ix)) 2.1530; 
         3 round(N61(ix)) round(N62(ix)) 1.9517; 
         1 (NODES_NUMBER-1) 0 1];
```

![Fig. 2. Frequency response of the initial circuit of the low-pass filter](image)
The first column represents the type of the component: resistor – 1, inductor – 2 or capacitor – 3. The last column represents the value of the component. N11 to N62 are the variables, which represent every node in the circuit. An internal cycle calculates the Input matrix for every individual ix, as the independent input variables are vectors of generated values [4]. The values are rounded, because the actual generated numbers are not integer numbers, which is needed for the number of the node representation. The components are not allowed to connect to node lp1, which is fixed:

```plaintext
if(round(N1_2(ix)) == 1
  N12(ix) = 2;
else
  N12(ix) = round(N1_2(ix));
end
```

The Y-matrix of the circuit is constructed on the base of the Input matrix using the MNA. The I- and U-matrices of the circuit are constructed and the matrix circuit equation is solved in the following way:

Y(1, NODES_NUMBER) = 1; %Inserting the voltage source into the matrices
Y(NODES_NUMBER, 1) = 1;
I(NODES_NUMBER, 1) = Vin;
U = pinv(Y)*I;
Vout_real(ix, 1) = real(U((NODES_NUMBER-1), 1)); %The output voltage is captured
Vout_imag(ix, 1) = imag(U((NODES_NUMBER-1), 1));
Vout_module(ix, 1) = abs(U((NODES_NUMBER-1), 1));

`Vout_real`, `Vout_imag` and `Vout_module` vectors are used to build the frequency response of the circuit. Once built, this frequency response is compared to the input frequency response and the GA is optimizing the difference between the two frequency responses to be minimal. The purpose function is presented in the following way:

```plaintext
g_fun = g_fun + abs(Vout_real - real(Vout_initial(i))) + abs(Vout_imag - imag(Vout_initial(i)));
```

The goal is `g_fun` to be zero (0). A simple verification of the algorithm can be performed first, fixing the parameters of the initial circuit from Fig. 1 into the matrices `FieldD` and `Input`. The result for the `g_fun` should be constant 0 for every iteration. The result for several iterations is presented in Fig. 3. Once the verification is performed successfully, the values of the independent variables are controlled from the GA in order to find the structure, which best fits the input requirements. The GA iterations and the achievement of the best value for the purpose function for each of the input variables are presented in Fig. 4 and Fig. 5.

The last generated Input matrix for this optimization is:

```plaintext
Input = [1.00000 1.00000 2.00000 1.00000;
         2.00000 3.00000 4.00000 0.97690;
         2.00000 2.00000 3.00000 1.26470;
         3.00000 0             4.00000 1.37080;
         3.00000 3.00000 4.00000 0.40870;
         3.00000 3.00000 0             2.15300;
         3.00000 0             2.00000 1.95170;
         1.00000 4.00000 0             1.00000];
```

The corresponding circuit is shown in Fig. 6. The frequency response of the initial circuit in Fig. 1 and the frequency response of the optimized circuit, shown in Fig. 6, are presented in Fig. 7. It is interesting to mention that in some GA generations the GA finds exactly the same structure like the one in Fig. 1.

5 Structural-Parametric Synthesis of a Low-Pass Filter

In the structural synthesis only the topology is altered by the GA, whether the component values are fixed. In the structural-parametric synthesis the
The component values are variables and only the type of the components, fixed in the first column of the Input matrix, is constant. The Input matrix has the following representation for optimizing the structure and the component values simultaneously:

\[
\text{Input} = \begin{bmatrix}
1 & 1 & 2 & 1; \\
2 & \text{round}(N11(ix)) & \text{round}(N12(ix)) & N7_1(ix); \\
2 & \text{round}(N21(ix)) & \text{round}(N22(ix)) & N7_2(ix); \\
3 & \text{round}(N31(ix)) & \text{round}(N32(ix)) & N8_1(ix); \\
3 & \text{round}(N41(ix)) & \text{round}(N42(ix)) & N8_2(ix); \\
3 & \text{round}(N51(ix)) & \text{round}(N52(ix)) & N9_1(ix); \\
3 & \text{round}(N61(ix)) & \text{round}(N62(ix)) & N9_2(ix); \\
1 & \text{NODES\_NUMBER}-1 & 0 & 1;
\end{bmatrix}
\]

The variables N7_1 to N9_2 represent the component values. In the FieldD matrix the following ranges for the component values are defined:

\[
\begin{align*}
N7_1 &= 0.8 \div 1.2 \text{ represents } Ll \\
N7_2 &= 1.0 \div 1.5 \text{ represents } Lc \\
N8_1 &= 1.0 \div 1.6 \text{ represents } Cj \\
N8_2 &= 0.2 \div 0.6 \text{ represents } Ch \\
N9_1 &= 2.0 \div 2.3 \text{ represents } Ce \\
N9_2 &= 1.8 \div 2.2 \text{ represents } Ck
\end{align*}
\]

All other conditions are the same as in the structural synthesis implementation of the GA. The GA iterations and the achievement of the best value for the purpose function for each of the input variables are presented in Fig. 8, Fig. 9 and Fig. 10. The last generated Input matrix for this optimization is:

\[
\text{Input} = \begin{bmatrix}
1.00000 & 1.00000 & 2.00000 & 1.00000; \\
2.00000 & 3.00000 & 4.00000 & 1.14803; \\
2.00000 & 3.00000 & 0 & 1.19819; \\
3.00000 & 0 & 2.00000 & 1.29676; \\
3.00000 & 3.00000 & 2.00000 & 0.32001; \\
3.00000 & 3.00000 & 0 & 2.15922; \\
3.00000 & 0 & 4.00000 & 1.87362; \\
1.00000 & 4.00000 & 0 & 1.00000;
\end{bmatrix}
\]
The corresponding circuit is shown in Fig. 11. Another interesting circuit from one of the GA generations is shown in Fig. 12. The structure of the circuit is the same and the component values are almost the same like the ones in Fig. 1. The frequency response of the initial circuit (Fig. 1) and the frequency responses of the optimized circuits from Fig. 11 and Fig. 12 are presented in Fig. 13.

6 Conclusion
A GA optimization approach to passive circuit synthesis has been developed in the present paper. The synthesis procedure is realized using the GA toolbox in MATLAB. The obtained results are in agreement with previously published passive circuit synthesis optimizations and have a very good accuracy.

References: