First International Conference
Modelling and Development of Intelligent Systems
Sibiu - Romania, 22-25 October, 2009

An Approach to Analog Circuit Synthesis using Genetic Algorithms

Vladislav Durev, Elissaveta Gadjeva

Abstract

An approach to analog circuit synthesis using Genetic Algorithms (GA) is proposed in the present paper. The Modified Nodal Analysis (MNA) is used for the construction of the admittance circuit matrices. A circuit of Operational Amplifiers (OA) based high-pass filter and a circuit of band-pass filter are used for the demonstration of the method. The algorithm is implemented in MATLAB environment.

1 Introduction

The automated synthesis of analog circuits is an important problem that calls for solution in the recent years and many papers are published with different approaches, including the use of evolutionary algorithms. There is not a common approach to analog circuit synthesis, but different approaches, applied to different circuit types and applications [1-4].

As the analog circuit synthesis needs a mathematical representation of the circuit, which has a number of variables with different ranges of variation, it is reasonable to use GA [5] to aid the synthesis procedure.

An approach to automated analog circuit synthesis, based on the application of Genetic Algorithms (GA) is presented in the present paper. The software is implemented in MATLAB using the GA Toolbox [6] functionality. The approach optimizes both the structure and the parameter values in the analog circuit. It constructs the mathematical model of the circuit using MNA [7] and the embryonic electrical circuit concept [1, 8]. 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 the structure of the circuit using several matrices.

Two examples of synthesized circuits are presented – a third order unity-gain Bessel high-pass filter, based on OA and an asymmetric band-pass filter.

2 Application of GA in the synthesis of Operational Amplifier (OA) based circuits

The schematic of a third order unity-gain Bessel high-pass filter, based on OA, is shown in Fig. 1. The frequency response of this schematic is used as an input data for the GA, which is programmed to synthesize a schematic, based on the input data, according to a given purpose function. The task is to design a third-order unity-gain Bessel high-pass filter with the corner frequency f_c = 1 kHz.

The *Input* matrix is introduced, which gives the connection between the components types and values and the structure of the schematic. The structure of the *Input* matrix is like the structure of *PSpice* netlist file:

$$[Input] = \begin{bmatrix} type\_1 & node\_11 & node\_12 & value\_1 \\ type\_2 & node\_21 & node\_22 & value\_2 \\ ... & ... & ... & ... \\ type\_i & node\_i1 & node\_i2 & value\_i \end{bmatrix} \quad (1)$$

Every row from this matrix represents one passive component in the schematic, which is of *type_i* (resistor – 1, inductor – 2 or capacitor – 3), between the nodes *node_i1* and *node_i2*, with value *value_i*. All the variables in the *Input* matrix can be optimized from the GA or have fixed values. It depends on the user and the application.

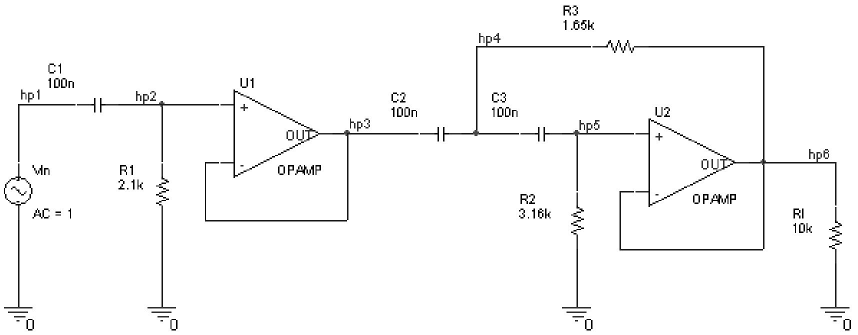

Fig. 1. Initial schematic of a third order unity-gain Bessel high-pass filter, based on OA

The *FieldD* matrix [6] determines the range of variation for every variable in the *Input* matrix:

$$[FieldD] = \begin{bmatrix} PRECI & ... & PRECI \\ 0 & ... & COMP\_VALUE\_\min \\ NODES\_NUMBER-1 & ... & COMP\_VALUE\_\max \\ 1 & ... & 1 \\ 0 & ... & 0 \\ 1 & ... & 1 \\ 1 & ... & 1 \end{bmatrix} \quad (2)$$

The value of *PRECI* determines the precision factor for representing the numbers in the GA, *NODES_NUMBER* contains the number of the nodes in the schematic. The number of the columns in the *FieldD* matrix corresponds to the number of variables, which are optimized by the GA. The first column in (2) represents a number of a component node, whose value is varied from 0 (*GND*) to (*NODES_NUMBER*-1). It is assumed that the number of the components is equal to the number of nodes. If the two terminals of a passive component are connected to one and the same node, this component is excluded from the generated schematic. The last column in (2) represents a component value, which will be varied by the GA in the range (*COMP_VALUE_min* ÷ *COMP_VALUE_max*). Thus, the ranges of variation of the variables in the *Input* matrix are fixed in the *FieldD* matrix [6].

The admittance *Y*-matrix of the schematic is built from the *Input* matrix, using MNA. The values, generated for the components types and the nodes numbers are rounded. The influence of the OAs is defined in the *Y*-matrix and the influence of the source of the schematic is defined in the *Y*-matrix and the *I*-matrix. The nodes voltages are obtained in the *U*-matrix, using the equation (3) [7]:

$$[U] = [Y]^{-1}[I] \tag{3}$$

In the following example the input source *Vin* and the load resistor *Rl* are fixed using the embryonic circuit concept [1, 8]. Moreover, the components *C*1, *R*1, *U*1 and *U*2 are fixed and the rest of the components are optimized by the GA (Fig. 1). The following variation ranges are written in the *FieldD* matrix for the components *R*2, *R*3, *C*2, *C*3 correspondingly:

*N*5_1 = 0 ÷ 5 kΩ
*N*5_2 = 0 ÷ 5 kΩ
*N*6_1 = 0 ÷ 200 nF
*N*6_2 = 2 ÷ 200 nF

The variables *N*5_1 ÷ *N*6_2 are members of the *Input* matrix, together with the numbers of the nodes of the optimized components:

```
Input = [1  2                    0                 2.1e3;
         1  round(N11(ix))       round(N12(ix))  N5_1(ix);
         1  round(N21(ix))       round(N22(ix))  N5_2(ix);
         3  1                    2               100e-9;
         3  round(N31(ix))       round(N32(ix))  N6_1(ix);
         3  round(N41(ix))       round(N42(ix))  N6_2(ix);
         1  (NODES_NUMBER-1)  0                  10e3];
```

For example the second row of the *Input* matrix represents a resistor, connected between nodes with numbers round (*N*11(*ix*)) and round(*N*12(*ix*)) and resistance value *N*5_1(*ix*) Ω. Only the type of this component is fixed and the nodes numbers and the component value are optimized by the GA. The fourth row represents a capacitor between nodes 1 and 2 with value 100 nF. All the properties of this component are fixed. It depends on the user which values in the *Input* matrix are fixed or optimized. An internal cycle calculates the *Input* matrix for every individual *ix*, as the independent input variables are defined as vectors of generated values, and the number of these values is equal to the number of the individuals in the population [6].

As the OAs *U*1 and *U*2 (Fig. 1) are assumed ideal amplifiers, their impact in the *Y*-matrix, together with the impact of the input source *Vin* is given in (4):

$$[Y] = \begin{vmatrix} \dots & \dots & \dots & \dots & \dots & \dots & 1 & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & 1 \\ 1 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & 1 & -1 & \dots & \dots & \dots & \dots & \dots & \dots \\ \dots & \dots & \dots & \dots & 1 & -1 & \dots & \dots & \dots \end{vmatrix} \tag{4}$$

The corresponding MATLAB code is:

```
Y((NODES_NUMBER+1), 2) =  1; %U1 input +
Y((NODES_NUMBER+1), 3) = -1; %U1 input -
Y((NODES_NUMBER+2), 5) =  1; %U2 input +
Y((NODES_NUMBER+2), 6) = -1; %U2 input -
Y(3, (NODES_NUMBER+1)) =  1; %U1 output
Y(6, (NODES_NUMBER+2)) =  1; %U2 output
```

Similarly, the impact of the input source *Vin* in the *Y*- and the *I*-matrix can be represented, using the following code:

```
Y(1, NODES_NUMBER) = 1;
Y(NODES_NUMBER, 1) = 1;
I(NODES_NUMBER, 1) = Vin;
```

The GA is optimizing the circuit according to a given purpose function. The goal of the purpose function is to minimize the difference between the initial and the optimized circuit, i.e. between the initial frequency response of the circuit in Fig. 1 and the frequency response of the optimized circuit. The expression for the purpose function takes into account the values of the real and the imaginary parts of the initial (*Vout_initial*) and the current (*Vout_current*) output voltage (between the nodes *hp*6 and *GND*, Fig. 1) in the frequency domain, using the least squares values method:

$$G_{fun} = \sum_{i=1}^{n} \left[\Re\left(V_{out\_current}\right) - \Re\left(V_{out\_initial}\right)\right]^2 + \sum_{i=1}^{n} \left[\Im\left(V_{out\_current}\right) - \Im\left(V_{out\_initial}\right)\right]^2 \tag{5}$$

The corresponding MATLAB code for (5) is:

```
g_fun = g_fun + ((real(Vout_current) - real(Vout_initial(i))).*(real(Vout_current) - real(Vout_initial(i)))) + ((imag(Vout_current) - imag(Vout_initial(i))).*(imag(Vout_current) - imag(Vout_initial(i)))); %Least squares approx method
```

The value of the purpose function (5) is optimized, taking into account the following parameters of the GA: number of individuals in the population *NIND* = 200; maximal number of iterations *MAXGEN* = 1000; number of input variables *NVAR* = 12, precision factor *PRECI* = 200, generation gap *GGAP* = 0.7 [6]. The last generated *Input* matrix is:

```
Input = [ 1.00000  2.00000  0.00000  2100
          1.00000  6.00000  4.00000  2069.21630055992
          1.00000  0.00000  5.00000  4435.65198208762
          3.00000  1.00000  2.00000  1e-7
          3.00000  4.00000  3.00000  5.37061759121239e-8
          3.00000  5.00000  4.00000  1.05775362735491e-7
          1.00000  6.00000  0.00000  10000];
```

The circuit, which corresponds to the given *Input* matrix is shown in Fig. 2. The frequency and phase responses of the initial (Fig. 1) and the synthesized (Fig. 2) circuits are given in Fig. 3 and marked with *Initial* and *GA* correspondingly. Both the circuits are simulated in *PSpice* for verification of the proposed approach. An excellent agreement was achieved without visual difference between the frequency and the phase responses.

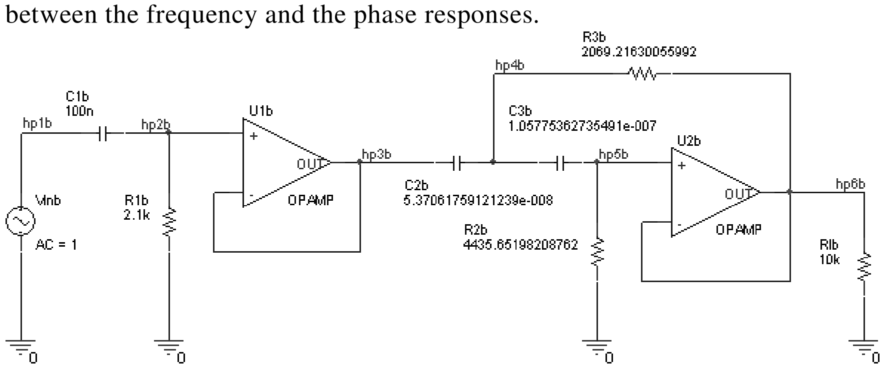

Fig. 2. GA synthesized third order unity-gain Bessel high-pass filter, based on OA

3 Application of GA in the synthesis of asymmetric band-pass filter

The approach used in section 2. can be applied to every passive analog schematic with independent voltage or current sources. Optimization of passive filter in respect to the number of the components is a problem, where the GA can be applied in the synthesis procedure.

The presented approach is applied to the Nielsen filter problem [9], which is widely used to test the effectiveness of the circuit synthesis procedures. Nielsen's band-pass filter is targeted for a modem application where one band of frequencies (31.2 to 45.6 kHz) must be isolated from another (69.6 to 84.0 kHz). This asymmetric filter is difficult to design and the standard design procedure needs 10th order elliptic function. It is shown in [1] that the filter can be synthesized automatically, but the number of components for the best suited circuit is 38. Better result is shown in [4], where the filter is synthesized with only 9 components, but additional numerical optimization is applied together with the GA. The frequency response of this 9-component filter is used as an input data for our approach.

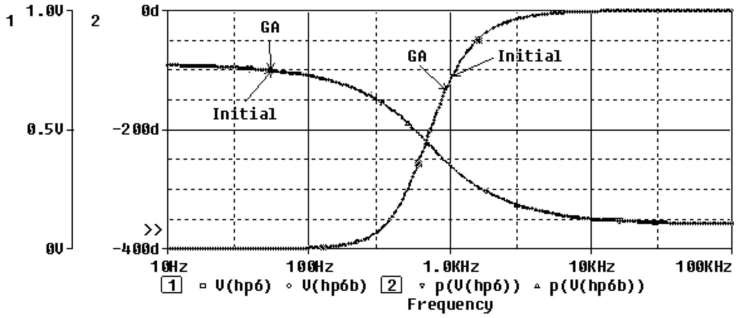

Fig. 3. Frequency and phase responses of the initial circuit from Fig. 1 (Initial) and the GA synthesized circuit from Fig. 2 (GA)

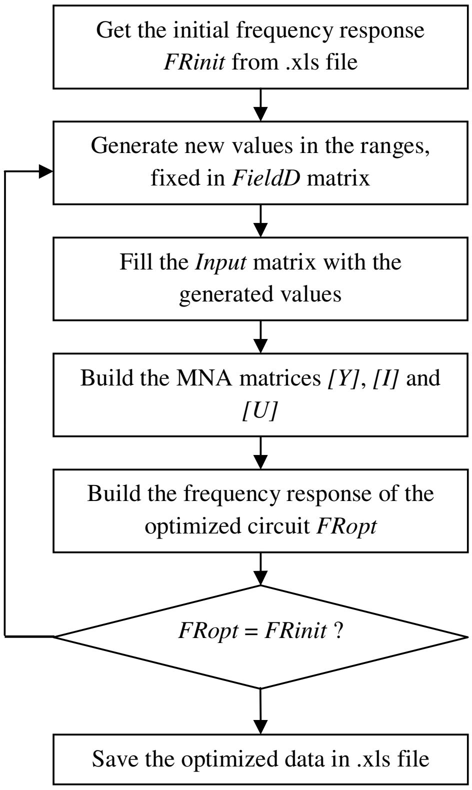

Fig. 4. The sequence of steps, used in the GA-based analog circuit synthesis approach

The expression of the purpose function in our case will optimize the schematic in respect to the frequency response:

$$G_{fun} = W.\sum_{i=1}^{n} \left| \left[M\left(V_{out\_current}\right) - M\left(V_{out\_initial}\right) \right] \right| \tag{6}$$

The parameters of the GA are: *NIND* = 200, *MAXGEN* = 1000, *NVAR* = 27, *PRECI* = 200, *GGAP* = 0.7 for 9 optimized passive components and 7 nodes. The component values are normalized and the range of variation for every component value is (0 ÷ 12). The *Input* matrix is represented as:

```
Input = [ 1  1                      2               1;
          2  round(N11(ix))         round(N12(ix))  N10_1(ix);
          2  round(N21(ix))         round(N22(ix))  N10_2(ix);
          2  round(N31(ix))         round(N32(ix))  N11_1(ix);
          2  round(N41(ix))         round(N42(ix))  N11_2(ix);
          2  round(N51(ix))         round(N52(ix))  N12_1(ix);
          3  round(N61(ix))         round(N62(ix))  N12_2(ix);
          3  round(N71(ix))         round(N72(ix))  N13_1(ix);
          3  round(N81(ix))         round(N82(ix))  N13_2(ix);
          3  round(N91(ix))         round(N92(ix))  N14_1(ix);
          1  (NODES_NUMBER-1)  0                    1];
```

It includes 9 components (5 inductors, 4 capacitors, like the obtained circuit in [4]), whose values and connections are optimized and 2 resistors for termination. The synthesized circuit is shown in Fig. 5.

The frequency responses of the initial circuit from [4] and the synthesized circuit from Fig. 5 are simulated in *PSpice* for verification and the results are shown in Fig. 6.

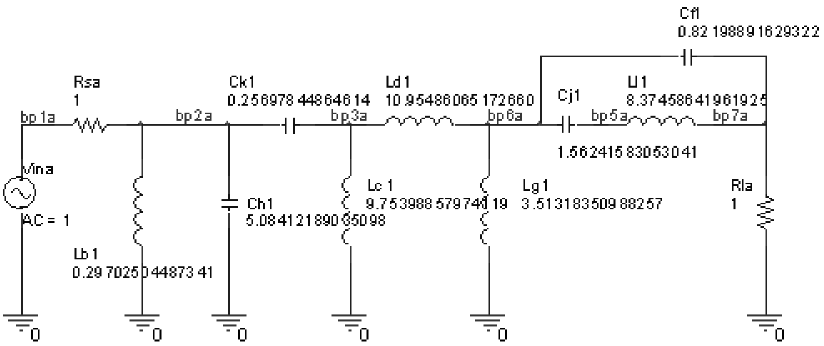

Fig. 5. The synthesized circuit of the band-pass filter

It is obvious from Fig. 6 that the presented approach finds the desired band and its width and it is compatible with the requirements, excluding the attenuation for the upper stop band. This is achieved only using the presented GA approach without any numerical optimization. The GA approach can give enough information though and the filter can fit the constraints if 3 additional poles are added to improve the upper stop band attenuation.

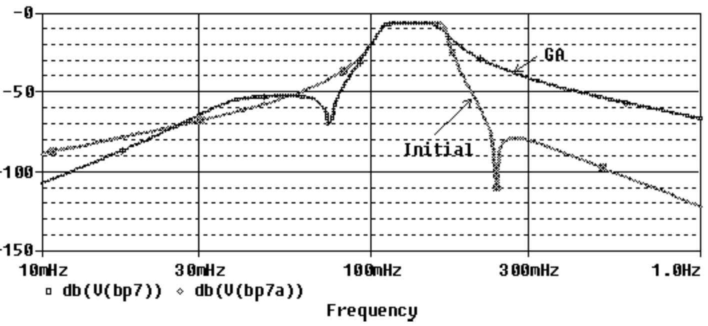

Fig. 6. The frequency responses of the band-pass filter in [4] (Initial) and the frequency response after applying the presented circuit synthesis procedure (GA)

The effectiveness of the presented approach depends mainly on the definition of the purpose function and the definition of the ranges for the input values. The structure of the schematic and the parameter values are optimized simultaneously, but in practice after some experiments is was proved that the structure of the schematic is optimized only during the first 50 iterations.

4 Conclusion

An approach to analog circuit synthesis using GA in the MATLAB environment is presented in the present paper, based on the use of embryonic circuit concept and MNA for building the circuit's equations. The presented procedure takes around 1 hour in order to synthesize a circuit with 10 components for 1000 iterations on Core 2 Duo architecture. It is shown that the obtained results are in agreement with the previous published data. The main advantage of the presented approach is the direct connection to the physical nature of the analog circuit using the MNA representation in the GA body.

The approach is successfully tested for circuits with up to 30 passive components.

References

[1] J. Koza, F. Bennet III, D. Andre, M. Keane, *Four Problems for which a Computer Program Evolved by Genetic Programming is Competitive with Human Performance*, Proceedings of the 1996 IEEE International Conference on Evolutionary Computation, 20-22 May, 1996, Nagoya, Japan, pp. 1-10, ISBN: 0-7803-2902-3.

[2] S. Zhao, L. Jiao, J. Zhao, Y. Wang, *Evolutionary design of analog circuits with a uniform-design based multi-objective adaptive genetic algorithm*, Proc. NASA/DoD Conference on Evolvable Hardware, 29 June-1 July 2005, pp. 26 – 29, ISBN: 0-7695-2399-4.

[3] C. Reinhold, P. Kralicek, W. John , J. Teich, *Synthesizing Passive Networks by Applying Genetic Programming and Evolution Strategies*, The 2003 Congress on Evolutionary Computation, 8 – 12 Dec., 2003, vol. 3, pp. 1740 – 1747, ISBN: 0-7803-7804-0.

[4] J. Grimbleby, *Automatic analogue circuit synthesis using genetic algorithms*, IEE proceedings. Circuits, devices and systems, 2000, vol. 147, no. 6, pp. 319-323, ISSN 1350-2409.

[5] S. Sivanandam, S.Deepa, *Introduction to Genetic Algorithms*, Springer, 2008, ISBN 978-3-540-73189-4.

[6] A. Chipperfield, P. Fleming, H. Pohlheim, C. Fonseca, *Genetic Algorithm Toolbox for use with MATLAB*, User's Guide Version 1.2, Department of Automatic Control and Systems Engineering, University of Sheffield.

[7] J. Vlach, K. Singhal, *Computer Methods for Circuit Analysis and Design*, Kluwer Academic Publishers, 2003, ISBN 0-442-01194-6.

[8] J. Koza, F. Bennett, J. Lohn, F. Dunlap, M. Keane, D. Andre, *Automated Synthesis of Computational Circuits Using Genetic Programming*, IEEE International Conference on Evolutionary Computation, 13 – 16 April, 1997, Indianapolis, IN, USA, pp. 447-452, ISBN: 0-7803-3949-5.

[9] I. Nielsen, *A C-T Filter Compiler – From Specification to Layout*, Analog Integrated Circuits and Signal Processing, vol. 7, issue 1, pp. 21-33, 1995, Kluwer Academic Publishers, ISSN:0925-1030.

VLADISLAV DUREV
Technical University - Sofia
Department of Electronics and Electronic Technology
8 Kliment Ohridski Str., 1000 Sofia
BULGARIA
E-mail: v_p_durev@yahoo.com

ELISSAVETA GADJEVA
Technical University - Sofia
Department of Electronics and Electronic Technology
8 Kliment Ohridski Str., 1000 Sofia
BULGARIA
E-mail: egadjeva@tu-sofia.bg