SOFTWARE IMPLEMENTATION OF METHODS FOR OPTIMAL DESIGN OF POWER ELECTRONIC DEVICES

Bogdan Gilev (1), Nikolay Hinov (2), Stanislav Stefanov (3)*

(1) Faculty of Applied Mathematics and Informatics, Technical University of Sofia;
(2) Faculty of Electronic Engineering and Technologies, Technical University of Sofia;
(3) Faculty of Transport, University of Architecture Civil Engineering and Geodesy, Sofia, Bulgaria

* Corresponding Author, e-mail: hinov@tu-sofia.bg

Abstract: The manuscript presents a software implementation of methods for the optimal design of power electronic devices based on MATLAB. Model-based optimization is one of the effective approaches to guarantee the performance of power electronic devices and systems. Through its application, the efficiency of the conversion of electrical energy is increased when setting a certain target function. Several examples are presented to demonstrate the effectiveness of the proposed optimization.

Key words: optimal design, power electronic converters, math software.

1. INTRODUCTION

Power electronic devices are a major tool in the transition to a carbon neutral society. In this sense, the problems related to their analysis, design, modeling and prototyping are key to the realization of the energy policy of modern countries [1-3]. This determines the huge interest in applying different methods and approaches for their design and operation. Developments in computational mathematics, modeling, data science, hardware, and software provide new methods and tools in the study of electronic transducers, such as the application of various artificial intelligence techniques. The design process is a complex intellectual activity that begins with the formulation of the initial concepts and ends with their realization through the designed real objects. A key stage in the overall design process is the creation of a mathematical model of the designed system, the implementation of this model in a suitable programming environment and the performance of numerical experiments with it. As a result of these experiments, alternative solutions are generated, which makes it possible to choose the best solution in a certain sense. Model-based optimization is one of the effective approaches to guarantee the performance of power electronic devices and systems [4]. The aim of the present work is to present the software implementation of methods for optimal design of power electronic devices based on MATLAB.

2. OVERVIEW OF DIFFERENT OPTIMIZATION METHODS AND ALGORITHMS

2.1. Classification and characteristics of optimization methods

Optimal design in power electronics applies a variety of methods and algorithms to achieve the best balance between various parameters, such as: efficiency, reliability, weight and size indicators, energy consumption, price, and others. In the design of power electronic devices, it is first necessary to choose the appropriate topology of the power circuit, to find the initial values of the circuit elements, then to determine the optimal values of the components, to synthesize and adjust a suitable controller. To achieve the optimal design in power electronics, various methods are used such as: analytical, numerical modeling and simulations, experimental testing, and optimization algorithms. In the optimization process, the operating conditions of the device should also be taken into account, such as temperature, humidity, pressure, vibration and other factors that can affect its operation. Optimization methods are divided into the following three main groups:

- Mathematical optimization methods. These methods use mathematical algorithms to find the best solution within certain constraints. Here are some of the most commonly used mathematical optimization methods: Gauss-Newton method - used to solve least-squares problems where the goal is to find a linear regression that best describes the existing data; Lagrange's method - used to find a conditional extremum of functions when there are constraints; Newton-Raphson method - used to find the exact minimum or maximum of functions; Simplex method - used to find optimal values of multidimensional functions by creating a simplex method; Genetic algorithms - used to optimize complex problems by simulating an evolutionary process to find optimal solutions; Swarm intelligence algorithms - used to optimize functions by reproducing the behaviors of swarms of animals or insects; Stochastic methods - used to optimize functions, using random values to explore the solution space.
- Heuristic optimization methods. These are computer methods for finding solutions to problems where there is no known exact algorithm for finding an optimal solution. They are usually based on the use of various search strategies that try to find a good solution by using heuristics, i.e., general rules, experience, or intuition that are used to limit the set of possible solutions. Heuristic optimization methods include Genetic algorithms – these are optimization methods that are based on the biological process of evolution. They use a process of selection and mutation to create new solutions that are evaluated against quality criteria and compared to previous solutions until the best solution is found; The method of crime points - this is a method that is based on searching for solutions by skipping some points in the possible solution. These points are called crime points and can be used to determine better decisions; Simulated cooling method - this method is based on the physical cooling process. It uses temperature as a parameter to search for solutions, gradually decreasing the temperature to approach the optimal solution; Tabu search method - this method is based on the idea of avoiding certain solutions that have proven ineffective in the past. It uses a "taboo" list of inadmissible solutions that is updated during the search to avoid repetitions of inefficient solutions.

• Metaheuristic optimization methods. These methods are based on observations in nature and are implemented through computer algorithms. Some of the most famous metaheuristic methods for optimization are: Genetic algorithms (Genetic algorithms) - these methods are inspired by the evolutionary process and are based on the concept of natural selection and the passing of genetic information from one generation to another; Method of ants (Ant colony optimization) - this method is based on the behavior of ants in search of food and is particularly effective in solving problems related to routing tasks; Artificial bees (Artificial bee colony) - this method is inspired by the behavior of bees in search of food and is based on three types of bees - workers, extraordinary bees and queen; Artificial immune systems - these methods are inspired by the human immune system and are used to solve optimization problems related to classification and pattern recognition; Simulated annealing - this method is inspired by the cooling process of metals and is used to solve optimization problems related to complex objective functions.

Due to the ever-increasing complexity of optimization procedures, these methods can be combined and modified depending on the specific task to be solved.

2.2. Command in MATLAB environment for parametric numerical

This optimization problem was solved in the MATLAB environment through author's m-file. The optimization is performed with the "fmincon" command [MathWorks, MATLAB Optimization Toolbox™ Users Guige], i.e.

[x,Fval]=fmincon(@Model,x0,[ ],[ ],[ ],xlb,xub,@Constr)

In a subprogram @Model the differential equations are solved with which the model is formally described. For this purpose, it is used the help of solver "ode45". The objective function is also set in the function @Model. This function usually is the sum of squares. The nonlinear constraints are set in the function @Constr. The minimum and maximum values of their parameters "x" are set in "xlb" and "xub".

3. EXAMPLES OF MODEL-BASED OPTIMIZATION IN MATLAB

3.1. Scheme of transformer less resonant DC-DC converter

3.1.1. Mathematical model

The power scheme of resonant DC-DC converter is shown in Figure 1.

The scheme parameters are:

Ud=110 V - input voltage; L=6 μH - inductance in the resonant circuit; C=116 nF - capacitance in the resonant circuit; C_f=20 μF - capacitance in the load circuit; R=5 Ω - load resistance; f=1/T=200 kHz – switching frequency of the transistors.

Kirchhov's laws and the fact that the convertor is a system with a changing structure are used and the following mathematical model of the converter is obtained:

$$L\frac{di}{dt}+u_{Cf}\ sign(i)+u_C=U_d\ control(t);\ C_f\frac{du_{Cf}}{dt}+\frac{u_{Cf}}{R}=i\ \ sign(i);C\frac{du_C}{dt}=i$$
$$control(t)=\begin{cases}-1, \text{ for an odd half period}\\ 1, \text{ for an even half period}\end{cases}\quad\text{and}\quad sign(i)=\begin{cases}+1, \text{ for } i\geq 0\\ -1, \text{ for } i<0\end{cases} \tag{1}$$

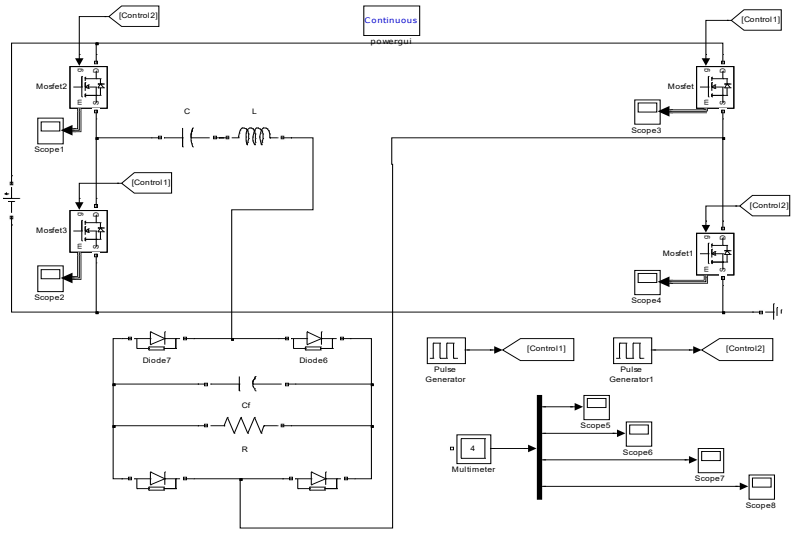

Figure 1. Power schematic in MATLAB environment

Obviously, this model is obtained using differential equations and the switching functions *control(t)* and *sign(i)*.

With the built-in MATLAB command "ode45" you can create code that implements the model (1). Since simulations with the model created in this way are very slow, a different approach is used here.

First, model (1) is reworked in matrix form:

$$\begin{pmatrix} L\dfrac{di}{dt} \\ C\dfrac{du_C}{dt} \\ C_f\dfrac{du_{Cf}}{dt} \end{pmatrix} = \begin{pmatrix} 0 & -1 & -sign(i) \\ 1 & 0 & 0 \\ sign(i) & 0 & -1/R \end{pmatrix} \begin{pmatrix} i \\ u_C \\ u_{Cf} \end{pmatrix} + \begin{pmatrix} U_d\ control(t) \\ 0 \\ 0 \end{pmatrix} \tag{2}$$

$$\text{i.e., } \dot{X} = AX + B \tag{3}$$

where

$$X = \begin{pmatrix} i \\ u_C \\ u_{Cf} \end{pmatrix},\ A = \begin{pmatrix} 0/L & -1/L & -sign(i)/L \\ 1/C & 0 & 0 \\ sign(i)/C_f & 0 & -1/(RC_f) \end{pmatrix} \text{ and } B = \begin{pmatrix} U_d\ control(t)/L \\ 0 \\ 0 \end{pmatrix}.$$

We then divide the simulation time into steps, i.e., $t_{k+1}=t_k+h$. Then the values of the state variables X at these times can be calculated (approximately) by the formula

$$X_{k+1} \approx e^{A_k h} X_k + B_k h \tag{4}$$

where A_k and B_k are the values at the moment t_k. Formula (4) is obtained by simplifying Cauchy's formula. For sufficiently small steps h (for example, on the order of h=T/10) it works very quickly and very reliably.

The code in MATLAB that implements the calculation of $\{X_k\}$ is:

```
Ud = 110; L=6e-6; C=116e-9; Cf=20e-6; R=5; tp =5e-006
h=1e-7;
t=0:h:4e-4;
n=length(t)
x=[0;0;0];
xx(1:3,1)=x;
for k=2:n
   contrI=sign(xx(1,k-1));
   contrU=sign(sin(t(k-1)*2*pi/tp));
   contrU2=sign(sin((t(k)+h)*2*pi/tp));
   A=[ 0,  -1, -contrI; 1, 0, 0; contrI, 0, -1/R];
   B=[contrU*Ud; 0 ; 0];
   AA=diag([1/L,1/C,1/Cf])*A;
   BB=diag([1/L,1/C,1/Cf])*B;
   Amat=expm(AA*h);
   x=Amat*x+h*BB;
   xx(1:3,k)=x;
end
```

By itself, this code would not be very useful, because in a Simulink/MATLAB environment, direct simulations can be done /fig.1/ and the values of the state space variables can be obtained. In the next point, however, this code will be included in an optimization procedure (based on the command fmincon) and through which an optimal (in a certain case) value of the filter capacitor C_f will be found.

3.1.2. Optimization problem

Optimization is the process of obtaining the best result in a certain sense under given constraints. In electronic converters, these are most often: minimal losses, critical-aperiodic transition process, the realization of certain dynamics, set operating mode and so on.

We will aim to realize certain dynamics that should be evaluated by criterion I(x), which is the root-mean-square error between a reference and an actual realized trajectory of a state variable. The criterion I(y) depends on the choice of some parameter y of the model.

After simulating the model (1) for the voltage u_C we get the result shown in Figure 2. Apparently, the u_C peak during the transient exceeds approximately 5 times the u_C set value. This overvoltage is dangerous to circuit elements. For this, the following optimization task is formulated - limiting the voltage u_C at start-up, through the optimal selection of the filter capacitor C_f and at the same time preventing large ripples of the load voltage in the established mode.

Because u_C oscillates there is no way to set a suitable reference curve for u_C to follow. To solve this problem, we connect all the voltage maxima and form the curve shown in Figure 2. We will call this curve the “wrap” of the voltage and denote it by wrap(u_C).

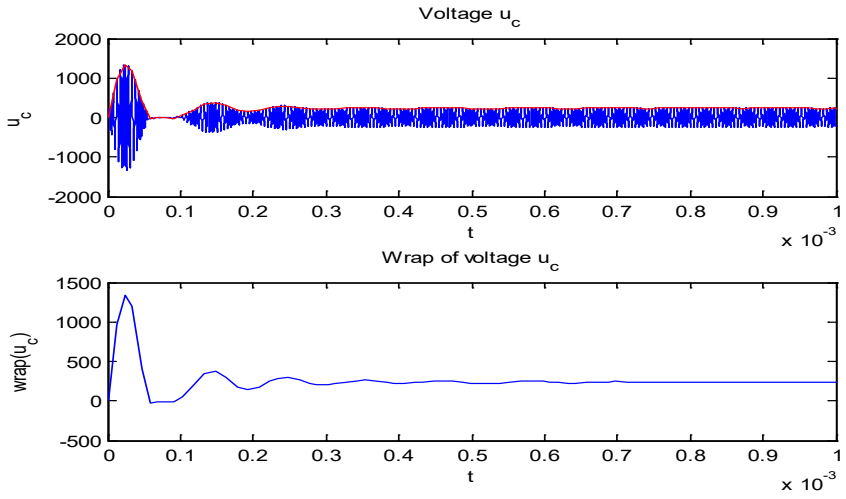

Figure 2. Voltage u_C and its wrap(u_C)

These is done with the code:

```
k=1;
x2(k)=0;
t2(k)=0;
for i=2:length(t)-1
   if ( (xx(2,i-1)<xx(2,i)) && (xx(2,i)>xx(2,i+1)) )
      k=k+1;
      x2(k)=xx(2,i);
      t2(k)=t(i);
      if ( (t2(k)-t2(k-1))<1e-5 ),   k=k-1;    end
   end
end
x3=interp1([t2,1e+4],[x2,233],t);
```

Now we can make the voltage "envelope" follow a selected reference trajectory. For this purpose, we choose the reference trajectory of u_{C_ref} shown in Figure 3.

The analytical expression of the reference trajectory is:

$$u_{C\,ref} = 237(1 - e^{-t/T_{\text{trans}}}), \text{ for } T_{\text{trans}} = 1.10^{-4}.$$

where 237 is the established value for u_C, and 1e-4 is the time for which 40% of the established value is reached.

The selected trajectory will be used to search for a suitable value of the filter capacitor C_f so that the difference between the reference and the wrap of u_c is minimal. For this purpose, the functionality is minimized:

$$I(C_f) = \int_0^{t_{end}} \left(wrap(u_C) - u_{C,ref}\right)^2 dt \underset{Cf}{\rightarrow} \min \qquad (5)$$

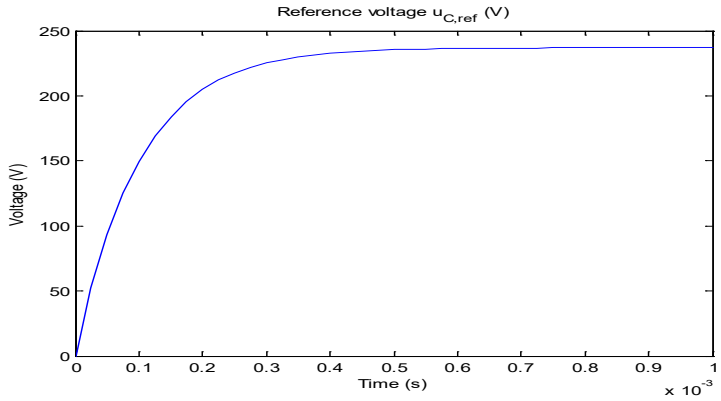

Figure 3. Reference for wrap of u_C

This optimization problem is solved with two constraints of inequality type, which formulate a region of variation of value of output filter capacitor C_f.

$$C_{\min} - C_f \le 0 \; and \; C_f - C_{\max} \le 0 \qquad (6)$$

We will repeat again that this optimization problem cannot be solved with a built-in procedure in MATLAB because the voltage u_C oscillates.

The task is solved in the MATLAB environment, and for this purpose an author's program (m-file) is compiled. The program includes: the objective function (2), equality type constraints - differential equations (1) and inequality type constraints (3). The optimization itself is done with the command

val=fmincon(@OptFun,100e-7,[ ],[ ],[ ],[ ],20e-7,20e-6,[ ],options).

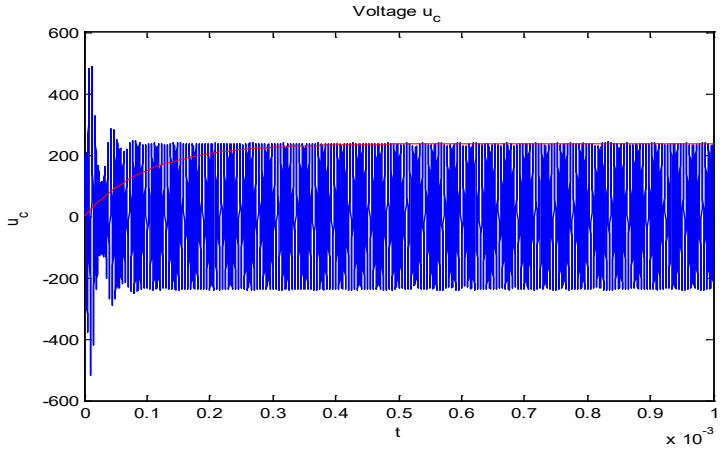

Figure 4. The voltage u_C- blue, the reference wrap of u_C - red

In the program, the differential equations are solved with the help of formula (4) and, accordingly, its code. The wrapper is separated with the code shown next. Integral (3) is replaced by a sum of squares of the form $I = \sum(\quad)^2$, where the differences

between points of the wrap of and the etalon of this wrap are included in parentheses. The corresponding code in the program is:

```
S=0;
for n=1:length(t)
   S=S+(x3(n)-237*(1-exp(-t(n)/1e-4)))^2;
end
```

After executing the program, the result is shown in Figure 4. The resulting optimal value for C_f = 2.0382e-007F.

3.2. Scheme of ZVS single-ended DC/AC converter

3.2.1. Mathematical model

The power scheme of single-ended DC/AC converter is shown in Figure 5.

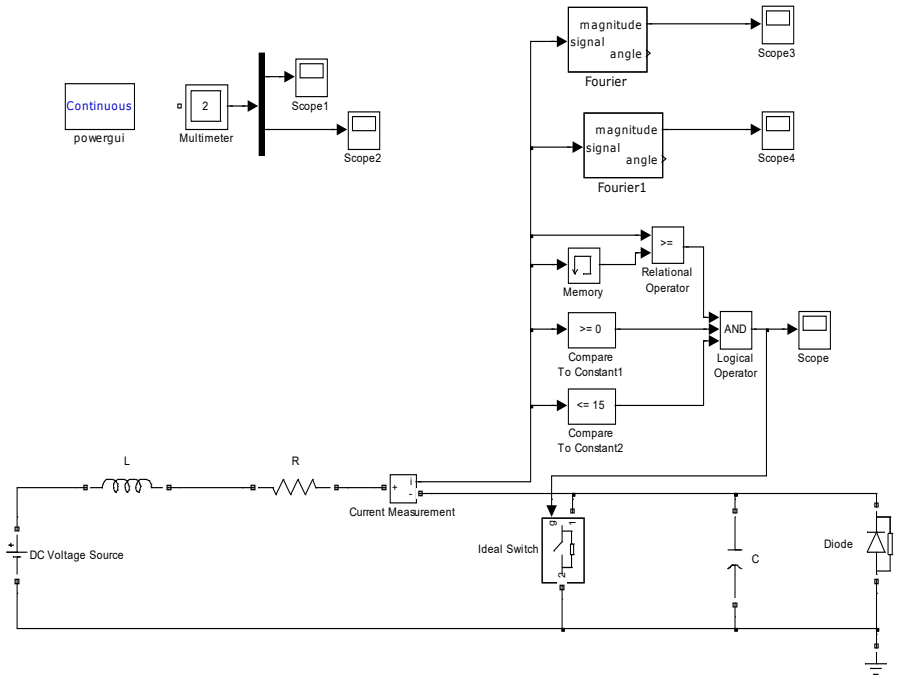

Figure 5. Power schematic in MATLAB environment

The scheme parameters are:

Ud=25 V - input voltage; L=12 μH - inductance in the resonant circuit; C=1 μF - capacitance in the resonant circuit; R=5 Ω - load resistance; f=1/T=25 kHz – switching frequency of the transistors.

Kirchhov's laws and the fact that the convertor is a system with a changing structure are used and the following mathematical model of the converter is obtained:

$$
\begin{aligned}
&L\frac{di}{dt} + R\,i + u = U_d \\
&C\frac{du}{dt} = i \;\; control(t) \\
&control(t) = \begin{cases} 1, & for \quad \left((i > 0 \; and \; i < 15 \; and \; i \uparrow) or (u < 0 \; and \; i < 0)\right) \\ 0, & otherwise \end{cases}
\end{aligned}
\tag{7}
$$

Again, the mathematical model is obtained using differential equations and the switching function *control(t)*.

In contrast to the previous scheme /fig.1/ here the built-in MATLAB command "ode45" works fast enough and reliable. For this, model (7) is implemented using this command and the following code:

```
global R L C Ud
L=10e-6, C=1e-6, R=1, Ud=25
X0=[0;0];
options=odeset('MaxStep',1e-6);
[t,y] = ode45(@SS,[0 5e-3], X0,options);
function dx = SS(t,x)
global R L C Ud
contr=1;
if (((-x(1,1)*R-x(2,1)+Ud)>=0 & x(1,1)>=0 & x(1,1)<=15) | (x(1,1)<=0 & x(2,1)<=0)),
contr=0; end
dx(1,1) =(-x(1,1)*R-x(2,1)+Ud)/L;
dx(2,1) =(contr*x(1,1))/C;
```

We will use this code again by embedding it in an appropriate optimization procedure.

3.2.2. Optimization problem

The scheme of Figure 5 is characterized by the fact that it does not have a transient respond. Therefore, no unwanted (dangerous) overvoltages can occur here during the transient respond. For this, here we consider another optimization task where we aim to maintain the most favorable operating mode during the established mode. Specifically, we want to find values of the resonant elements L and C, for which the first harmonic of the current is maximal, i.e., we want to maximize the functionality:

$$
I(L,C) = \frac{2\pi}{T}\left(\left(\int_0^T i(L,C)\sin\omega t dt\right)^2 + \left(\int_0^T i(L,C)\cos\omega t dt\right)^2\right) \underset{L,C}{\rightarrow} \max
$$

which is equivalent to minimizing the functional:

$$
I(L,C) = -\left(\int_0^T i(L,C)\sin\omega t dt\right)^2 - \left(\int_0^T i(L,C)\cos\omega t dt\right)^2 \underset{L,C}{\rightarrow} \min \tag{8}
$$

The code with which we calculate the functional (8) is:

```
global L C R Ud f
```

```
[tt,xx] = ode45(@SS,[0 5e-3],y0,options);
a=quad(@(x)funA(x,tt,xx),0,T)*2/T
b=quad(@(x)funA(x,tt,xx),0,T)*2/T
J=-sqrt(a^2+b^2)
 % Fourier coefficients
function a=funA(t,tt,xx)
global L C R Ud f
 ff=interp1(tt,xx(1,:),t);
a=(ff).*cos(2*pi*f*t);
  % Fourier coefficients
function b=funB(t,tt,xx)
global L C R Ud f
 ff=interp1(tt,xx(1,:),t);;
b=(ff).*sin(2*pi*f*t);
```

The natural constraints (type inequality) are imposed

$$L_{\min} \le L \le L_{\max} \quad , \quad C_{\min} \le C \le C_{\max} \tag{9}$$

From the point of view of achieving good design parameters, the additional restrictions are also imposed:

$$\max(u) - 3U_d < 0\,;\ R - 2\sqrt{L/C} < 0\,;\ 0.25 - \nu < 0\,;\ \nu - 0.65 < 0$$

where

$$\nu = \frac{2\pi f}{\sqrt{\frac{1}{LC} - \left(\frac{R}{2L}\right)^2}} \tag{10}$$

Analogously to before, we get the following optimization task: to minimize the objective function (8), under the constraints of the type of equalities - the differential equations (7) and the type of inequalities (9-10). The task is solved in the MATLAB environment, and for this purpose an author's program (m-file) is compiled. The optimization itself is done with the built-in fmincon command and the code:

```
xlb=[1e-6;  0.1e-7];
xub=[100e-6; 10e-7];
x0=[L; C];
options= optimset('fmincon');
options = optimset(options,'Display','iter');
[x,Fval]=fmincon(@Optim,x0,[],[],[],[],xlb,xub,@Const,options)
```

n the subprogram “Optim”, the objective function is set. It is implemented with the code associated with formula (8). Constraints (10) are set in the “Const| subprogram, they are implemented with the code:

```
function [Cnoeq,Ceq]=Const(x)
global L C R Ud f
L=x(1)
C=x(2)
y0=[0,0];
options=odeset('MaxStep',1e-6);
[tt,xx] = ode45(@SS,[0 5e-3],y0,options);
Xmax=max(xx(:,2))
```

```
omega0=sqrt(1/C/L-(R/L/2)^2)
ni=2*pi*f/omega0
Cnoeq=[ Xmax-3*Ud, R-2*sqrt(L/C), 0.25-ni, ni-0.65 ]
Ceq=[ ];
```

After the execution of the program, the result shown in Figure 6 is obtained.

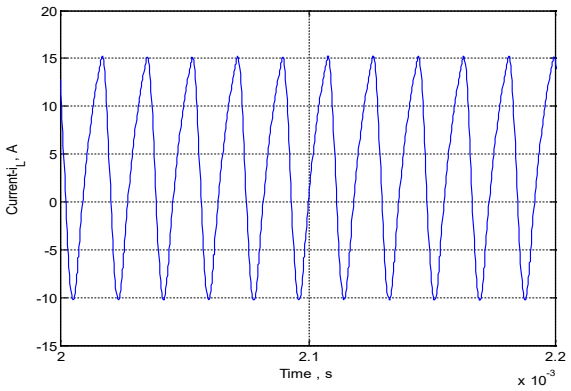

Figure 6. Current in the resonant circuit

The result of the program execution is:

Max Iter	F-count	f(x)	Line search constraint	Directional steplength	First-order derivative	optimality	Procedure
0	3	-13.8077	122.6				
1	6	-11.3932	58.99	1	8.34e+005	3.14e+007	
2	9	-8.65327	18.24	1	-2.28e+006	4.16e+007	
3	12	-3.8208	2.582	1	2.53e+006	1.24e+007	
4	15	-3.60795	0.05686	1	1.77e+006	1.65e+006	
5	18	-3.61701	3.173e-006	1	9.42e+005	1.64e+006	

fmincon stopped because the size of the current search direction is less than twice the default value of the step size tolerance and constraints were satisfied to within the selected value of the constraint tolerance.

Active inequalities (to within options.TolCon = 1e-006):

Lower	Upper	Ineglin	Inegnolin
2	2	4	

The optimal values obtained are: L = 8.3197e-006H and C = 5.0671e-007F.

4. CONCLUSION

The manuscript provides research related to the software implementation of optimal design of power electronic devices. Two specific examples are discussed, through which the applicability of the model-based optimization method is demonstrated. The obtained results prove the possibilities of applying modern information and communication technologies for designing in power electronics. This is useful both for application by power electronics designers and for training needs.

ACKNOWLEDGEMENT

This work was supported by the European Regional Development Fund within the Operational Programme "Science and Education for Smart Growth 2014 - 2020" under the Project CoE "National center of mechatronics and clean technologies" BG05M2OP001-1.001-0008

REFERENCES

[1] J. Chen et al., "Fixed Frequency LCC Resonant Converter Modeling and Optimal Design for High-Voltage Capacitor Charging Power Supply in Constant Power Control," in IEEE Transactions on Industry Applications, doi: 10.1109/TIA.2023.3264221.

[2] ZHAO, Shuai; BLAABJERG, Frede; WANG, Huai. An overview of artificial intelligence applications for power electronics. IEEE Transactions on Power Electronics, 2020, 36.4: 4633-4658.

[3] A. Triviño-Cabrera, J. C. Quiró, J. M. González-González and J. A. Aguado, "Optimized Design of a Wireless Charger Prototype for an e-Scooter," in IEEE Access, vol. 11, pp. 33014-33026, 2023, doi: 10.1109/ACCESS.2023.3243958.

[4] Delhommais, Mylène. "Review of optimization methods for the design of power electronics systems." 2020 22nd European Conference on Power Electronics and Applications (EPE'20 ECCE Europe). IEEE, 2020.

[5] Gill, P.E., W. Murray, and M.H.Wright, Practical Optimization, Academic Press, London, 1981.

[6] Nelder, J.A. and R. Mead, "A Simplex Method for Function Minimization," Computer J., Vol .7, pp. 308-313, 1965.

[7] Shanno, D.F., "Conditioning of Quasi-Newton Methods for Function Minimization," Mathematics of Computing, Vol. 24, pp. 647-656, 1970.

[8] Levenberg, K., "A Method for the Solution of Certain Problems in Last Squares," Quart. Appl. Math. Vol. 2, pp. 164-168, 1944.

[9] Marquardt, D., "An Algorithm for Least-Squares Estimation of Nonlinear Parameters," SIAM J. Appl. Math. Vol. 11, pp. 431-441, 1963

[10] Schittowski, K., "NLQPL: A FORTRAN-Subroutine Solving Constrained Nonlinear Programming Problems," Annals of Operations Research, Vol. 5, pp. 485-500, 1985.

Information about the authors:

Bogdan Nikolov Gilev – Associated Professor in Department of mathematical modeling and numerical methods, Faculty of Applied Mathematics and Informatics, Technical University of Sofia, Sofia, Bulgaria. Current research interests: mathematical and software models, optimal design, innovation method of control of electric devices.

Nikolay Lyuboslavov Hinov – Associated Professor in Department of Power Electronics, Faculty of Electronic Engineering and Technologies, Technical University of Sofia, Bulgaria. Current research interests: development and design of power electronic converters with application in industrial technologies, electric vehicles, decentralized generation of electricity and energy storage.

Stanislav Toshkov Stefanov – Chef Assistant in Department of Mathematics, Faculty of Transport, University of Architecture Civil Engineering and Geodesy, Sofia, Bulgaria, Current research interests: mathematical modeling and geometric maps.

Manuscript received on 15 April 2023