[1.] Mihaela OSACI, [1.] Corina Daniela CUNȚAN, [1.] Ioan BACIU

LUA PROGRAMMING FOR FEMM APPLICATIONS (FINIT ELEMENT METHOD MAGNETICS)

[1.] University Politehnica Timişoara, Faculty of Engineering of Hunedoara, Electrical Engineering and Industrial Informatics Department, Hunedoara, ROMANIA

Abstract: The paper presents, through case studies, the importance of Lua script programming for electromagnetic field applications numerically solved with FEMM (Finite Element Method Magnetics). Going over the simplicity of the language and the "open source" character, Lua can solve a number of aspects that FEMM cannot offer through its visual tools in the preprocessor and postprocessor, among which: interactive modification of the values of some parameters, interactive modification of the geometry of the problem, facilities for saving graphic results, ... It gets, by Lua programming, to the point where no graphics tool of the FEMM suite is required.
Keywords: finite element method, scripting language, Finite Element Method Magnetics, Lua programming, electromagnetic field

INTRODUCTION

Electromagnetic field theory shows that very few problems of electromagnetism have analytical solutions, that is, solutions in the form of mathematical functions describing the behaviour of the electromagnetic field in a region from space to time and space [1], [2], [3]. That is why in most problems involving the design of electromagnetic devices, numerical methods are used. Of the numerical methods existing in the literature, the finite element method is one of the most used for the numerical calculation of the electromagnetic field at industrial frequencies and of the electrostatic, magnetostatic, or stationary magnetic field [4].

Currently, there is available a set of graphical interface applications for electromagnetic problems based on the finite element method, of which are mentioned here: Ansys, Comsol Multiphysics, FEMM (Finite Element Method Magnetics) [5], [6], [7].

Finite Element Method Magnetics (FEMM) is an open source software application for solving electromagnetic problems based on the finite element method [7], [8]. This application can be easily used to numerically solve electrostatic problems, magneto-static problems, and stationary magnetic, respectively low frequency magnetic problems. FEMM is a product much used both in science and engineering and in academia [9], [10], [11], [12].

A scripting language is a runtime programming language that automates the execution of tasks that are typically performed by a human operator. Tasks that can be automated by scripting can be related to software applications, text editors, web pages, operating system shells, embedded systems and computer games [13].

Lua is a powerful scripting language, easy to use, efficient and built-in. It supports procedural programming, object-oriented programming, functional programming, data-driven programming and data description. Lua combines simple procedural syntax with powerful data description structures [14].

In addition to the general instructions, the Lua language presents a series of FEMM application-specific instructions that can achieve a series of dynamic effects at both the pre-processor and postprocessor levels. This paper presents a series of case studies that aim to highlight the importance of scripting in solving FEMM electromagnetic field problems [8].

THE IMPORTANCE OF LUA SCRIPT PROGRAMMING FOR ELECTROMAGNETIC FIELD APPLICATIONS SOLVED NUMERICALLY WITH FEMM THROUGH CASE STUDIES

FEMM with the help of its tools can numerically solve an electromagnetic field problem for a given geometry. Here we can come up with an example, presented as a tutorial also in the specialized literature [15], the calculation of the electromagnetic force of interaction that is exerted on an iron ball located at a distance given by a cylindrical coil without magnetic core travelled by constant current (figure 1).

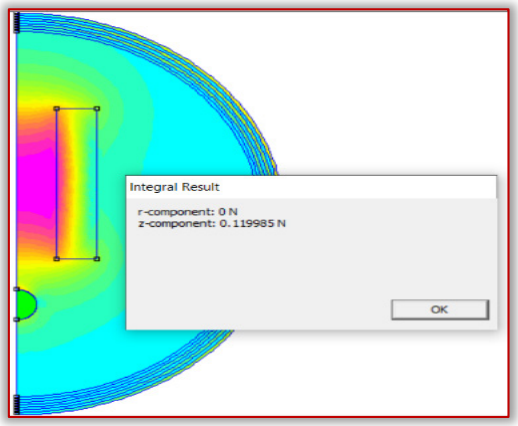

Figure 1. Result in FEMM postprocessor for the electromagnetic force of attraction between an iron ball and a coil travelled by constant current

If one were to calculate the electromagnetic force for several positions of the iron ball, classically, for each position of interest of the ball, the geometry of the ball would have to be reconstructed in the postprocessor and perform the classical operations of obtaining the solution, then the solution, respectively the value of the force and the coordinate of the position of the ball should be manually noted somewhere. This modality is expensive, time-consuming, and can also generate errors when transcribing the solution. The calculation of the electromagnetic force exerted on the ball, for different positions of the ball, can be automated by a Lua script.

In order to program Lua, the iron ball's belonging to a group is set in the pre-processor, for example, the group with the number 1 (figure 2). A group number is a label that can be applied to different lines, arcs, and block labels in a geometry. The advantage of introducing elements of the model into a group is that then all the members of the group can be selected either with a single click in the user interface, or with a single command in a Lua script.

As facilities the script shows the console Lua, by showconsole(), instruction so that the results are visible. The full geometry of the problem is loaded into FEMM and then saved with a different file name, temporarily, so that the original geometry is not overwritten later.

```
mydir="./"
open(mydir .. "bobinaLua.fem")
mi_saveas(mydir .. "tempbobinaLua.fem")
mi_seteditmode("group")
```

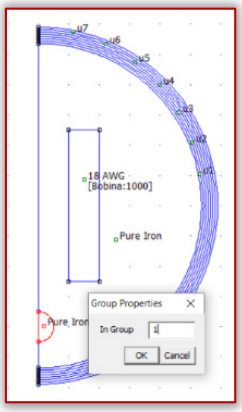

Figure 2. Labelling the moving part of the geometry

Next is the group editing mode through mi seteditmode("group"). The script will then evaluate the geometry of the ball in a number of different positions, displaying the force from each position on the Lua console. To assess the force, all elements associated with the block labels in Group 1 (which we use to represent parts of the moving ball) are selected using the command mo groupselectblock(1). The force is calculated by invoking fz=mo_blockintegral(19), which returns the directed component z of the force calculated from the tensor of voltages. The ball is moved by the instruction mi_movetranslate(0,0.1) by prior selection of the corresponding group by mi_selectgroup(1)

```
for n=0,15 do
  mi_analyze()
  mi_loadsolution()
  mo_groupselectblock(1)
  fz=mo_blockintegral(19)
  print((15-n)/10,fz)
  if (n<15) then
    mi_selectgroup(1)
    mi_movetranslate(0,0.1)
  end
end
```

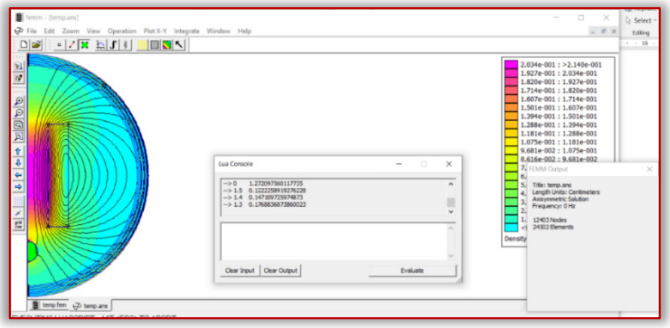

Figure 3. Results in the Lua console

The Lua script can be written in a text file that can be saved with the "take" extension and can be launched into execution from femm environment, File menu, with open lua script command (Figure 3).

If you want to keep the geometries from the postprocessor in each of the successive positions of the ball, this can be done by saving a screenshot in the graphic file with the bmp extension in the current folder, for example. For this, after the geometry change sequence, the following lines of code can be inserted:

```
fisier=tostring(n)
mo_savebitmap(mydir .. fisier..".bmp")
```

Results are shown in Figure 4. Figure 5 there can be viewed some of the 15 graphic situations generated for the calculation of the electromagnetic force.

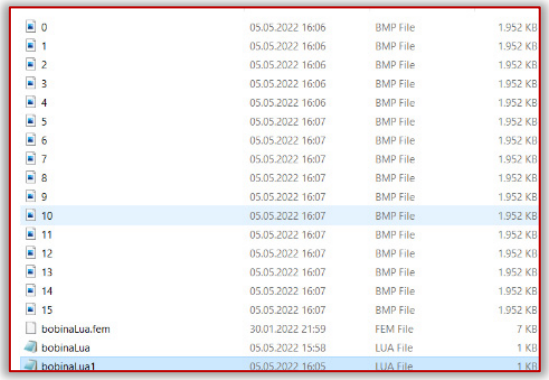

Figure 4. Created graphic files

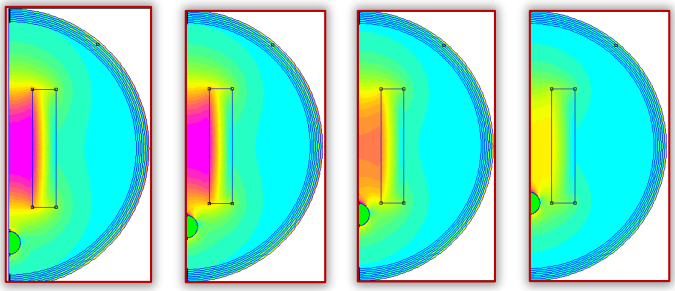

Figure 5. Generated graphic situations

A second case study will highlight the possibility of automatic modification of some parameters of the problem through the script take. The problem highlights the skin effect in a massive conductor driven by alternating current of a given frequency (figure 6).

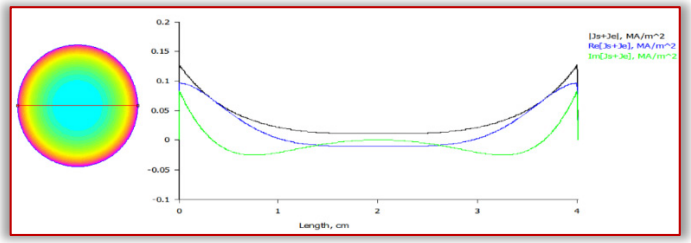

Figure 6. FEMM study of the film effect in massive conductors

If the study of the abutment of current density lines toward the periphery of the massive conductor at different frequencies of current passing through the conductor, would be wanted, you would have to change the frequency in the pre-processor in problem definition (figure 7), and do, in order, a series of operations as the launch in execution of the solver, drawing in the post-processor a contour line along the diameter of the conductor and constructing the current density graph according to length.

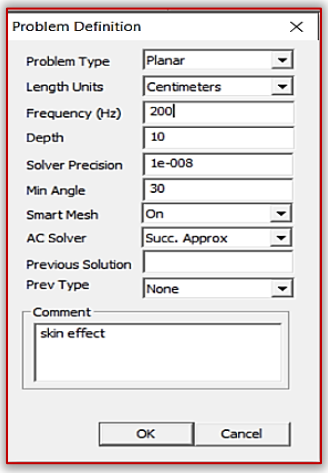

Figure 7. Manual frequency change in the FEMM pre-processor

With Lua code, written in a notepad text file with the extension Lua, saved in the current directory where the FEMM file was saved, it can automate these procedures. Basically one opens the console take with the statement showconsole(), then one opens file in which we defined the FEMM problem and copy this file to a temporary file in which the changes will be made automatically.

```
mydir="./"
open(mydir .. "skin.fem")
mi_saveas(mydir .. "skintemp.fem")
```

In a for cycle with the frequency f meter varying from 200 to 1000 Hz, the frequency changes in the definition of the problem with mi_prodef(f), the mi_analyze() solver is executed, and the solution is loaded into the post-processor with mi_loadsolution(). In the post processor, the edit mode is set on contour with mo_setedmode("contour"), the contour is drawn with mo_addcontour(-2,0) and mo_addcontour(2,0), respectively. To display the results, the frequency value is taken into a variable, it is converted to string with file=tostring(f), then with mo_makeplot(8,200, mydir .. file..".bmp") the current density graph is made in a bmp file with the name frequency (figure 8) value for f=200,1000,100 do

```
mi_probdef(f)
mi_analyze()
mi_loadsolution()
mo_seteditmode("contour")
mo_addcontour(-2,0)
mo_addcontour(2,0)
fisier=tostring(f)
mo_makeplot(8,200, mydir .. fisier..".bmp")
end
```

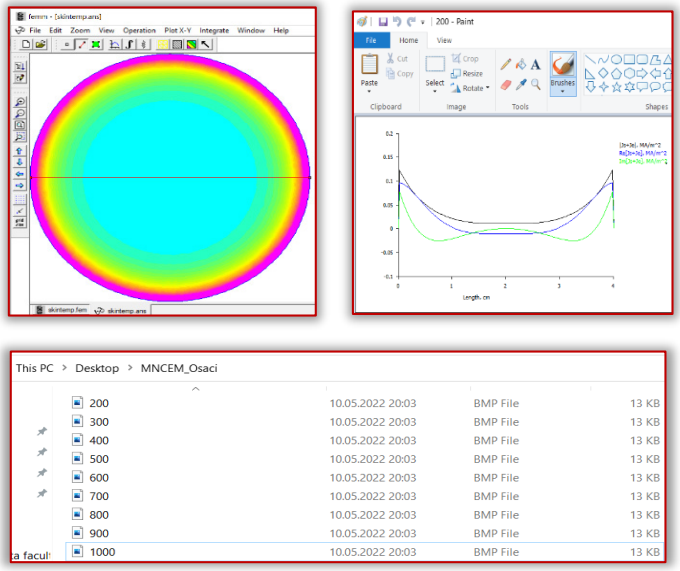

Figure 8. Results obtained with Lua script

The third case study shows how all visual actions performed in FEMM to obtain a numerical result for a problem are done using a Lua script. The problem is the distribution of the electrostatic field between the fittings of a cylindrical capacitor. Through the la console, constructive parameters of the capacitor are introduced, namely: The radius of the outer armature, the radius of the inner armature and the length of the capacitor (figure 9). A fee file is then opened in the electrostatic pre-processor which is named according to the constructive parameters.

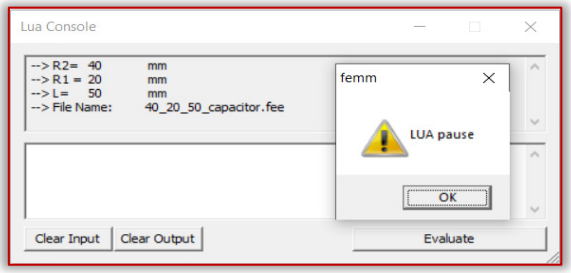

Figure 9. Introduction of the capacitor parameters through the Lua console

Also through the console, the name of the material between the fittings and the electrical permittivity of the material (figure 10) is introduced.

After which from the Lua code the planar geometry of the capacitor is achieved, the conductors for the fittings are defined and the fittings are attached, the material between the fittings is attached to the geometry, the mesh is generated, the solver is launched in execution, the solution is loaded into the post-processor and the result is obtained,

that is, the distribution of the electrostatic field between the capacitor fittings in the colour code-figure 11 and figure 12.

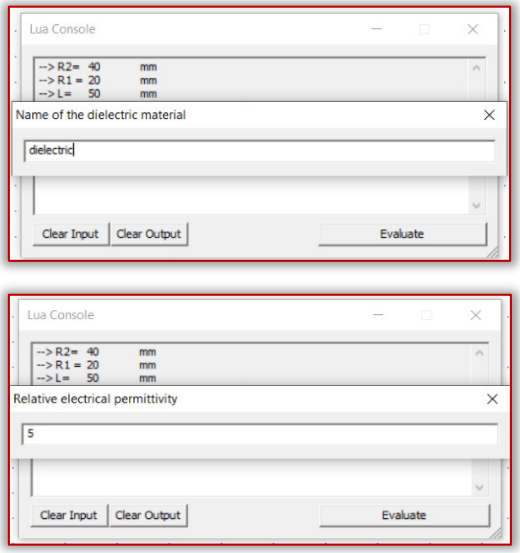

Figure 10. Introduction of the dielectric parameters through the Lua console

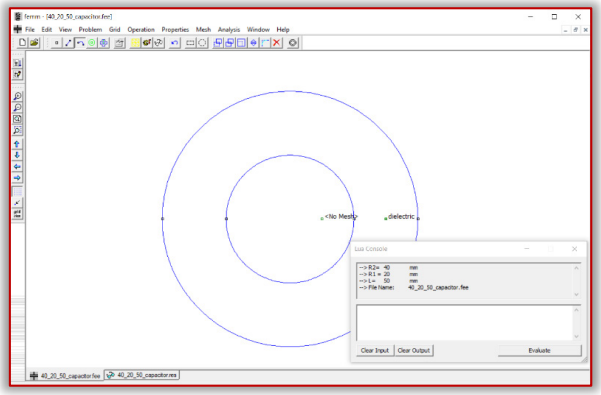

Figure 11. Fee file in electrostatic pre-processor

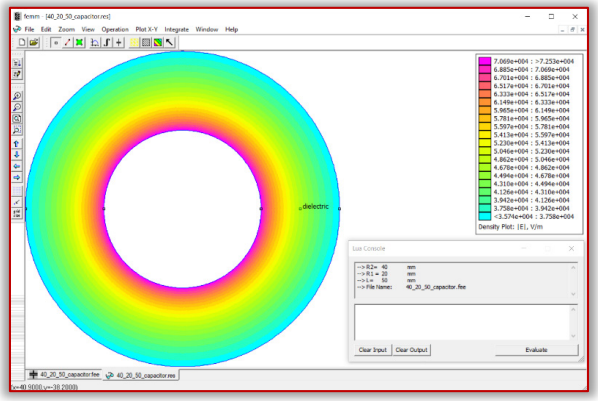

Figure 12. The res file with the simulation result in the electrostatic post-processor

CONCLUSIONS

The paper presents, through case studies, the utility of the Scriptural programming of the Lua for the dynamic implementation of some facilities that cannot be achieved through the implementation of visual FEMM. The first case study presents an automation of a calculation on a dynamic geometry, the second case study presents the possibility of changing the parameters of a problem dynamically, and the third case study presents the implementation of an electrostatic FEMM problem by the AI code. It is noted the ease of using the language, the fact that from few lines of code, both numerical results and very consistent graphic results are obtained, and last but not least the possibility of dynamic implementation of behaviours that, classically, only with FEMM tools could not be achieved.

References:

[1] R.S. Kshetrimayum, Electromagnetic Field Theory, Cengage India; 1st edition, 2012;
[2] Bhag Singh Guru, Hüseyin R. Hiziroglu, Electromagnetic Field Theory Fundamentals, 2nd Edition, Cambridge University Press, 2009;
[3] M. Osaci, Teoria câmpului electromagnetic, Editura Politehnica Timișoara, 2021
[4] Anders Bondeson Thomas Rylander Pär Ingelström, Computational Electromagnetics, Springer, 2005;
[5] *** https://www.ansys.com/
[6] *** https://www.comsol.com/
[7] *** https://www.femm.info/wiki/HomePage
[8] *** https://www.femm.info/wiki/Documentation/
[9] Baltzis, K. B., The FEMM Package: A Simple, Fast, and Accurate Open Source Electromagnetic Tool in Science and Engineering, Journal of Engineering Science & Technology Review, Jan2008, Vol. 1 Issue 1, p83-89;
[10] Konstantinos B. Baltzis, The finite element method magnetics (FEMM) freeware package: May it serve as an educational tool in teaching electromagnetics?, Education and Information Technologies, vol. 15, pages19–36 (2010);
[11] G.T.Mohanraj, M.R.Rahman, Sharnappa Joladarashi, Harish Hanumanthappa, Bharath Kumar Shanmugam, Harsha Vardhan, Shahid AzamRabbani, Design and fabrication of optimized magnetic roller for permanent roll magnetic separator (PRMS): Finite element method magnetics (FEMM) approach, Advanced Powder Technology, Volume 32, Issue 2, February 2021, Pages 546-564;
[12] Tariq Benamimour, Amar Bentounsi, Hind Djeghloud, Study of Wind Turbinebased Variable Reluctance Generator Using Hybrid FEMM-MATLAB Modeling, International Journal of Electrical and Computer Engineering (IJECE) Vol.7, No.1, February 2017, pp. 31~41;
[13] Rich Morin and Vicki Brown, Scripting Languages, MacTech, vol. 15, Issue 9, 1999;
[14] *** http://www.lua.org/
[15] *** https://www.femm.info/wiki/CoilGun

ISSN: 2067-3809