INTRODUCTION TO THE FINITE-ELEMENT METHOD - MODELING, CALCULATION, AND VISUALIZATION - A TUTORIAL

C Schlensok, D van Riesen, G Henneberger

Aachen University (RWTH), Germany

In this paper a tutorial developed at the Department of Electrical Machines (IEM) at Aachen University (RWTH Aachen) used as an introduction to the Finite-Element Method (FEM) is presented. The tutorial explains the modeling, calculation, and visualization of FEM problems on the basis of the benchmark problem 20 for the TEAM workshop (1). For modeling the commercial tool ANSYS (2) is used. The calculation and the visualization are performed by the self-developed open-source software *i*MOOSE (3). The tutorial itself is also part of *i*MOOSE and open source.

INTRODUCTION

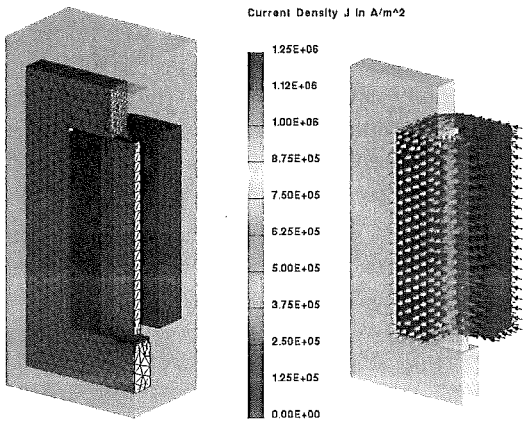
The solving of a FEM problem is divided into three steps:

1. pre-processing, which is the modeling and the discretization of the geometry,
2. processing or solving, which is the electro-magnetic calculation in this case, and
3. post-processing, which is the calculation of forces, torque, the visualization, etc.

In the following these three steps are explained using the benchmark problem 20 for the TEAM workshop as an example.
TEAM workshop No. 20 consists of a yoke which is open at the top. Below this opening there is the center pole which is surrounded by a winding. The whole geometry is surrounded by air. Due to the symmetry it is sufficient to model only a quarter of the structure (Fig. 1(a)). The coil is current driven at four points of operation: $I_1 = 1000\,\mathrm{A}$, $I_2 = 3000\,\mathrm{A}$, $I_3 = 4500\,\mathrm{A}$, and $I_4 = 5000\,\mathrm{A}$. The problem is to be solved with 3D static electromagnetic computation. In the tutorial the geometry is also modeled and calculated in 2D which is a very strong simplification and results in wrong values for the analysis. But this way both 2D and 3D modeling and calculation are explained. The paper only presents the 3D problem.

PRE-PROCESSING

For the pre-processing the commercially distributed program ANSYS is used. It can be controlled by a scripting language. There is the possibility either to type each command into the command line or to load a script file as input. Script files are regular text files which contain the commands. Using scripts makes the modeling very clearly structured. Here one main script contains the parameter script and the modeling scripts as input files.
There are two in principle different ways of modeling: **Top-Down Modeling** (TDM) or **Bottom-Up Modeling** (BUM). In case of TDM the entities (lines, area, volumes) of which the model consists are created directly by boolean operations, subtracting and adding entities. If BUM is used, at first the key points of the geometry are created. These are then connected with lines which are then combined to areas. The volumes are then created of the areas. The BUM is much more time intensive due to the many "manual" operations. But sometimes ANSYS has tolerance problems with TDM. Of course it is possible to mix both principles. In the tutorial both methods are explained. Fig. 1(a) shows the meshed geometry.



(a) Geometry (b) Current-Density

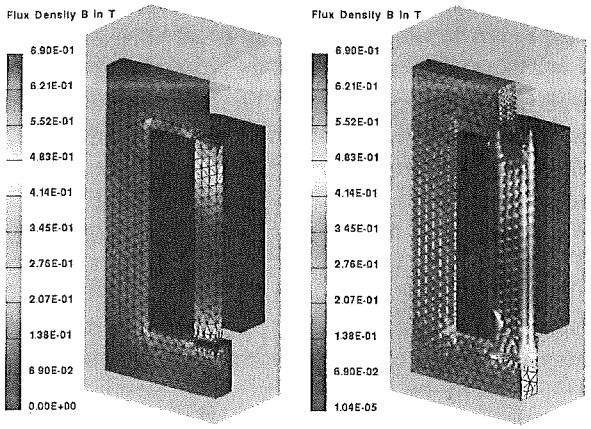
Figure 1: Geometry and Current-Density Distribution

SOLVING

In a next step the problem is solved. For the four different current excitations four computations have to be performed. For solving the 3D problem the solver *i*MOOSE.stat3d is used. The solver gets the problem definition from a problem file, which is a text file. In this file the solver specific parameters, the materials, the excitation, and the boundary conditions are de-

© 2004 The Institution of Electrical Engineers. Printed and published by the IEE, Michael Faraday House, Six Hills Way, Stevenage, Herts SG1 2AY, UK

fined. The ANSYS mesh is read and the currents are assigned to the cross areas of the coil as Fig. 1(b) shows for $I_1 = 1000\,\mathrm{A}$. The resulting flux-density distribution is depicted in Fig. 2.

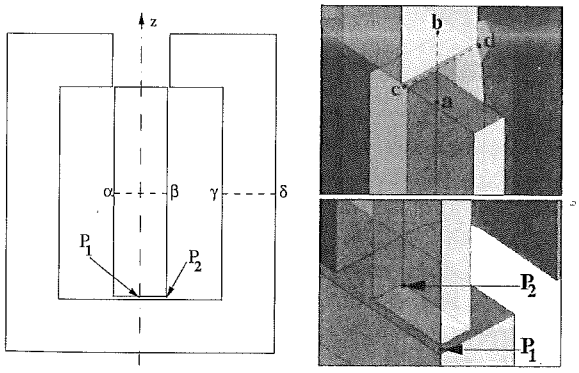


(a) Scalar (b) Vector

Figure 2: Flux-Density Distribution for $I_1 = 1000\,\mathrm{A}$

POST-PROCESSING

For post-processing *i*MOOSE.trinity is used. It is a command-line based tool which can also read a script and run in the background. Next to the visualization, its results are shown in Fig. 1 and Fig. 2, *i*MOOSE.trinity can also perform post-processing by using the commands of the programming language Python (4). The analysis points P_1 and P_2 and the analysis lines $\alpha - \beta$ and $\gamma - \delta$ are depicted in Fig. 3(a), the lines $a - b$ and $c - d$ inf Fig. 3(b). At these locations the flux density is quantified.



(a) Analysis Locations (b) Locations

Figure 3: Analysis Locations for Flux Density

The results of the z-component B_z of the flux density at the locations P_1 and P_2 are depicted in Fig. 4. They are compared with measurements. The relative error is also shown. For current excitations ($I \geq 3000\,\mathrm{A}$) the relative error is $\leq 2.5\,\%$. In case of $I = 1000\,\mathrm{A}$ especially the flux density of P_1 shows higher error (11.3 %).

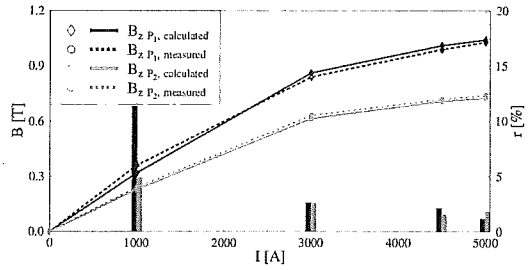


Figure 4: Flux Density B_z at P_1 and P_2

Exemplarily the z-component B_z of the flux density along the line $c - d$ for all four excitations is shown in Fig. 5. The result for $I = 5000\,\mathrm{A}$ is compared to the measurement. Due to the meshing and the geometry discretization the numerical results differ strongly in some points, because the nodes and the elements do not fit exactly to the required coordinates of the measurement. But the behavior shows the right tendency. In the tutorial the quantities of the other locations are computed and presented as well.

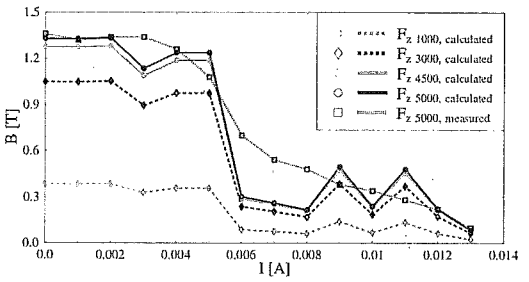


Figure 5: Flux Density B_z Along the Line $c - d$.

CONCLUSION

This paper gives a short overview of a tutorial which is provided by the open-source project *i*MOOSE by the IEM at Aachen University. The tutorial introduces the FEM by explaining the three steps modeling, solving, and post-processing by the example of TEAM-Workshop No. 20.

1. International Compumag Society - ICS 2001, http://ee.ascs3.uakron.edu/team
2. ANSYS Inc., http://www.ansys.com
3. *i*MOOSE, Innovative Modern Object-Oriented Solving Environment - *i*MOOSE, http://www.imoose.de
4. Python, http://www.python.org