Computationally Efficient Representation and Implementation of the Preisach-model of Hysteresis

Z. Sari

University of Pécs,
Pollack Mihály Faculty of Engineering and Information Technology,
Pécs, Hungary Pécs, Rókus 2., e-mail: ski@morpheus.pte.hu

Abstract. The paper deals with the details of representation and implementation of the Preisach-model of hysteresis in order to be efficiently applicable in numerical computations, particularly in finite element modelling and analysis.

Keywords: hysteresis modelling, numerical analysis, COMSOL-Matlab

1. Introduction

The main goal of the paper is to describe the representation and implementation details of a hysteresis model into finite element analysis of a field calculation problem in the COMSOL-Matlab environment. It will be shown, that the quite difficult task of representation and implementation of a complicated hysteresis operator can be handled relatively easily by means of an appropriate description of the hysteresis operator and the efficient usage of the high level software environment, what the interaction of the COMSOL and Matlab packages can offer.

Since there are numerous engineering fields, where the proper application of a hysteresis operator is required, (like ferromagnetics, porous media flow, fatigue of mechanical structures, vapour-liquid phase transitions etc.) and the 'mainstream' numerical tool for the solution of these kind of problems is the Finite Element Method (FEM), it is a naturally arising need, that the hysteresis operator of choice should be ready to use, directly in the high-level FEM packages. Since the commercial FEM packages do not provide hysteresis models directly, it relies on the user building his/her implementation. In the paper a simple hysteretic diffusion problem is solved as a case study in the COMSOL Multiphysics finite element analysis software, and by the aid of this case study, the details of application of a hysteresis model in the COMSOL-Matlab environment is presented.

2. The Comsol FEM package and the Matlab environment

As the primary tool for the finite element analysis of the problem, the COMSOL Multiphysics engineering simulation software is applied. The COMSOL package is a high level modelling and simulation environment providing tools for all steps of modelling (geometry definition, mesh generation, physics definition, solution, post-

processing etc.), and having a set of very advantageous properties, which enables a highly customized way of building models, and running simulations. One of the most important features of the package is the equation-based modelling, facilitating the weak form representation of the partial differential equations (PDEs) of the problem, which is really advantageous, because by the application of the weak form, the user has full control over the physics defined on various domains of the modelled geometry. Another extremely useful feature is the capability of using functions written in the Matlab environment. This way, practically any kind of function or multi-valued operator can be implemented into the finite element model. As a consequence of the fundamental nature of the connection between COMSOL and Matlab, implementing a hysteresis operator requires a fair amount of initialization tasks preceding the actual usage of the operator, but after that, it can be used easily, as if it was a part of the FEM package.

3. The Preisach Operator

The defining equation of the operator applied here is based on the work of P. Krejci et al. [2]

$$\mathcal{P}[\lambda,u](t) = \int_0^\infty g(r,\wp[\lambda,u](t))dr, \tag{1}$$

$$g(r,v) = \int_0^v \varphi(r,z)dz, \tag{2}$$

where u is the input of the operator, r is the abscissa of the phase space, the auxiliary function g contains the integration of the distribution function φ (the usual Preisach-distribution) over the phase space, where the upper bound $\wp[\lambda,u](t)$ of integration (2) is the output of the memory operator $\wp$, and λ is a function of the Λ phase space. The definition of the phase space is

$$\Lambda = \{\lambda : \mathbb{R}_+ \to \mathbb{R} : |\lambda(r) - \lambda(s)| \le |r-s| \ \forall r,s \in \mathbb{R}_+ : \lim_{r\to\infty} \lambda(r) = 0\}, \tag{3}$$

which represents a set of functions with the property of having a maximum absolute steepness of one, and having value of zero at $r \to \infty$. This definition basically describes an infinitely large triangle bounding the functions of the phase space. The actual state of the phase space (the concrete λ function) represents the memory of the hysteresis operator. Putting (1) and (2) together gives a double integral defining the Preisach-operator as

$$\mathcal{P}[\lambda,u](t) = \int_0^\infty \int_0^{\wp[\lambda,u](t)} \varphi(r,z)dzdr, \tag{4}$$

where the integration takes place over a subdomain of the phase space, defined by the actual state function λ, which can be obtained from the memory operator.

4. The implementation of the hysteresis operator

In order to implement the operator (4) into a numerical computation it is advantageous to make some practical modifications. First of all, in order to apply an operator in a

numerical environment, the phase space has to be finite. In other words, infinitely large inputs are not allowed, thus an upper bound value u_s has to be introduced. Besides the construction of the finite phase space defined as

$$\Lambda = \{\lambda : \mathbb{R}_+ \to \mathbb{R} : |\lambda(r) - \lambda(s)| \leq |r - s| \; \forall r, s \in \mathbb{R}_+ : \lim_{r \to u_s} \lambda(r) = 0\}, \tag{5}$$

I have introduced a memory operator ν, which is responsible for the computation of the new state (memory function) in view of the actual state λ, the input u, and the time derivative $\dot{u}$ of the input. The defining equation (4) of the Preisach-operator thus replaced by

$$\mathcal{P}[\lambda, u, \dot{u}](t) = \int_0^{u_s} \int_{r-u_s}^{\nu[\lambda,u,\dot{u}](t)} \varphi(r, z) dz dr, \tag{6}$$

which contains the memory operator ν as the upper bound of the inner integration having the same role as the memory operator $\wp$ in (4), but with a different realization.

5. The structure of the phase space and the behaviour of the memory operator

The phase space defined by (5) is basically a usual Preisach-triangle rotated by 45 degrees counter-clockwise. The role of the memory operator $\nu[\lambda, u, \dot{u}](t)$ is to calculate the new state function λ based on the actual state and the input. The behaviour of the operator in the phase space (Preisach-triangle) can be tracked by the aid of Fig. 1, which shows a possible state function and the corresponding hysteresis curve.

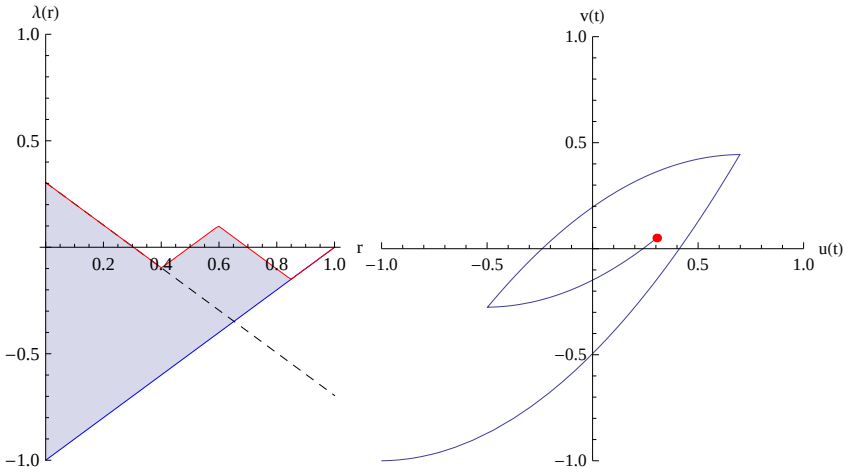

Figure 1. A state function in the phase space with the corresponding hysteresis curve

In the left subfigure the memory function $\lambda(r)$ can be seen as the upper boundary of the filled region of the phase space, the lower boundary is $l(r) = r - u_s$, where the u_s is normalized to $\{-1,1\}$ in the figure.

Integration (6) takes place over the filled subdomain of the phase space bounded by the fixed "lower" side of the triangle and the continuously changing memory function resulting from the memory operator. The input-output relation of the operator appears as a hysteresis curve, which can be seen in the right hand side subfigure, where $u(t)$ and $v(t)$ denotes the input and the output of the operator (6) respectively. The memory operator utilizes an auxiliary line $e(r) = -\mathrm{sgn}(\dot{u})r + u$ (dashed line in figure) for the purpose of tracking the changes of the phase space. The intersecting point of the $e(r)$ line and the $\lambda(r)$ axis represents the actual value of the input $u(t)$, and starting from a given state (for example the one depicted in Fig. 1.) the change of the input of the hysteresis operator causes the moving of the $e(r)$ line upwards or downwards depending on the sign of the time derivative of the input. The moving of $e(r)$ results in a new intersection point of the $e(r)$ line with the actual state function $\lambda(r)$, and the new memory function is constructed simply by "concatenating" the segment of $e(r)$ preceding the intersection with the segment of $\lambda(r)$ beyond the intersection.

Fig. 2 shows the new state and the corresponding hysteresis curve after decreasing the value of the input $u(t)$.

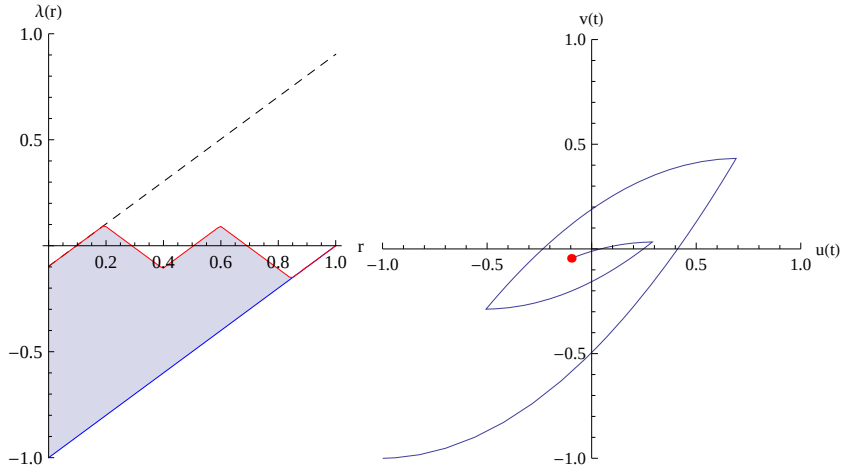

Figure 2. State function and hysteresis curve after decresing the input from state depicted in Fig. 1

Further decreasing the input results in "wiping out" the portion of memory (state function), which represents the inside minor loop, which can be seen in Fig. 3.

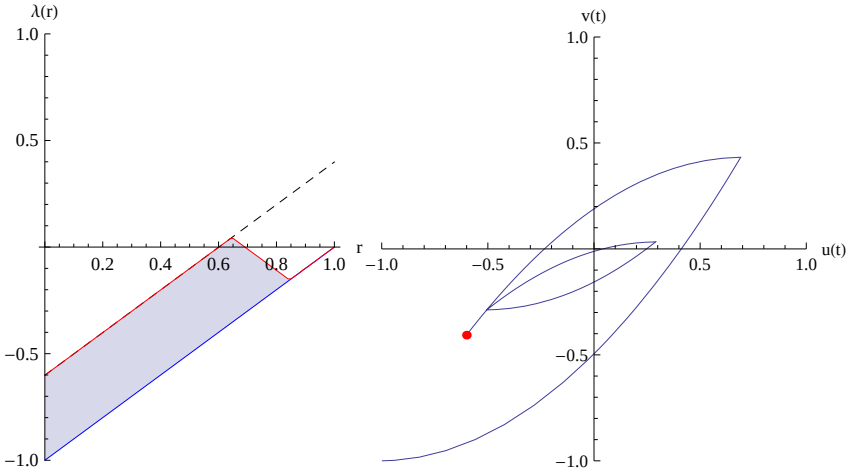

Figure 3. Phase space and hysteresis curve after a "wipe out"

The main advantage of this representation is, that the complicated changes in the phase space can be tracked very easily by the aid of the introduced memory operator, since the line $e(r)$ can be constructed without any problems in view of the input of the hysteresis operator and the state function representing the memory is described by a single-valued function, which is very easy to handle as well.

The Matlab (or any suitable numerical tool) implementation of the operator described above consists of the discrete representation of $\lambda(r)$ in a form of a finite dimensional vector, and the main task to perform is to find the intersection of $e(r)$ and $\lambda(r)$ in order to construct the new state function. The method of finding the intersection is the following

$$i_m = \left\langle \frac{1}{2}\left|\Delta\left[\operatorname{sgn}\left(\mathbf{e}-\boldsymbol{\lambda}\right)\right]\right|, \mathbf{i} \right\rangle, \tag{7}$$

where $\mathbf{e}$ and $\boldsymbol{\lambda}$ are the vectors of $\mathbb{R}^m$ corresponding to the discretized $e(r)$ and $\lambda(r)$ respectively, Δ is the difference operator, $\mathbf{i} = [1, 2, \ldots m]$ is an index vector, $\langle .,. \rangle$ denotes the usual dot product, and i_m is the index of the intersection point. After finding the location of the intersection, the construction of the new state function is straightforward.

The representation outlined here can be immediately applicable to define a Preisach-operator in a numerical environment, and furthermore – resulting from the nature of the memory operator applied – the algorithm is easily vectorizable, so it can be implemented into finite element computation as is. One further point is, that the integration of the distribution over the phase space can be calculated off-line, thus during the actual computation it can be substituted by interpolation, which significantly speeds up the procedure of numerical computation.

6. Implementation of the hysteresis operator into finite element computation

The finite element implementation is demonstrated through a case study in the COMSOL environment. Since COMSOL is highly customizable, can handle complicated nonlinear problems, and it is possible to use Matlab functions directly from the user interface without explicit usage of Matlab, it is a really good tool for solving nolinear problems involving hysteresis phenomena.

The problem solved in the case study is a one-dimensional nonlinear diffusion problem defined as

$$\dot{H} - \frac{1}{\sigma\mu_0}\nabla^2 H = -\dot{M}, \qquad M = \mathcal{P}\{H\}, \tag{8}$$

where H is the magnetic field strength, M is the magnetization, and $\mathcal{P}\{.\}$ denotes the Preisach-operator. In the COMSOL environment, after defining the diffusion equation (8) as the governing physics, the hysteretic relation has to be defined. It can be managed by defining an auxiliary dependent variable for M, and providing the weak-form equation connecting the auxiliary variable M and the hysteresis operator, which is represented by a variable assigned to the predefined Matlab function. The Matlab function definitions can also be done from the COMSOL environment in the global definitions section.

Solving the problem defined by (8) on a one-dimensional domain results in the time evolution of the H and M fields. In Fig. 4 and Fig. 5 minor hysteresis curves can be seen corresponding to different locations in space, which were obtained by solving the nonlinear diffusion problem with the application of the Preisach-operator using the representation described above. (The value of magnetization is normalized to $[0,1]$.)

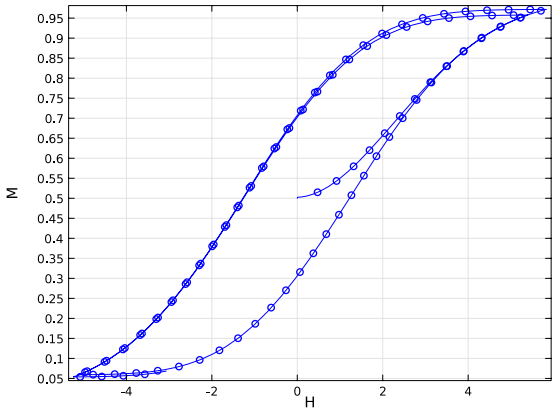

Figure 4. Minor hysteresis curves

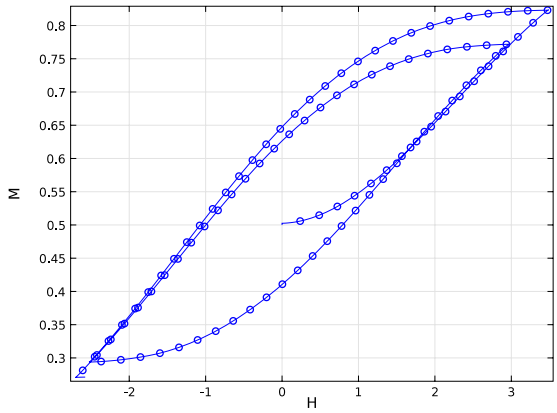

Figure 5. Minor hysteresis curves

It is important to emphasize, that after initializing the Preisach-model in the Matlab environment, it can be used similarly to a built-in function in COMSOL by the aid of an auxiliary dependent variable defined by a weak expression, and this way most of the programming difficulties of the Matlab implementation can be avoided, since the hysteresis operator can be used directly from the COMSOL user interface, and all of the modelling tasks (solution, post-processing) can also be accomplished without explicit usage of Matlab, which has the clear advantage of using a high level user-friendly interface instead of tweaking with the subtleties of programming.

7. Conclusion

As a concluding remark it can be stated, that the representation and the implementation outlined in the paper can be a productive way of application of the Preisach-model of hysteresis for handling nonlinear numerical field calculation problems involving hysteresis phenomena

References

[1] Iványi, A.: *Hysteresis Models in Electromagnetic Computation*, Budapest, Academic Press, (1997).

[2] Brokate, M., Eleuteri, M., Krejci, P.: *On a model for electromagnetic processes in ferromagnetic materials with hysteresis and space dependent electric conductivity*, Mathematical Methods in the Applied Sciences, (2008), pp. 1545–1567.