Polytechnique Montréal

Power-Equivalent Model Tutorial

Author:

Gregory Giard

Last modified:

June 20th 2023

1 Introduction

This document serves as a detailed tutorial on the usage of the Power-Equivalent Model (PEM), starting from material magnetic data, until the computation of temperature-dependant equivalent permeability. The goal of the PEM is to generate permeability curves $\overline{\mu}(H)$ for non-linear and hysteretic materials that can be used in a time-harmonic electromagnetic simulation, in particular for induction heating problems. A detailed mathematical description of the PEM can be found in [1, 2]. We describe here the steps required to compute the equivalent permeability data that can then be used in a general multiphysics simulation.

2 General code structure

The code is divided into two main steps. The first step of the code solves the 1-D time-transient slab diffusion problem with the scalar Preisach model to compute the eddy current and hysteresis losses as a function of x. The problem is solved with Fortran 90 using the finite element method (FEM). The losses curve are then stored in the folder. The current values used in our software represent magnetic measurements done on AISI4340 steel at Polytechnique Montreal. Corresponding loss curves are stored in Results/Transient_results. A user could describe other materials with this model.

The second step uses the losses created from tje previous step as inputs to solve the Power-Equivalent model. The equivalent permeability curves are then stored in Results/Mu_results.

Here is a more detailed description of the different functions in the code:

1) **slabProblem.m:** This function launches the fortran code to solve the 1-D slab problem for every H_0 and material data chosen by the user. This algorithm was designed specifically to simulate the B-H behaviour of a material using Preisach's model. See [3, 4] for more details on the model. The averaged Foucault (Joule) and hysteresis losses are then stored in Results/Transient_results

2) **powerEquivalentModel.m:** This is the main function deals with the computation of the equivalent permeability curves, solving the equations laid out in [2] using the transient losses previously calculated.

    a) **readLosses.m:** This function is used to read the loss data calculated during the slab problem transient simulation.

    b) **fem1D.m:** This function deals with solving the non-linear H equation, i.e.

    $$P_{tot} = \frac{\rho}{2}\left[H\frac{\mathrm{d}^2 H}{\mathrm{d}x^2} + \left(\frac{\mathrm{d}H}{\mathrm{d}x}\right)^2\right], \tag{1}$$

    where $P_{tot} = P_{Joule} + P_{Hyst}$ and ρ is the resistivity of the material. The algorithm uses the 1-D finite element method and a Newton-Raphson's iterative method to solve this equation. Here are the functions used to compute the FEM solution:

        i) **mesh.m:** generates the 1-D mesh.
        ii) **numbering.m:** numbers the DoFs.
        iii) **assemble.m:** assembles the FEM matrix for a given iteration of the N-R method.
        iv) **solveSystem.m:** solves the matrix system for a δ_H value
        v) **updateField.m:** updates the $H(x)$ field with $H^{i+1} = H^i + \delta_H^i$.

    c) Once $H(x)$ is known, the values of $\frac{\mathrm{d}\phi}{\mathrm{d}x}$, $\frac{\mathrm{d}^2\phi}{\mathrm{d}x^2}$, $\mathrm{Re}\{\overline{\mu}\}$ and $\mathrm{Im}\{\overline{\mu}\}$ are calculated directly in powerEquivalentModel.m

    d) **correctMu.m:** Apply the weak-field correction to $\mathrm{Re}\{\overline{\mu}\}$ (if asked by the user, see [1, 2]).

    e) **displayResults.m:** Display the 1-D distributions of the different quantities (if asked by the user).

3 Code execution steps

3.1 Clone the repository

The source code for the Power-Equivalent Model can be found in a git repository and cloned with the following command:

```
$ git clone https://github.com/giardg/PowerEquivalentModel.git
```

The main script to run is called PEM_IO.m. The general structure of the code and the user input data will be described in the next few sections.

3.2 User input data

In the main script PEM_IO.m, a user can modify some parameters to generate equivalent permeability curves from their own data.

First, there are constant input parameters that can be easily modified (see Fig. 1 as an example):

- **freq:** frequency of the applied field. During the development of the PEM, we observed that, even though this quantity is used in the different equations to be solved, it does not influence the equivalent permeability curves (under the assumption that the material properties are constant with frequency).
- **L:** length of the 1-D domain considered. This quantity must be higher than the penetration depth of the field.
- **rho:** resistivity of the material. During the development of the PEM, we observed that, even though this quantity is used in the different equations to be solved, it does not influence the equivalent permeability curves. Therefore, we do not have to worry about the thermal-dependence of this quantity while solving the PEM.
- **H0_list:** list of all the applied field amplitude to be considered. As described in [1, 2], the value of H_0 is an important quantity to take into account while using the PEM.
- **mattype:** type of material to be considered. All the different types are described below.
- **output:** folder to store the results.
- **flag_real:** allows to compute the equivalent permeability using only a real part. The idea is to still calculate the averaged transient Joule and hysteresis losses, but then compute the equivalent permeability using $P_{Joule} + P_{Hyst} \rightarrow P_{Joule}$ and $P_{Hyst} \rightarrow 0$. For example, this flag can be turned on if the software used for induction heating simulations does not allow complex permeability curves.

```
%% Input parameters
freq         = 10e3;                  %Hz (No influence on the permeability curves)
L            = 2.5e-3;                %m (adapt L to have a long enough domain with penetration depth)
rho          = 2.5e-7;                %Ohm/m (No influence on the permeability curves)
H0_list      = [1:4,5:5:100]*1e3;     %A/m
mattype      = 7;                     %1-7
output       = '\Results\';           %Output folder
flag_real    = false;                 %Flag if we want to approximate with only a real permeability
                                      %(i.e. Pj+Ph -> Pj)
```

Figure 1: Example of the input parameters used for the AISI 4340 case.

Another very important input parameter to consider is the material data. Seven types of magnetic materials are implemented in the code (not all of them are equally useful):

- **Type 1: Linear material.** This is the simplest case, where the material is described simply by a constant permeability, i.e.
$$B = \mu_0 \mu_r H \,. \tag{2}$$
For this case, the flag_linear parameter should be turned on, as the permeability is real. Only μ_r must be defined in this case. Note that there is no advantages in using this case, because the PEM is not necessary for linear materials.

- **Type 2: Non-linear material, described by an arctan curve.** In this case, the *B*-*H* behaviour of the material follows an arctan curve, i.e.
$$B = \mu_0 H + \frac{2B_{sat}}{\pi} \arctan\left(\frac{\pi\mu_0(\mu_{r_{max}} - 1)}{2B_{sat}} H\right) \,. \tag{3}$$
For this case, the flag_linear parameter should be turned on, as the permeability is real. Two parameters must be defined in this case: $\mu_{r_{max}}$ and B_{sat}.

- **Type 3: Preisach $3n$ coefficients non linear hysteretic model.** This material types describe a non-linear and hysteretic materiel using Presach scalar model with $3n$ coefficients, as described in [4]. The descending and ascending parts of the hysteresis are described as follow:
$$\begin{cases} B_+ = \sum_i a_i \arctan\left(\frac{H+c_i}{b_i}\right) \,, \\ B_- = \sum_i a_i \arctan\left(\frac{H-c_i}{b_i}\right) \,. \end{cases} \tag{4}$$
A set of n values of a_i, b_i and c_i must be defined in the input file. Note that a fitting tool is also provided in /Material_computation/Preisach3n/FittingToolHyst.m in order to compute the parameters from experimental data in a .txt file.

- **Type 4: High field limit case (not used very much).** This model describes an anhysteretic material with the following *B*-*H* behaviour:
$$\begin{cases} B = \mu_0 H + B_{sat} & \text{if } H > H_{sat} \\ B = \mu_0 H - B_{sat} & \text{if } H < -H_{sat} \\ \text{3rd order polynomial} & \text{if } -H_{sat} < H < H_{sat} \,. \end{cases} \tag{5}$$
The parameters H_{sat} and B_{sat} must be defined in this case.

- **Type 5: Preisach Model with an ellipsoïd as the major cycle (not used very much).** This model uses the EFG identification algorithm (see [4]) on a major hysteresis defined by an ellipsoïd in the *B*-*H* domain. The input parameters are the real and imaginary parts of the permeability describing this hysteresis.

- **Type 6: Preisach Model with ellipsoïds as the major and minor cycles (not used very much).** This model describes the linear complex case, i.e.
$$B = \mu_0(\mu_r' + j\mu_r'')H \,. \tag{6}$$
The input parameters are the real and imaginary parts of the permeability describing the hysteresis. Note that there is no advantages in using this case, because the PEM is not necessary for linear materials.

- **Type 7: Preisach 4 parameters non linear hysteretic model.** This model described in [4] uses only 4 parameters of a major hysteresis curve to define the Everett function in the Preisach model: B_{sat} (saturation flux density), B_r (remanent flux density), H_c (coercive field) and s (adimensionnal parameter describing the "squareness" of the hysteresis). It had been shown in [1] that an analytic equation links s and W_h (the area inside the hysteresis curve). Therefore, 4 parameters must be defined in the input file: B_{sat}, B_r , H_c and W_h.

The parameters describing the material behaviour are defined in lists, such that the whole algorithm can be executed for varying material properties as a function of temperature. Therefore, a list of considered temperatures must also be provided. These temperatures are then associated respectively to each material parameters provided in their own lists. Note that for the material type 3, the a_i, b_i and c_i parameters are defined in a cell structure (MATLAB), such that the object located at the index k in the cell is a list of n values.

In the git repository, we show the case of the AISI 4340 for which the equivalent permability curves were computed. The data available for this material is shown in Fig. 2. From this data, one can then use the material type 7 described above to compute the equivalent permeability.

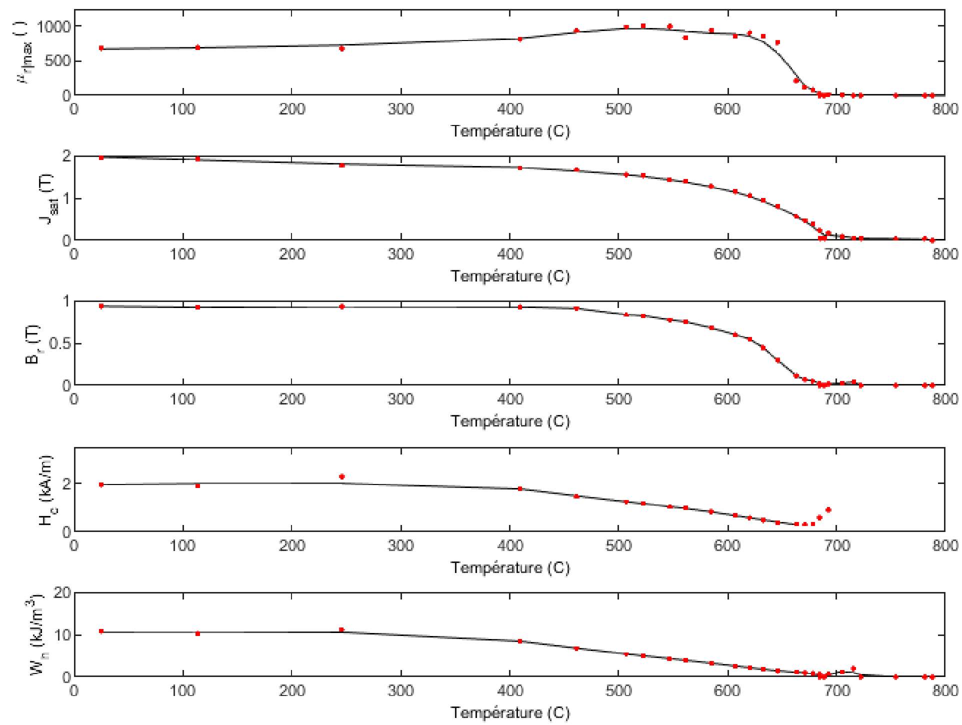

Figure 2: Magnetic properties measured for AISI 4340 steel (data from [5]).

Ideally, a new user should only modify the input parameters in PEM_IO.m and run this main script (see Fig. 1 for the template case). The material data can be defined in the next section of the code.

3.3 Output results

The transient loss distributions are stored in the folder Result/Transient_result as .txt files named such as: "H0 *H0*_Losses *mattype*_*L* mm_*T* deg_*freq* kHz.txt". Finally, all the permeability curves are stored in the folder Results/Mu_results. The data is stored in .txt files for each H_0 and temperature required by a user, named such as: "H0 *H0*_mu *mattype*_*T* deg_*freq* kHz.txt". Each file consists only of 3 columns: H, $\mathrm{Re}\{\overline{\mu}\}$ and $\mathrm{Im}\{\overline{\mu}\}$ (if flag_real was set, only the first two columns are stored). Depending on the application, a user could want to create a look-up table for various H_0 and T. The matlab file createMuTable.m could be used as an example to generate such table.

4 References

[1] G. Giard, "Modélisation en régime harmonique de la perméabilité de matériaux magnétiques, hystérétiques et non-linéaires pour le chauffage par induction," Master's thesis, Polytechnique Montréal, 2022.

[2] G. Giard, M. Tousignant, K. McMeekin, F. Bellotto, P. Bocher, and F. Sirois, "Power-equivalent complex permeability model for nonlinear and hysteretic materials in the frequency domain for induction heating applications," *IEEE Transactions on Magnetics*, vol. 58, no. 6, pp. 1–9, 2022.

[3] M. Tousignant, F. Sirois, G. Meunier, and C. Guerin, "Incorporation of a vector preisach–mayergoyz hysteresis model in 3-d finite element analysis," *IEEE Transactions on Magnetics*, vol. 55, no. 6, pp. 1–4, 2019.

[4] M. Tousignant, *Modélisation de l'hystérésis et des courants de Foucault dans les circuits magnétiques par la méthode des éléments finis.* PhD thesis, Polytechnique Montréal, 2019.

[5] K. McMeekin, "Mesure d'hystérésis magnétique volumique de l'acier 4340 en fonction de la température," Master's thesis, Polytechnique Montréal, 2016.