SOME BASIC MATHEMATICAL ELEMENTS ON STEEL HEAT TREATING: MODELING, FREEWARE PACKAGES AND NUMERICAL SIMULATION*

J. M. Díaz Moreno, M. T. González Montesinos, C. García Vázquez, F. Ortegón Gallego and G. Viglialoro

1 Introduction

Every year since 2007, in June, a Modeling Week is held within the master program of the faculty of Mathematics of Universidad Complutense of Madrid (UCM) in cooperation with the Institute of Interdisciplinary Mathematics (IIM). During this week, students work in small groups on real industrial problems proposed by companies under supervision of one or two qualified instructors. As it was announced in 2013, the main purpose of the *VII Modeling Week* was to promote the use of mathematical methods and models in research, industry, innovation, and management in the knowledge economy. That year, Prof. Ortegón Gallego was invited to the seventh edition of this event as an instructor of one intensive course on the subject *Some mathematical models and numerical simulation of heat treatment of steel*. During this course, the basic elements of the mathematical modeling of steel heat treating were described, a very simple 3D model was considered, without taking into account mechanical effects, for the heating-cooling industrial procedure applied to a helical gear (Fig. 10). Then we perform some numerical simulations of this model according to the industrial heating technique: induction or flame hardening. To do so, we provide the students with some efficient noncommercial software packages, namely `Freefem++` ([1, 12]), `MEdit` ([7]) and `gmsh` ([9]).

`Freefem++` is used for preprocessing (building the tetrahedralization of the helical gear, the coil, etc.) and processing (in order to obtain the numerical aproximation). `Freefem++` also provides the user with some tools for postprocessing, mainly for the graphical representation of the solution of 2D problems, but it is more convenient to use `gmsh` for the solution of 3D problems. `MEdit` is a useful package for 3D visualization and can be used within `Freefem++` whereas `gmsh` should be run in its own session.

This paper is organized as follows. In Section 2 we first consider a 2D mathematical model governed by a partial differential equation with mixed boundary conditions. Then, we deduce the equivalent variational formulation for this problem and apply the finite element method (FEM) in order to obtain the numerical approximation. In Section 3 we present the freeware package `Freefem++`. Here we implement the numerical resolution of the model given in the previous section To do so, we introduce some fundamental keywords of the `Freefem++` language, how to build a 2D triangulation, how to use different types of approximation, how the problem is written, how it can be solved, and the solution can be saved or depicted. The numerical resolution of an evolution problem is also discussed here and also its implementation with `Freefem++`.

Section 4 describes the main properties of steel and the basic ideas of the heating-cooling process in order to produce hardness over critical parts of the workpiece. In Section 5 we describe a simplified mathematical model for the induction heating stage and for the cooling stage applied to a helical gear. Section 6 is devoted to the step-by-step construction of the 3D tetrahedralization of a helical gear with `Freefem++`. This shows the versatility, functionality and easy-to-use of the `Freefem++` language. Finally, Section 7 present some numerical results on both induction hardening and flame hardening, performed with `Freefem++`. All these results are shown here with the help of `gmsh`.

2 A first mathematical model

In this section we consider a stationary boundary value problem governed by a partial differential equation in 2D. This will be used to illustrate some basic notions on the mathematical theory and how to obtain numerical approximations.

2.1 Description of a mixed boundary value problem

Let $\Omega \subset \mathbb{R}^2$ be the domain depicted in Figure 1. Here Ω_1 and Ω_2 stand for two different materials with corresponding thermal conductivities k_1 and k_2, respectively. We want to compute the stabilized temperature in Ω, that is a function $u\colon \Omega \mapsto \mathbb{R}$, knowing that (a) the temperature is prescribed on Γ_0, (b) the heat flux over Γ_1 is proportional to the difference between the room temperature and the temperature on Γ_1, and (c) a heating source f is applied in the whole Ω. This situation is modeled by the following problem:

$$\begin{cases} -\nabla \cdot (k\nabla u) = f \quad \text{in } \Omega, \\ \quad u = g \quad \text{on } \Gamma_0, \\ k\dfrac{\partial u}{\partial n} = \alpha(u_{\text{ext}} - u) \quad \text{on } \Gamma_1, \end{cases} \tag{1}$$

*A reduced version of this paper can be found in [6] (`http://www.thermalprocessing.com/archives/`, 2014 Fall).

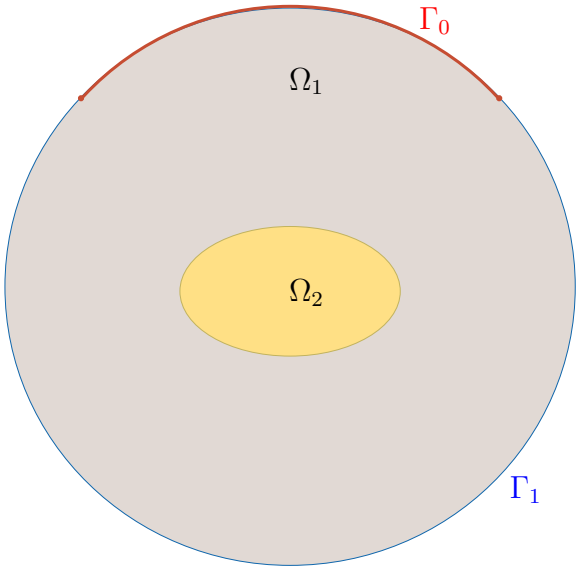


Figure 1: Domain $\Omega = \Omega_1 \cup \bar{\Omega}_2$ where the boundary value problem is set. Γ_0 is the Dirichlet boundary whereas Γ_1 is the Robin one.

where α is the transfer coefficient, u_{ext} is the known room temperature, and k is the thermal conductivity in Ω, that is

$$k = \begin{cases} k_1 & \text{in } \Omega_1, \\ k_2 & \text{in } \Omega_2. \end{cases} \tag{2}$$

The mathematical analysis of elliptic partial differential equations assures the existence of a unique solution to problem (1). Indeed, if we introduce the space $L^2(\Omega)$ of all Lebesgue measurable functions that are square integrable in Ω, that is

$$L^2(\Omega) = \left\{ v \colon \Omega \mapsto \mathbb{R},\ v \text{ is meas. in } \Omega \text{ and } \int_\Omega |v|^2 \, \mathrm{d}x < \infty \right\},$$

and the one order Sobolev space $H^1(\Omega)$ given by

$$H^1(\Omega) = \left\{ v \in L^2(\Omega) \,/\, \frac{\partial v}{\partial x_1}, \frac{\partial v}{\partial x_2} \in L^2(\Omega) \right\},$$

and assuming the following assumptions on the data: $f \in L^2(\Omega)$, $g \in H^1(\Omega)$, k_1 and k_2 are two bounded measurable functions such that $\min(k_1, k_2) \geq \underline{k} > 0$ where $\underline{k}$ is some constant, $u_{\text{ext}} \in L^2(\Gamma_1)$ and α is a nonnegative, bounded and measurable function in Γ_1, then it is well known that the problem (1) has a unique solution $u \in H^1(\Omega)$. It is straightforward to notice that a problem like (1) cannot be solved analytically for general data f, g, k, α and u_{ext}. In that case, we have to compute a numerical solution in order to obtain an approximation to the exact solution. One way to do this is to apply a special case of Galerkin's method, namely, the so-called finite element method. In order to introduce this method, we need to express the original problem (1) in what is called its (equivalent) variational formulation.

2.2 Variational formulation

In order to deduce the equivalent variational formulation of problem (1), we first multiply the differential equation in Ω by a test function $v \in H^1(\Omega)$, integrate in Ω, and do an integration by parts. In doing so, it yields

$$\int_\Omega k \nabla u \nabla v = \int_\Omega f v + \int_{\partial\Omega} k \frac{\partial u}{\partial n} v.$$

Taking into account the second boundary condition, we deduce

$$\int_{\partial\Omega} k \frac{\partial u}{\partial n} v = \int_{\Gamma_0} k \frac{\partial u}{\partial n} v + \int_{\Gamma_1} \alpha (u_{\text{ext}} - u) v.$$

And by choosing the test functions $v \in H^1(\Omega)$ such that $v = 0$ on Γ_0 we obtain

$$\int_{\partial\Omega} k \frac{\partial u}{\partial n} v = \int_{\Gamma_1} \alpha u_{\text{ext}} v - \int_{\Gamma_1} \alpha u v.$$

Finally, we deduce the variacional formulation of problem (1):

$$\begin{cases} \text{To find } u \in H^1(\Omega) \text{ such that } u = g \text{ on } \Gamma_0 \text{ and} \\ \displaystyle\int_\Omega k \nabla u \nabla v + \int_{\Gamma_1} \alpha u v = \int_\Omega f v + \int_{\Gamma_1} \alpha u_{\text{ext}} v, \\ \quad \text{for all } v \in H^1(\Omega) \text{ such that } v = 0 \text{ on } \Gamma_0. \end{cases} \tag{3}$$

Calling $a(u, v)$ the bilinear part in (3) and also $L(v)$ the linear part, that is,

$$a(u, v) = \int_\Omega k \nabla u \nabla v + \int_{\Gamma_1} \alpha u v,$$

$$L(v) = \int_\Omega f v + \int_{\Gamma_1} \alpha u_{\text{ext}} v,$$

the variational formulation can be written as

$$\begin{cases} \text{To find } u \in H^1(\Omega) \text{ such that } u = g \text{ on } \Gamma_0 \text{ and} \\ a(u, v) = L(v), \ \text{ for all } v \in H^1(\Omega) \text{ with } v = 0 \text{ on } \Gamma_0. \end{cases} \tag{4}$$

Remark 1 *When solving a problem like (3) with* `Freefem++`*, it is absolutely necessary to know the expression of both the bilinear and linear forms. This is described below in Section 3.*

2.3 Discrete variational formulation

Obviously, the solution to problem (4) belongs to an infinite dimensional affine space of functions, namely $\{w \in H^1(\Omega)$ such that $w = g$ on Γ_0 $\}$. The discretization of this problem will consist in the introduction of approximate problems defined in a finite dimensional setting. Indeed, consider a finite dimensional

subspace of functions $X_h \subset H^1(\Omega)$. At this stage, we do not worry about the shape of the functions in this subspace, but this will be very important when X_h is built along with the FEM technique. In general, we also need to approximate the function g by $g_h \in X_h$ (by interpolation, projection or whatever). Finally, we consider the subspace $X_{0h} \subset X_h$ of all functions that vanish on Γ_0, that is

$$X_{0h} = \{v_h \in X_h \,/\, v_h = 0 \text{ on } \Gamma_0\},$$

which is obviously another finite dimensional subspace in $H^1(\Omega)$. The discrete version of (4) now follows,

$$\begin{cases} \text{To find } u_h \in g_h + X_{0h} \text{ such that} \\ a(u_h, v_h) = L(v_h), \text{ for all } v_h \in X_{0h}. \end{cases} \tag{5}$$

It is straightforward to show that (5) is equivalent to a system of M linear equations $A_h U_h = F_h$, M being the dimension of X_{0h}, the matrix of this system having the entries

$$(A_h)_{ij} = a(\phi_j, \phi_i), \quad 1 \le i, j \le M,$$

and the components of the right hand side being

$$(F_h)_i = L(\phi_i) - a(g_h, \phi_i), \quad 1 \le i \le M,$$

where $\{\phi_1, \ldots, \phi_M\}$ is a base of X_{0h}.

At this point, due to computational and memory cost, we see that the choice of the base $\{\phi_1, \ldots, \phi_M\}$ really matters. Indeed, it is desirable that the space X_{0h} admits a base $\{\Phi_1, \ldots, \Phi_M\}$ such that

(a) the entries $a(\Phi_j, \Phi_i), 1 \le i, j \le M$, $L(\Phi_i)$ and $a(g_h, \Phi_i)$, $1 \le i \le M$ can be easily computed at a low computational cost; and

(b) $a(\Phi_j, \Phi_i) = 0$ for many values of the indices i, j. For instance, $a(\Phi_j, \Phi_i) = 0$ whenever $|i - j| \ge p$ (in this case, the matrix A_h has a band structure near the main diagonal);

In the FEM case, these two properties are fulfilled. To do this, we first consider that the domain Ω has been previously meshed. This means that we have a triangulation of $\bar{\Omega} = \bigcup_{k=1}^{K} T_k$, where every $T_k \subset \mathbb{R}^2$ is a closed triangle, and if $1 \le k_1 < k_2 \le K$, then $T_{k_1} \cap T_{k_2}$ is the empty set, or else a common vertex or else a common side of both triangles.

Remark 2 *If the domain Ω is not polygonal, it is impossible to build a triangulation for this domain, in the sense described above. In that case, another approximation is needed: prior to the application of the method, Ω has to be approached by a polygonal domain Ω_h. This is the case shown in Figure 2.*

Denoting by $\mathcal{T}_h = \{T_k\}_{1 \le k \le K}$ the triangulation of Ω (or, if necessary, that of Ω_h), we may define finite dimension subspaces of $H^1(\Omega)$, according to the FEM technique, as follows

$$X_h = \left\{ v_h \in C^0(\bar{\Omega}) / v_h|_T \in \mathcal{P}_m(T), \text{ for all } T \in \mathcal{T}_h \right\} \tag{6}$$

where $\mathcal{P}_m(T)$ is the space of real-valued polynomials of degree m defined in T. Typical values of m are $m = 1$ and $m = 2$.

Remark 3 *The subscript h in this notation stands for the mesh diameter, that is,*

$$h = \max_{T \in \mathcal{T}_h} h_T, \quad h_T = \max_{x,y \in T} |x - y|.$$

In particular, $h \to 0$ means that the triangles becomes smaller, and thus, the number of them increases.

Consider the space X_h defined in (6) and let $m = 1$. Take a function $v \in X_h$ and $T \in \mathcal{T}_h$. Then, $v_h|_T$ is a polynomial of degree one, that is, there exist some constants a_T, b_T, $c_T \in \mathbb{R}$ such that $v_h(x, y) = a_T x + b_T y + c_T$ for all $(x, y) \in T$. In particular, $v_h|_T$ is completely determined in T if we only know the values of $v_h(x_0, y_0)$, $v_h(x_1, y_1)$ and $v_h(x_2, y_2)$, as soon as the points (x_0, y_0), (x_1, y_1) $(x_2, y_2) \in T$ are noncollinear. A usual choice is the three vertices of T. This means, that any function $v_h \in X_h$ is completely determined by the set of values $\{v_h(a_k)\}_{k=1}^{M}$, where $\{a_1, \ldots, a_M\}$ is the set of the M vertices of the triangulation $\mathcal{T}_h$. In this case, the vertices are said to be the degrees of freedom of the functions of X_h. Notice that this yields $\dim X_h = M$, the number of vertices of $\mathcal{T}_h$. Now consider the so-called hat functions $\ell_1, \ldots, \ell_M \in X_h$ defined as

$$\ell_i(a_j) = \delta_{ij}, \quad 1 \le i, j \le M,$$

where δ_{ij} is the Kronecker symbol (or the identity matrix). It is straightforward that the set $\{\ell_1, \ldots, \ell_M\}$ is a base of the space X_h. It is very easy to check that the hat functions fulfilled both conditions (a) and (b) above.

If we let $m = 2$ in the definition of X_h, then it is an easy task to verify that the degrees of freedom of the $\mathcal{P}_2$ approximation are the set of vertices together with the mid points of sides of all the triangles in $\mathcal{T}_h$, and also we can exhibit a base of X_h verifying conditions (a) and (b).

3 Solving PDEs boundary value problem with `Freefem++`

In this section we introduce the `Freefem++` programming language. `Freefem++` is an integrated software for obtaining the FEM numerical solution to boundary value problems for PDE of the elliptic type. By means of some semidiscretization in time scheme, `Freefem++` may also deal with parabolic/hyperbolic evolution problems. This includes problems arising in solid mechanics, fluid mechanics, plasma physics, etc. This software can be freely downloaded from the Internet (`http://www.freefem.org/ff++/index.htm`) and it is fully documented ([1]). Though initially developed by O. Pironneau (Laboratoire Jacques-Louis Lions, Université Pierre et Marie Curie, Paris) in the 1980s, today it is maintained by F. Hecht, (Laboratoire JLL).

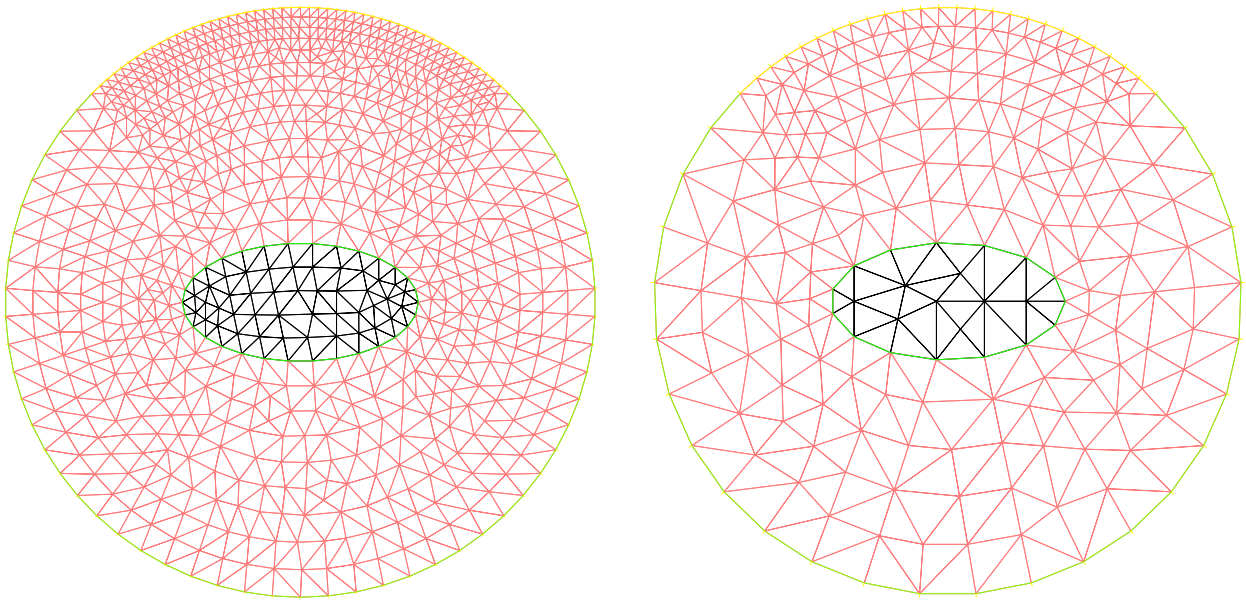


Figure 2: These pictures show two examples of a triangulation of the domain Ω appearing in Figure 1, or rather of Ω_h. Both meshes are automatically generated from a selected set of points on the borders (boundary and interface). On the left we have used double the number of points used on the mesh shown on the right. Consequently, the resulting mesh on the right is coarser than the one on the left.

3.1 A stationary problem

Suppose we want to compute the numerical solution of (1) by the FEM. We already know that the equivalent variational formulation is given by (3). Naturally, we need to provide all the necessary data taking part in the definition of this problem. But we also need to build a mesh and to decide the approximation type of the FEM to be used (which determines the discrete space X_h).

```
1  real radius=5.;
2
3  border Gamma0(t=pi/4,3*pi/4){x=radius*cos(t);
4                               y=radius*sin(t)
                                   ;};
5  border Gamma1(t=3*pi/4,2*pi+pi/4){
6                 x=radius*cos(t);
7                 y=radius*sin(t);};
8  border gamma(t=0,2*pi){x=2*cos(t);y=sin(t);};
9
10 plot(Gamma0(50)+Gamma1(50) + gamma(30),
11      ps="points-on-the-borders.eps");
12
13 mesh Th = buildmesh(Gamma0(50)
14           + Gamma1(50) + gamma(30));
15 mesh Thcoarser = buildmesh(Gamma0(25)
16           + Gamma1(25) + gamma(15));
17
18 plot(Th,ps="circle-ellipse.eps");
19 plot(Thcoarser,
20      ps="circle-ellipse-coarser.eps");
```

The first line introduces a new real variable, `radius` and assigns it the value 5 (this is the radius of the circle).

Lines 3 through 8 declare the borders we are going to use in this problem. Notice that the boundary of Ω is formed by the two border pieces `Gamma0` and `Gamma1`: you just give their respective parametrizations. On the other hand, `gamma` defines here the interface between the two different materials from which Ω is made of. Lines 10 and 11 open a new graph window to represent the borders `Gamma0`, `Gamma1` and `gamma` with 50, 50 and 30 points on them, respectively (Figure 3). It also saves the result in a PostScript file.

Lines 13 through 16 automatically generate the meshes `Th` and `Thcoarser` (see Figure 2) from the three borders already declared. Notice that `Thcoarser` is a coarser mesh since in this case the number of points on every border is half the number of the border points for the mesh `Th`. Lines 18 through 20 plot successively in the graph window the two meshes previously generated and save them in PostScript format.

```
21 fespace Xh(Th,P2);
22 Xh u, v;
23
24 func fsource = 25-0.7*(x^2+8*x*y^2);
25 func gDirichlet = 350.;// in K
26
27 real k1=1.e2, k2=1.e4;// diffusion coefficient
28 real alpha = 5.;// transfert coefficient
29 func uext = 300.;// in K
```

Line 21 introduces the finite element space X_h given in (6), noted as `Xh`, the order of approximation being here $m = 2$.

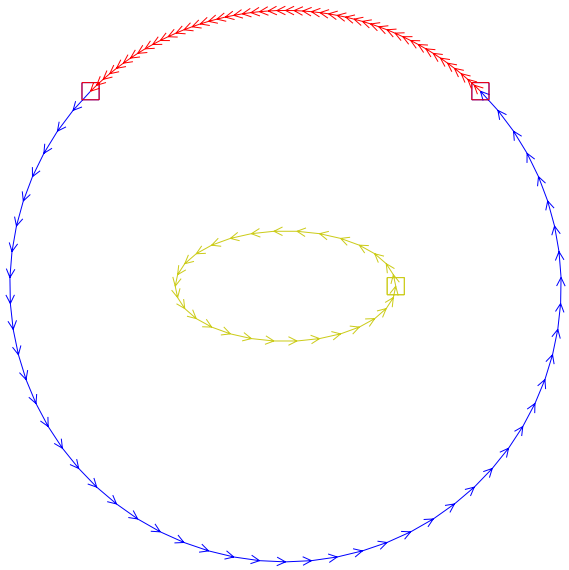


Figure 3: Borders with regular distributed points on. The arrows indicate that the arcs are traversed counterclockwise.

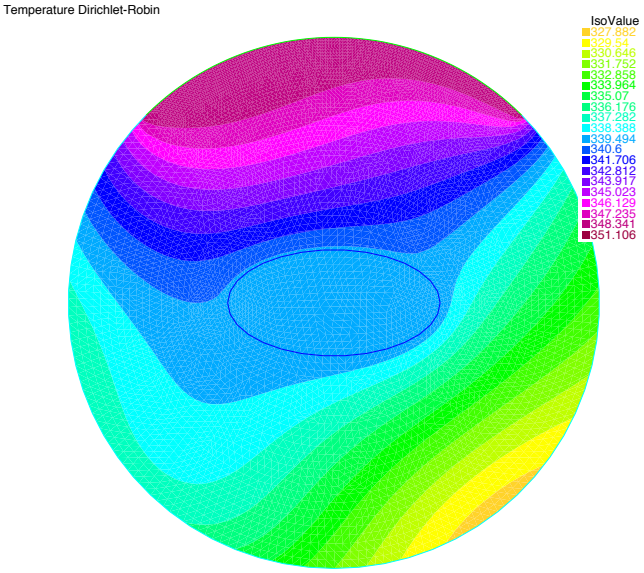


Figure 4: Solution to problem (1) computed with the `Freefem++` script described in this section. The plot directive `fill=1` fills in different colors the isovalues of the solution, whereas `value=1` puts a legend with a table of the isovalues and the corresponding colors used in this graphical representation.

The next line declares new variables of the `Xh` type.

Line 24 defines a new user function, `fsource`, that implements the function $f(x, y) = 25 - 0.7(x^2 + y^2)$. This is the right hand side of our PDE. In the same way, `gDirichlet` implements the constant function $g(x, y) = 350$, the boundary Dirichlet datum on Γ_0.

```
30 int dom1 = Th(0.9*radius,0.).region;
31 int dom2 = Th(0.,0.).region;
32
33 cout << "Domain 1 has reference value " <<
       dom1 << endl;
34 cout << "Domain 2 has reference value " <<
       dom2 << endl;
35
36 fespace Vh(Th,P0);
37 Vh kdiffusion;
38
39 kdiffusion = k1*(region == dom1) + k2*(region
       == dom2);
```

Line 30 creates a label for the subdomain Ω_1 and assigns this value to the new integer variable `dom1`. Idem for Ω_2 and `dom2`. These two values are needed in order to implement the function $k(x, y)$ given in (2). To do so, we first introduce the space V_h, (`Vh` in our script) of piecewise constant functions relative to $\mathcal{T}_h$, namely $V_h = \left\{ v_h \in L^2(\Omega) / v_h|_T \in \mathcal{P}_0(T), \text{ for all } T \in \mathcal{T}_h \right\}$.

Finally, line 39 implements the required thermal conductivity $k(x, y)$.

```
40 problem diffuDR(u,v) =
41     int2d(Th)(kdiffusion*(dx(u)*dx(v)
42                 + dy(u)*dy(v)))
43   + int1d(Th,Gamma1)(alpha*u*v)
44   + int1d(Th,Gamma1)(alpha*uext*v)
45   - int2d(Th)  (fsource*v)
46     + on(Gamma0,u=gDirichlet);
47
48
49 diffuDR;
50
51 plot(u,fill=1,value=1,
52      cmm="Temperature Dirichlet-Robin",
53      ps="tempDR-stationary.eps");
```

We are ready for the setting of the problem in the `Freefem++` language. This is done in lines 40 through 47. One can easily identify how every term in the variational formulation (3) is written in `Freefem++`. Moreover, the Dirichlet boundary condition $u = g$ on Γ_0 is just added in the definition of the problem in a quite simple way as `on(Gamma0,u=gDirichlet)`.

Line 49 calls for the resolution of the problem. The solution is in `u`. Finally, the solution is plotted in the graph window (see Figure 4) and is saved in PostScript format.

3.2 An evolution problem

We still consider the same domain Ω given in Figure 1, but with a different thermal conductivity. We want to compute the temperature distribution in Ω during a time interval, say $[0, T]$ where $T > 0$ is given, and with the same type of boundary conditions. The temperature at any point $x \in \Omega$ and at any time $t \in [0, T]$ verifies an initial condition/boundary value problem

for a PDE of parabolic type, namely

$$\begin{cases} \rho c_\epsilon \dfrac{\partial u}{\partial t} - \nabla \cdot (k \nabla u) = f \quad \text{in } \Omega \times (0,T), \\ \quad u = g \quad \text{on } \Gamma_0 \times (0,T), \\ k \dfrac{\partial u}{\partial n} = \alpha(u_{\text{ext}} - u) \quad \text{on } \Gamma_1 \times (0,T), \\ u|_{t=0} = u_0 \quad \text{in } \Omega, \end{cases} \tag{7}$$

where ρ is the density and c_ϵ is the specific heat at constant strain. To simplify the exposition, we assume that ρ, c_ϵ, α and u_{ext} are constant functions in Ω. The thermal conductivity k is again like in (2), but this time we take $k_1 \gg k_2$. Finally, u_0 stands for the initial temperature, which is supposed to be known.

In order to obtain the numerical solution to this problem using `Freefem++`, the first thing to do is a semidiscretization in time, that is, we apply a finite difference scheme to approximate the time derivative term. For instance, we can use a Crank-Nicolson scheme: let $M > 1$, $\Delta t = T/M$ and for, $j = 0, \ldots, M-1$, define $u^{j+1}(x) \simeq u(x,(j+1)\Delta t)$ as the unique solution to the elliptic problem

$$\begin{cases} \rho c_\epsilon \dfrac{u^{j+1} - u^j}{\Delta t} - \nabla \cdot \left[k \nabla \left(\dfrac{u^{j+1} + u^j}{2} \right) \right] \\ \qquad = \dfrac{f^{j+1} + f^j}{2} \quad \text{in } \Omega, \\ u^{j+1} = g^{j+1} \quad \text{on } \Gamma_0, \\ k \dfrac{\partial u^{j+1}}{\partial n} = \alpha(u_{\text{ext}} - u^{j+1}) \quad \text{on } \Gamma_1, \end{cases} \tag{8}$$

where $u^0 = u_0$. Putting $w^{j+1} = (u^{j+1} + u^j)/2$, it is an easy task to verify that w^{j+1} solves the following problem

$$\begin{cases} \rho c_\epsilon w^{j+1} - \nabla \cdot (\frac{\Delta t}{2} k \nabla w^{j+1}) = F^{j+1} \quad \text{in } \Omega, \\ \quad w^{j+1} = G^{j+1} \quad \text{on } \Gamma_0, \\ k \dfrac{\partial w^{j+1}}{\partial n} = \alpha(u_{\text{ext}} - w^{j+1}) \quad \text{on } \Gamma_1, \end{cases} \tag{9}$$

where $f^j(x) = f(x, j\Delta t)$, $g^j(x) = g(x, j\Delta t)$ and then

$$F^{j+1} = \frac{\Delta t}{4}(f^{j+1} + f^j) + \rho c_\epsilon u^j, \text{ and } G^{j+1} = \frac{g^{j+1} + g^j}{2}.$$

Thus, w^{j+1} can be computed if we already know u^j, then we the new iteration is given by $u^{j+1} = 2w^{j+1} - u^j$ and continue till j gets the value $M-1$. Consequently, we need to compute a sequence of elliptic problems. This can be easily done with `Freefem++` by using a typical `for`-block structure. In this example, lines 86 and 87 introduce the new user function `fsource` standing for the right hand side

$$f(x,y,t) = 25 - 0.7t(x^2 + y^2),$$

whereas `gDirichlet` given in the next two lines stands for the Dirichlet datum

$$g(x,y,t) = 273. + 6t(10 + x).$$

```
real rho = 1.e3; // density
real cspecheat = 50.; // specific heat
real Tfinal = 5.5; // Final time

int Niter = 55; // Number of iterations
real deltat = Tfinal/Niter,
     deltat2 = deltat/2. ;

real radius=5.;
border Gamma0(t=pi/4,3*pi/4){
                x=radius*cos(t);
                y=radius*sin(t);};
border Gamma1(t=3*pi/4,2*pi+pi/4){
                x=radius*cos(t);
                y=radius*sin(t);};
border gamma(t=0,2*pi){x=2*cos(t);
                       y=sin(t);};
mesh Th = buildmesh(Gamma0(50)
         + Gamma1(50) + gamma(30));
fespace Xh(Th,P2);
Xh u, un1, v, Fsource, uaux, GDirichlet;

real k1=2.e5, k2=1.e3;// diffusion coeff.
real alpha = 5.;// transfert coefficient
func uext = 300.;// room temperature

int dom1 = Th(0.9*radius,0.).region;
int dom2 = Th(0.,0.).region;

cout << "Domain 1 has reference value " <<
    dom1 << endl;
cout << "Domain 2 has reference value " <<
    dom2 << endl;

func real fsource(real x, real y, real t)
     {return (25-0.7*t*(x^2+y^2));} // Right
         hand side f
func real gDirichlet(real x, real y, real t)
     {return 273.+6*t*(10+x);} // Dirichlet
         datum

fespace Vh(Th,P0);
Vh kdiffusion;

kdiffusion = k1*(region == dom1) + k2*(region
    == dom2);

problem heat(u,v) = // problem for w^{j+1}
   int2d(Th)(rho*cspecheat*u*v)
 + int2d(Th)(kdiffusion*deltat2
          *(dx(u)*dx(v)+dy(u)*dy(v)))
 + int1d(Th,Gamma1)(deltat2*alpha*u*v)
 - int1d(Th,Gamma1)(deltat2*alpha*uext*v)
 - int2d(Th)(Fsource*v)
   + on(Gamma0,u=GDirichlet);
```

```
104 func uinit = 273.; // Initial temperature
105 uaux = uinit; // Initialize the iterative
      process
106  for(real t=0;t<=Tfinal-deltat;t+=deltat)
107  {
108  plot(uaux,cmm="Temperature t="+t, fill=1,
       value=1, wait=0,
109       ps="temperature"+t+".eps");
110
111  Fsource = deltat*(fsource(x,y,t+deltat)+
       fsource(x,y,t))/4.
112                   +rho*cspecheat*uaux;
113  GDirichlet = (gDirichlet(x,y,t+deltat)+
       gDirichlet(x,y,t))/2.;
114
115  heat; // solves for w^{j+1}
116
117  un1 = 2*u - uaux; // new iteration u^{j+1}
118  uaux = un1;
119  }
120  plot(uaux,cmm="Temperature t="+Tfinal,
121       fill=1, value=1,
122       ps="temperature"+Tfinal+".eps");
```

In this script, what is really new can be found in the `for`-block in lines 106 through 119. This is the iterative scheme (9). Figure 5 shows the transient temperature distribution at time instants $t = 1$, $t = 2.5$, $t = 4$ and $t = 5.5$. As one may expects, the response of the two materials is unequal due to their different thermal conductivities.

4 Some facts about steel: phases and heat treating

Steel is an alloy of iron and carbon. Steel used for industrial purposes, has a carbon content up to about 2 wt%. Other alloying elements may be present, such as Cr and V in tools steels, or Si, Mn, Ni and Cr in stainless steels. Most structural components in mechanical engineering are made of steel. Certain of these components, such as wheel gears, bevel gears, pinions and so on, engaged each others in order to transmit some kind of (rotational or longitudinal) movement. As a result the contact surfaces of these components are particularly stressed. The goal of heat treating of steel is to attain a satisfactory hardness. Prior to heat treating, steel is a soft and ductile material. Without a hardening treatment, and due to the surface stresses, the gear teeth will soon get damaged and they will no longer engage correctly. Thus, an industrial procedure for heat treating is necessary in order to increase the hardness of the steel along the tooth profile and at the same time maintain the rest of the workpiece soft and ductile in order to reduce fatigue (Figure 7).

Solid steel may be present at different phases, namely austenite, martensite, bainite, pearlite and ferrite. The phase diagram of steel is shown in Figure 6. For a given wt% of carbon content up to 2.11, all steel phases are transformed into austenite provided the temperature has been raised up to a certain range. The minimum austenization temperature (727°) is attained for a carbon content of 0.77 wt% (eutectoid steel). Upon cooling, the austenite is transformed back into the other phases (see Figure 8), but its distribution depends strongly on the cooling strategy ([5, 17]). Martensite is the hardest constituent in steel, but at the same time is the most brittle, whereas pearlite is the softest and more ductile phase. Martensite derives from austenite and can be obtained only if the cooling rate is high enough. Otherwise, the rest of the steel phases will appear.

The hardness of the martensite phase is due to a strong supersaturation of carbon atoms in the iron lattice and to a high density of crystal defects. From the industrial standpoint, heat treating of steel has a collateral problem: hardening is usually accompanied by distortions of the workpiece. The main reasons of these distortions are due to (1) thermal strains, since steel phases undergo different volumetric changes during the heating and cooling processes, and (2) experiments with steel workpieces under applied loading show an irreversible deformation even when the equivalent stress corresponding to the load is in the elastic range. This effect is called transformation induced plasticity.

Depending on the workpiece geometry and size, the heating stage can be accomplished by an induction procedure or by direct flame. Induction techniques have been successfully used in industry since the last century. During a time interval, a high frequency current passes through a coil generating a high alternating magnetic field which induces eddy currents in the workpiece, which is placed close to the coil. The eddy currents dissipate energy in the workpiece producing the necessary heating. Figure 9 shows an induction machine during the heating process applied to a gear; we can see the coil encircling the gear.

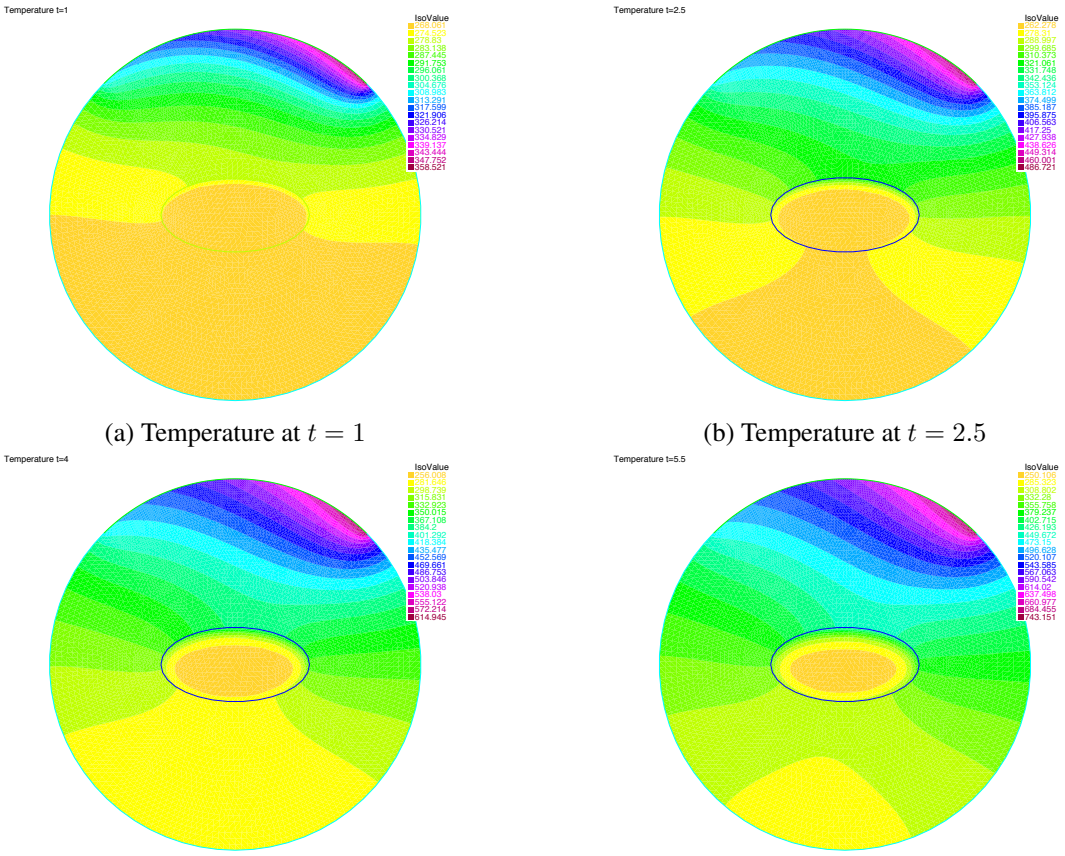
5 Induction hardening of a helical gear

There are many industrial hardening procedures for gears. This may depend on the size of the gear and on the final purpose of the workpiece. Induction hardening is a very general procedure widely used in the manufacture of gears. It can be applied in different manners ([2, 3, 20, 21]): (a) total induction around the workpiece, by the use of a coil surrounding the gear, or (b) local induction, by a tooth-by-tooth or a gap-by-gap techniques. In the first case, the encircling coil hardens the teeth from the tips downward. This pattern may be acceptable for certain types of gears such as splines and some gearings. Another induction technique producing a uniform contour hardening makes use of multifrequency induction ([13, 14, 15, 20]).

In this paper, we first consider a helical gear (Figure 10 left) encircled by a coil (made of copper) as shown like in Figure 10 right.

5.1 The mathematical model

Here we consider a very simple model for the induction hardening of a helical gear without taking into account mechanical



(a) Temperature at $t = 1$

(b) Temperature at $t = 2.5$

(c) Temperature at $t = 4$

(d) Temperature at $t = 5.5$

Figure 5: Numerical solution to the evolution problem (7): temperature distribution at four different time instants.

effects describing the deformation of the workpiece and keeping only the dynamics of the austenite and martensite phase fractions. For a more complete model, the reader is referred to [8, 13, 15, 19].

The mathematical model is divided into two stages: (a) heating, and (b) cooling. During the first stage, heat is produced by electromagnetics, letting pass a high frequency current through the coil. Here, the main variables are the electric potential, the magnetic potential, the temperature and phase fraction corresponding to the austenite transformation. At the beginning of the cooling stage, the current is switched off, and the main variables become the temperature and the phase fractions corresponding to austenite and martensite.

The heating stage

Since we are dealing with high frequency current, we may consider that our model is already written in the harmonic regime ([4, 10]). We denote by φ the complex-valued electric potential, $\boldsymbol{A} = (A_1, A_2, A_3)$ the complex-valued magnetic vector potential, u the temperature and a the austenite phase fraction. In this situation, the model reads as follows.

$$\nabla \cdot (\sigma(u)\nabla\varphi) = 0 \text{ in } \Omega \times (0, T_a), \tag{10}$$

$$\frac{\partial \varphi}{\partial n} = 0 \text{ on } \partial\Omega \times (0, T_a), \tag{11}$$

$$\sigma(u)\frac{\partial \varphi}{\partial n}\bigg|_{\Gamma_1 \times (0,T_a)} = \boldsymbol{j},\ \sigma(u)\frac{\partial \varphi}{\partial n}\bigg|_{\Gamma_2 \times (0,T_a)} = -\boldsymbol{j}, \tag{12}$$

$$\begin{aligned} i\omega\sigma_0(u)\boldsymbol{A} + \nabla \times \left(\frac{1}{\mu}\nabla \times \boldsymbol{A}\right) - \delta\nabla(\nabla \cdot \boldsymbol{A}) \\ = -\sigma_0(u)\nabla\varphi = 0 \text{ in } D \times (0, T_a), \end{aligned} \tag{13}$$

$$\boldsymbol{A} = \boldsymbol{0} \text{ on } \partial D \times (0, T_a), \tag{14}$$

$$\begin{aligned} \rho c_\epsilon \frac{\partial u}{\partial t} - \nabla \cdot (\kappa(u)\nabla u) = \frac{\sigma_0(u)}{2}|\omega \boldsymbol{A}|^2 \\ -\rho L_1 \frac{\partial a}{\partial t} \text{ in } \Omega \times (0, T_a), \end{aligned} \tag{15}$$

$$\frac{\partial u}{\partial n} = 0 \text{ on } \partial\Omega \times (0, T_a), \tag{16}$$

$$u(\cdot, 0) = u_0 \text{ in } \Omega, \tag{17}$$

$$\frac{\partial a}{\partial t} = f_1(u, a) \text{ in } \Omega \times (0, T_a), \tag{18}$$

$$a(x, 0) = 0 \text{ in } \Omega, \tag{19}$$

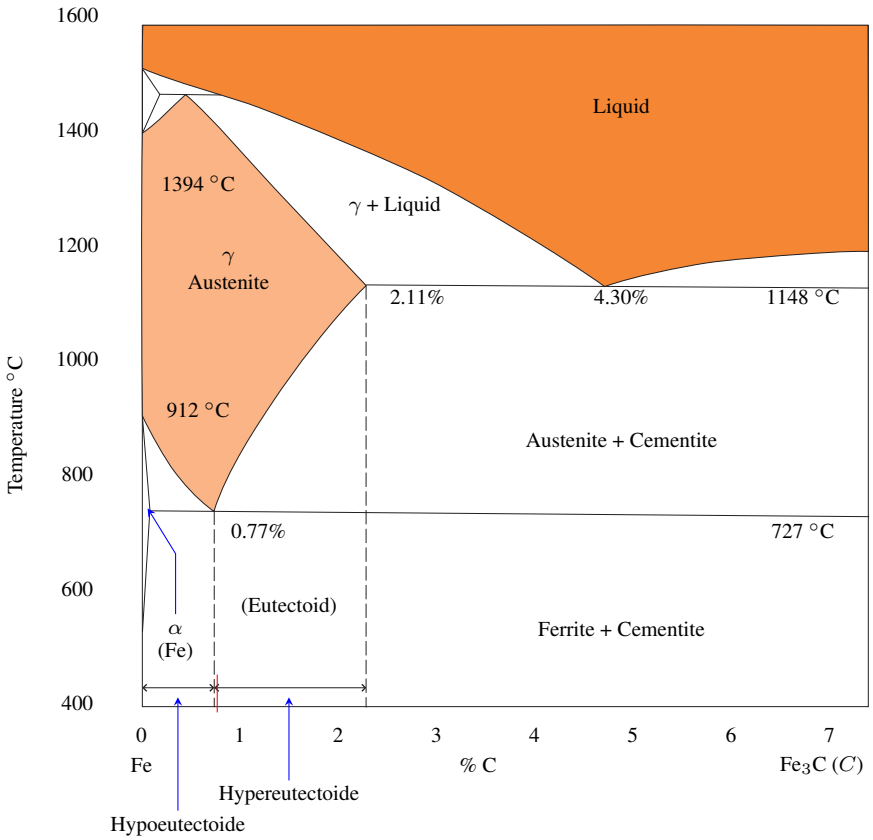


Figure 6: Iron-carbide phase diagram.

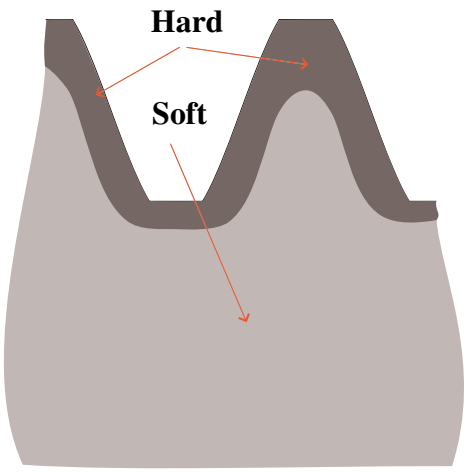


Figure 7: Martensite layer near the workpiece tooth boundary.

where Ω stands for the gear, D is a large enough box containing both the gear and the coil (where it is defined the magnetic vector potential), $T_a > 0$ is the final time of the heating stage (when austenization is happening) and Γ_1, Γ_2 are two opposite faces on the coil boundary, $\sigma(u)$ is the temperature dependent electric conductivity. The Neumann boundary datum $\boldsymbol{j}$ is the external source current density. The function $\sigma_0 \colon D \times (0, T_a) \times \mathbb{R} \mapsto \mathbb{R}$ is given as follows

$$\sigma_0(x,t,s) = \begin{cases} \sigma(x,t,s) \text{ if } (x,t) \in \Omega^{\text{cond}} \times (0,T_a),\ s \in \mathbb{R}, \\ 0 \qquad\qquad \text{elsewhere.} \end{cases}$$

where $\Omega^{\text{cond}} = \Omega \cup \Omega^{\text{coil}}$ is the set of conductors (that is, the steel gear together with the copper coil).

The function $\mu \colon D \mapsto \mathbb{R}$ stands for the magnetic permeability and is given by

$$\mu(x) = \mu_1 \chi_\Omega + \mu_2 \chi_{\Omega^{\text{coil}}} + \mu_3 \chi_{D \setminus \Omega^{\text{cond}})},$$

where μ_i, $1 \leq i \leq 3$, are constant values such that $0 < \mu_2 < \mu_3 \ll \mu_1$; ω is the frequency, $\delta > 0$ is a small parameter, ρ is the density, c_ϵ the specific heat capacity at constant strain and κ is the temperature dependent thermal conductivity. L_1 is the latent heat and u_0 is the initial temperature. Finally, the function $f_1(u,a)$ describes the dynamics of the austenite production. Here, a Leblond-Devaux model is considered ([18]), namely

$$f_1(u,a) = \max\left\{\frac{1}{\tau_a(u)}(a_{\text{eq}}(u) - a), 0\right\} \mathcal{H}(u - A_s).$$

where A_s is the austenite starting temperature, $\mathcal{H}$ stands for the Heaviside function $\mathcal{H}(s) = 1$ if $s \geqslant 0$ and is zero elsewhere. The functions $a_{\text{eq}}(u)$, $\tau_a(u)$ can be determined from experimental data.

The first source term in the right hand of the equation (15) is the so-called Joule's heating.

The cooling stage

During this stage the current is stopped, and the workpiece is rapidly cooled down by aquaquenching. In this stage, the austenite is transformed into martensite. The model reads as follows.

$$\rho c_\epsilon \frac{\partial u}{\partial t} - \nabla \cdot (\kappa(u) \nabla u) = -\rho L_1 \frac{\partial a}{\partial t} - \rho L_2 \frac{\partial m}{\partial t} \text{ in } \Omega \times (T_a, T_m), \tag{20}$$

$$\frac{\partial u}{\partial n} = \alpha(u_{\text{aq}} - u) \text{ on } \partial\Omega \times (T_a, T_m), \tag{21}$$

$$u|_{t=T_a} = u(\cdot, T_a) \text{ in } \Omega. \tag{22}$$

$$\frac{\partial a}{\partial t} = f_1(u,a) - \frac{\partial m}{\partial t} \text{ in } \Omega \times (T_a, T_m), \tag{23}$$

$$a|_{t=T_a} = a(\cdot, T_a) \text{ in } \Omega, \tag{24}$$

$$\frac{\partial m}{\partial t} = f_2\left(a, m, u, \frac{\partial m}{\partial t}\right) \text{ in } \Omega \times (T_a, T_m), \tag{25}$$

$$m(x,0) = 0 \text{ in } \Omega, \tag{26}$$

Here, T_m is the aquaquenching final time, L_2 is the latent heat relative to the martensite transformation and u_{aq} is the temperature of the liquid used in the aquaquenching. On the other hand, the dynamics of the martensite phase fraction may be described by the function f_2 as follows (see[16])

$$f_2\left(a, m, u, \frac{du}{dt}\right) = c_m(1-m)\mathcal{H}\left(-\frac{du}{dt}\right)\mathcal{H}(M_s - u),$$

where the M_s is the martensite starting temperature from the austenite transformation, and c_m is a constant determined from experimental data.

6 Building a 3D triangulation of a helical gear

This section is devoted to the tetrahedralization of a helical gear. As it was stated above, we only use noncommercial software: `Freefem++`, `TetGen` ([11]), `MEdit` and `gmsh`.

The first thing to do is a triangulation of the gear base. The base has two borders, namely, a tooth boundary and an inner circle. The way to produce the tooth boundary is to use a truncation of a cosine function oscillating around an outer circle. Let R be the outer circle radius, then a parametrization of the teeth is of the form

$$\begin{cases} x = (R + d(t)) \cos t, \\ y = (R + d(t)) \sin t, \quad t \in [0, 2\pi) \end{cases} \tag{27}$$

where $d(t)$ is an oscillating function. For instance, if we want a m-tooth gear we may take

$$d(t) = \min(0.8l, \max(-0.8l, l\cos(mt + \pi))),$$

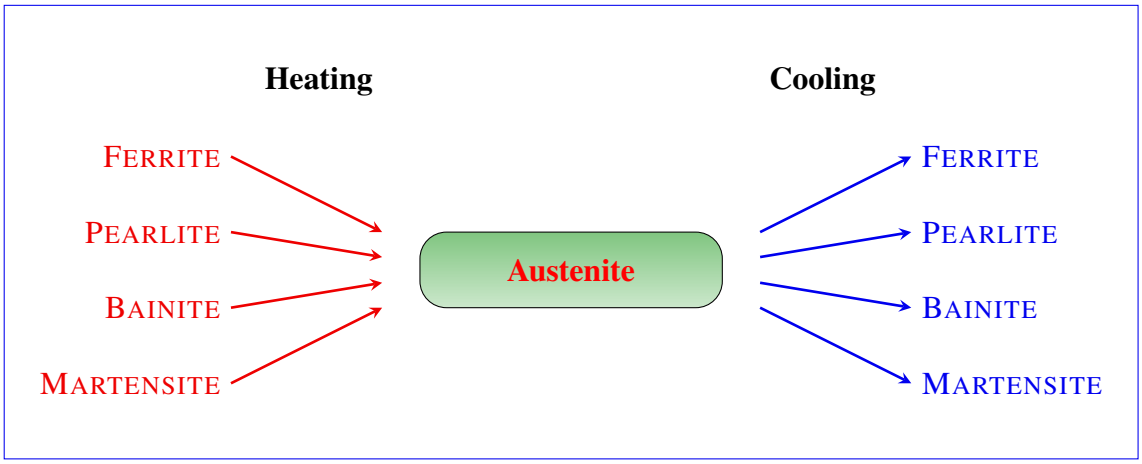


Figure 8: Phase transformation of steel.

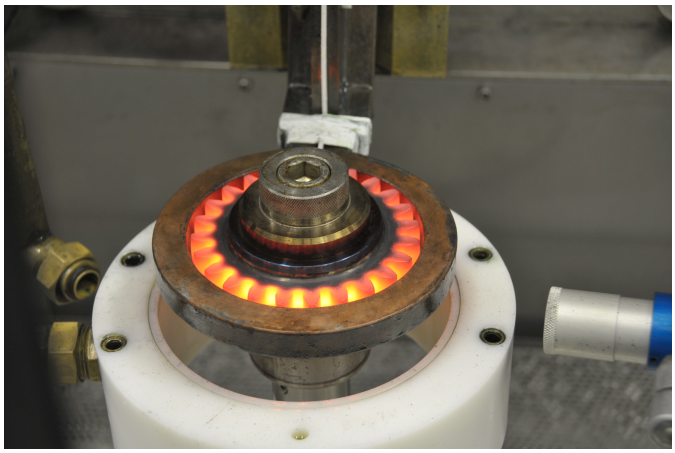


Figure 9: Induction hardening machine at work during the heating process applied to a wheel gear. Courtesy from the Stiftung Institut für Werkstofftechnik - Bremen, Germany (Foundation Institute of Materials Science. IWT Bremen).

where l is the amplitude. Notice that, in order to produce a flat tip and bottom, we are also truncating the cosine by 20% its height and depth (Figure 11).

To start up, we load some necessary packages for 3D meshes.

```
// 3D tetrahedralization of a gear

load "msh3" // add general 3D functionality
load "tetgen" // 3D tetrahedralization
load "medit" // we may call MEdit just
             // for 3D visualization
```

Then, we introduce some useful parameters.

```
real radius = 0.2;// gear radius (cm)
int numteeth = 20;// number of teeth
real toothheight = radius/8.;
//
//The inner cylinder has radius
// equals to radiusportion*radius
real radiusportion = 0.7;

// Half of the gear thickness (cm)
real thickness = 0.04;

//Number of points in the tooth boundary
int npteeth = 200;

// Number of points in the inner ring
int npinnerring = 30;

//Number of layers in the 3D mesh
// of the gear and coil
int nlayer = 7;
//
```

Here comes the implementation of the functions like (27) describing the boundaries.

```
//
// Truncated teeth by 20% tip and bottom
func real tooth(real t){return
    min(0.8*toothheight,
     max(-0.8*toothheight,
      toothheight*cos(numteeth*t+pi)));}

func real coronax(real t){return
       (radius + tooth(t))*cos(t);}

func real coronay(real t){return
       (radius + tooth(t))*sin(t);}

//Outer boundary
border dentadura(t=0,2*pi){
        x=coronax(t);
        y=coronay(t);label=123;}
```

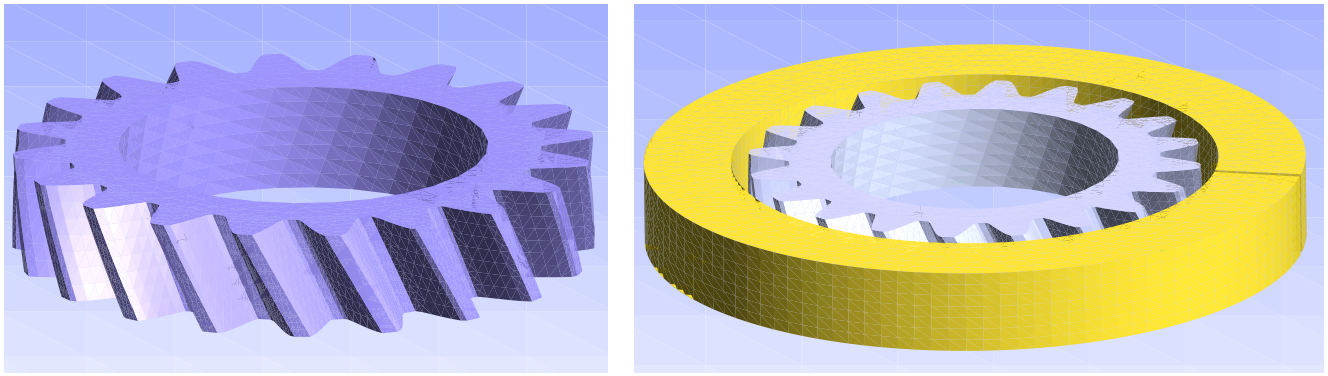


Figure 10: A specimen of a helical gear. Typical setting of the coil and the helical gear for induction hardening. These pictures have been obtained by using the postprocessing software `gmsh`([9]).

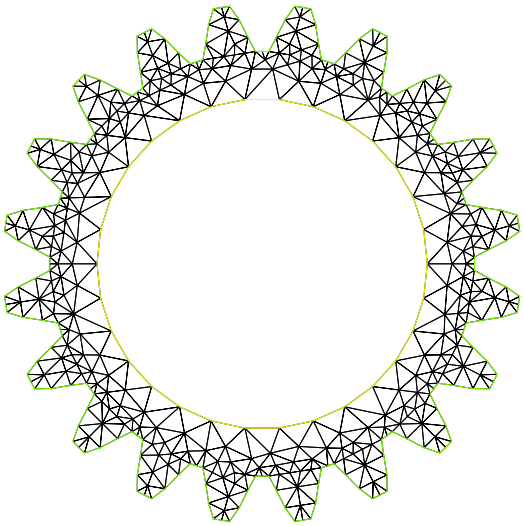


Figure 11: The gear base. The outer boundary is parametrized according to (27).

The inner boundary is just a circle.

```
//Inner boundary
border circle1(t=0,2*pi){
    x=radiusportion*radius*cos(t);
    y=radiusportion*radius*sin(t);}

plot(dentadura(npteeth)
      + circle1(npinnerring));
```

Finally, we generate the mesh. Notice that the number of the boundary points on the border `circle1` is negative. This means that this arc is traversed clockwise and this results in a hole inside the domain. Also, the directive `fixeborder=1` means that during the automatic generation of the mesh, the original points on the boundary will be vertices of the triangulation. This is a very important issue when dealing with 3D meshes.

```
mesh Th=buildmesh(dentadura(npteeth) +
       circle1(-npinnerring),fixeborder=1);

plot(Th,ps="gearbase.eps");
//
```

In order to build the gear mesh, we need to generate a 3D mesh of the whole workpiece boundary. In this case, we may distinguish four different surfaces: the base, the lid, the cylindrical inner wall and the tooth surface. Thus we build a two dimensional mesh in 3D for each of these surfaces. Then we stick out these meshes together forming a closed 3D mesh of the gear skin. This closed 3D mesh is used as a datum to generate the final gear tetrahedralization via `TetGen`.

Here we transform the 2D mesh `Th` into 3D (flat) objets, namely the base (`Thbase`) and the lid (`Thtapa`). These surfaces are assigned the reference numbers 76 and 77, respectively.

```
int[int] labels = [0,76];
mesh3 Thbase = movemesh23(Th,
       transfo=[x,y,-thickness],
       label=labels);
labels = [0,77];
mesh3 Thtapa = movemesh23(Th,
       transfo=[x,y,thickness],
       label=labels);
```

Then we generate the cylindrical inner wall. To do so, we first build a 2D rectangular mesh and transform it into the desired 3D mesh with reference number 79.

```
//
//Inner wall
real x0=0., x1=2.*pi,
     y0=-thickness, y1=thickness;
//Auxiliary mesh of a 2D rectangle
mesh Thauxi =
      square(npinnerring,nlayer,
       [x0+(x1-x0)*x,y0+(y1-y0)*y]);
func XX3 = radiusportion*radius*cos(x);
func YY3 = radiusportion*radius*sin(x);
```

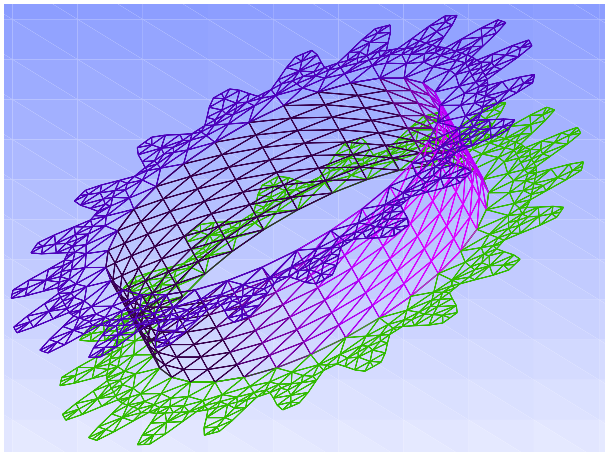


Figure 12: 3D mesh of base, inner cylindrical wall and lid of the gear.

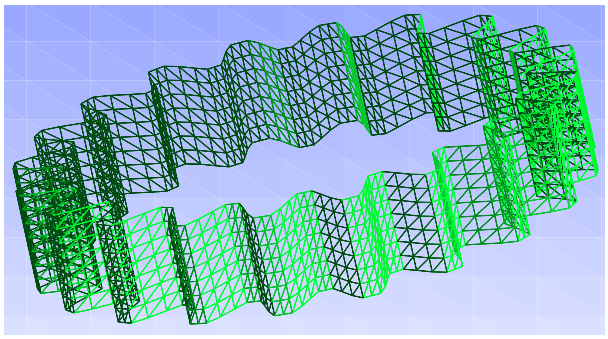


Figure 13: 3D mesh of the gear tooth surface.

```
func ZZ2 = y;
labels = [0,79];
mesh3 Thlaterali = movemesh23(Thauxi,
    transfo=[XX3,YY3,ZZ2],
    label=labels);
```

We stick these three meshes together, just to see that everything is OK. You can do so by saving this new mesh in a file that can be read with `gmsh` (or you just can call MEdit).

```
mesh3 Thprogress = Thbase
       + Thlaterali + Thtapa;
savemesh(Thprogress,"boundaryprogress.mesh");
// Interactive call for MEdit
//medit("Boundary progress", Thprogress);
```

Figure 12 shows the resulting mesh `Thprogress`. In the same way, the lateral tooth surface is obtained by a suitable transformation of a 2D rectangular mesh.

```
// Tooth surface
mesh Thauxe =
  square(npteeth,nlayer,
    [x0+(x1-x0)*x,y0+(y1-y0)*y]);
func XX4 = coronax(x);
func YY4 = coronay(x);
```

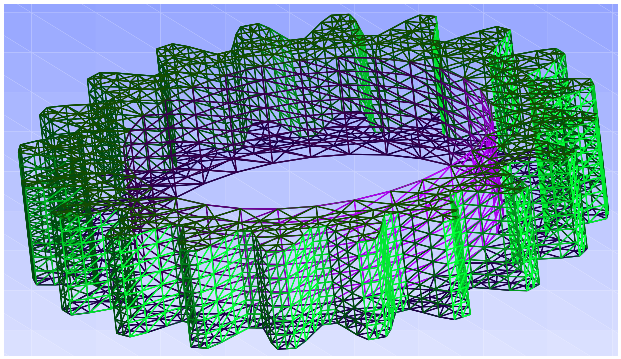


Figure 14: 3D mesh of the full gear boundary.

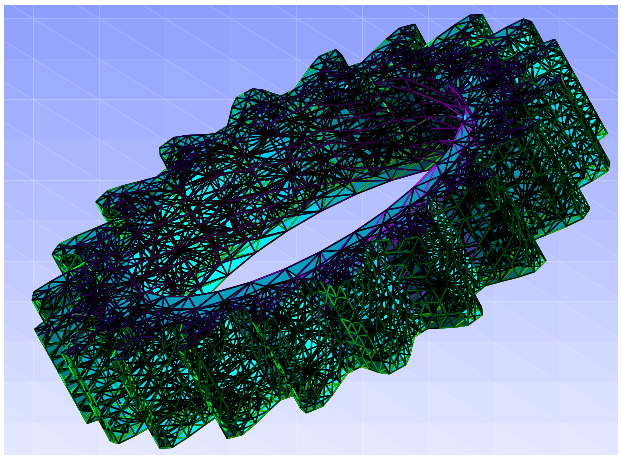


Figure 15: 3D tetrahedra mesh of a straight gear obtained with `TetGen` using the closed 3D boundary in Figure 14 as a datum.

```
labels = [0,78];
mesh3 Thlateralo = movemesh23(Thauxe,
    transfo=[XX4,YY4,y], label=labels);
```

Now we may stick out the four surface meshes together to produce the full gear skin (Figure 14). Notice that the contact points must fit in well.

```
//The full gear skin
mesh3 GearSkin =  Thbase + Thtapa
    + Thlaterali + Thlateralo;
```

Using the 3D skin mesh `GearSkin` as a datum to `TetGen`, a straight gear tetrahedralization is generated as shown in Figure 15.

```

//3D gear mesh (tetrahedra)
real[int] domain =
  [1.05*radiusportion*radius,0.,0.,145,0.1];

mesh3 ThGear3 =
  tetg(GearSkin,    switch="paAAQY",
   nbofregions=1,
   regionlist=domain);
```

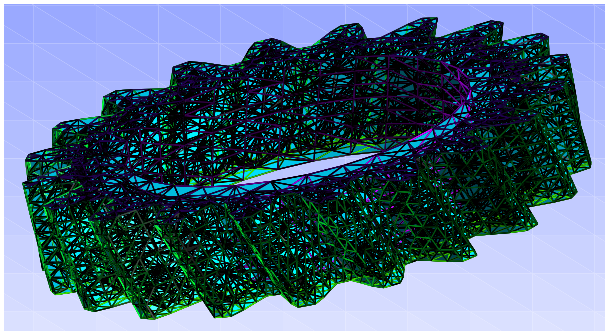


Figure 16: Final 3D tetrahedra mesh of the helical gear obtained by twisting the mesh shown in Figure 14.

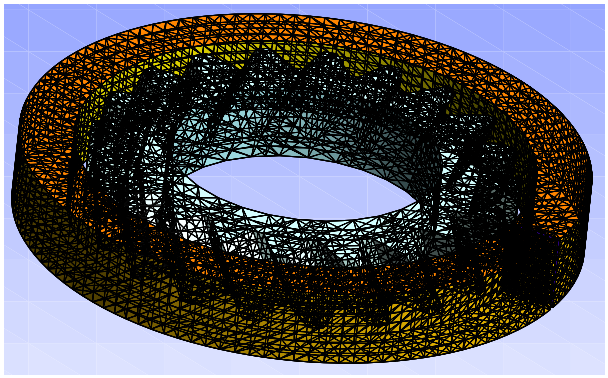


Figure 17: Coil-gear setting.

The final step is to produce a twist onto the straight gear mesh resulting in the helical gear mesh shown in Figure 16.

```
//The helical twist
func XX5 =  cos(2*z)*x + sin(2*z)*y;
func YY5 = -sin(2*z)*x + cos(2*z)*y;

mesh3 ThGear =
  movemesh3(ThGear3,
    transfo=[XX5,YY5,z]);

//medit("Helical Gear", ThGear);

// We save the mesh
savemesh(ThGear,"gear.mesh");
```

Remark 4 *We have not explain every* `Freefem++` *keyword appearing in these scripts. Most of them are self-explanatory but there are many fine points and keywords that admit their own different options. To learn more about this, the reader is referred to the various documentation available in the Internet (see, for instance, [1, 7, 9, 11]).*

In the same way, we may generate a 3D tetrahedra mesh for the coil. Figure 17 shows the 3D skin mesh of the coil-gear setting. Finally, we need a large enough set containing both, the gear and the coil. This is the set D where the magnetic vector potential is to be computed according to the equations (13) and (14) during the heating stage. We may choose the shape of this set as simple as possible, for instance, a box or cube. Figure 18 shows the whole setting cube-coil-gear.

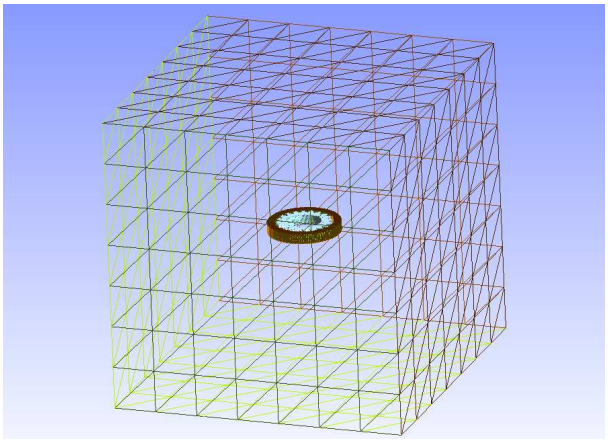


Figure 18: Cube-coil-gear setting.

	Vertices	**Triangles**	**Tetrahedra**
Gear	2598	4696	8137
Coil	8936	7536	41496
Cube	14961	14134	89720

Table 1: Number of vertices, triangles and tetrahedra in each of the three meshes.

Table 1 gives some information about the three generated meshes. It took `Freefem++` 0.016275 seconds to compile the script and 2.92382 seconds to execute it on a MacBook Pro, taking into account the time spent on transferring the mesh files to the hard disk.

7 Numerical simulations

In this section we show some numerical simulations of the model (10)-(26) using the meshes and setting described in the previous section. Every PDE is first written in its own variational formulation. Notice that this is a strongly coupled system of nonlinear PDEs. The numerical solution is computed by means of an iterative method (semidiscretization in time by a finite difference scheme). This allows us to compute successively every unknown at a certain time $t = (j+1)\Delta t$ assuming the rest of variables are known at $t = j\Delta t$. Since the dynamics of the austenite and martensite are modeled by ordinary differential equations with initial values, namely (18), (19), (23)-(26), these phase fractions can be computed by the application of a finite difference scheme (that is, there is no variational formulations for these variables since there is no diffusion term in the equations (18), (23) and (25)).

The heating stage takes $T_a = 5.5$ seconds and the frequency is $\omega = 900$ Hz. The cooling stage takes another 9.5 seconds,

that is $T_m = 15$ seconds.

Figures 19:(a)-(f) show some numerical results at two different instants of time, namely at the end of the heating stage, $t = 5.5$, and at the end of the cooling stage, $t = 15$ for the austenite, martensite and final temperature. As it is expected, induction hardening heats up the gear from the tooth tip downward, so that the whole tooth is austenized (figures 19:(a),(b)) and transform into martensite after cooling (figures 19:(d),(e)). Figure 19:(c) shows that almost all the austenite has been transformed, retaining the workpiece a very small amount of this phase fraction.

Another industrial technique used in the heat treatment of gears is flame hardening. This procedure can be used for both small and large gears. Usually, two techniques can be utilized: spinning and progressive heating ([20, 21]). Imagine that the gear tooth surface is heated up uniformly by a set of burners placed around the workpiece. This situation can be modeled as follows.

$$\rho c_\epsilon \frac{\partial u}{\partial t} - \nabla \cdot (\kappa(u)\nabla u) = -\rho L_1 \frac{\partial a}{\partial t} \text{ in } \Omega \times (0, T_a), \quad (28)$$

$$u = g \text{ on } \Gamma_0 \times (0, T_a), \quad (29)$$

$$\frac{\partial u}{\partial n} = 0 \text{ on } \Gamma_1 \times (0, T_a), \quad (30)$$

$$u(\cdot, 0) = u_0 \text{ in } \Omega, \quad (31)$$

$$\frac{\partial a}{\partial t} = f_1(u, a) \text{ in } \Omega \times (0, T_a), \quad (32)$$

$$a(x, 0) = 0 \text{ in } \Omega, \quad (33)$$

where we keep the same notation as above and also Γ_0 is the gear tooth surface, $\Gamma_1 = \partial\Omega \setminus \Gamma_0$ and the function g describes the action of the burners on the gear tooth surface. The cooling stage is just the same as before.

Figures 20:(a)-(f) show some numerical results corresponding to a flame hardening simulation using the modeling (28)-(33) for the heating stage, and (20)-(26) for the cooling stage. A more uniform contour can be observed in this case for both the austenite and martensite profiles, which is in good agreement with experimental results.

Conclusions

Some simplified mathematical models arising in the industrial processes of steel heat treating have been considered and applied for the numerical simulation of the hardening (induction or flame) of a helical gear. The numerical simulations have been performed with freeware packages available in the Internet, namely `Freefem++`, `TetGen` and `gmsh`. We have shown some features of `Freefem++` as a preprocessing, processing and postprocessing software. We have described how to do, step by step, a 3D mesh for a helical gear. This presentation should be considered at an academic level. We have tried to put in evidence the potentiality and performance of `Freefem++` as a useful FEM tool to carry out fast and efficient numerical simulations in a complex 3D geometry. Of course, it is possible to consider more sophisticated models for the description of the industrial processes of steel heat treating. For instance, by including mechanical effects, deformations, stresses, etc., other phase fractions such as bainite and pearlite. This does not represent a constraint to `Freefem++` since it can handle more complicate situations. Though the 3D meshes used here are coarse, in general the numerical results are in good agreement with known experimental results. We hope that the reader gives a try to these packages and convinces himself of its easy-to-use, versatility, robustness and efficiency.

Acknowledgements

This research was partially supported by Ministerio de Economía y Competitividad under grant MTM2010-16401 with the participation of FEDER, and Consejería de Educación y Ciencia de la Junta de Andalucía, research group FQM-315.

The authors wish to thank Dipl.-Ing. Dawid Nadolski, from the Foundation Institute of Materials Science (IWT Bremen), for granted us permission to use the illustration of the induction hardening machine shown in Fig. 9. The authors are in debt with Prof. Dr. Dietmar Hömberg, from the Weiertraß Institut für Angewandte Analysis und Stochastik, WIAS-Berlin (Weierstraß Institute for Applied Analysis and Stochastics). His expertise and skills have led us to learn much about the mathematical modeling of steel heat treating.

We also thank our dearest colleague José Rafael Rodríguez Galván, from Universidad de Cádiz, who provide us with the file `ff++listings.sty` used in formatting the `Freefem++` scripts of this paper.

Prof. Ortegón Gallego addresses his gratitude to Dr. Thomas Petzold, from WIAS-Berlin, who first learned him about multifrequency induction hardening for gear components and for letting him set contact with his colleague from Bremen Dipl.-Ing. Dawid Nadolski. He also thanks the organizers of the VII Modeling Week, Prof. Valeri Markov, from UCM, and Prof. J. I. Díaz, from IIM and UCM, for letting him participate in the 2013 edition as an instructor about the subject dealt in this paper.

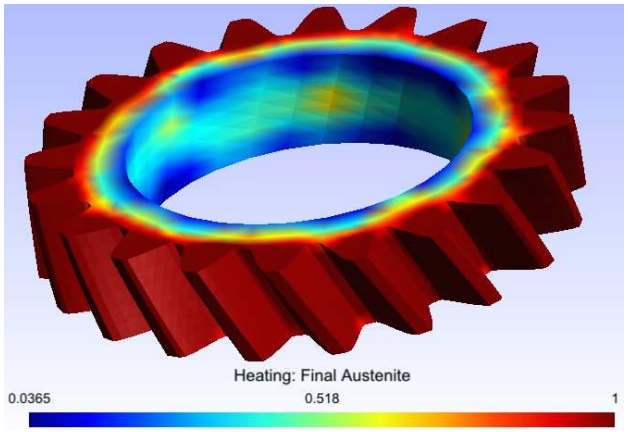
References

[1] S. Auliac, A. Le Hyaric, J. Morice, F. Hecht, K. Ohtsuka, O. Pironneau, *FreeFem++*. Third Edition, Version 3.31-2, 2014. `http://www.freefem.org/ff++/ftp/freefem++doc.pdf`

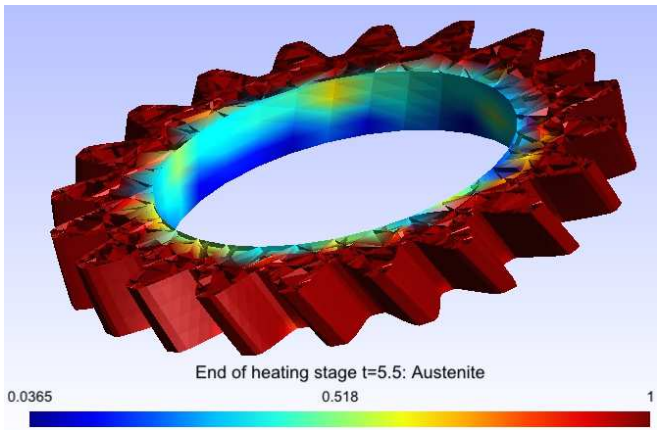
[2] N. Barka, A. Chebak, A. El Ouafi, *Simulation of Helical Gear Heated by Induction Process Using 3D Model*, Advanced Materials Research, **658**, 266–270, 2013.

[3] N. Bugliarello, BĠeorge, D. Giessel, D. McCurdy, R. Perkins, S. Richardson, C. Zimmerman, *Heat treat processes for gears*, Gear Solutions 38–51, July 2010.

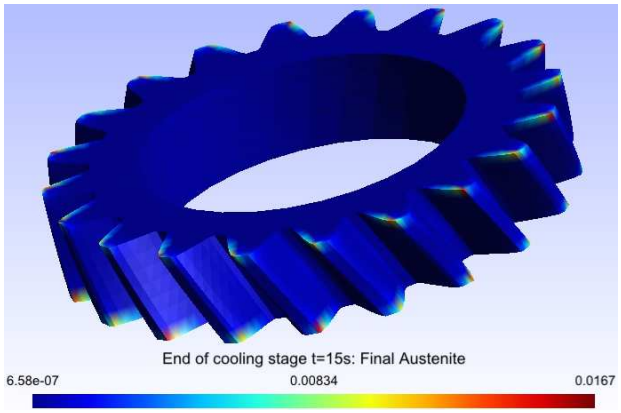
[4] A. Bermúdez, D. Gómez, M.C. Muñiz, P. Salgado, Transient numerical simulation of a thermoelectrical problem in cylindrical induction heating furnaces, Adv. Comput. Math. 26 (2007) 39–62.



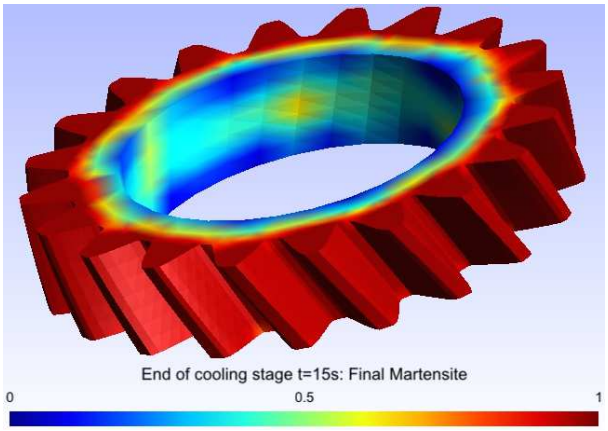
(a) Austenite at the end of the heating stage $t = 5.5$ s.



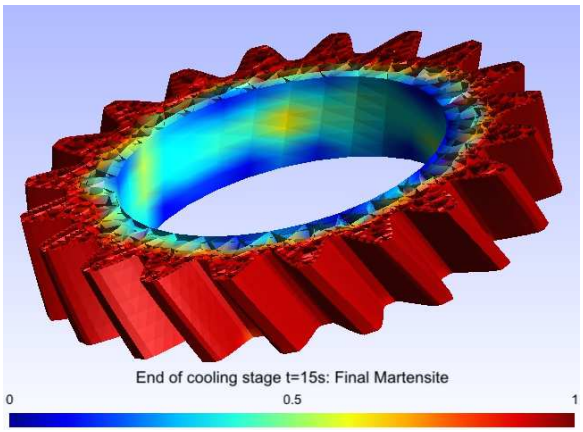
(b) Austenite at the end of the heating stage $t = 5.5$ s. Clipping image.



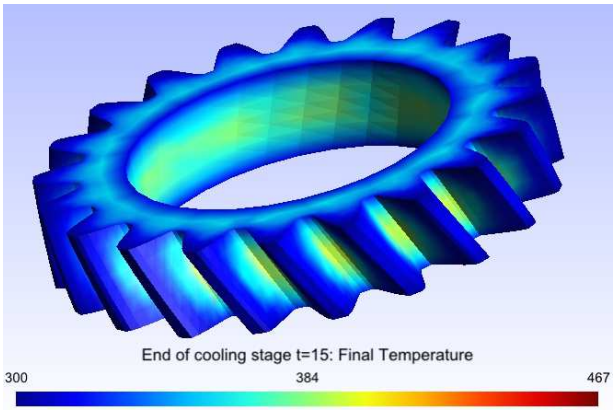
(c) Austenite at the end of the cooling stage $t = 15$ s.



(d) Martensite at the end of the cooling stage $t = 15$ s.

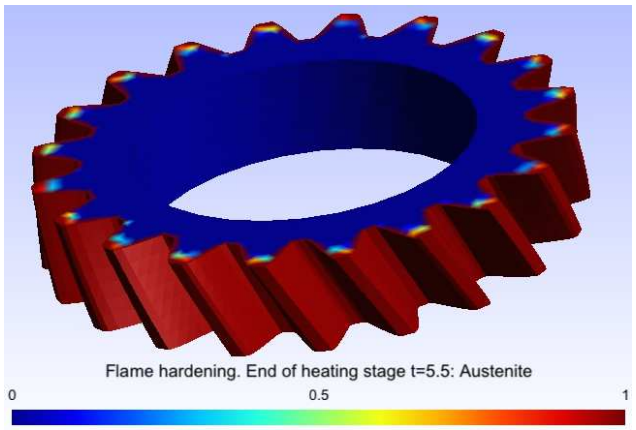


(e) Martensite at the end of the cooling stage $t = 15$ s. Clipping image.

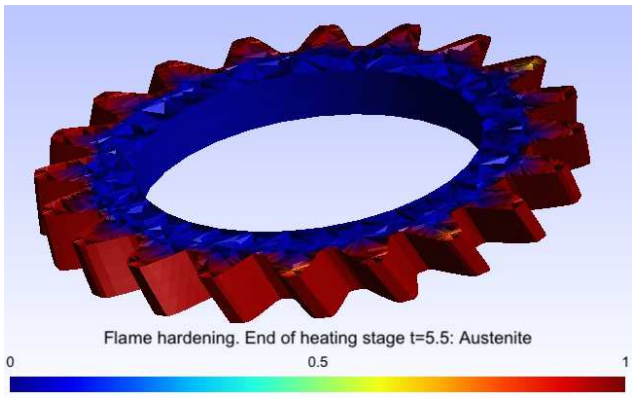


(f) Temperature at the end of the cooling stage $t = 15$ s.

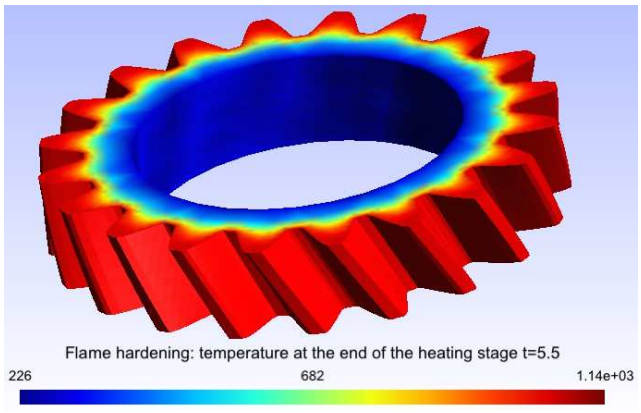
Figure 19: Induction hardening.



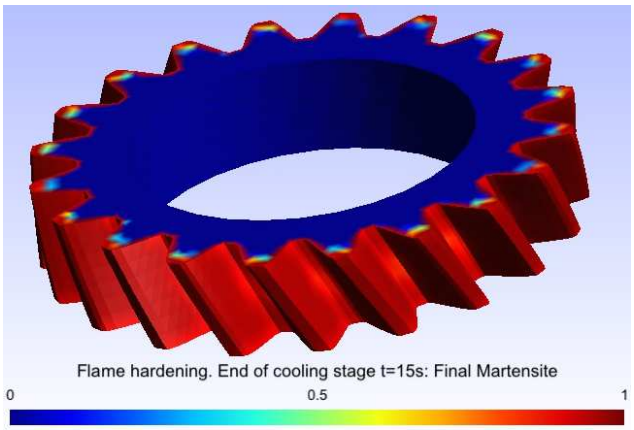
(a) Austenite at the end of the heating stage $t = 5.5$ s.



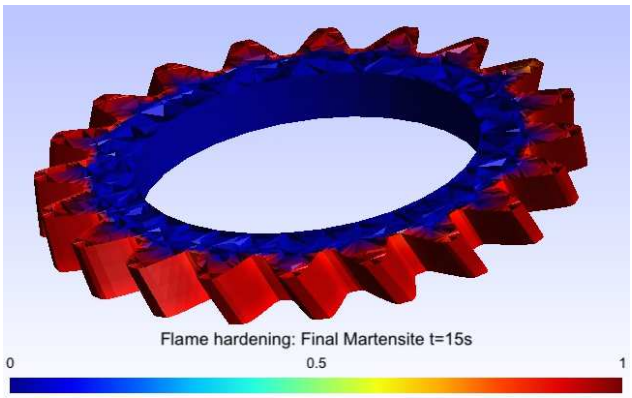
(b) Austenite at the end of the heating stage $t = 5.5$ s. Clipping image.



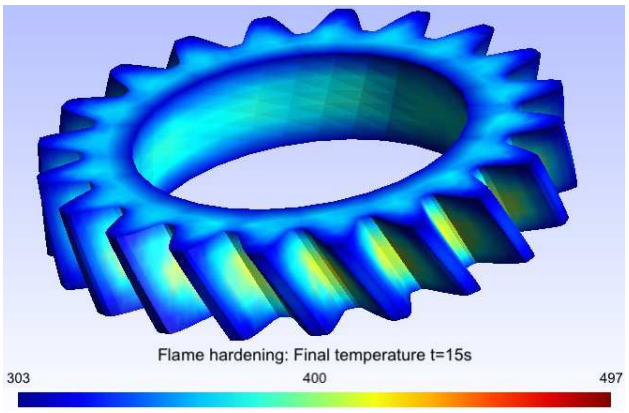
(c) Temperature at the end of the heating stage $t = 5.5$ s.



(d) Martensite at the end of the cooling stage $t = 15$ s.



(e) Martensite at the end of the cooling stage $t = 15$ s. Clipping image.



(f) Temperature at the end of the cooling stage $t = 15$ s.

Figure 20: Flame hardening.

[5] J. R. Davis *et al. ASM Handbook: Heat Treating*, vol. **4**, ASM International, USA, 2007.

[6] J. M. Díaz Moreno, M. T. González Montesinos, C. García Vázquez, F. Ortegón Gallego & G. Viglialoro, *Some Basic Mathematical Elements On Steel Heat Treating: Modeling, Freeware Packages And Numerical Simulation,* Thermal Processing for Gear Solutions, 2014-Fall, 42–47.

[7] P. J. Frey, *MEdit: An interactive mesh visualization software.* Rapport technique INRIA Nº. 0253, 2001.
`http://www.ann.jussieu.fr/~frey/publications/RT-0253.pdf`

[8] J. Fuhrmann, D. Hömberg and M. Uhle, *Numerical simulation of induction hardening of steel,* COMPEL, **18**, No. 3, 482–493, 1999.

[9] C. Geuzaine J. F. Remacle, *Gmsh Reference Manual*, 2014.
`http://geuz.org/gmsh/doc/texinfo/gmsh.pdf`

[10] M. T. González Montesinos, F. Ortegón Gallego, *On an induction?conduction PDEs system in the harmonic regime.* Nonlinear Analysis: Real World Applications, **15**, 58–66, 2014.

[11] Hang Si, *TetGen: A Quality Tetrahedral Mesh Generator and a 3D Delaunay Triangulator.*
`http://tetgen.berlios.de/`, 2009.

[12] Hecht, F. *New development in FreeFem++.* J. Numer. Math. **20** (2012), no. 3-4, 251–265, 65Y15.

[13] D. Hömberg, *A mathematical model for induction hardening including mechanical effects*, Nonlinear Analysis: Real World Applications, **5**, 55–90, 2004.

[14] D. Hömberg , Thomas Petzold, Elisabetta Rocca, Simulation of multi-frequency-induction-hardening including phase transitions and mechanical effects. WIAS Preprint No. 1975, 2014.

[15] D. Hömberg, Q. Liu, J. Montalvo-Urquizo, D. Nadolski, T. Petzold, A. Schulz , Analysis and simulation of multifrequency induction hardening, WIAS Preprint No. 1910, 2014.

[16] D. P. Koistinen, R. E. Marburger, *A general equation prescribing the extent of the austenite-martensite transformation in pure iron-carbon alloys and plain carbon steels*, Acta Met., **7**, 59–60, 1959.

[17] G. Krauss, *Steels: Heat Treatment and Processing Principles*, ASM International, USA, 2000.

[18] J. B. Leblond and J. Devaux, *A new kinetic model for anisothermal metallurgical transformations in steels including effect of austenite grain size*, Acta metall., **32**, No. 1, 137–146, 1984.

[19] J. Montalvo Urquizo, Q. Liu, A. Schmidt, *Simulation of quenching involved in induction hardening including mechanical effects*, Computational Materials Science, **79**, 639-649, 2013.

[20] F. J. Otto, D. H. Herring, *Gear Heat Treatment. Part I*, Heat Treating Progress, June, 2002.

[21] V. Rudnev, D. Loveless, R. Cook, M. Black, *Induction Hardening of Gears: a Review*, Heat Treatment of Metals, **4**, 97–103, 2003.

[22] H. M. Yin, Regularity of weak solution to Maxwell's equations and applications to microwave heating, J. Differential Equations **200**, 137-161, 2004.

ABOUT THE AUTHORS

J. M. Díaz Moreno, C. García Vázquez and F. Ortegón Gallego are professors in the Department of Mathematics,
Universidad de Cádiz,
11510 Puerto Real (SPAIN).

M. T. González Montesinos is professor in the Department of Applied Mathematics I,
Universidad de Sevilla,
41012 Sevilla (SPAIN).

G. Viglialoro is professor in the Department of Mathematics and Informatics,
Università di Cagliari,
Sardinia (ITALY).

`josemanuel.diaz@uca.es,`
`concepcion.garcia@uca.es,`
`francisco.ortegon@uca.es,`
`mategon@us.es,`
`giuseppe.viglialoro@unica.it.`