High-Temperature Superconductor in GetDP
Various Finite Element Formulations

Julien Dular

University of Liège, Liège, Belgium

April 22, 2021

Introduction

Objective:
Present and analyze various Finite Element (FE) formulations for modelling HTS and their implementation in GetDP .
We will follow the GetDP philosophy:

- we will focus on building the weak form,
- and exploit the flexible function space possibilities, specifically for global variables.

$\Rightarrow$ we will cover technical details.

Important remark:
One does not have to deal with these details for running GetDP on existing templates (e.g. using Onelab).

Details are however fundamental for investigating new models and/or understanding the code.

Outline

Outline

Outline

Magnetodynamics

- In the modelled domain, magnetodynamic (quasistatic) equations

$$\boxed{\mathrm{div}\, \boldsymbol{b} = 0, \qquad \mathbf{curl}\, \boldsymbol{h} = \boldsymbol{j}, \qquad \mathbf{curl}\, \boldsymbol{e} = -\partial_t \boldsymbol{b},}$$

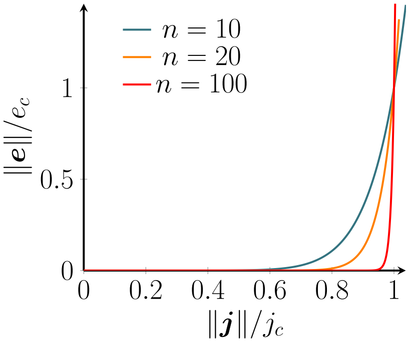
  with
  $\boldsymbol{b}$, the magnetic flux density (T),
  $\boldsymbol{h}$, the magnetic field (A/m),
  $\boldsymbol{j}$, the current density (A/m^2),
  $\boldsymbol{e}$, the electric field,
  (the displacement current $\partial_t \boldsymbol{d}$ is ignored).

- Need constitutive relationships relating $\boldsymbol{b}$ to $\boldsymbol{h}$ and $\boldsymbol{e}$ to $\boldsymbol{j}$.
- Need boundary conditions (BC).

Constitutive laws

1. High-temperature superconductors (HTS):

$$\boldsymbol{e} = \rho(\|\boldsymbol{j}\|)\boldsymbol{j} \quad \text{and} \quad \boldsymbol{b} = \mu_0 \boldsymbol{h},$$



where the electrical resistivity is given as

$$\rho(\|\boldsymbol{j}\|) = \frac{e_c}{j_c}\left(\frac{\|\boldsymbol{j}\|}{j_c}\right)^{n-1},$$

with $e_c = 10^{-4}$ V/m,
j_c, the critical current density,
n, the flux creep exponent,
$n \in [10, 1000]$.

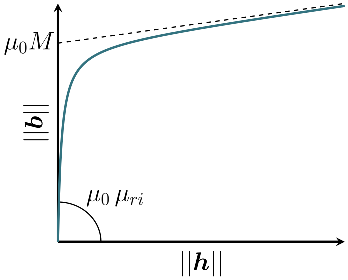
C.J.G. Plummer and J. E. Evetts, IEEE TAS **23** (1987) 1179.

E. Zeldov et al., Appl. Phys. Lett. **56** (1990) 680.

Constitutive laws

2. Ferromagnetic materials (FM):

$$\boldsymbol{b} = \mu(\boldsymbol{b})\, \boldsymbol{h} \quad \text{and} \quad \boldsymbol{j} = \boldsymbol{0}.$$



Typical values (supra50):

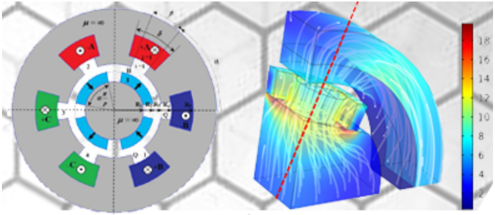
- initial relative permeability $\mu_{ri} = 1700$,
- saturation magnetization $\mu_0 M = 1.3$ T.

Eddy currents are neglected.

3. Air:

$$\boldsymbol{b} = \mu_0\, \boldsymbol{h} \quad \text{and} \quad \boldsymbol{j} = \boldsymbol{0}.$$

Constitutive laws, extensions



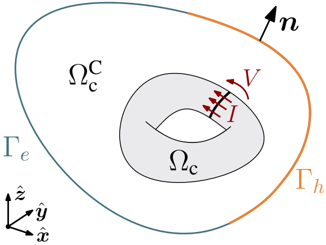
One can also consider

- normal conductors and coils,
- permanent magnets,
- ferromagnetic materials with hysteresis,
  Jacques, K. (2018). Doctoral dissertation, University of Liège.
- type-I superconductors (need a London length).

Boundary conditions and global variables

Domain Ω decomposed into:

- $\Omega_{\rm c}$, the conducting domain ($\Omega_{\rm c} = \cup_{i=1}^{N} \Omega_{{\rm c}_i}$),
- $\Omega_{\rm c}^{\rm C}$, the complementary non-conducting domain.



Boundary conditions are of two types:

1. Local conditions. On domain boundary $\partial\Omega = \Gamma$:
   - $\boldsymbol{h} \times \boldsymbol{n} = \bar{\boldsymbol{h}} \times \boldsymbol{n}$, imposed on Γ_h,
   - $\boldsymbol{e} \times \boldsymbol{n} = \bar{\boldsymbol{e}} \times \boldsymbol{n}$ (or $\boldsymbol{b} \cdot \boldsymbol{n} = \bar{\boldsymbol{b}} \cdot \boldsymbol{n}$), imposed on Γ_e $(= \Gamma \backslash \Gamma_h)$.
2. Global conditions. Either the applied current I_i, or voltage V_i is imposed (or a relation between them, not covered here) on each separate conducting region $\Omega_{{\rm c}_i}$,
   - $I_i = \bar{I}_i$, imposed for $i \in C_I$, a subset of $C = \{1, \dots, N\}$,
   - $V_i = \bar{V}_i$, imposed for $i \in C_V$, the complementary subset.

Summary

- Equations in Ω:

$$\boxed{\operatorname{div}\boldsymbol{b} = 0, \quad \mathbf{curl}\,\boldsymbol{h} = \boldsymbol{j}, \quad \mathbf{curl}\,\boldsymbol{e} = -\partial_t \boldsymbol{b}.}$$

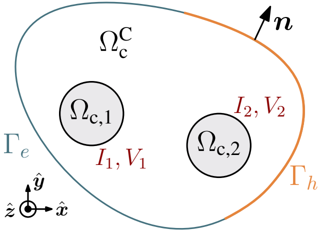
- Constitutive laws:

$$\boxed{\boldsymbol{e} = \rho \boldsymbol{j}, \quad \boldsymbol{b} = \mu \boldsymbol{h}.}$$

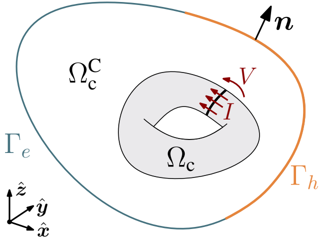
- Boundary conditions:

$$\boxed{\begin{aligned} (\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \boldsymbol{0}, \quad (\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \boldsymbol{0},\\ I_i = \bar{I}_i \text{ for } i \in C_I, \quad V_i = \bar{V}_i \text{ for } i \in C_V. \end{aligned}}$$

2D:



3D:



Finite element formulations

GetDP solves the problem with the finite element method.

Two classes of formulations:

- h-conform, e.g. h-formulation,
  - enforces the continuity of the tangential component of $\boldsymbol{h}$,
  - involves $\boldsymbol{e} = \rho \boldsymbol{j}$ and $\boldsymbol{b} = \mu \boldsymbol{h}$,
  - much used for HTS modelling.
- b-conform, e.g. a-formulation,
  - enforces the continuity of the normal component of $\boldsymbol{b}$,
  - involves $\boldsymbol{j} = \sigma \boldsymbol{e}$ and $\boldsymbol{h} = \nu \boldsymbol{b}$, ($\sigma = \rho^{-1}$, $\nu = \mu^{-1}$)
  - much used in electric rotating machine design.

Nonlinear constitutive laws involved in opposite ways $\Rightarrow$ very different numerical behaviors are expected… and observed.

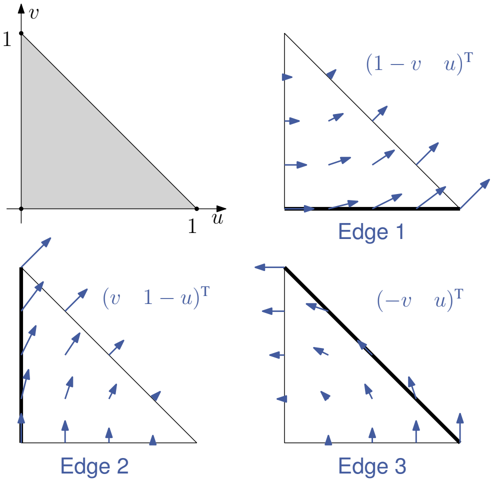
Differential forms

In GetDP , we discretize the fields as differential k-forms.
The exterior derivative d applied on a k-form gives a $k+1$-form.

- 0-form, (e.g. ϕ, v):
  - continuous scalar fields (conform),
  - generated by **nodal** functions ψ_n,
    value (point evaluation) at node $\tilde{n} = \delta_{n\tilde{n}}$,
  - exterior derivative is $\mathbf{grad}$.
- 1-form, e.g. $\boldsymbol{h}, \boldsymbol{e}, (\boldsymbol{a}, \boldsymbol{t})$:
  - vector fields with continuous tangential trace (curl-conform),
  - generated by **edge** functions $\boldsymbol{\psi}_e$,
    circulation (line integral) along edge $\tilde{e} = \delta_{e\tilde{e}}$,
  - exterior derivative is $\mathbf{curl}$.
- 2-form, e.g. $\boldsymbol{b}, \boldsymbol{j}$:
  - vector fields with continuous normal trace (div-conform),
  - generated by **facet** functions $\boldsymbol{\psi}_f$,
    flux (surface integral) through facet $\tilde{f} = \delta_{f\tilde{f}}$,
  - exterior derivative is div .

Differential forms - Illustration

Edge functions (1-form fields) for a linear triangular finite element:



Their **curl** (2-form fields) are constant.

Outline

Derivation of the a-formulation

Introduce the vector potential $\boldsymbol{a}$, and the electric potential v:

$$\boldsymbol{b} = \mathbf{curl}\, \boldsymbol{a}, \qquad \boldsymbol{e} = -\partial_t \boldsymbol{a} - \mathbf{grad}\, v.$$

Define $\boldsymbol{a}$ in Ω and v in Ω_{c} (discontinuous across electrodes):

- $\boldsymbol{a}$ as a 1-form and v as a 0-form,
- satisfying the local BC $(\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \mathbf{0}$,
- and global BC $V_i = \bar{V}_i$ for $i \in C_V$ (i.e. the circulation of $-\mathbf{grad}\, v$ around conducting domain Ω_{c_i} is equal to $\bar{V}_i$).

This strongly satisfies

$$\operatorname{div} \boldsymbol{b} = 0, \quad \mathbf{curl}\, \boldsymbol{e} = -\partial_t \boldsymbol{b}, \quad (\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \mathbf{0}, \quad V_i = \bar{V}_i \text{ for } i \in C_V.$$

What remains (and will be imposed weakly) is:

$$\mathbf{curl}\, \boldsymbol{h} = \boldsymbol{j}, \quad \boldsymbol{j} = \sigma \boldsymbol{e}, \quad \boldsymbol{h} = \nu \boldsymbol{b}, \quad (\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \mathbf{0}, \quad I_i = \bar{I}_i \text{ for } i \in C_I.$$

Choosing $\boldsymbol{a}$ and v

We still have freedom on the choice of $\boldsymbol{a}$ and v. Indeed, for any scalar field ϕ, the substitution

$$\boldsymbol{a} \rightarrow \boldsymbol{a} + \int_0^t \mathbf{grad}\, \phi \, dt$$
$$v \rightarrow v - \phi$$

lets the physical solution, $\boldsymbol{b}$ and $\boldsymbol{e}$, unchanged.

We present here one possibility for gauging $\boldsymbol{a}$ and v in:
(1) 2D case with in-plane $\boldsymbol{b}$, (2) 3D case.

In both cases, **one** global shape function $v_{d,i}$ in each Ω_{c_i} is sufficient for representing a unit voltage in Ω_{c_i}, s.t. we have:

$$\mathbf{grad}\, v = \sum_{i=1}^{N} V_i \, \mathbf{grad}\, v_{d,i}.$$

Choosing $\boldsymbol{a}$ and v, cont'd

$$\boldsymbol{b} = \mathbf{curl}\, \boldsymbol{a}, \qquad \boldsymbol{e} = -\partial_t \boldsymbol{a} - \mathbf{grad}\, v, \qquad \mathbf{grad}\, v = \sum_{i=1}^{N} V_i\, \mathbf{grad}\, v_{d,i}$$

1. 2D with in-plane $\boldsymbol{b}$:

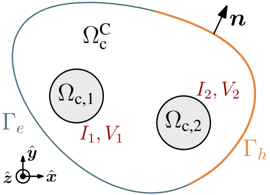
- We choose $\boldsymbol{a}$ along $\hat{z}$,

$$\boldsymbol{a} = \sum_{n \in \Omega} a_n\, \psi_n \hat{\boldsymbol{z}},$$

  with ψ_n the node function of node n.

  NB: It is a Coulomb gauge $\operatorname{div} \boldsymbol{a} = 0$.

- $\mathbf{grad}\, v_{d,i}$ is along $\hat{z}$ and constant ($= 1$) in each Ω_{c_i}. (V is a voltage per unit length.)
- Remaining constant fixed by BC.



GetDP $\boldsymbol{a}$ in 2D, with in-plane $\boldsymbol{b}$

$$\boldsymbol{a} = \sum_{n \in \Omega} a_n \, \psi_n \hat{\boldsymbol{z}},$$

```
FunctionSpace {
    // Perpendicular edge functions (1-form field in the out-of-plane direction).
    { Name a_space_2D; Type Form1P;
        BasisFunction {
            { Name psin; NameOfCoef an; Function BF_PerpendicularEdge;
                Support Omega_a_AndBnd; Entity NodesOf[All]; }
        }
        Constraint {
            { NameOfCoef an; EntityType NodesOf; NameOfConstraint a; }
        }
    }
}
```

grad v in 2D, with in-plane $\boldsymbol{b}$

$$\mathbf{grad}\ v = \sum_{i=1}^{N} V_i\, \mathbf{grad}\ v_{d,i} = \sum_{i=1}^{N} V_i\, \hat{z}_i$$

```
FunctionSpace {
    { Name grad_v_space_2D; Type Form1P;
        BasisFunction {
            // Constant per region and along z. Corresponds to a voltage per unit length.
            { Name zi; NameOfCoef Vi; Function BF_RegionZ;
                Support Region[OmegaC]; Entity Region[OmegaC]; }
        }
        GlobalQuantity {
            // Associated global quantities to be used in the formulation.
            { Name V; Type AliasOf; NameOfCoef Vi; }
            { Name I; Type AssociatedWith; NameOfCoef Vi; }
        }
        Constraint {
            { NameOfCoef V; EntityType Region; NameOfConstraint Voltage; }
            { NameOfCoef I; EntityType Region; NameOfConstraint Current; }
        }
    }
}
```

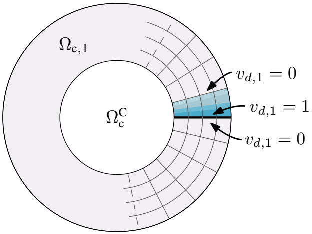
Choosing $\boldsymbol{a}$ and v

2. 3D:

- In Ω_c, define $v_{d,i}$ to be zero everywhere except on a transition layer in Ω_{c_i}: layer of one element, on one side of the electrodes, in each Ω_{c_i} (v has no longer a physical interpretation),

$$\mathbf{grad}\, v = \sum_{i=1}^{N} V_i \,\mathbf{grad}\, v_{d,i}.$$

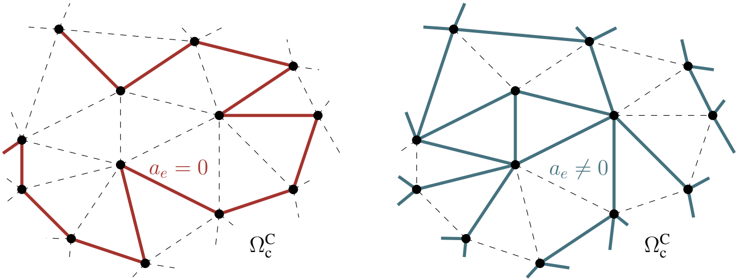
- $\boldsymbol{a}$ is generated by edge functions.
- In Ω_c, $\boldsymbol{a}$ is unique, e.g. outside the transition layer, $\boldsymbol{e} = -\partial_t \boldsymbol{a}$ (reduced vector potential).
- In Ω_c^C, $\boldsymbol{a}$ is made unique with a co-tree gauge…



Co-tree gauge for $\boldsymbol{a}$ in Ω_c^C in 3D

- In Ω_c^C, only $\mathbf{curl}\, \boldsymbol{a} = \boldsymbol{b}$ has a physical meaning. One DOF per facet is sufficient (and necessary), instead of one DOF per edge.
- The support entities of the 1-form $\boldsymbol{a}$ are the edges.
- To associate a unique edge to each facet: consider only edges in a **co-tree**, i.e. the complementary of a tree:

$$\boldsymbol{a} = \sum_{e \in \Omega_c \cup (\textbf{co-tree} \text{ in } \Omega_c^C)} a_e\, \boldsymbol{\psi}_e .$$



NB: Be careful on the conducting domain boundary $\partial\Omega_c$, no gauge there because $\boldsymbol{a}$ is already unique.

GetDP $\boldsymbol{a}$ in 3D

$$\boldsymbol{a} = \sum_{e \in \Omega_c \cup (\textbf{co-tree} \text{ in } \Omega_c^C)} a_e \, \psi_e$$

```
FunctionSpace {
    { Name a_space_3D; Type Form1;
        BasisFunction {
            // Usual edge functions everywhere (decomposed to handle BndOmegaC) correctly
            { Name psie ; NameOfCoef ae ; Function BF_Edge ;
                Support Omega_a_AndBnd ; Entity EdgesOf[ All , Not BndOmegaC ] ; }
            { Name psie2 ; NameOfCoef ae2 ; Function BF_Edge ;
                Support Omega_a_AndBnd ; Entity EdgesOf[ BndOmegaC ] ; }
        }
        Constraint {
            { NameOfCoef ae; EntityType EdgesOf; NameOfConstraint a; }
            { NameOfCoef ae2; EntityType EdgesOf; NameOfConstraint a; }
            { NameOfCoef ae; EntityType EdgesOfTreeIn; EntitySubType StartingOn;
                NameOfConstraint GaugeCondition; }
}}}
```

```
Constraint {
    { Name GaugeCondition ; Type Assign ;
        Case {
            // Zero on edges of a tree in Omega_CC, containing a complete tree on Surf_a_noGauge.
            {Region Omega_a_OmegaCC ; SubRegion Surf_a_noGauge; Value 0.; }
        }
    }
}
```

GetDP v in 3D

$$\textbf{grad}\, v = \sum_{i=1}^{N} V_i\, \textbf{grad}\, v_{d,i}$$

```
FunctionSpace{
    { Name grad_v_space_3D; Type Form1;
        BasisFunction {
            // Global unit voltage shape function. Support limited to only one side of the electrodes.
            { Name vi; NameOfCoef Vi; Function BF_GradGroupOfNodes;
                Support ElementsOf[OmegaC, OnPositiveSideOf Electrodes];
                Entity GroupsOfNodesOf[Electrodes]; }
        }
        GlobalQuantity {
            // Associated global quantities to be used in the formulation.
            { Name V; Type AliasOf; NameOfCoef Vi; }
            { Name I; Type AssociatedWith; NameOfCoef Vi; }
        }
        Constraint {
            { NameOfCoef V;
                EntityType GroupsOfNodesOf; NameOfConstraint Voltage; }
            { NameOfCoef I;
                EntityType GroupsOfNodesOf; NameOfConstraint Current; }
        }
    }
}
```

Choosing $\boldsymbol{a}$ and v, other possibilities

Many other possibilities can also be implemented in 3D.

Examples:

- Distributed support for v, via a preliminary FE resolution.

  [S. Schöps, et al. (2013) COMPEL: The international journal for computation and mathematics in electrical and electronic engineering, 2013.]

- Coulomb gauge in $\Omega_{\mathrm{c}}^{\mathrm{C}}$ via a Lagrange multiplier.

  Creusé, et al. (2019). Computers & Mathematics with Applications, 77(6), 1563-1582.

Derivation of the a-formulation, cont'd

What remains is:

$$\underbrace{\mathbf{curl}\,\boldsymbol{h} = \boldsymbol{j}, \quad \boldsymbol{j} = \sigma \boldsymbol{e}, \quad \overbrace{\boldsymbol{h} = \nu \boldsymbol{b}}_{}}_{\Rightarrow\ \mathbf{curl}\,(\nu\,\mathbf{curl}\,\boldsymbol{a}) = -\sigma\,(\partial_t \boldsymbol{a} + \mathbf{grad}\,v)\ \circledast}\overbrace{, \quad (\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \boldsymbol{0}}^{\diamond}, \quad \underbrace{I_i = \bar{I}_i \text{ for } i \in C_I}_{\updownarrow}$$

- Multiply $\circledast$ by a test function $\boldsymbol{a}'$, in the same space than $\boldsymbol{a}$ but with homogeneous BC, and integrate over Ω,

$$\left(\mathbf{curl}\,(\nu\,\mathbf{curl}\,\boldsymbol{a})\,,\boldsymbol{a}'\right)_{\Omega} + \left(\sigma\,(\partial_t \boldsymbol{a} + \mathbf{grad}\,v)\,,\boldsymbol{a}'\right)_{\Omega_c} = 0$$

$$\Rightarrow \quad \left(\nu\,\mathbf{curl}\,\boldsymbol{a}\,,\mathbf{curl}\,\boldsymbol{a}'\right)_{\Omega} - \langle \underbrace{\nu\,\mathbf{curl}\,\boldsymbol{a} \times \boldsymbol{n}}_{\text{natural BC } \diamond}\,,\ \boldsymbol{a}'\rangle_{\Gamma_h}$$

$$+ \left(\sigma\,\partial_t \boldsymbol{a}\,,\boldsymbol{a}'\right)_{\Omega_c} + \left(\sigma\,\mathbf{grad}\,v\,,\boldsymbol{a}'\right)_{\Omega_c} = 0$$

Derivation of the a-formulation, cont'd

What remains is:

$$\underbrace{\mathbf{curl}\,\boldsymbol{h} = \boldsymbol{j}, \quad \boldsymbol{j} = \sigma \boldsymbol{e}, \quad \overbrace{\boldsymbol{h} = \nu \boldsymbol{b},}}_{\Rightarrow\ \mathbf{curl}\,(\nu\,\mathbf{curl}\,\boldsymbol{a}) = -\sigma\,(\partial_t \boldsymbol{a} + \mathbf{grad}\,v)\ \circledast} \overbrace{\quad (\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \boldsymbol{0},}^{\diamond} \quad \underbrace{I_i = \bar{I}_i \text{ for } i \in C_I}_{\updownarrow}$$

- Multiply $\circledast$ by a test function $\mathbf{grad}\,v'$, and integrate over Ω_c,

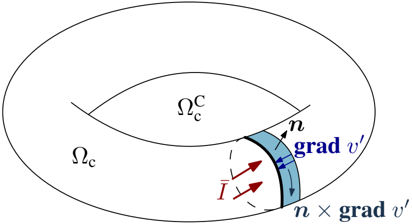
$$\left(\mathbf{curl}\,(\nu\,\mathbf{curl}\,\boldsymbol{a})\,,\mathbf{grad}\,v'\right)_{\Omega_c} + \left(\sigma\,\partial_t \boldsymbol{a}\,,\mathbf{grad}\,v'\right)_{\Omega_c} + \left(\sigma\,\mathbf{grad}\,v\,,\mathbf{grad}\,v'\right)_{\Omega_c} = 0$$

$$\Rightarrow \quad -\underbrace{\left\langle \nu\,\mathbf{curl}\,\boldsymbol{a} \times \boldsymbol{n}\,,\mathbf{grad}\,v' \right\rangle_{\partial\Omega_c}}_{\updownarrow \dots} + \left(\sigma\,\partial_t \boldsymbol{a}\,,\mathbf{grad}\,v'\right)_{\Omega_c} + \left(\sigma\,\mathbf{grad}\,v\,,\mathbf{grad}\,v'\right)_{\Omega_c} = 0$$

Derivation of the a-formulation, cont'd

- The surface term simplifies

$$
\begin{aligned}
\left\langle \nu\,\mathbf{curl}\,\boldsymbol{a} \times \boldsymbol{n} \,, \mathbf{grad}\; v' \right\rangle_{\partial\Omega_c} &= \left\langle \boldsymbol{h} \times \boldsymbol{n} \,, \mathbf{grad}\; v' \right\rangle_{\partial\Omega_c} \\
&= \left\langle \boldsymbol{h} \,, \boldsymbol{n} \times \mathbf{grad}\; v' \right\rangle_{\partial\Omega_c} \\
&= \left\langle \boldsymbol{h} \,, \boldsymbol{n} \times \mathbf{grad}\; v' \right\rangle_{\partial(\text{transition layer})} \\
&= I\; V' = \bar{I}\; V' \qquad (\text{Ampère's law} + \updownarrow).
\end{aligned}
$$



a-formulation

Finally, the a-formulation amounts to find $\boldsymbol{a}$ and v in the chosen function spaces such that, $\forall \boldsymbol{a}'$ and v',

$$\begin{aligned}\left(\nu\, \mathbf{curl}\, \boldsymbol{a}\ , \mathbf{curl}\, \boldsymbol{a}'\right)_{\Omega} - \left\langle \bar{\boldsymbol{h}} \times \boldsymbol{n}_{\Omega}\ , \boldsymbol{a}'\right\rangle_{\Gamma_h} \\ + \left(\sigma\, \partial_t \boldsymbol{a}\ , \boldsymbol{a}'\right)_{\Omega_c} + \left(\sigma\, \mathbf{grad}\ v\ , \boldsymbol{a}'\right)_{\Omega_c} = 0,\end{aligned}$$

$$\left(\sigma\, \partial_t \boldsymbol{a}\ , \mathbf{grad}\ v'\right)_{\Omega_c} + \left(\sigma\, \mathbf{grad}\ v\ , \mathbf{grad}\ v'\right)_{\Omega_c} = \sum_{i=1}^{N} I_i \mathcal{V}_i(v'),$$

with $I_i = \bar{I}_i$ for $i \in C_I$,
and $\mathcal{V}_i(v') = V_i'$ (i.e. the DOF associated with the unit voltage function $v_{d,i}$).

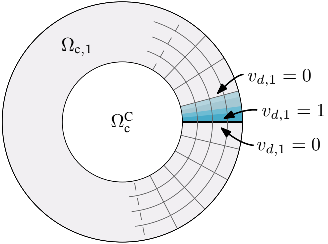
a-formulation - Interpretation

When the test function $v' = v_{d,i}$ is chosen ($\mathcal{V}_i(v_{d,i}) = 1$), the second equation reads

$$\left(\sigma\left(\partial_t \boldsymbol{a} + \textbf{grad}\, v\right), \textbf{grad}\, v_{d,i}\right)_{\Omega_c} = I_i$$

$$\Rightarrow \quad \left(\sigma\, \boldsymbol{e}, -\textbf{grad}\, v_{d,i}\right)_{\Omega_c} = I_i.$$

"Flux of $\sigma\boldsymbol{e}$ ($= \boldsymbol{j}$) averaged over a transition layer = total current".



NB: The flux of $\sigma\boldsymbol{e}$ depends on the chosen cross-section as $\sigma\boldsymbol{e}$ is not a 2-form (as $\boldsymbol{j}$ should be). Conservation of current is weakly satisfied.

Outline

Derivation of the h-formulation

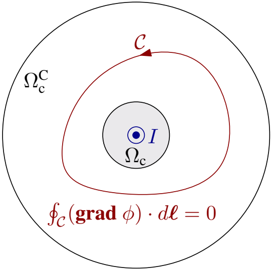
Choose $\boldsymbol{h}$ such that

- it is a 1-form,
- $(\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \boldsymbol{0}$,
- $\textbf{curl}\, \boldsymbol{h} = \boldsymbol{0}$ in Ω_c^C (this is the **key** point),
- and express $\boldsymbol{j}$ directly as $\boldsymbol{j} = \textbf{curl}\, \boldsymbol{h}$ in Ω_c, with $\boldsymbol{h}$ generated by edge functions.

What are the functions $\boldsymbol{h}$ that satisfy $\textbf{curl}\, \boldsymbol{h} = \boldsymbol{0}$ in Ω_c^C?

$\Rightarrow$ Surely gradients of scalar functions!

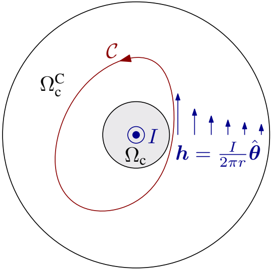
- If $\boldsymbol{h} = \textbf{grad}\, \phi$, then $\textbf{curl}\, \boldsymbol{h} = \boldsymbol{0}$, $\forall \phi$.
- However, choosing only $\boldsymbol{h} = \textbf{grad}\, \phi$ does not allow to represent a net current intensity (necessary if Ω_c^C is multiply connected).
- We need additional functions...



Derivation of the h-formulation, cont'd

- **One** global shape function $\boldsymbol{c}_i$ for each Ω_{c_i} is enough for representing a unit current intensity in Ω_{c_i}.
- As with the a-formulation, we have freedom on the choice of these functions. The only constraint is that

$$\oint_{\mathcal{C}_i} \boldsymbol{c}_j \cdot d\boldsymbol{\ell} = \delta_{ij}.$$



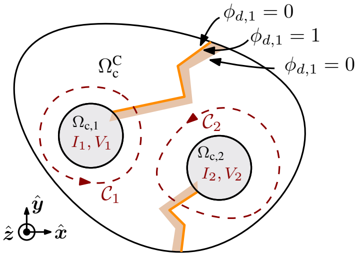
In Ω_c^C, we therefore have

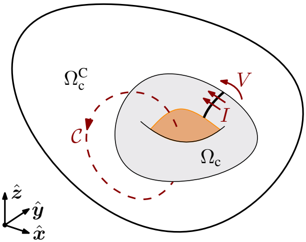
$$\boldsymbol{h} = \mathbf{grad}\,\phi + \sum_{i=1}^{N} I_i\,\boldsymbol{c}_i.$$

Choice of the global functions

One possibility for choosing the $\boldsymbol{c}_i$ functions, the cut functions:

- Introduce cuts to make Ω_c^C simply connected.
- Define the $\boldsymbol{c}_i$ on transition layers: layer of one element on one side of the cut, for each cut.
- $\boldsymbol{c}_i = \mathbf{grad}\, \phi_{d,i}$, with $\phi_{d,i}$ a discontinuous scalar potential.





NB: Gmsh has an automatic cohomology solver for generating cuts in complicated geometries (e.g. helix windings).

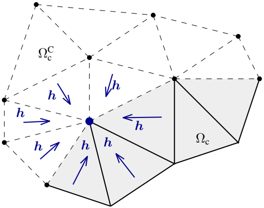
[M. Pellikka, et al. SIAM Journal on Scientific Computing 35(5), pp. 1195-1214, 2013.]

Summary and shape function supports

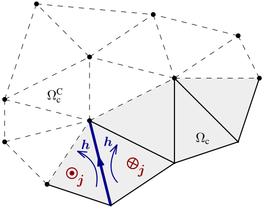
In Ω we have

$$\boldsymbol{h} = \sum_{n \in \Omega_\mathrm{c}^\mathrm{C}} \phi_n \, \mathbf{grad} \, \psi_n + \sum_{e \in \Omega_\mathrm{c} \setminus \partial\Omega_\mathrm{c}} h_e \, \boldsymbol{\psi}_e + \sum_{i=1}^{N} I_i \, \boldsymbol{c}_i.$$

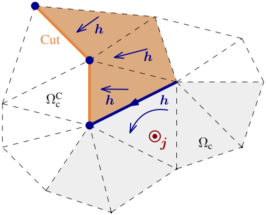
Gradient of node functions.



Classical edge functions.



Global cut function. Net current $\neq 0$.



Note: Gray areas $= \Omega_\mathrm{c}$.

GetDP $\boldsymbol{h}$ in 2D or 3D

$$\boldsymbol{h} = \sum_{n\in\Omega_c^C} \phi_n \,\mathbf{grad}\, \psi_n + \sum_{e\in\Omega_c\setminus\partial\Omega_c} h_e \,\boldsymbol{\psi}_e + \sum_{i=1}^{N} I_i \,\boldsymbol{c}_i.$$

```
FunctionSpace{
    { Name h_space; Type Form1;
        BasisFunction {
            // Nodal functions
            { Name gradpsin; NameOfCoef phin; Function BF_GradNode;
                Support Omega_h_OmegaCC_AndBnd; Entity NodesOf[OmegaCC]; }
            { Name gradpsin; NameOfCoef phin2; Function BF_GroupOfEdges;
                Support Omega_h_OmegaC; Entity GroupsOfEdgesOnNodesOf[BndOmegaC]; }
            // Edge functions
            { Name psie; NameOfCoef he; Function BF_Edge;
                Support Omega_h_OmegaC_AndBnd; Entity EdgesOf[All, Not BndOmegaC]; }
            // Cut functions
            { Name ci; NameOfCoef Ii; Function BF_GradGroupOfNodes;
                Support ElementsOf[Omega_h_OmegaCC, OnPositiveSideOf Cuts];
                Entity GroupsOfNodesOf[Cuts]; }
            { Name ci; NameOfCoef Ii2; Function BF_GroupOfEdges;
                Support Omega_h_OmegaC_AndBnd;
                Entity GroupsOfEdgesOf[Cuts, InSupport TransitionLayerAndBndOmegaC]; }
        }
        GlobalQuantity {
            { Name I ; Type AliasOf        ; NameOfCoef Ii ; }
            { Name V ; Type AssociatedWith ; NameOfCoef Ii ; }
        }
        Constraint {
            { [...] }
            { [...] }
}}}
```

Dealing with global variables, other possibilities

Many other possibilities can also be implemented.

Examples:

- Winding functions ($\Rightarrow$ see Erik Schnaubelt talk tomorrow),

  [S. Schöps, et al. (2013) COMPEL: The international journal for computation and mathematics in electrical and electronic engineering, 2013.]

- Large resistivity ($\approx 1\ \Omega$m) in Ω_c^C and integral constraint on the current (simple but much more DOF).

  [Shen, B., et al. (2020). IEEE access, 8, 100403-100414.]

Derivation of the h-formulation, cont'd

With the chosen $\boldsymbol{h}$, we strongly satisfy

$$\textbf{curl}\, \boldsymbol{h} = \boldsymbol{j}, \quad (\boldsymbol{h} - \bar{\boldsymbol{h}}) \times \boldsymbol{n}|_{\Gamma_h} = \boldsymbol{0}, \quad I_i = \bar{I}_i \text{ for } i \in C_I.$$

What remains (and will be imposed weakly) is:

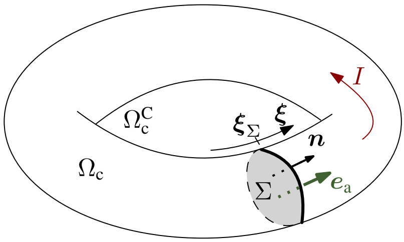
$$\text{div}\, \boldsymbol{b} = 0, \quad \textbf{curl}\, \boldsymbol{e} = -\partial_t \boldsymbol{b}, \quad \boldsymbol{e} = \rho \boldsymbol{j}, \quad \boldsymbol{b} = \mu \boldsymbol{h},$$
$$(\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \boldsymbol{0}, \quad V_i = \bar{V}_i \text{ for } i \in C_V.$$

We model an external applied voltage V by a localized $\boldsymbol{e}_\mathrm{a}$ field in a modified Ohm's law:

$$\boldsymbol{e} = \boldsymbol{e}_\mathrm{a} + \rho \boldsymbol{j},$$

with $\boldsymbol{e}_\mathrm{a} = V\delta(\boldsymbol{\xi} - \boldsymbol{\xi}_\Sigma)\boldsymbol{n}$ so that we globally have a net E.M.F.

($\delta(\cdot)$ is the Dirac distribution)



NB: Also see [Geuzaine, C. (2001). Phd thesis.]

Derivation of the h-formulation, cont'd

What remains is:

$$\overset{\Rightarrow\ \mathbf{curl}\ (\rho\,\mathbf{curl}\ \boldsymbol{h})+\mathbf{curl}\ \boldsymbol{e}_{\mathrm{a}}=-\partial_t(\mu\boldsymbol{h})\ \circledast}{}$$

$$\operatorname{div} \boldsymbol{b} = 0, \quad \overbrace{\mathbf{curl}\ \boldsymbol{e} = -\partial_t \boldsymbol{b}, \quad \boldsymbol{e} = \boldsymbol{e}_{\mathrm{a}} + \rho \boldsymbol{j}, \quad \boldsymbol{b} = \mu \boldsymbol{h}}^{\Rightarrow\ \mathbf{curl}\ (\rho\,\mathbf{curl}\ \boldsymbol{h})+\mathbf{curl}\ \boldsymbol{e}_{\mathrm{a}}=-\partial_t(\mu\boldsymbol{h})\ \circledast},$$

$$\underbrace{(\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \boldsymbol{0}}_{\diamond}, \quad \underbrace{V_i = \bar{V}_i \text{ for } i \in C_V}_{\ddagger}.$$

- Multiply $\circledast$ by a test function $\boldsymbol{h}'$, in the same space than $\boldsymbol{h}$ but with homogeneous BC, and integrate over Ω,

$$\big(\partial_t(\mu\boldsymbol{h})\ ,\boldsymbol{h}'\big)_\Omega + \big(\mathbf{curl}\ (\rho\,\mathbf{curl}\ \boldsymbol{h})\ ,\boldsymbol{h}'\big)_\Omega + \big(\mathbf{curl}\ \boldsymbol{e}_{\mathrm{a}}\ ,\boldsymbol{h}'\big)_\Omega = 0,$$

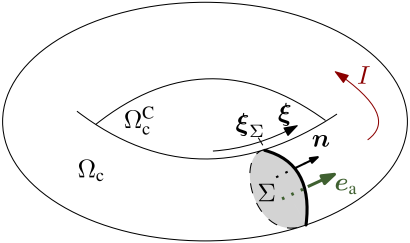
$$\Rightarrow \big(\partial_t(\mu\boldsymbol{h})\ ,\boldsymbol{h}'\big)_\Omega + \big(\rho\,\mathbf{curl}\ \boldsymbol{h}\ ,\mathbf{curl}\ \boldsymbol{h}'\big)_{\Omega_c} + \underbrace{\big(\boldsymbol{e}_{\mathrm{a}}\ ,\mathbf{curl}\ \boldsymbol{h}'\big)_{\Omega_c}}_{\ddagger\ldots}$$

$$- \langle \underbrace{(\boldsymbol{e}_{\mathrm{a}} + \rho\,\mathbf{curl}\ \boldsymbol{h}) \times \boldsymbol{n}}_{\text{natural BC } \diamond}\ ,\ \boldsymbol{h}'\rangle_{\Gamma_e} = 0$$

Derivation of the h-formulation, cont'd

- The third term simplifies

$$
\begin{aligned}
\left(\boldsymbol{e}_{\mathrm{a}}\,,\mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_{\mathrm{c}}} &= V\left(\delta(\boldsymbol{\xi}-\boldsymbol{\xi}_{\Sigma})\boldsymbol{n}\,,\mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_{\mathrm{c}}} \\
&= V\left\langle \boldsymbol{n}\,,\mathbf{curl}\,\boldsymbol{h}'\right\rangle_{\Sigma} \\
&= V\oint_{\partial\Sigma}\boldsymbol{h}'\cdot d\boldsymbol{\ell} \\
&= VI' = \bar{V}I' \qquad \text{(Ampère's law + ⓘ).}
\end{aligned}
$$



Derivation of the h-formulation, cont'd

What about $\operatorname{div} \boldsymbol{b} = 0$?

- Taking $\boldsymbol{h}' = \mathbf{grad}\, \phi'$ in the formulation yields

$$\begin{aligned}
&\left(\partial_t(\mu \boldsymbol{h}) \,, \mathbf{grad}\, \phi'\right)_\Omega + \left(\mathbf{curl}\, (\boldsymbol{e}_\mathrm{a} + \rho\, \mathbf{curl}\, \boldsymbol{h}) \,, \mathbf{grad}\, \phi'\right)_\Omega = 0, \\
\Rightarrow &- \left(\operatorname{div}\, (\partial_t(\mu \boldsymbol{h})) \,, \phi'\right)_\Omega + \left\langle \partial_t(\mu \boldsymbol{h}) \cdot \boldsymbol{n} \,, \phi' \right\rangle_{\Gamma_e} \\
&- \left\langle \bar{\boldsymbol{e}} \times \boldsymbol{n} \,, \mathbf{grad}\, \phi' \right\rangle_{\Gamma_e} = 0.
\end{aligned}$$

One can show that $\langle \partial_t(\mu \boldsymbol{h}) \cdot \boldsymbol{n} \,, \phi' \rangle_{\Gamma_e} = \langle \boldsymbol{e} \times \boldsymbol{n} \,, \mathbf{grad}\, \phi' \rangle_{\Gamma_e}$, so with $(\boldsymbol{e} - \bar{\boldsymbol{e}}) \times \boldsymbol{n}|_{\Gamma_e} = \mathbf{0}$, what remains is

$$\partial_t\Big(\left(\operatorname{div}\, (\mu \boldsymbol{h}) \,, \phi'\right)_\Omega \Big) = 0,$$

such that $\operatorname{div} \boldsymbol{b} = 0$ is (weakly) verified if the initial condition $\boldsymbol{h}_{t_0}$ is such that $(\operatorname{div}\, (\mu\, \boldsymbol{h}_{t_0}) \,, \phi')_\Omega = 0$.

h-formulation

Finally, the h-formulation amounts to find $\boldsymbol{h}$ in the chosen function space such that, $\forall \boldsymbol{h}'$,

$$\left(\partial_t(\mu \boldsymbol{h}) \,, \boldsymbol{h}'\right)_{\Omega} + \left(\rho \, \mathbf{curl}\, \boldsymbol{h} \,, \mathbf{curl}\, \boldsymbol{h}'\right)_{\Omega_c} - \left\langle \bar{\boldsymbol{e}} \times \boldsymbol{n} \,, \boldsymbol{h}' \right\rangle_{\Gamma_e} + \sum_{i=1}^{N} V_i \mathcal{I}_i(\boldsymbol{h}') = 0,$$

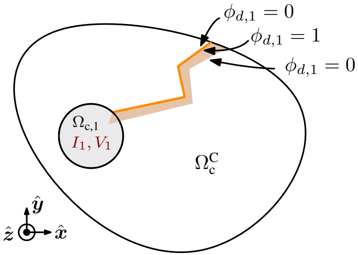
with $V_i = \bar{V}_i$ for $i \in C_V$,
and $\mathcal{I}_i(\boldsymbol{h}') = I_i'$ (i.e. the DOF associated with the cut function $\boldsymbol{c}_i$).

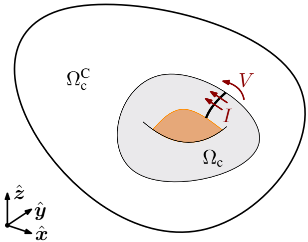
h-formulation - Interpretation

When the test function $\boldsymbol{c}_i$ ($\mathcal{I}_i(\boldsymbol{c}_i) = 1$) is chosen, we get the equation:

$$(\partial_t(\mu \boldsymbol{h}) \, , \boldsymbol{c}_i)_{\Omega} + (\rho \, \mathbf{curl} \, \boldsymbol{h} \, , \mathbf{curl} \, \boldsymbol{c}_i)_{\Omega_c} = -V_i.$$

"Flux change $\mu \boldsymbol{h}$ ($= \boldsymbol{b}$) $+$ circulation of $\rho \boldsymbol{j}$ ($= \boldsymbol{e}$), both averaged over a transition layer $=$ total voltage".





NB: The flux of $\mu \boldsymbol{h}$ depends on the chosen cut as $\mu \boldsymbol{h}$ is not a 2-form (as $\boldsymbol{b}$ should be). Same for $\rho \boldsymbol{j}$.

Outline

Structure of the resolution

- After spatial discretization, we get time-varying and non-linear matrix systems,

$$\boldsymbol{A}(\boldsymbol{x}, t) \cdot \boldsymbol{x} = \boldsymbol{b}(t),$$

where $\boldsymbol{x} = (\boldsymbol{a}, v)$ or $x = (\boldsymbol{h})$.

- Resolution: two imbricated loops.
  - Time-stepping: Implicit Euler with adaptative time steps;
  - Iterative solution of the non-linear system: Newton-Raphson or fixed point (Picard).

Outline

Implicit Euler

Time derivatives at time step t_n are explicitly expressed as:

$$\frac{du}{dt}(t_n) = \frac{u(t_n) - u(t_{n-1})}{\Delta t},$$

with $u(t_n)$ containing the DOF and $u(t_{n-1})$ being known.

Other possibilities can be implemented:

- Explicit Euler,
- Crank-Nicholson,
- Higher-order schemes...

$\Rightarrow$ Just explicitly write the scheme in the GetDP formulation.

GetDP Implicit Euler in the formulation

Syntax:

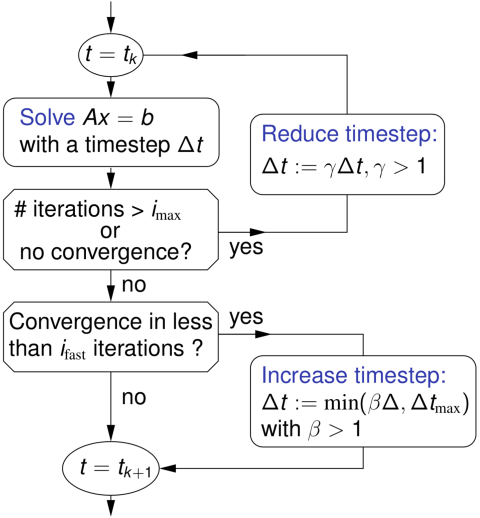
- `Dof{h}`: DOF at the current time step n (and iteration),
- `{h}[i]`: saved/known solution of $\boldsymbol{h}$ at time step $n-$ `i`,
- `{h}`: solution at the previous iteration (see later).

Example: flux variation term $(\partial_t(\mu\boldsymbol{h})\ ,\boldsymbol{h}')_\Omega$ in h-formulation

$$\left(\frac{\mu\boldsymbol{h}_n}{\Delta t}\ ,\boldsymbol{h}'\right)_\Omega - \left(\frac{\mu\boldsymbol{h}_{n-1}}{\Delta t}\ ,\boldsymbol{h}'\right)_\Omega$$

```
Formulation {
    { Name MagDyn_htot; Type FemEquation;
        Quantity {
            { Name h; Type Local; NameOfSpace h_space; }
            { [...] }
        }
        Equation {
            // Flux variation term (on the linear magnetic domain)
            Galerkin { [ mu[] * Dof{h} / $DTime , {h} ];
                In MagnLinDomain; Integration Int; Jacobian Vol;  }
            Galerkin { [ - mu[] * {h}[1] / $DTime , {h} ];
                In MagnLinDomain; Integration Int; Jacobian Vol;  }
            [...]
        }
    }
}
```

Adaptive time-stepping



Parameters:

- $\gamma = 1/2$
- $\beta = 2$
- $i_{\text{fast}} = i_{\max}/4$
- Fixed-point:
  $i_{\max} = 400$
- Newton-Raphson
  $i_{\max} = 50$

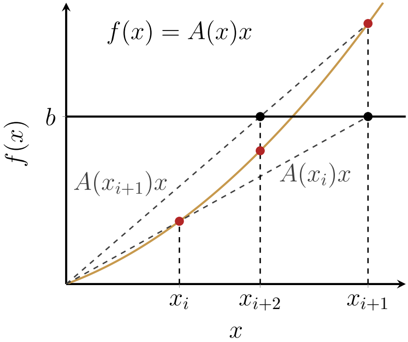
GetDP Adaptive time-stepping in resolution

```
Resolution {
  { Name MagDyn;
    System { {Name A; NameOfFormulation MagDyn_htot;} }
    Operation {
      [...]
      // Initialize
      SetTime[ timeStart ]; SetDTime[ dt ]; SetTimeStep[ 0 ];
      // Overall time loop
      While[$Time < timeFinalSimu && $DTime > 1e-10]{
        SetTime[ $Time + $DTime ]; SetTimeStep[ $TimeStep + 1 ];

        // Customized iterative loop
        Call CustomIterativeLoop;

        // If converged (= less than iter max and not diverged)...
        Test[ $iter < iter_max && ($res / $res0 <= 1e10)]{
          SaveSolution[A];
          Test[ $iter < iter_max / 2 && $DTime < dt_max]{
            Evaluate[ $dt_new = Min[$DTime * 2, dt_max] ];
            SetDTime[$dt_new];
          }
        }
        // ... otherwise, decrease the time step and start again
        {
          RemoveLastSolution[A];
          Evaluate[ $dt_new = $DTime / 2 ];
          SetDTime[$dt_new];
          SetTime[$Time - $DTime]; SetTimeStep[$TimeStep - 1];
        }
      }
    }
  }
}
```

Outline

Solving a non-linear equation: $f(x) = b$

1. Picard iteration method (a fixed point method):

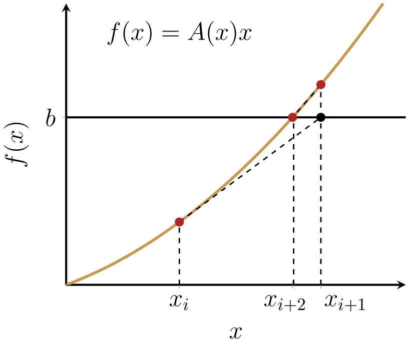


- Write $f(x)$ as $f(x) = A(x)x$.
- Get a first estimate x_0.
- At each iteration i:
  - solve $A(x_{i-1})x = b$,
  - $x_i := x$,
  - $i := i + 1$ and loop.
- Stop when convergence criterion is met.

- May converge for wide range of first estimates x_0.
- Convergence is slow!

Solving a non-linear equation: $f(x) = b$

2. Newton-Raphson iterative method:



- Get a first estimate x_0.
- At each iteration i, solve for x_i:

$$\frac{df}{dx}(x_{i-1})\,(x_i - x_{i-1}) = f(x_{i-1}).$$

- Stop when convergence criterion is met.

- Quadratic convergence, if the initial est. x_0 is close enough.
- Relaxation factors can also be implemented.
- If x is a vector, $\frac{df}{dx}$ is a matrix (Jacobian matrix)...

Jacobian for isotropic constitutive laws

- Consider a constitutive law of the form

$$\boldsymbol{a}(\boldsymbol{x}) \quad = \quad g(\|\boldsymbol{x}\|)\,\boldsymbol{x}.$$

Example: $\boldsymbol{e} = \rho\boldsymbol{j}$, or $\boldsymbol{b} = \mu\,\boldsymbol{h}$, ...

- The Newton-Raphson expansion can be cast in the form

$$\boldsymbol{a}(\boldsymbol{x}^i) \approx \boldsymbol{a}(\boldsymbol{x}^{i-1}) + \boldsymbol{J}(\boldsymbol{x}^{i-1}) \cdot \left(\boldsymbol{x}^i - \boldsymbol{x}^{i-1}\right),$$

where $\boldsymbol{J}$ is the 3×3 Jacobian matrix (i is the iteration index):

$$(\boldsymbol{J}(\boldsymbol{x}))_{jk} = \frac{\partial a_j}{\partial x_k} = \delta_{jk}\, g(\|\boldsymbol{x}\|) + x_j x_k \, \frac{\frac{dg(\|\boldsymbol{x}\|)}{d\|\boldsymbol{x}\|}}{\|\mathbf{x}\|}.$$

Examples in: Dular, J., et al. (2020) TAS 30 8200113.

- Example: $(\rho\,\mathbf{curl}\,\boldsymbol{h}\ , \mathbf{curl}\,\boldsymbol{h}')_{\Omega_c}$ in h-formulation, with $\mathbf{curl}\,\boldsymbol{h} = \boldsymbol{j}$:

$$\left(\rho(\boldsymbol{j}^{i-1})\boldsymbol{j}^{i-1}\ , \mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c} + \left(\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{j}}(\boldsymbol{j}^{i-1})\,\boldsymbol{j}^i\ , \mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c} - \left(\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{j}}(\boldsymbol{j}^{i-1})\,\boldsymbol{j}^{i-1}\ , \mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c}$$

GetDP Picard and Newton-Raphson in formulation

Example: nonlinear term $(\rho\,\mathbf{curl}\,\boldsymbol{h}\ ,\mathbf{curl}\,\boldsymbol{h}')_{\Omega_c}$ in *h*-formulation

N-R: $$\left(\rho(\boldsymbol{j}^{i-1})\,\boldsymbol{j}^{i-1}\ ,\mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c} + \left(\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{j}}(\boldsymbol{j}^{i-1})\,\boldsymbol{j}^{i}\ ,\mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c} - \left(\frac{\partial \boldsymbol{e}}{\partial \boldsymbol{j}}(\boldsymbol{j}^{i-1})\,\boldsymbol{j}^{i-1}\ ,\mathbf{curl}\,\boldsymbol{h}'\right)_{\Omega_c}$$

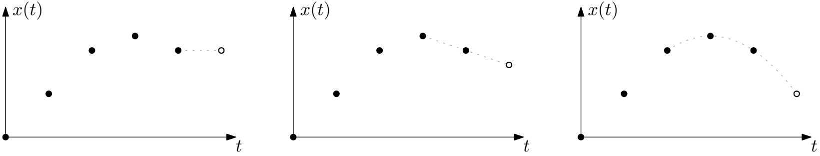
```
Formulation {
    { Name MagDyn_htot; Type FemEquation;
        Quantity {
            { Name h; Type Local; NameOfSpace h_space; }
            { [...] }
        }
        Equation {
            // (1) Picard
            Galerkin { [ rho[{d h}]] * Dof{d h} , {d h} ];
                In NonLinOmegaC; Integration Int; Jacobian Vol;  }

            // (2) Newton-Raphson
            Galerkin { [ rho[{d h}] * {d h} , {d h} ];
                In NonLinOmegaC; Integration Int; Jacobian Vol;  }
            Galerkin { [ dedj[{d h}] * Dof{d h} , {d h} ];
                In NonLinOmegaC; Integration Int; Jacobian Vol;  }
            Galerkin { [ - dedj[{d h}] * {d h} , {d h} ];
                In NonLinOmegaC ; Integration Int; Jacobian Vol;  }
            [...]
}}}
```

Syntax:

- `{h}`: solution of the previous iteration,
- `{d h}`: exterior derivative of $\boldsymbol{h}$. Here for $\boldsymbol{h}$ it is its **curl** .

First estimate

- We propose a series of possibilities:



(a) Zeroth-order extrapolation (b) First-order extrapolation (c) Second-order extrapolation

In Resolution: `SetExtrapolationOrder[ n ];` ($n \in \mathbb{N}$).

- It can strongly affect the required number of iterations!

Convergence criterion

- The residual $\boldsymbol{b} - \boldsymbol{A}(\boldsymbol{x}_i)\boldsymbol{x}_i$ is sometimes misleading.
- We choose the electromagnetic power, P, as a (global) convergence indicator:

  h-formulation

  $$P = (\partial_t(\mu\, \boldsymbol{h})\ , \boldsymbol{h})_{\Omega} + (\rho\, \mathbf{curl}\, \boldsymbol{h}\ , \mathbf{curl}\, \boldsymbol{h})_{\Omega_{\mathrm{c}}}\ .$$

  a-formulation

  $$P = (\partial_t(\mathbf{curl}\, \boldsymbol{a})\ , \nu\, \mathbf{curl}\, \boldsymbol{a})_{\Omega} + (\sigma \boldsymbol{e}\ , \boldsymbol{e})_{\Omega_{\mathrm{c}}}\ ,$$

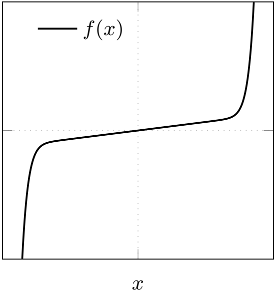
  with $\boldsymbol{e} = -\partial_t \boldsymbol{a} - \mathbf{grad}\ v$.
- We stop when $|\Delta P/P|$ is small enough:
  - $\approx 10^{-8}$ with Newton-Raphson,
  - $\approx 10^{-4}$ with Picard.

Outline

Nonlinearity in HTS for dual formulations

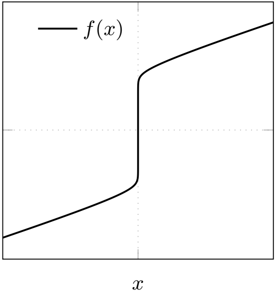
h-formulation $\boldsymbol{e} = \rho \boldsymbol{j}$

$$f(x) = |x|^{n-1}x + x$$



a-formulation $\boldsymbol{j} = \sigma \boldsymbol{e}$

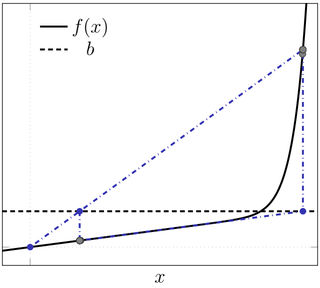
$$f(x) = |x|^{1/n-1}x + x$$



Different nonlinearities $\Rightarrow$ different numerical behaviors.

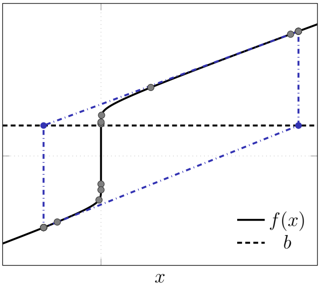
Beware of cycles

Cycles can occur in each method, depending on the shape of the function $f(x)$:



Picard iteration on
h-formulation

Prefer Newton-Raphson!



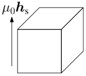
Newton-Raphson iteration on
a-formulation

Prefer Picard!

Relaxation factors can help, but no efficient solution (that we know of).

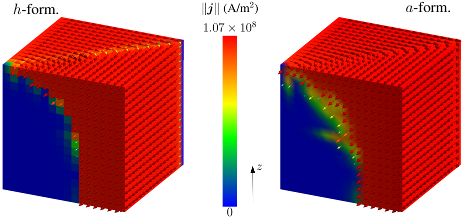
Illustration for a superconducting cube

System



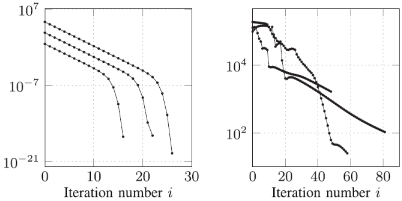
Side $a = 10$ mm.
$\mu_0 \boldsymbol{h}_s = \hat{z}\, B_0\, \sin(2\pi f t)$,
with $B_0 = 200$ mT,
$f = 50$ Hz,
$j_c = 10^8$ A/m^2 and
$n = 100$.

Current density distribution



Residual

- L_2 norm of $\boldsymbol{r} = \boldsymbol{A}\boldsymbol{x} - \boldsymbol{b}$
- Left: h-formulation
- Right: a-formulation



(a) Newton-Raphson technique. (b) Picard technique.

$\Rightarrow$ Much more efficient with Newton-Raphson (as is expected!).

Conclusion for HTS

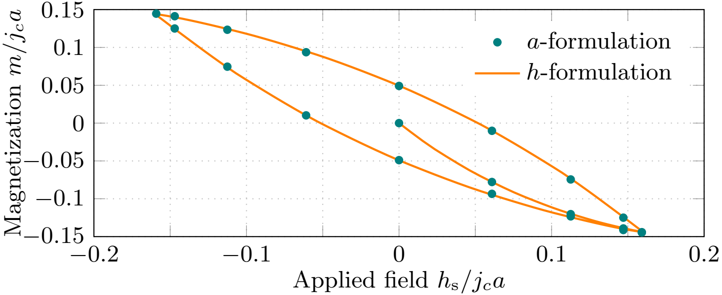
The diverging slope associated with $\boldsymbol{j} = \sigma \boldsymbol{e}$ for $\boldsymbol{j} \to 0$ is really difficult to handle.

$\Rightarrow$ Among the two simple formulations, the h-formulation is much more efficient for systems with HTS:

- with an adaptive time-stepping algorithm,
- solved with a Newton-Raphson method,
- with a first estimate obtained by 1^{st}-order extrapolation.

One particular case: "single time step"

- For large values of n, nearly a critical state model.
- Robustness of Picard on the $\boldsymbol{j} = \sigma \boldsymbol{e}$ law can help to reduce the number of time steps.



- Here, for a magnetization cycle (3D cube problem)
  - lines: h-formulation with 300 time steps,
  - dots: a-formulation with 20 time steps $\Rightarrow$ much faster!
- In practice, accurate for $\boldsymbol{j}$ and $\boldsymbol{b}$, but $\boldsymbol{e}$ is underestimated!

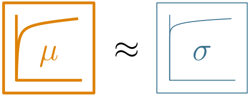
Outline

Outline

Ferromagnetic materials

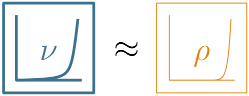
The nonlinearity is in the magnetic constitutive law.

- h-formulation the involved law is $\boldsymbol{b} = \mu \boldsymbol{h}$.



$\Rightarrow$ Easily enters cycles with Newton-Raphson.
OK with Picard, or N-R with relaxation factors but slow.

- a-formulation the involved law is $\boldsymbol{h} = \nu \boldsymbol{b}$.



$\Rightarrow$ Efficiently solved with Newton-Raphson.

The a-formulation is more appropriate for dealing with the nonlinearity, whereas for HTS, the dual formulation was best.

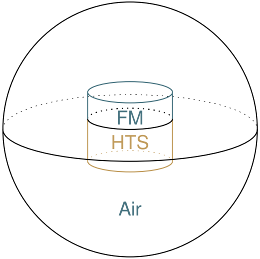
Coupled materials - h-a-formulation

Use the best formulation in each material

Decompose the domain Ω, for example into:

- $\Omega^h = \{\text{HTS}\}$
- $\Omega^a = \{\text{Ferromagnet, Air}\}$

and couple via $\Gamma_\text{m} = \partial(\text{HTS})$:



$$\left(\partial_t(\mu \boldsymbol{h})\,,\boldsymbol{h}'\right)_{\Omega^h} + \left(\rho\,\textbf{curl}\,\boldsymbol{h}\,,\textbf{curl}\,\boldsymbol{h}'\right)_{\Omega^h_c} + \left\langle \partial_t \boldsymbol{a} \times \boldsymbol{n}_{\Omega^h}\,,\boldsymbol{h}'\right\rangle_{\Gamma_\text{m}} = 0,$$

$$\left(\nu\,\textbf{curl}\,\boldsymbol{a}\,,\textbf{curl}\,\boldsymbol{a}'\right)_{\Omega^a} - \left\langle \boldsymbol{h} \times \boldsymbol{n}_{\Omega^a}\,,\boldsymbol{a}'\right\rangle_{\Gamma_\text{m}} = 0.$$

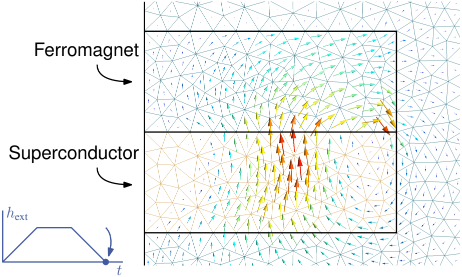
(For homogeneous natural BC)

$\Rightarrow$ see Erik Schnaubelt talk tomorrow

Results

Example:

- Stacked cylinders
- 2D axisymmetric
- External applied field



Number of iterations for three discretization levels:

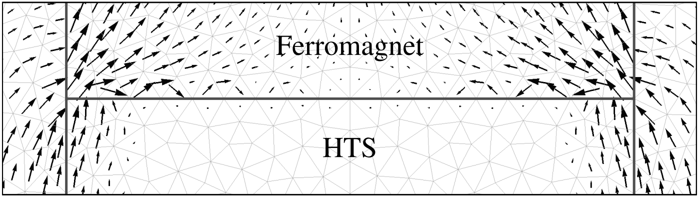
	h-formulation	a-formulation	h-a-formulation
Coarse	1878	4381	**1071**
Medium	3366	7539	**1931**
Fine	4422	14594	**3753**

In general, a speed-up from 1.2 to 3 is obtained.

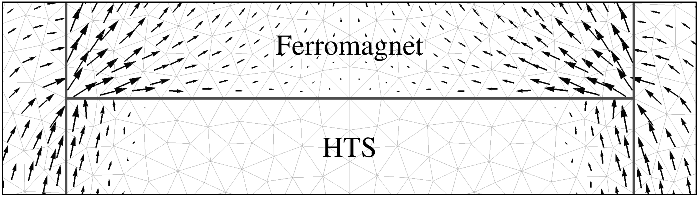
h-a-formulation Stability

The formulation is mixed (two unknown fields on Γ_{m})
$\Rightarrow$ Shape functions must satisfy an inf-sup condition.

- First-order functions for $\boldsymbol{h}$ and $\boldsymbol{a}$ (inf-sup KO):



- Second-order for $\boldsymbol{a}$, first-order for $\boldsymbol{h}$ (inf-sup OK):



Hierarchical functions

Example for 2nd-order shape functions for $\boldsymbol{a}$ (in 2D) on Γ_{m}:

```
FunctionSpace{
    { Name a_space_2D; Type Form1P;
        BasisFunction {
            // Usual first-order functions
            { Name psin; NameOfCoef an; Function BF_PerpendicularEdge;
                Support Omega_a_AndBnd; Entity NodesOf[All]; }
            // Second-order functions on BndOmega_ha only
            { Name psin2; NameOfCoef an2; Function BF_PerpendicularEdge_2E;
                Support Omega_a_AndBnd; Entity EdgesOf[BndOmega_ha]; }
        }
        Constraint {
            { NameOfCoef an; EntityType NodesOf; NameOfConstraint a; }
            { NameOfCoef an2; EntityType EdgesOf; NameOfConstraint a2; }
        }
    }
}
```

NB: This is for a locally enriched function space. Using 2nd-order elements on the whole domain can be done directly at the meshing step.

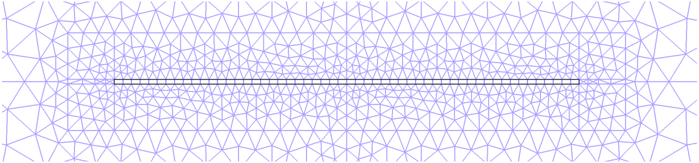
Command for 2D: `gmsh geometry.msh -2 -order 2`.

Outline

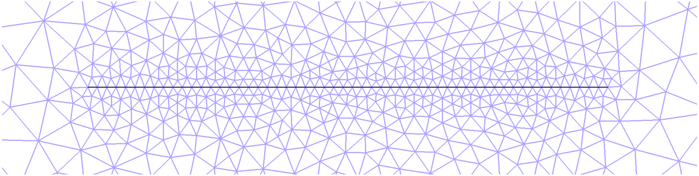
HTS tapes - t-a-formulation

To model thin superconducting tapes, two main possibilities:

1. Use the true geometry and the h-formulation with one-element across the thickness (quadrangle).



2. Perform the slab approximation and model the tape as a line ⇒ t-a-formulation.

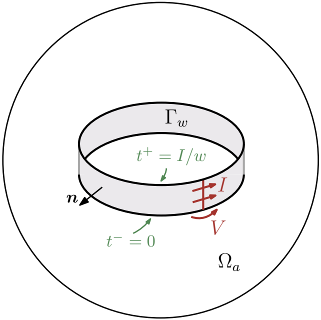


t-a-formulation

Consider a tape Γ_w of thickness w.

The current density is described by a current potential $\boldsymbol{t}$:

- such that $\boldsymbol{j} = \mathbf{curl}\, \boldsymbol{t}$,
- gauged by being defined along the normal of the tape, $\boldsymbol{t} = t\boldsymbol{n}$,
- with BC related to the total current I ($t^+ - t^- = I/w$).



In Ω_a, write the a-formulation and express the surface integral $\langle \boldsymbol{h} \times \boldsymbol{n}\ , \boldsymbol{a}' \rangle_{\Gamma_w}$ in terms of the surface current density $w\, \mathbf{curl}\, \boldsymbol{t}$.

t-a-formulation

Find $\boldsymbol{a}$ and $\boldsymbol{t}$ in the chosen function spaces such that, $\forall \boldsymbol{a}', \boldsymbol{t}'$,

$$\left(\nu\, \mathbf{curl}\, \boldsymbol{a}\ , \mathbf{curl}\, \boldsymbol{a}'\right)_{\Omega_a} - \left\langle \bar{\boldsymbol{h}} \times \boldsymbol{n}_\Omega\ , \boldsymbol{a}'\right\rangle_{\Gamma_h} - \left\langle w\, \mathbf{curl}\, \boldsymbol{t}\ , \boldsymbol{a}'\right\rangle_{\Gamma_w} = 0,$$

$$\left\langle w\, \partial_t \boldsymbol{a}\ , \mathbf{curl}\, \boldsymbol{t}'\right\rangle_{\Gamma_w} + \left\langle w\, \rho\, \mathbf{curl}\, \boldsymbol{t}\ , \mathbf{curl}\, \boldsymbol{t}'\right\rangle_{\Gamma_w} = -\sum_{i \in C} V_i \mathcal{I}_i(\boldsymbol{t}'),$$

with $V_i = \bar{V}_i$ for $i \in C_V$,
and $\mathcal{I}_i(\boldsymbol{t}') = I_i'$ (i.e. the DOF associated with the BC $w(t^+ - t^-)$).

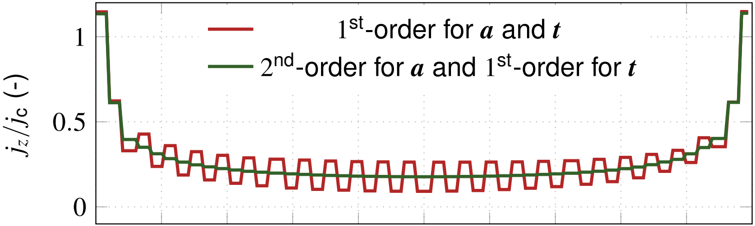
It is basically an h-a-formulation with a slab approximation.
See: [Bortot, L., et al. (2020). IEEE Trans. on App. Supercond., 30(5), 1-11].

t-a-formulation - Stability

The t-a-formulation is mixed (two unknown fields on Γ_w)
$\Rightarrow$ Shape functions must satisfy an inf-sup condition.

Similar conclusions than with the h-a-formulation.

Example for a 2D case, current density along the tape:



GetDP

Function space for $\boldsymbol{t}$

Defined as a scalar quantity in the `FunctionSpace`, the normal $\boldsymbol{n}$ is introduced in the formulation:

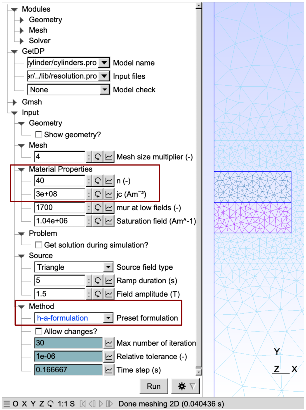
$$t = \sum_{n \in \Gamma_w \setminus \partial\Gamma_w} t_n \psi_n + \sum_{i=1}^{N} T_i \ell_i, \quad \text{with} \quad \boldsymbol{t} = t\boldsymbol{n}.$$

```
FunctionSpace{
    { Name t_space; Type Form0;
        BasisFunction {
            // Node functions except on the lateral edges of the tapes.
            { Name psin; NameOfCoef tn; Function BF_Node;
                Support Gamma_w; Entity NodesOf[All, Not LateralEdges]; }
            // Global shape function for representing a net current intensity.
            { Name elli; NameOfCoef Ti; Function BF_GroupOfNodes;
                Support Gamma_w_AndBnd; Entity GroupsOfNodesOf[PositiveEdges]; }
        }
        GlobalQuantity {
            // Global quantities to be used in the formulation.
            { Name T ; Type AliasOf        ; NameOfCoef Ti ; }
            { Name V ; Type AssociatedWith ; NameOfCoef Ti ; }
        }
        Constraint {
            { NameOfCoef V; EntityType GroupsOfNodesOf; NameOfConstraint Voltage; }
            { NameOfCoef T; EntityType GroupsOfNodesOf; NameOfConstraint Current_w; }
        }
    }
}
```

Final remark - Interface with Onelab

One can use existing GetDP templates and models without diving into the technical details.
In particular, we can use the Onelab interface. Example:

```
Function{
    // Choose the formulation
    DefineConstant[preset = {4, Highlight "Blue",
      Choices{
        1="h-formulation",
        3="a-formulation (small steps)",
        4="h-a-formulation"},
      Name "Input/5Method/0Preset formulation" }];
    // Superconductor parameters
    DefineConstant [ec = 1e-4];
    DefineConstant [jc = {3e8,
      Name "Input/3Material Properties/2jc (A/m2)"}];
    DefineConstant [n = {40,
      Name "Input/3Material Properties/1n (-)"}];
}
```



NB: Interface via Python scripts is also possible.

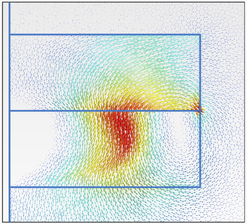
Conclusion

We presented four formulations in GetDP :

a-formulation | h-formulation | h-a-formulation | t-a-formulation

and discussed their relevance for HTS modelling.

Full examples are available on Life-HTS and Onelab:
`www.life-hts.uliege.be` and `onelab.info`



Cylinders model after magnetization.

Life-HTS

Liège university Finite Element models for High-Temperature Superconductors

This project contains model files for modeling systems containing high-temperature superconductors (HTS) with GetDP as a finite element solver and Gmsh as mesh generator.

Files are available here.

Several finite element formulations are implemented together with various linearization methods and iterative procedures. Simple models are proposed for practical applications (bulk and tapes HTS, coupling with ferromagnets...)

These models are developed at the University of Liège.

Thank you for your attention!

Main references

- Onelab website, with codes, examples, and tutorials: `onelab.info`
- Life-HTS website: `http://www.life-hts.uliege.be/`
- *Finite Element Formulations for Systems with High-Temperature Superconductors*,
  J. Dular, C. Geuzaine, and B. Vanderheyden, TAS 30 (2020) 8200113.
- *Stability of H-A and T-A Coupled Formulations*,
  J. Dular, M. Harutyunyan, L. Bortot, Sebastian Schöps, B. Vanderheyden, and C. Geuzaine (to be published).
- *Modélisation du champ magnétique et des courants induits dans des systèmes tridimensionnels non linéaires*,
  P. Dular, thesis (1996) U. Liège.
- *High order hybrid finite element schemes for Maxwell's equations taking thin structures and global quantities into account*,
  C. Geuzaine, thesis (2001) U. Liège.
- *The FEM method for electromagnetic modeling*,
  G. Meunier ed., Wiley, 2008.