MULTISIMO LAB

Technical University of Cartagena

Multi-material topology optimization for shape-morphing ultra-soft magneto-active structures

*C. Perez-Garcia, R. Ortigosa, **J. Martínez-Frutos**, and D. Garcia-Gonzalez*

Jesus.martinez@upct.es

May 2026

17th World Congress on Computational Mechanics. Munich,Germany.

Low σ

Stress Field

OUTLINE

Contents

01 Residual Mechanical Anisotropy in ultra-soft hMREs

02 NN constitutive modelling and staggered scheme

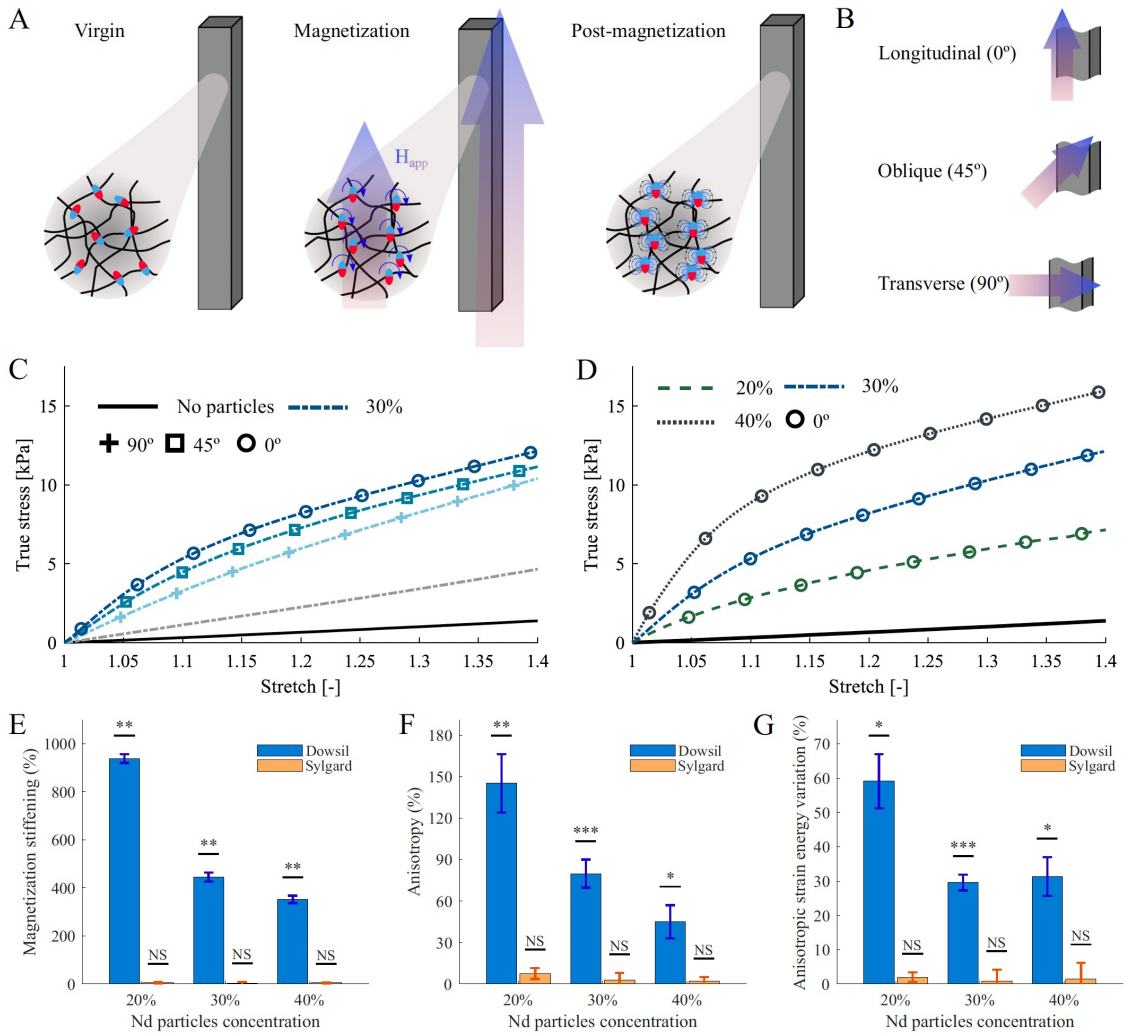
03 Multimaterial and Topology optimization algorithm

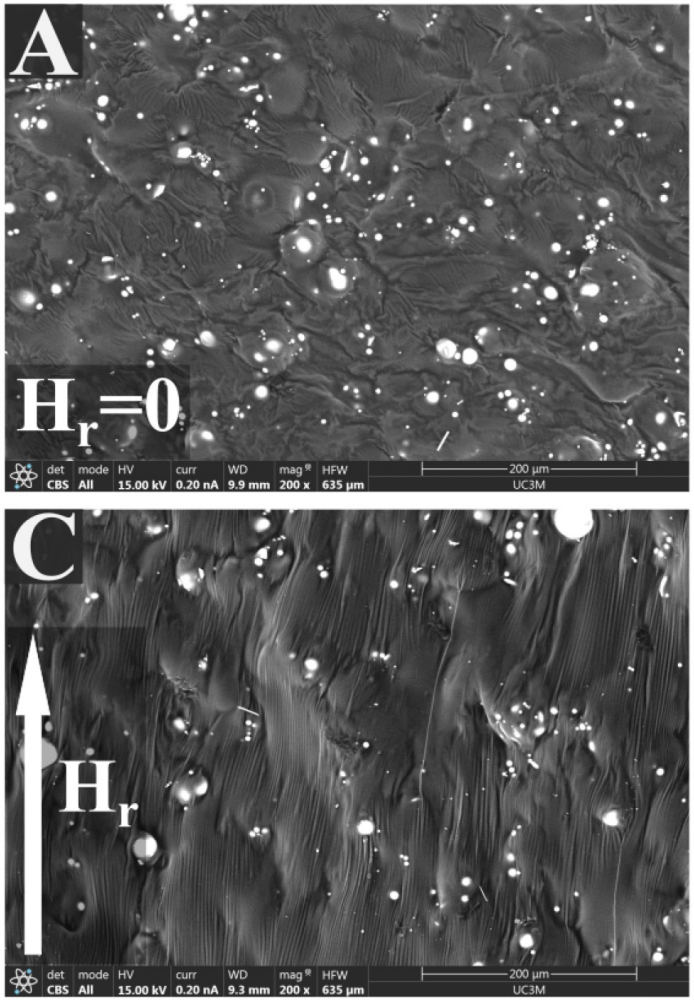
03 Julia implementation - HyperFEM

04 Acknowledgements

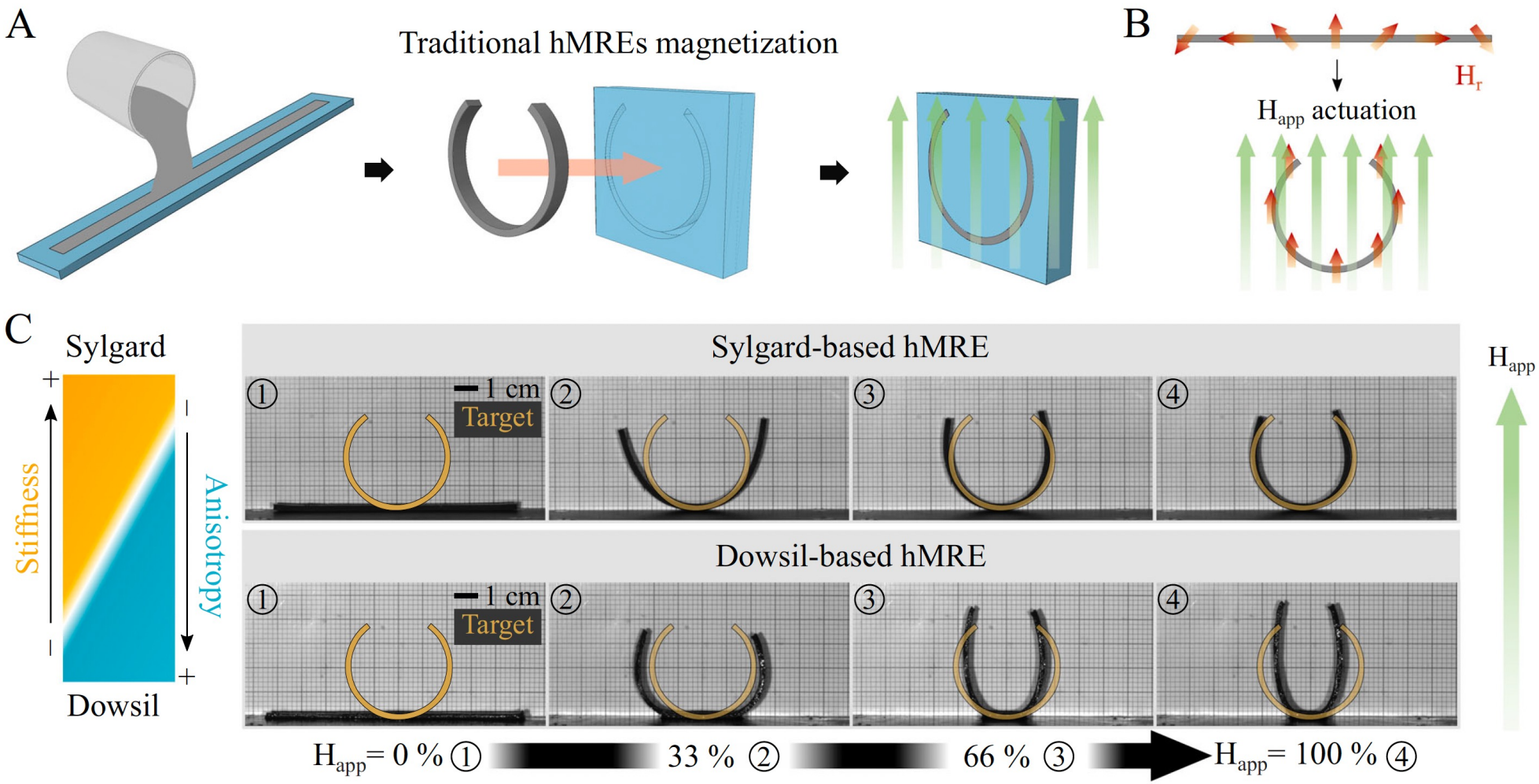
17th World Congress on Computational Mechanics. Munich,Germany.

Residual Mechanical Anisotropy in ultra-soft hMREs





Shape programming of ultra-soft hMRES

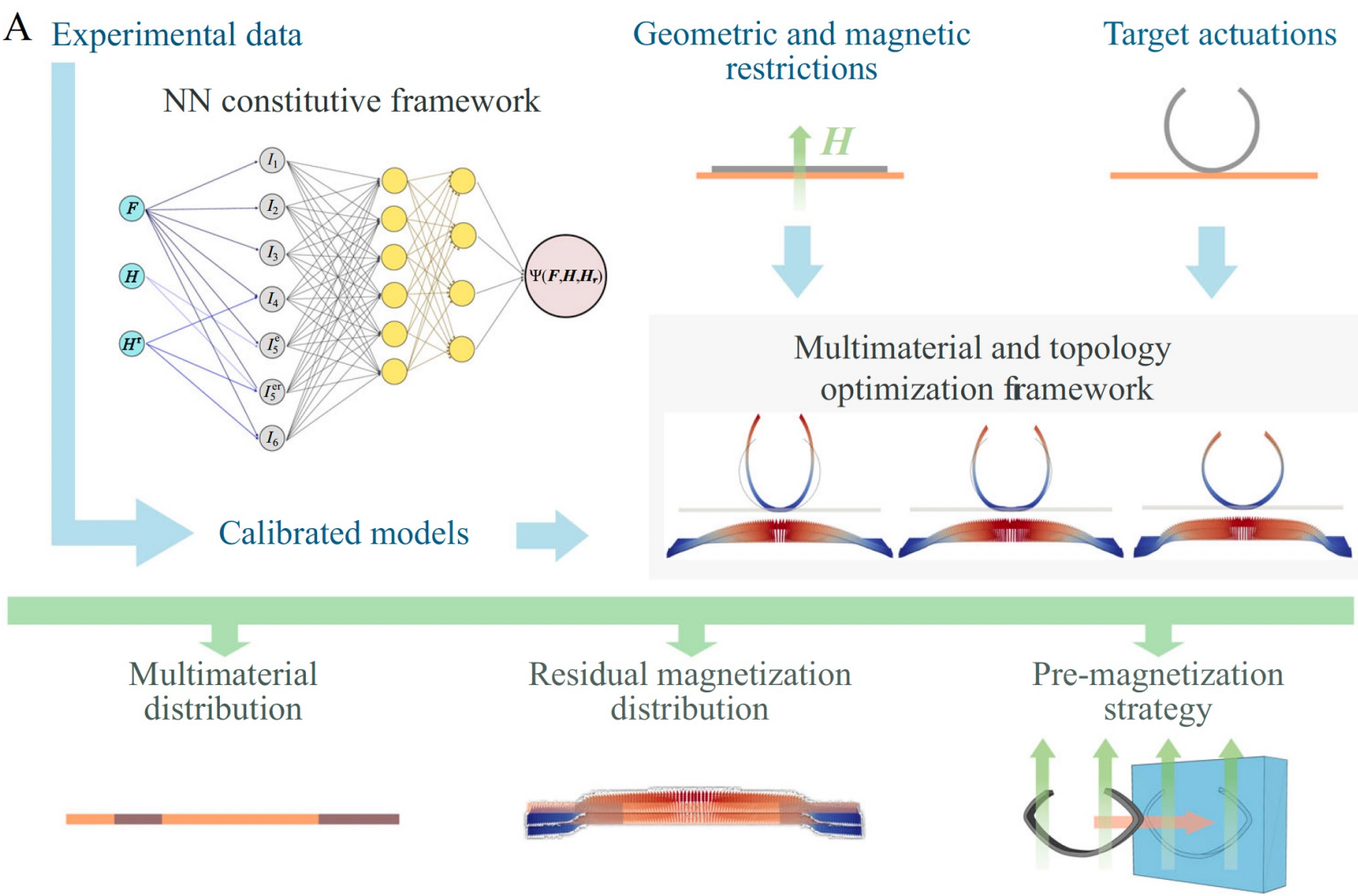


Sylgard 184 (≈2 MPa)

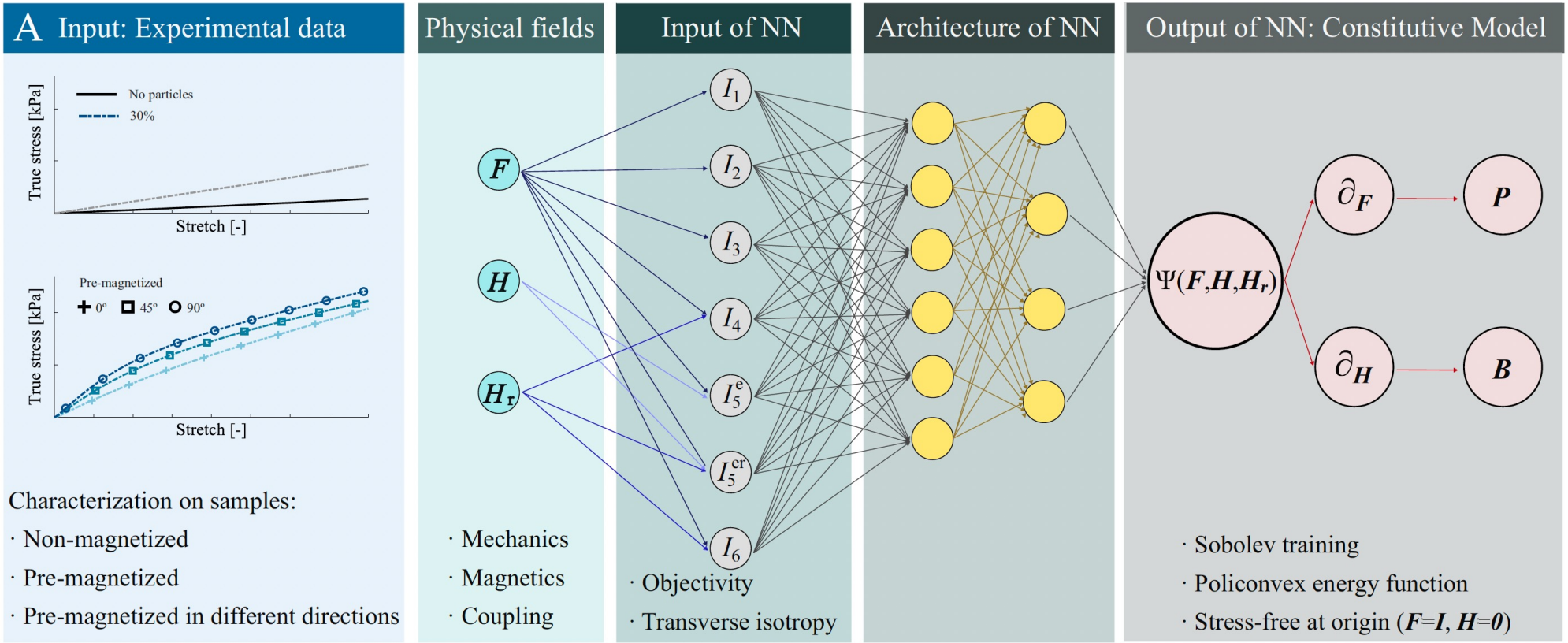
f SéNeCa(+)
Agencia de Ciencia y Tecnología
Región de Murcia
GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES
UNIÓN EUROPEA
FONDO EUROPEO DE DESARROLLO REGIONAL
"Una manera de hacer Europa"
AGENCIA ESTATAL DE INVESTIGACIÓN

Dowsil CY52-276 (≈3 kPa)

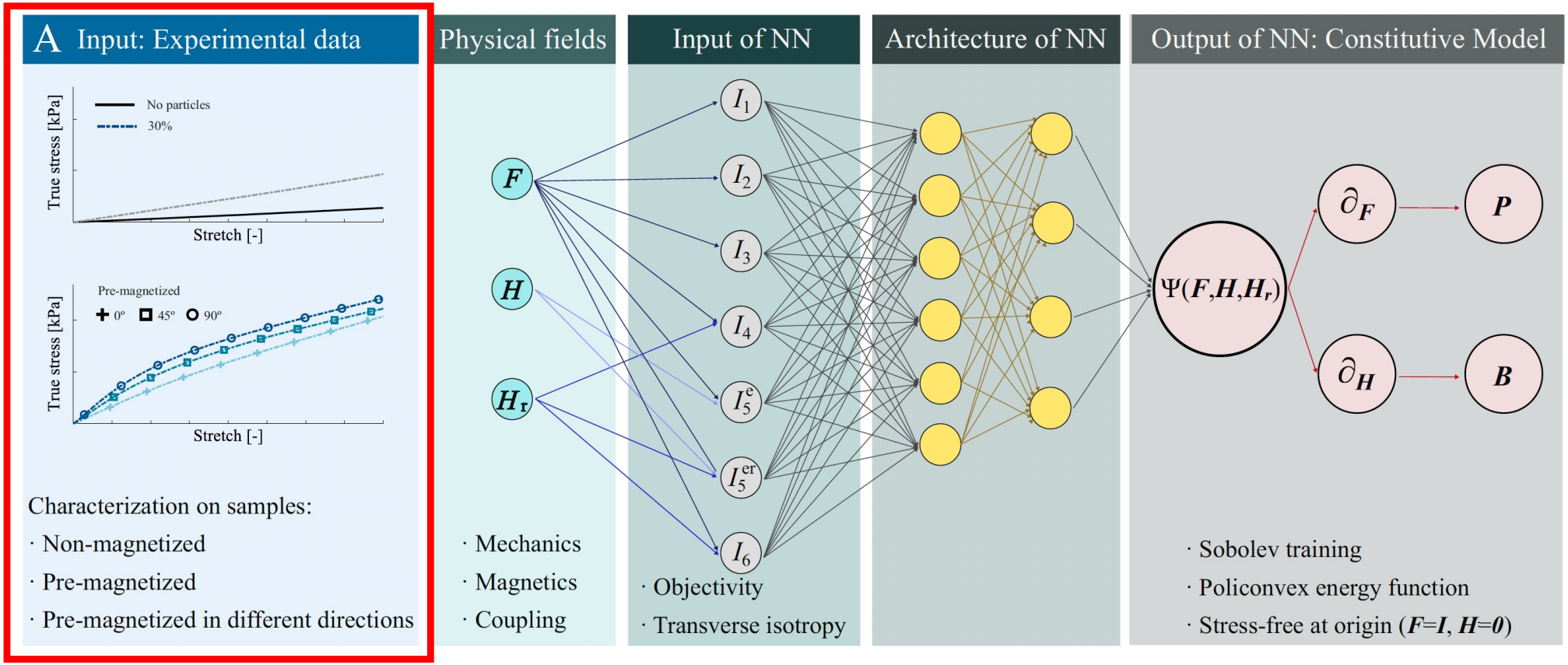
Computational framework



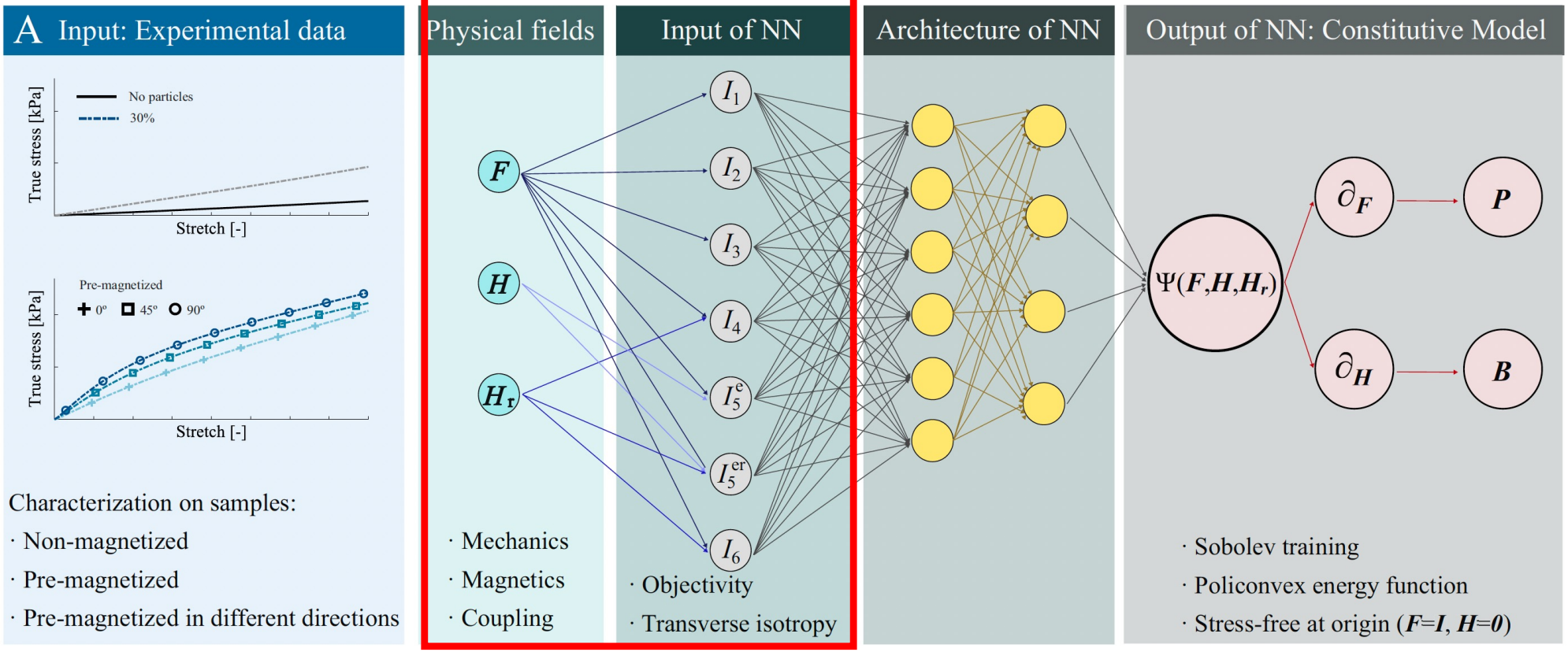
Physics-augmented neural networks constitutive model



Physics-augmented neural networks constitutive model



Physics-augmented neural networks constitutive model

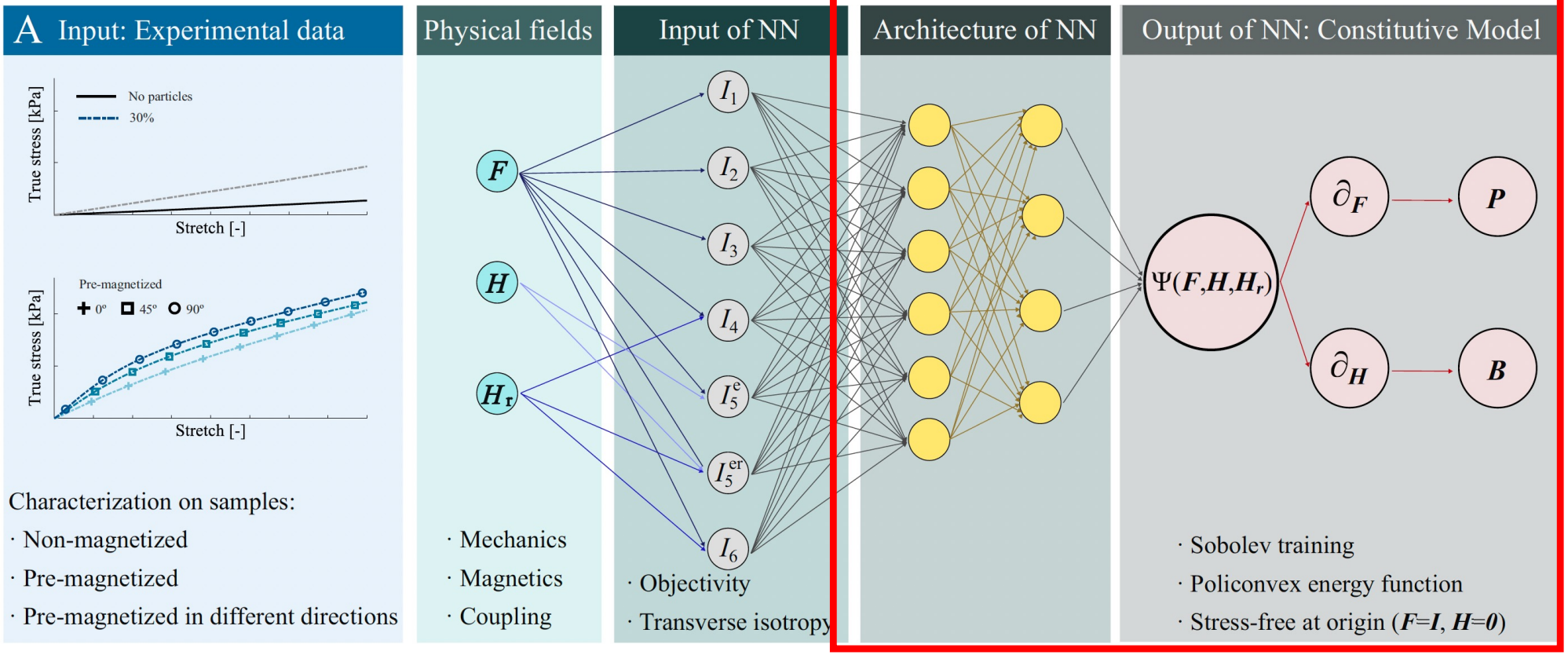


$$I_1 = \boldsymbol{F} : \boldsymbol{F}; \qquad I_2 = \mathrm{Cof}\boldsymbol{F} : \mathrm{Cof}\boldsymbol{F}; \qquad I_3 = \det \boldsymbol{F};$$

$$I_4 = \boldsymbol{F}\boldsymbol{H}_r \cdot \boldsymbol{F}\boldsymbol{H}_r; \qquad I_6 = \mathrm{Cof}\boldsymbol{F}\boldsymbol{H}_r \cdot \mathrm{Cof}\boldsymbol{F}\boldsymbol{H}_r;$$

$$I_5^e = \boldsymbol{H} \cdot \boldsymbol{C}^{-1} \cdot \boldsymbol{H} \qquad I_5^{er} = \boldsymbol{H} \cdot \boldsymbol{C}^{1/2} \cdot \boldsymbol{H}_r$$

Physics-augmented neural networks constitutive model

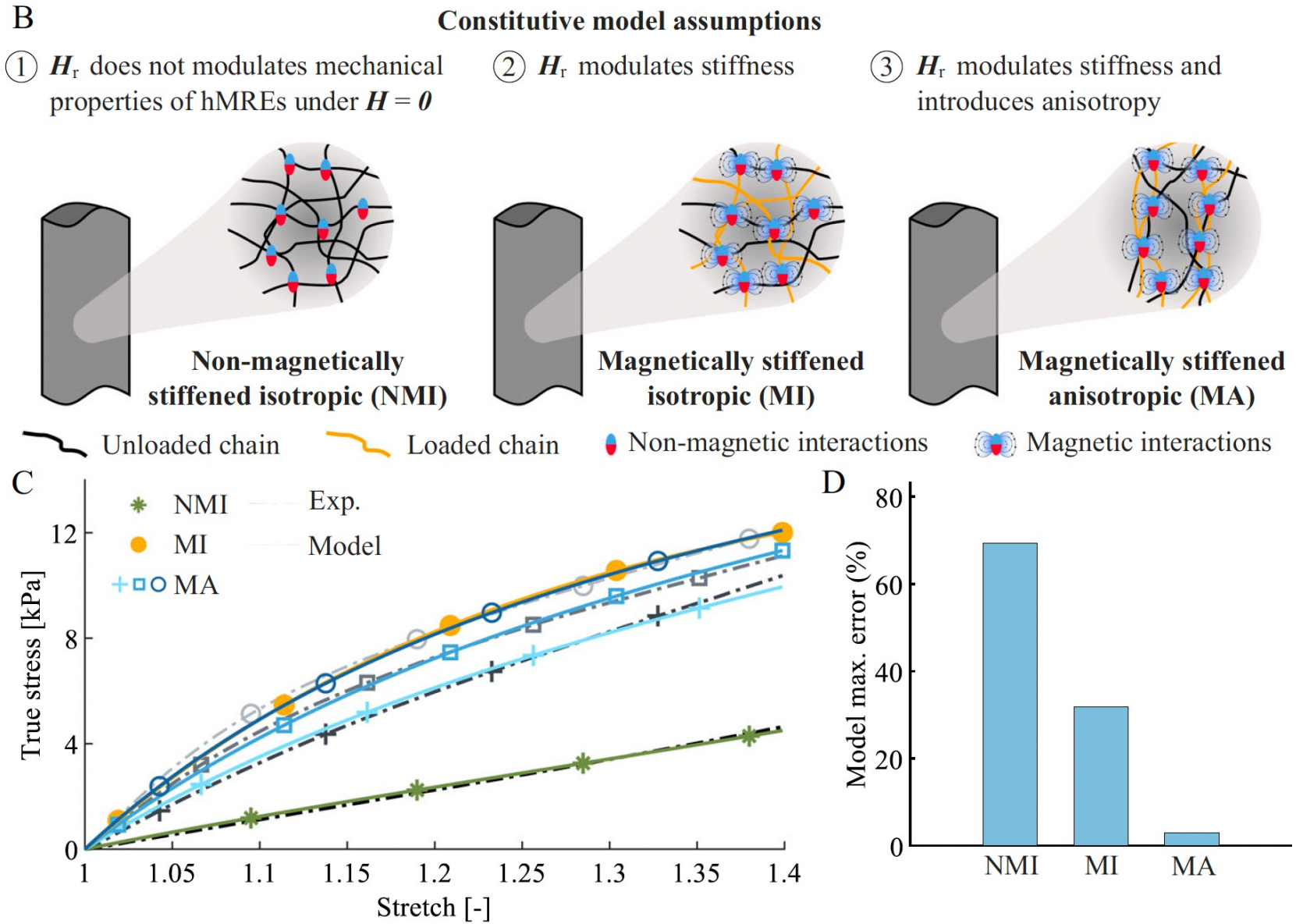


$$\Psi(\mathbf{F}, \mathbf{H}, \mathcal{W}) = \mathbb{U}(I_1, I_2, I_3, I_4, I_5, I_5^{er}, I_6; \mathcal{W})$$

$$\mathcal{W} = \{\mathbf{W}_1, \ldots, \mathbf{W}_{L+1}, \mathbf{b}_1, \ldots, \mathbf{b}_{L+1}\}$$

$$\mathcal{L}(\mathcal{W}) = \frac{\sum_{i=1}^{n_d} \left\| \mathbf{P}^i - \partial_{\mathbf{F}} \Psi(\mathbf{F}^i, \mathbf{0}, \mathcal{W}) \right\|^2}{\sum_{i=1}^{n_d} \left\| \mathbf{P}^i \right\|^2} + \frac{\kappa_r}{2} \sum_i \left(\min((\mathcal{W}_\nu)_i, 0)\right)^2$$

Physics-augmented neural networks constitutive model

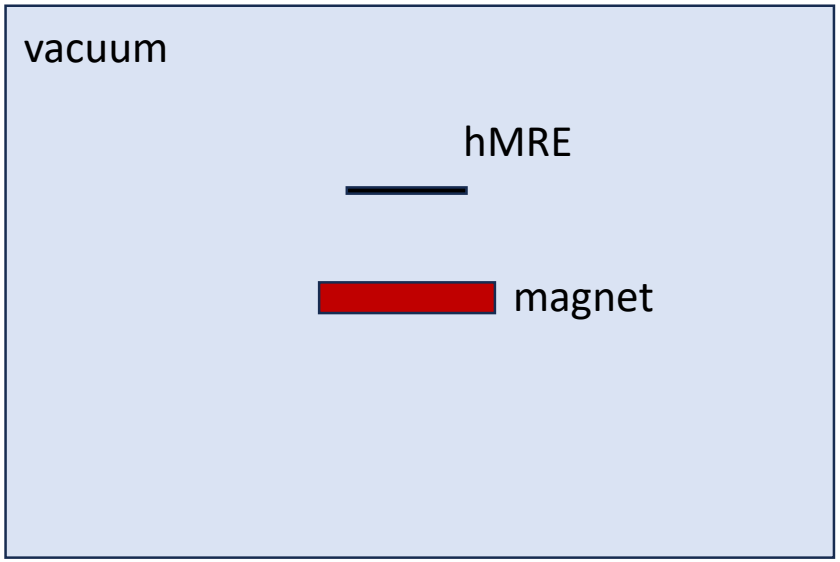


Staggered solver for magneto-mechanics in vacuum

The numerical resolution of the coupled magneto-mechanical problem was performed via a custom finite element implementation using a **staggered scheme.**

Step 1: Magnetostatic problem (scalar potential)
Solve for $\varphi \in \mathbb{V}^{\varphi}$ in $\Omega_0 \cup \Omega_{v0}$:

$$
\begin{aligned}
\mathrm{DIV}\,\mathbf{B}(\boldsymbol{X},\mathbf{F},\mathbf{H}) &= 0, \\
\mathbf{H} &= -\nabla_0 \varphi, \\
\mathbf{B} &= -\frac{\partial \Psi}{\partial \mathbf{H}} \quad \text{in } \Omega_0, \\
\mathbf{B} &= -\frac{\partial \Psi_{Maxw}}{\partial \mathbf{H}} \quad \text{in } \Omega_{v0}, \\
\varphi &= \varphi^{\star} \quad \text{on } \partial\Omega_{v0\varphi}.
\end{aligned}
$$



The scalar magnetic potential φ was computed in the entire domain (hMRE and vacuum). Continuity and boundary conditions were imposed across interfaces and on the boundary.

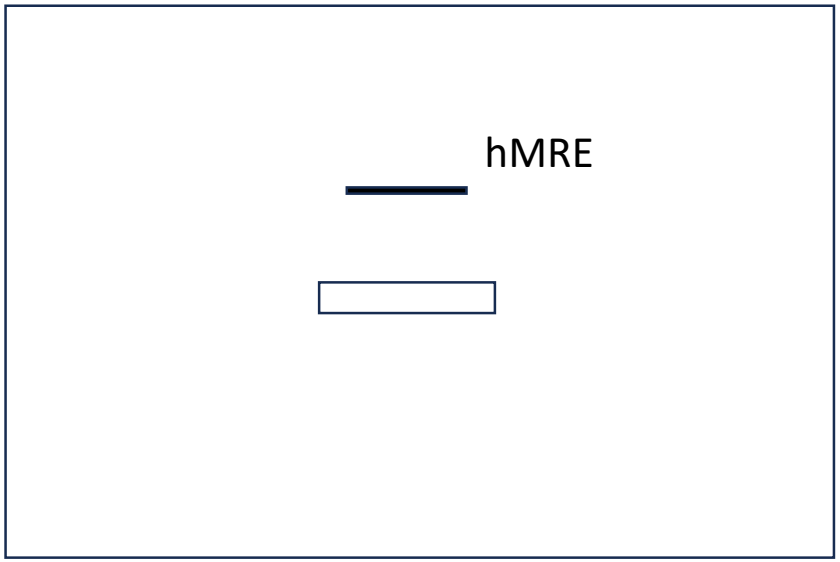
Staggered solver for magneto-mechanics in vacuum

The numerical resolution of the coupled magneto-mechanical problem was performed via a custom finite element implementation using a **staggered scheme.**

Step 2: Mechanical problem in the solid

Using $\mathbf{H}$ from Step 1, solve for $\boldsymbol{\phi} \in \mathbb{V}_s^{\boldsymbol{\phi}}$ in Ω_0:

$$
\begin{aligned}
\mathrm{DIV}\,\mathbf{P}(\boldsymbol{X}, \mathbf{F}, \mathbf{H}) &= \mathbf{F}_0, \\
\mathbf{F} &= \nabla_0 \boldsymbol{\phi}, \\
\boldsymbol{\phi} &= \boldsymbol{\phi}^\star \quad \text{on } \partial\Omega_{0\boldsymbol{\phi}}, \\
[\![\mathbf{P}]\!]\mathbf{N} &= \mathbf{0} \quad \text{on } \partial\Omega_0 \cap \partial\Omega_{v0}, \\
\mathbf{P} &= \frac{\partial \Psi}{\partial \mathbf{F}} \quad \text{in } \Omega_0.
\end{aligned}
$$



The deformation mapping $\boldsymbol{\phi}$ *was computed in the solid domain by solving the equilibrium equation. Dirichlet and Neumann boundary conditions were applied as appropriate, and* ***H is taken from Step 1.***

Staggered solver for magneto-mechanics in vacuum

The numerical resolution of the coupled magneto-mechanical problem was performed via a custom finite element implementation using a **staggered scheme.**

Step 3: Mechanical problem in the vacuum

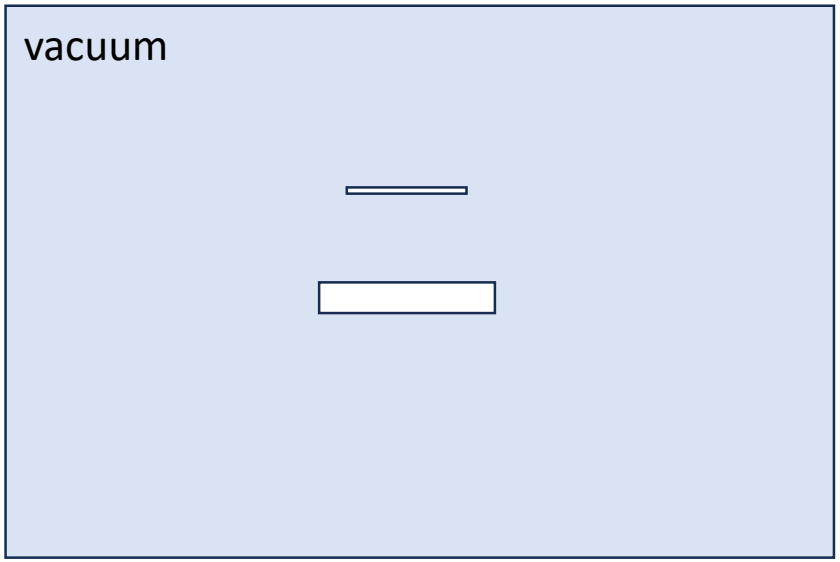
Enforce continuity with $\boldsymbol{\phi}$ and solve for $\boldsymbol{\phi}_v \in \mathbb{V}_v^{\boldsymbol{\phi}}$ in Ω_{v0}:

$$\mathrm{DIV}\left(\frac{\partial \Psi_{vm}}{\partial \mathbf{F}}(\mathbf{F})\right) = \mathbf{0},$$

$$\mathbf{F} = \nabla_0 \boldsymbol{\phi}_v,$$

$$\boldsymbol{\phi}_v = \mathbf{0} \quad \text{on } \partial\Omega_{v0\boldsymbol{\phi}},$$

$$\boldsymbol{\phi}_v = \boldsymbol{\phi} \quad \text{on } \partial\Omega_0 \cap \partial\Omega_{v0}.$$



A nonlinear elastic problem was solved in the vacuum region using a potential $\boldsymbol{\phi}$v, enforcing continuity with the solid deformation and applying clamped boundary conditions on its boundary.

Mesh movement stabilization

1) Fictitious energy model

$$\Psi_{vm}(\mathbf{F}) = \frac{\mu_1}{2p_1 3^{p_1-1}}\left(\hat{I}_1\right)^{p_1} + \frac{\mu_2}{2p_2 3^{p_2-1}}\left(\hat{I}_2\right)^{p_2} + \frac{\kappa}{2}\left(J^{p_3} + J^{-p_3}\right)$$

$\{\mu_1, \mu_2, \kappa\}$ represent fictitious elastic properties of Ω_{v0}

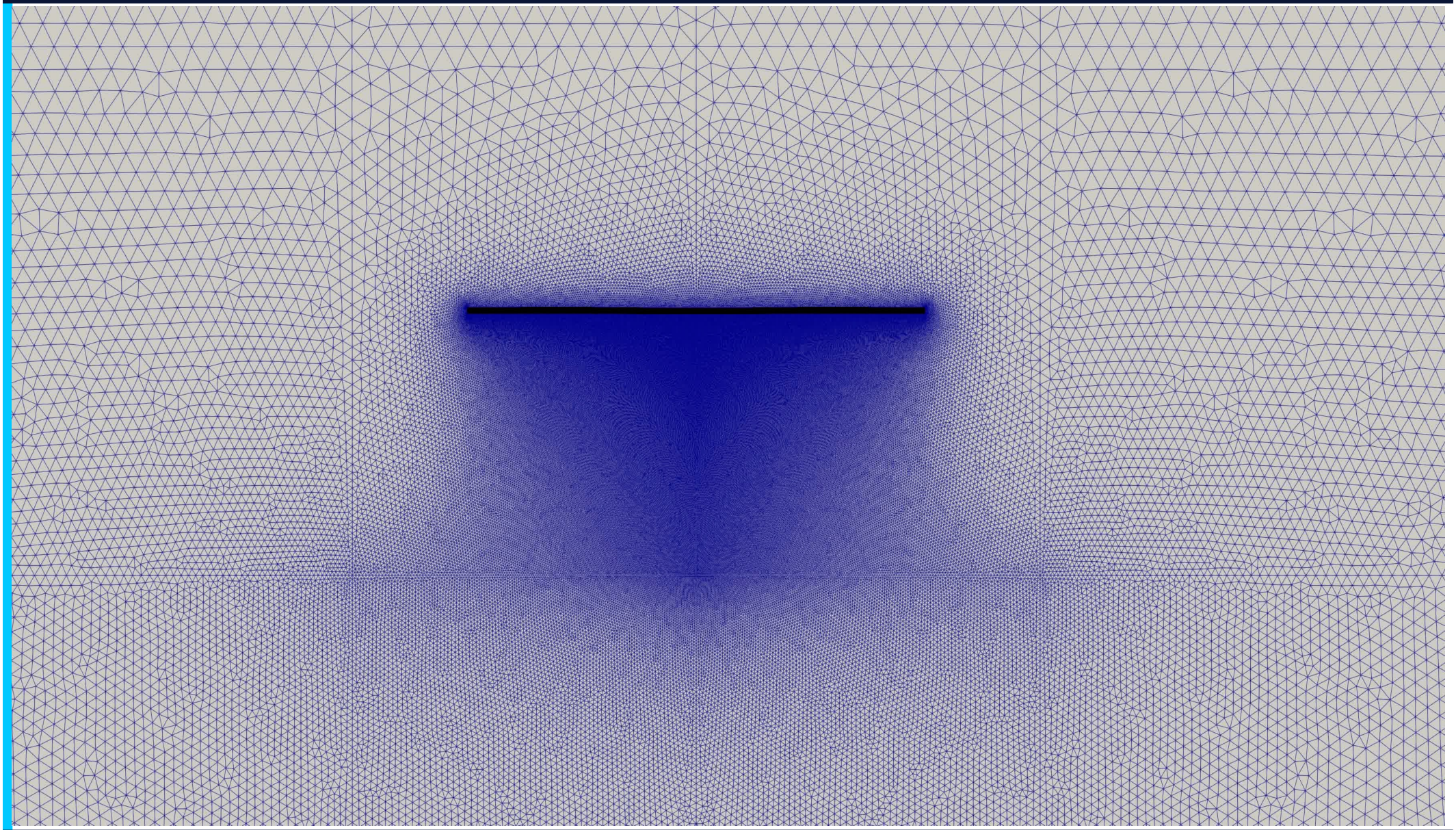
$\{p_1, p_2, p_3\}$ penalise in a nonlinear fashion the isochoric invariants and the volumetric invariant J

2) Stabilization of the elasticity tensor

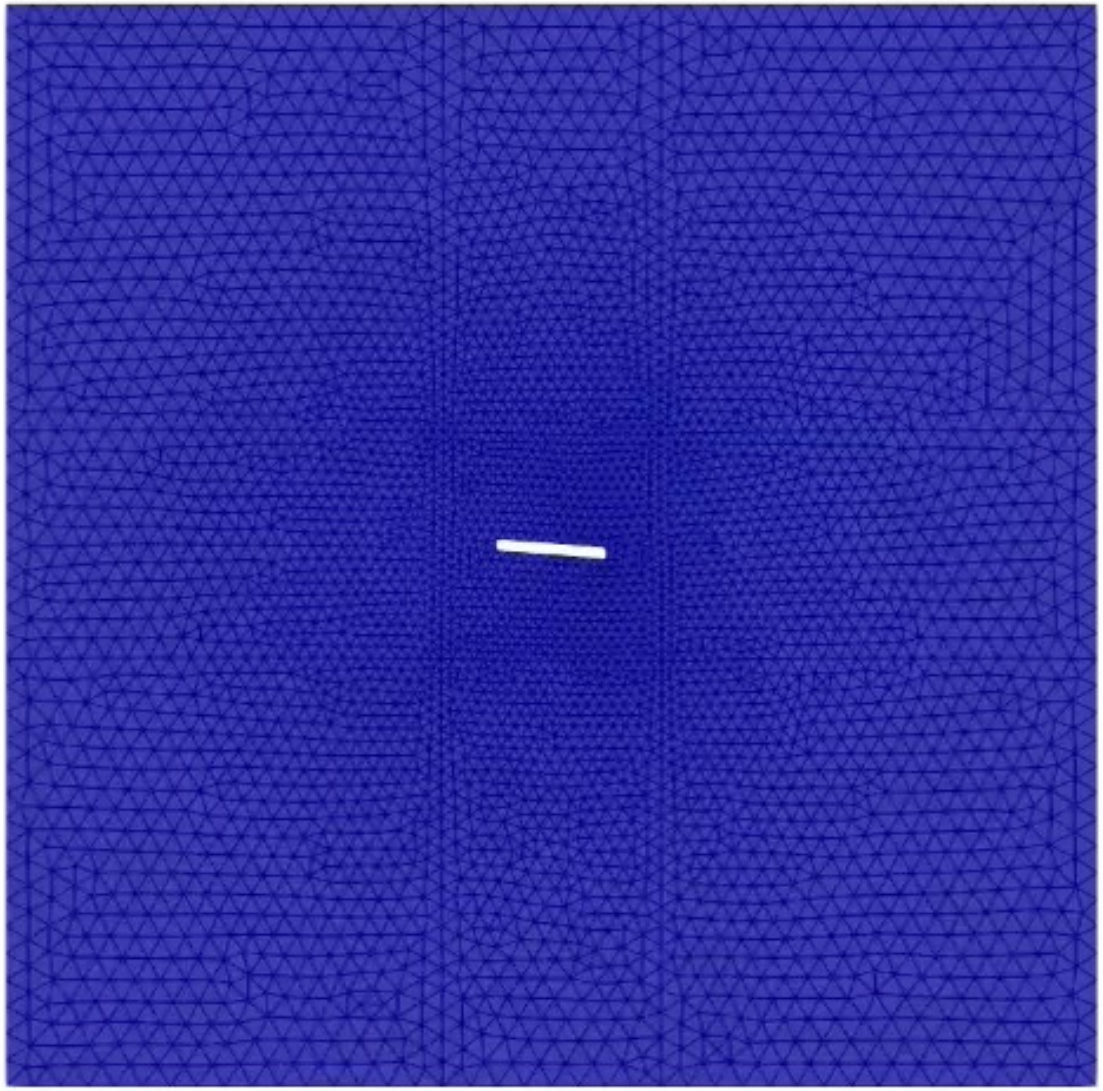
$$D\mathcal{W}_{\phi_v}(\boldsymbol{\phi}_v|_k)[\Delta\boldsymbol{\phi}_v] = \int_{\Omega_0} \boldsymbol{\nabla}_0\boldsymbol{v} : \boldsymbol{\mathcal{C}}_{\text{reg}}(\mathbf{F}) : \boldsymbol{\nabla}_0\Delta\boldsymbol{\phi}_v \, dV$$

$$\boldsymbol{\mathcal{C}}_{\text{reg}} = \sum_{i=1}^{9} \max\left(\lambda_i, 0\right) \boldsymbol{V}_i \otimes \boldsymbol{V}_i,$$

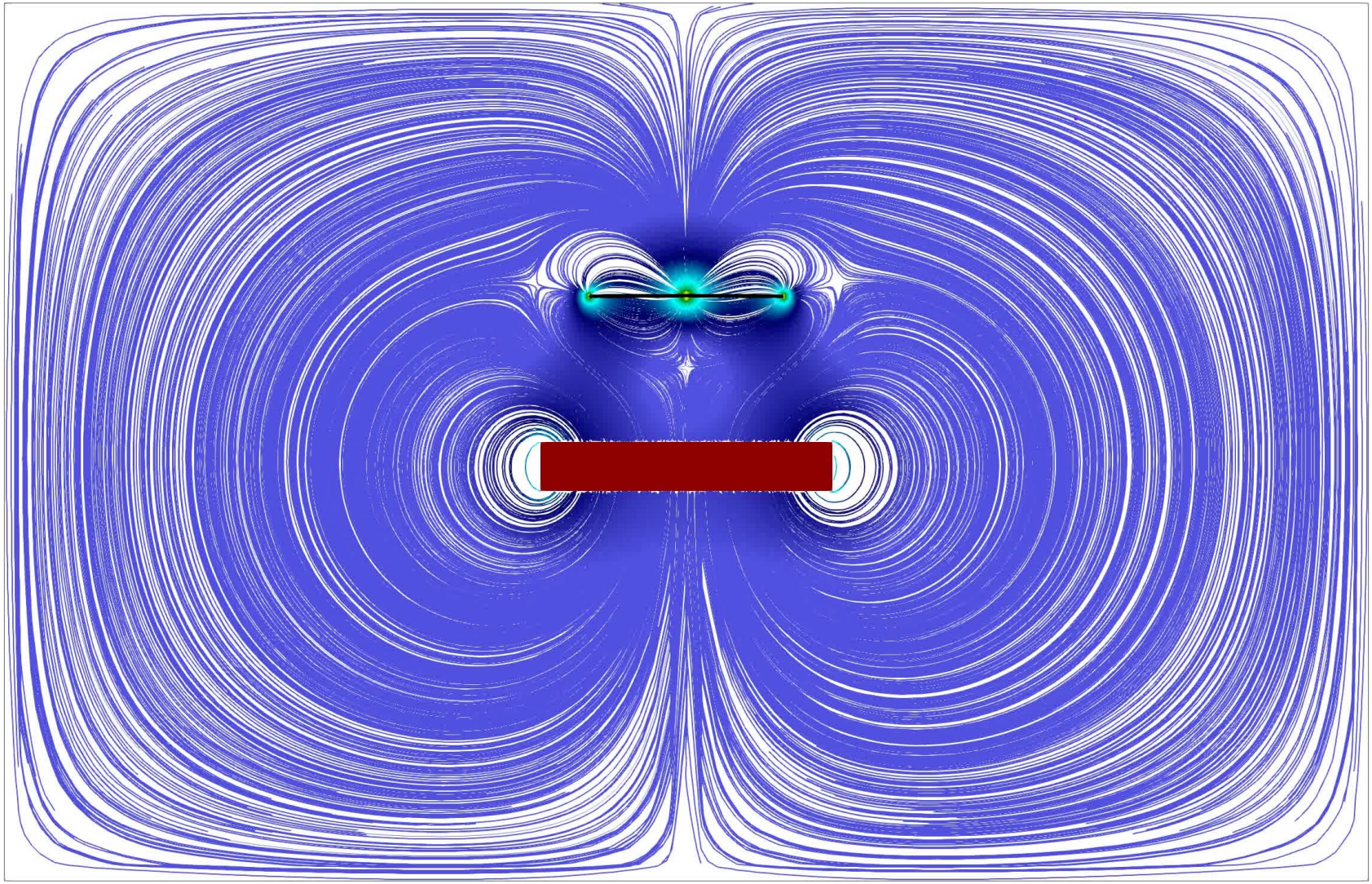
Mesh movement stabilization



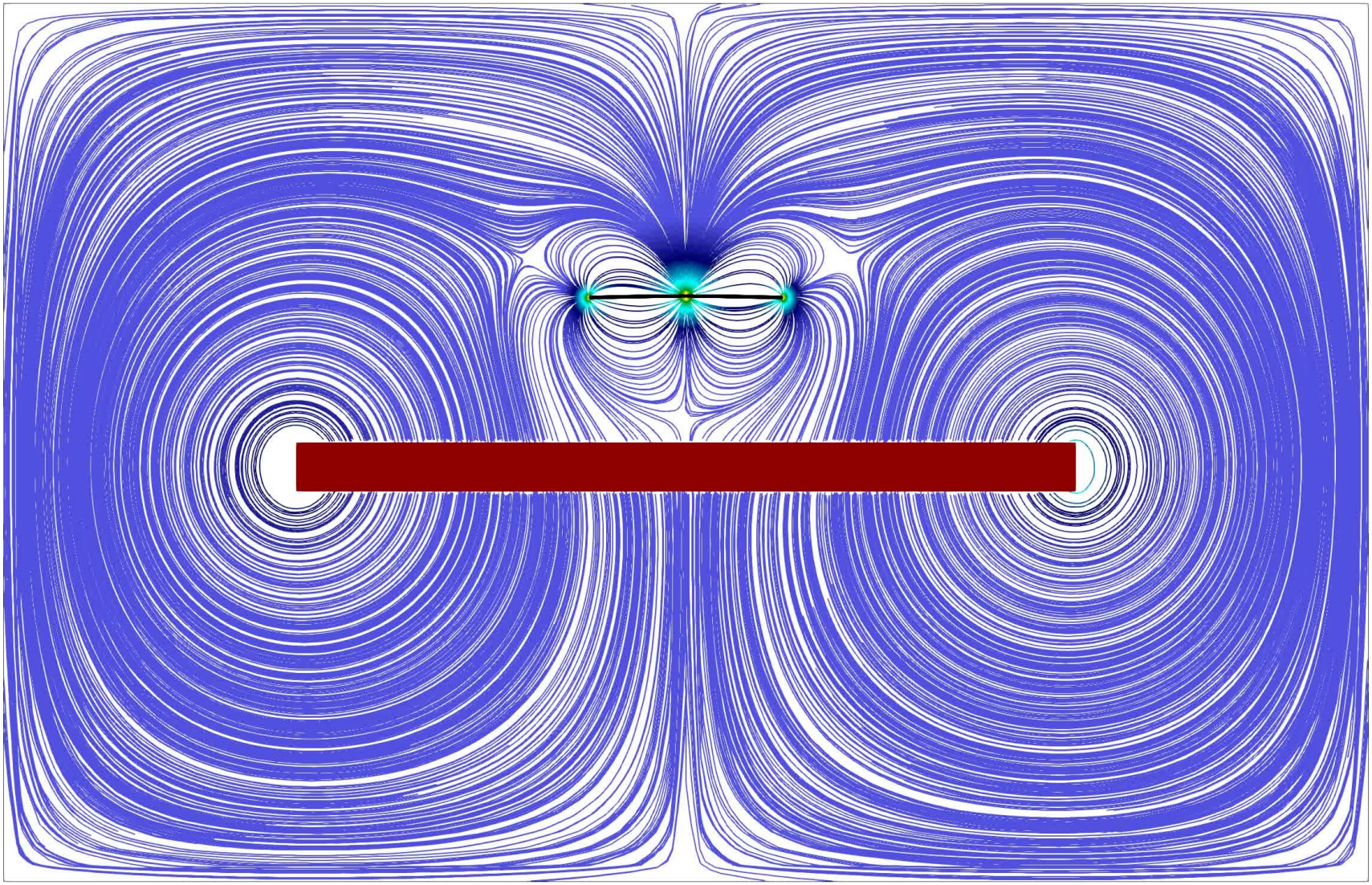
Mesh movement stabilization



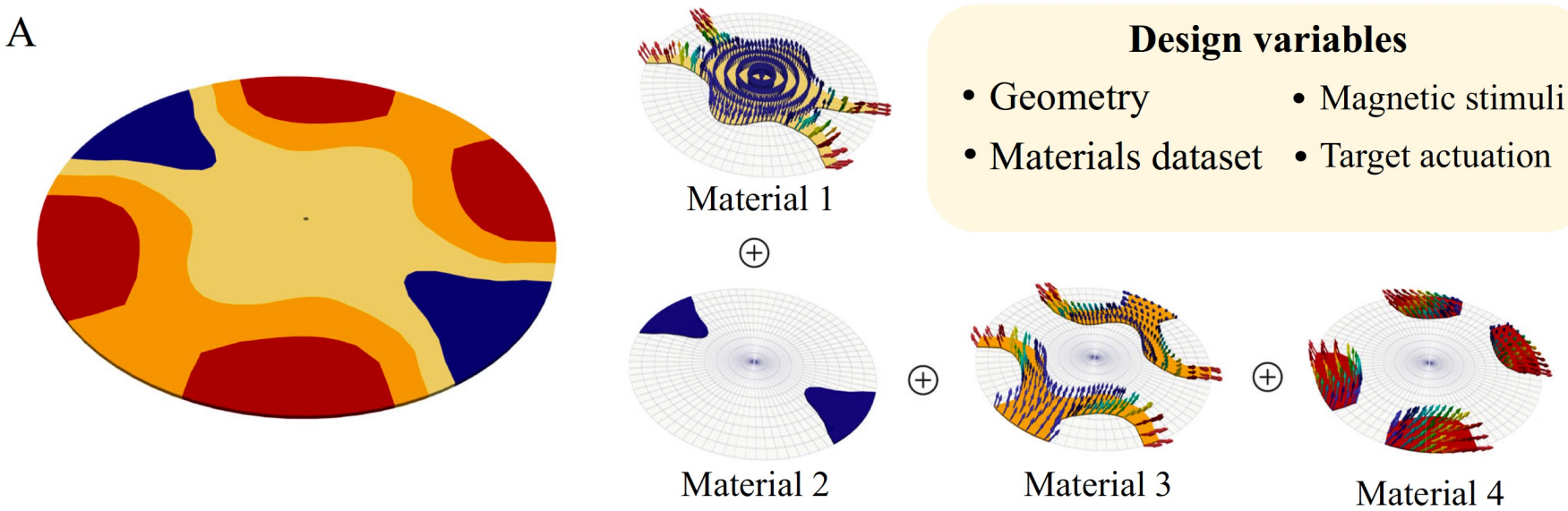
Staggered solver for magneto-mechanics in vacuum



Examples



Optimization framework



Design variables

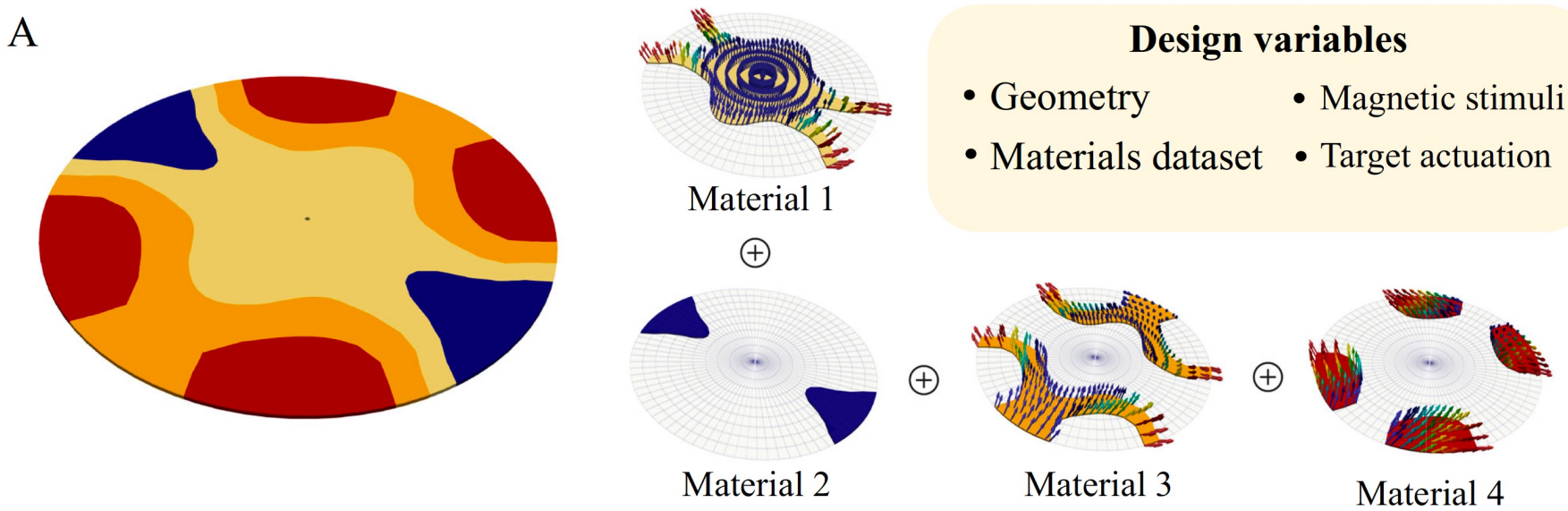
- Geometry
- Materials dataset
- Magnetic stimuli
- Target actuation

$$\mathbf{H}_r = H_r \mathbf{N}(\theta, \varphi), \qquad \mathbf{N}(\theta, \varphi) = \begin{bmatrix} \cos(\theta)\sin(\varphi) & \sin(\theta)\sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

$$\Psi(\xi, c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - \xi^{p_\xi})\, \Psi_p(\mathbf{F}) + \xi^{p_\xi}\, \Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F})$$

$$\Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - c^{p_c})\ \Psi|_{c=0.2}\,(\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app}) + c^{p_c}\ \Psi|_{c=0.3}\,(\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app})$$

Optimization framework



Design variables

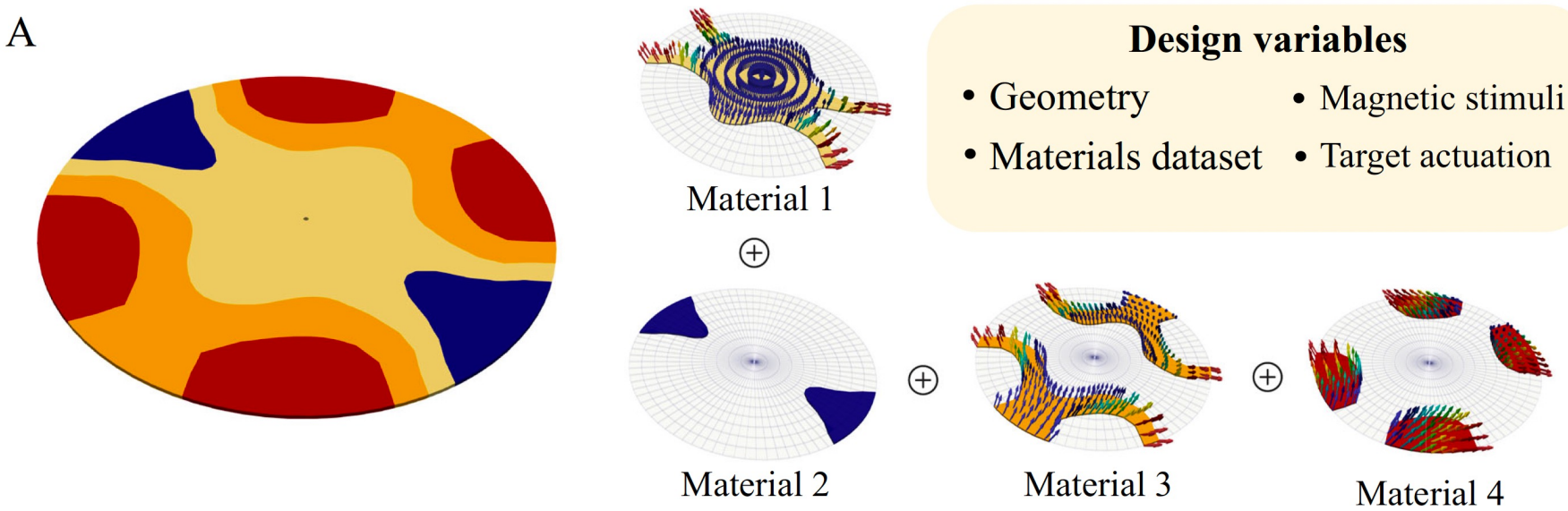
- Geometry
- Materials dataset
- Magnetic stimuli
- Target actuation

$$\mathbf{H}_r = H_r \mathbf{N}(\theta, \varphi), \qquad \mathbf{N}(\theta, \varphi) = \begin{bmatrix} \cos(\theta)\sin(\varphi) & \sin(\theta)\sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

$$\Psi(\xi, c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - \xi^{p_\xi})\, \Psi_p(\mathbf{F}) + \xi^{p_\xi}\, \Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F})$$

$$\Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - c^{p_c})\ \Psi|_{c=0.2}\,(\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app}) + c^{p_c}\ \Psi|_{c=0.3}\,(\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app})$$

Optimization framework



Design variables

- Geometry
- Materials dataset
- Magnetic stimuli
- Target actuation

$$\mathbf{H}_r = H_r \mathbf{N}(\theta, \varphi), \qquad \mathbf{N}(\theta, \varphi) = \begin{bmatrix} \cos(\theta)\sin(\varphi) & \sin(\theta)\sin(\varphi) & \cos(\varphi) \end{bmatrix}$$

$$\Psi(\xi, c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - \boxed{\xi^{p_\xi}})\, \Psi_p(\mathbf{F}) + \xi^{p_\xi}\, \Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F})$$

$$\Psi_a(c, \theta, \varphi, H_{app}, \mathbf{F}) = (1 - \boxed{c^{p_c}})\, \Psi|_{0.2} \quad (\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app}) + c^{p_c}\, \Psi|_{0.3} \quad (\mathbf{F}, \mathbf{H}_r(\theta, \varphi), H_{app})$$

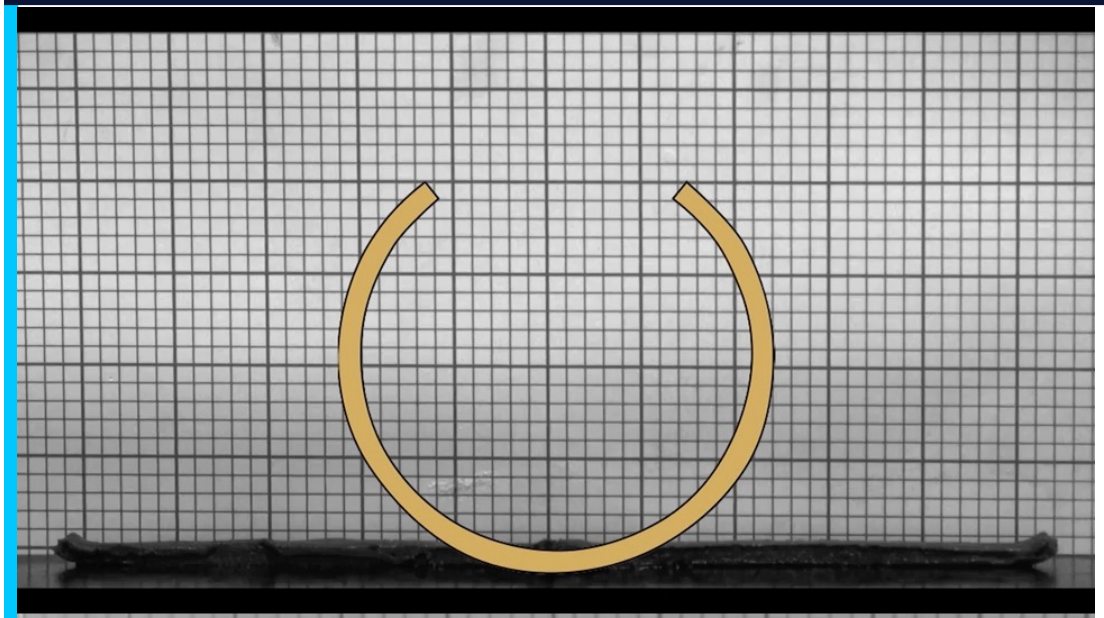
Optimization formulation

$$\text{Minimize in } (\xi, c, \theta, \varphi, H_r): \ \mathcal{J}(\xi, c, \theta, \varphi, H_{app}) := \int_{\Omega_0} \|\boldsymbol{\phi}(\xi, c, \theta, \varphi, H_{app}) - \boldsymbol{\phi}_d\|^2 \, dV$$

$$\text{subject to} \begin{cases} \begin{cases} \mathrm{DIV}\mathbf{P}(\xi, c, \theta, \psi, H_{app}, \mathbf{F}) = \mathbf{F}_0; & \text{in } \Omega_0; \\ \mathbf{F} = \boldsymbol{\nabla}_0 \boldsymbol{\phi} \Rightarrow \mathrm{CURL}\mathbf{F} = \mathbf{0}; & \text{in } \Omega_0 \\ \boldsymbol{\phi} = \boldsymbol{\phi}^\star; & \text{on } \partial\Omega_{0\boldsymbol{\phi}} \end{cases} \\ \begin{cases} \Psi(\xi, c, \theta, \psi, H_{app}, \mathbf{F}) = (1 - \xi^{p_\xi})\Psi_p(\mathbf{F}) + \xi^{p_\xi}\Psi_a(c, \theta, \psi, H_{app}, \mathbf{F}) \\ \Psi_a(c, \theta, \psi, H_{app}, \mathbf{F}) = \begin{cases} (1 - c^{p_c}) \ \Psi|_{c=0.2} (\mathbf{F}, \mathbf{H}_r(\theta, \psi), H_{app}) \\ + c^{p_c} \ \Psi|_{c=0.3} (\mathbf{F}, \mathbf{H}_r(\theta, \psi), H_{app}) \end{cases} \\ \mathbf{P}(\xi, c, \theta, \psi, H_{app}, \mathbf{F}) = \dfrac{\partial \Psi(\xi, c, \theta, \psi, \mathbf{F}, H_{app})}{\partial \mathbf{F}} \end{cases} \\ \xi \in [0, 1] \\ c \in [0, 1] \\ \theta \in [-2\pi, 2\pi] \\ \psi \in [-2\pi, 2\pi] \\ H_{app} \in [0, H_{app,max}] \end{cases}$$

Optimized Dowsil C-shape

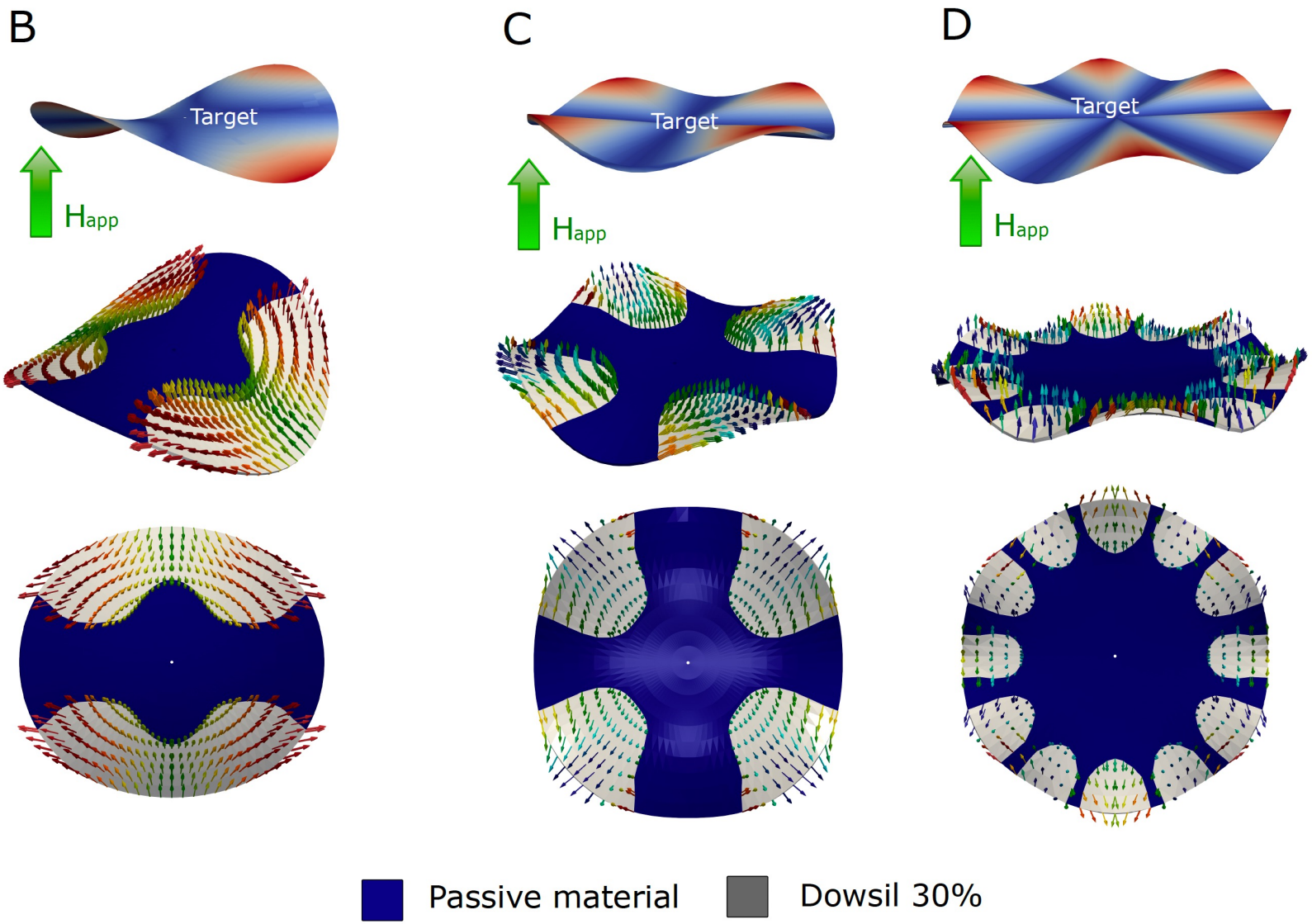
Optimized Dowsil C-shape



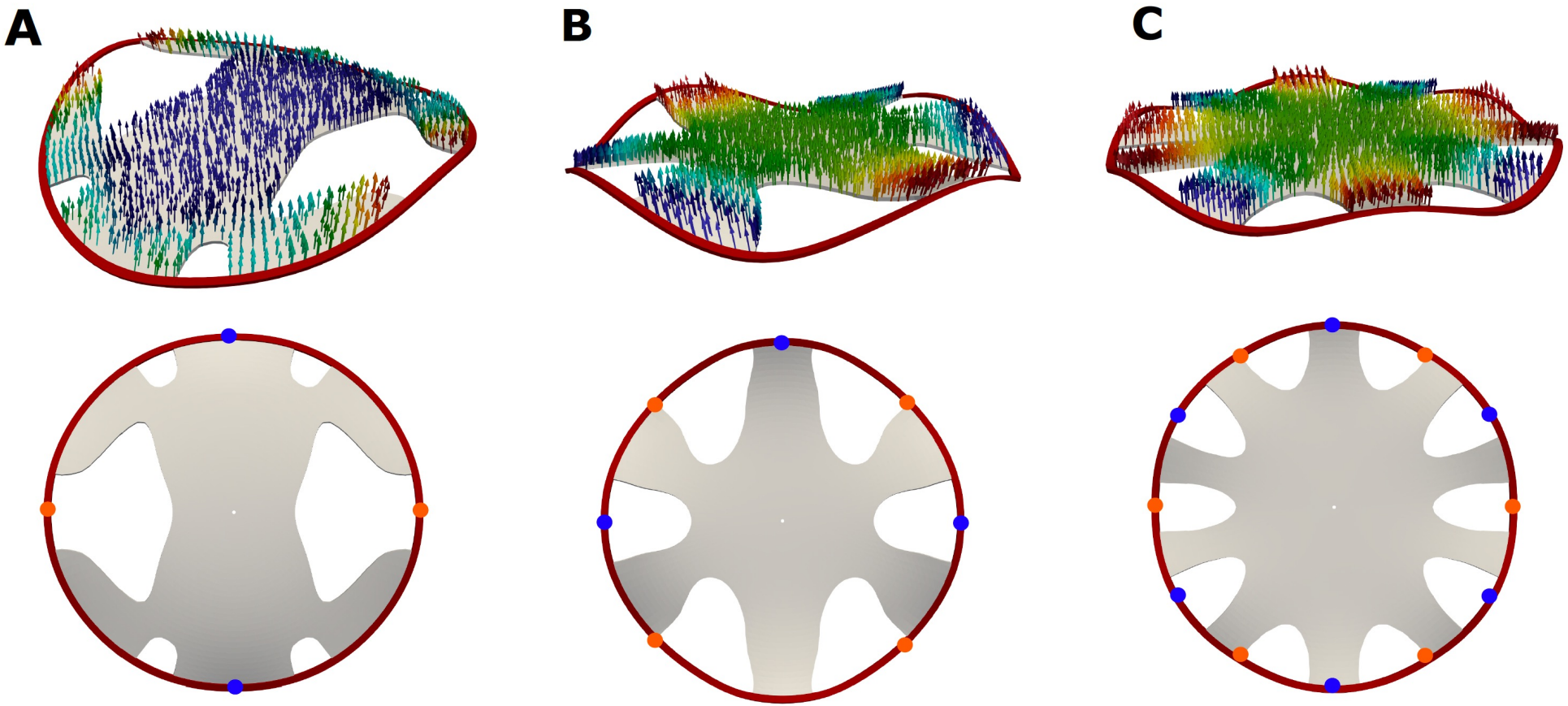
Topology and material optimization

f SéNeCa(+)
Agencia de Ciencia y Tecnología
Región de Murcia
GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES
UNIÓN EUROPEA
FONDO EUROPEO DE DESARROLLO REGIONAL
"Una manera de hacer Europa"
AGENCIA ESTATAL DE INVESTIGACIÓN

Staggered Approach



Staggered Approach



Optimizable region

Non optimizable region

Maximize out of plane displacement

Minimize out of plane displacement

HyperFEM (julia)

github.com

README MIT license

HyperFEM

docs stable | docs dev | Tests passing | codecov unknown

Multiphysics Simulation of Multifunctional Soft Materials

HyperFEM is a library within the Gridap ecosystem designed for the simulation of multiphysics problems involving multifunctional hyperelastic materials. The ultimate goal of HyperFEM is to provide a **high-level, expressive, and rapid prototyping tool** that accelerates the modeling stages of **Thermo–Electro–Magneto–Mechanical** multiphysics problems. To this end, it includes a comprehensive library of **analytically derived constitutive models** formulated through tensor algebra. In addition, HyperFEM provides abstractions for **monolithic and staggered solution schemes**, extending Gridap's capabilities in the context of **nonlinear solid mechanics**.

Installation

Open the Julia REPL, type `]` to enter package mode, and install as follows

```
pkg> add HyperFEM
```

Usage

First, include the main HyperFEM module:

```
using HyperFEM
```

Example of a Monolithic implementation for the simulation of Nonlinear ElectroMechanical deformation of dielectric elastomers

https://github.com/MultiSimOLab/Adv_Mater_Ultra_Soft_Magnet

HyperFEM (julia)

- **High-level multiphysics framework** within the Gridap ecosystem for nonlinear hyperelastic materials

- Designed to **bridge the gap between continuum-level modeling and algorithms based on PDEs** (optimization, control, ML, UQ...).

- Rapid prototyping of **thermo-electro-magneto-mechanical-acoustic-fluid** problems using analytically derived constitutive models

- Dedicated abstractions for **monolithic and staggered solution** schemes in nonlinear solid mechanics

HyperFEM (julia)

TensorAlgebra: high-level tensor operations generated through multiple dispatch and metaprogramming

```
"""
    ⊗₁₃²⁴(A::TensorValue{D}, B::TensorValue{D})::TensorValue{D*D}

Outer product of two second-order tensors (matrices), returning a fourth-order tensor
represented in a `D² x D²` flattened matrix using combined indices.
"""
@generated function (⊗₁₂³⁴)(A::TensorValue{D,D,Float64}, B::TensorValue{D,D,Float64}) where {D}
  str = ""
  for iB in 1:D*D
    for iA in 1:D*D
      str *= "A.data[$iA] * B.data[$iB], "
    end
  end
  Meta.parse("TensorValue{D*D,D*D, Float64}($str)")
end


"""
```

$$(I3 \otimes_{13}^{24} (N \otimes N))$$

HyperFEM (julia)

PhysicalModels: a high-level abstraction for constitutive models, providing analytically defined continuum energy functions and consistent derivatives

```julia
struct NeoHookean3D <: IsoElastic
  λ::Float64
  μ::Float64
  ρ::Float64

  function NeoHookean3D(; λ::Float64, μ::Float64, ρ::Float64=0.0)
    new(λ, μ, ρ)
  end

  function (obj::NeoHookean3D)(Threshold=0.01)
    λ, μ = obj.λ, obj.μ
    J(F) = det(F)
    H(F) = det(F) * inv(F)'
    Ψ(F) = μ / 2 * tr((F)' * F) - μ * logreg(J(F)) + (λ / 2) * (J(F) - 1)^2 - 3.0 * (μ / 2.0)

    ∂log∂J(J) = J >= Threshold ? 1 / J : (2 / Threshold - J / (Threshold^2))
    ∂log2∂J2(J) = J >= Threshold ? -1 / (J^2) : (-1 / (Threshold^2))

    ∂Ψ_∂J(F) = -μ * ∂log∂J(J(F)) + λ * (J(F) - 1)
    ∂Ψu(F) = μ * F + ∂Ψ_∂J(F) * H(F)
    ∂Ψ2_∂J2(F) = -μ * ∂log2∂J2(J(F)) + λ
    ∂Ψuu(F) = μ * I9 + ∂Ψ2_∂J2(F) * (H(F) ⊗ H(F)) + ∂Ψ_∂J(F) * ×ᵢ⁴(F)
    return (Ψ, ∂Ψu, ∂Ψuu)
  end
end
```

HyperFEM (julia)

PhysicalModels: a high-level abstraction for constitutive models, providing analytically defined continuum energy functions and consistent derivatives

```
modeliso = MooneyRivlin3D(λ=(1e5, μ1=1e3, μ2=1e3)
fibra1 = HGO_1Fiber(c1=c1[1], c2=c2[1])
fibra2 = HGO_1Fiber(c1=c1[2], c2=c2[2])
fibra3 = HGO_1Fiber(c1=c1[3], c2=c2[3])
fibra4 = HGO_1Fiber(c1=c1[4], c2=c2[4])
modelelec = IdealDielectric(ε=4.0)

constitutive_model= modeliso +[fibra1 fibra2 fibra3 fibra4]+ modelelec
```

C. Perez-Garcia, R. Ortigosa, J. Martínez-Frutos, and D. Garcia-Gonzalez, ***Topology and material optimization in ultra-soft magnetoactive structures: making advantage of residual anisotropies.*** *Adv. Mater. (2025)*

ACKNOWLEDGEMENTS

- Project **POTENTIAL** Grant PID2022-141957OA-C22 funded by MCIN/AEI/10.13039/501100011033 and by RDF A way of making Europe".
- UC3M ERC Starting Grant **4D-BIOMAP**
- MULTISIMO-60Nd partnership

GOBIERNO DE ESPAÑA
MINISTERIO DE CIENCIA, INNOVACIÓN Y UNIVERSIDADES

UNIÓN EUROPEA
FONDO EUROPEO DE DESARROLLO REGIONAL
"Una manera de hacer Europa"

AGENCIA ESTATAL DE INVESTIGACIÓN

Low σ

Stress Field

HyperFEM (julia)

ComputationalModels: an abstraction of numerical problem classes (e.g., static linear, static nonlinear, dynamic nonlinear, staggered schemes) designed to encapsulate PDE solvers and enable their integration within outer-loop workflows such as control, optimization, and machine learning

```
# Problem 1: residual and jacobian
res_mag(Λ) = (φ, vφ) -> -1.0 * ∫((∇(vφ) ⋅ (∂Ψs∂H0 ∘ (F ∘ (∇(uhsolid(Λ))'), ℋ₀ ∘ (∇(φ)), Nh))))dΩsolid -
                                ∫((∇(vφ) ⋅ (∂Ψv∂H0 ∘ (F ∘ (∇(uhvacuum(Λ))'), ℋ₀ ∘ (∇(φ))))))dΩvacuum_mag

jac_mag(Λ) = (φ, dφ, vφ) -> ∫(∇(vφ)' ⋅ ((∂Ψs∂H0H0 ∘ (F ∘ (∇(uhsolid(Λ))'), ℋ₀ ∘ (∇(φ)), Nh)) ⋅ ∇(dφ)))dΩsolid +
                                ∫(∇(vφ)' ⋅ ((∂Ψv∂H0H0 ∘ (F ∘ (∇(uhvacuum(Λ))'), ℋ₀ ∘ (∇(φ)))) ⋅ ∇(dφ)))dΩvacuum_mag

# Problem 2: residual and jacobian
res_mech(Λ) = (u, v) -> ∫((∇(v)' ⊙ (∂Ψs∂F ∘ (F ∘ (∇(u)'), ℋ₀ ∘ (∇(φh(Λ))), Nh))))dΩsolid -
                                ∫((v.⁺ ⋅ ((∂Ψv∂F ∘ (F ∘ (∇(uhvacuum(Λ))'), ℋ₀ ∘ (∇(φh(Λ))))).⁻ ⋅ nΓsf.⁺)))dΓsf

jac_mech(Λ) = (u, du, v) -> ∫(∇(v)' ⊙ ((∂Ψs∂FF ∘ (F ∘ (∇(u)'), ℋ₀ ∘ (∇(φh(Λ))), Nh)) ⊙ (∇(du)')))dΩsolid

# Problem 3: residual and jacobian
res_vacmech(Λ) = (u, v) -> ∫((∇(v)' ⊙ (∂Ψvm∂F ∘ (F ∘ (∇(u)')))))dΩvacuum_mec
jac_vacmech(Λ) = (u, du, v) -> ∫(∇(v)' ⊙ ((∂Ψvm∂FF ∘ (F ∘ (∇(u)'))) ⊙ (∇(du)')))dΩvacuum_mec
```

HyperFEM (julia)

ComputationalModels: an abstraction of numerical problem classes (e.g., static linear, static nonlinear, dynamic nonlinear, staggered schemes) designed to encapsulate PDE solvers and enable their integration within outer-loop workflows such as control, optimization, and machine learning

```
# ******************************************************
#             Computational problems
#******************************************************

# Problem 1
nls_mag = NewtonSolver(LUSolver(); maxiter=20, atol=1.e-9, rtol=1.e-8, verbose=true)
comp_model_mag = StaticNonlinearModel(res_mag, jac_mag, Uφ⁺, Vφ, Dφ; nls=nls_mag, xh=φh⁺)

# Problem 2
nls_mech = NewtonSolver(LUSolver(); maxiter=20, atol=1.e-6, rtol=1.e-2, verbose=true)
comp_model_mech = StaticNonlinearModel(res_mech, jac_mech, Uu_solid⁺, Vu_solid, Du_solid; nls=nls_mech, xh=uh_solid⁺)

# Problem 3
nls_vacmech = NewtonSolver(LUSolver(); maxiter=10,  rtol=2, verbose=true)
comp_model_vacmech = StaticNonlinearModel(res_vacmech, jac_vacmech, Uu_vacuum⁺, Vu_vacuum, Du_vacuum; nls=nls_vacmech, xh=uh_vacuum⁺)

# Staggered resolution:  Problem1-> Problem2-> Problem3
comp_model = StaggeredModel((comp_model_mag, comp_model_mech, comp_model_vacmech), (φh⁺, uh_solid⁺, uh_vacuum⁺), (φh⁻, uh_solid⁻, uh_vacuum⁻))
```