POLITECNICO
MILANO 1863

European Research Council
Established by the European Commission

NEMESIS
New generation methods for numerical simulations

Numerical codes for PolydG approximations: lymph & vulpes libraries

Ilario Mazzieri, *(MOX, Department of Mathematics, Politecnico di Milano)*

Towards Polytopal meshes in Gmsh Workshop

26–27 January 2026, Montpellier

Team members

Paola F. Antonietti

Stefano Bonetti

Michele Botti

Matteo Caldana

Mattia Corti

Nicola de March

Ivan Fumagalli

Caterina B. Leimer Saglio

Ilario Mazzieri

Motivations

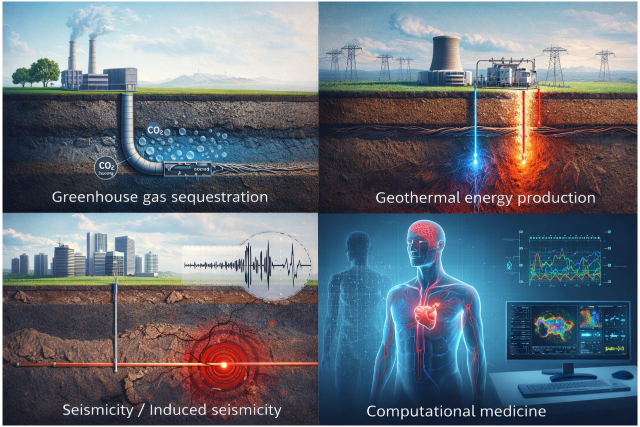
The solution of coupled multi-physics problems if of crucial importance nowadays.

Motivations:

→ Energy sustainability
→ Life sciences
→ Advanced manufacturing

(Some) Applications:

→ Greenhouse gas sequestration
→ Geothermal energy production
→ Seismicity / Induced seismicity
→ Computational medicine



Motivations

The solution of coupled multi-physics problems if of crucial importance nowadays.

Motivations:

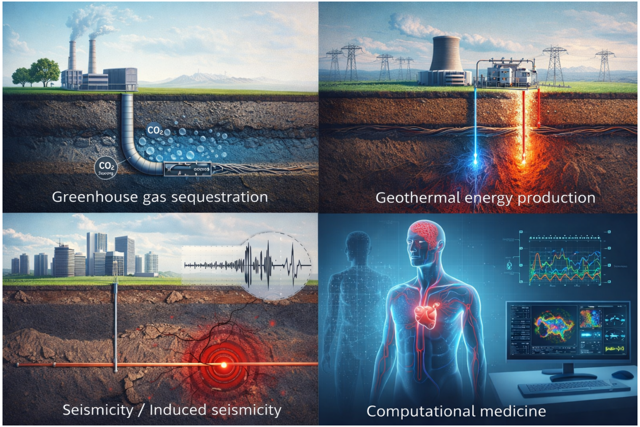
→ Energy sustainability
→ Life sciences
→ Advanced manufacturing

(Some) Applications:

→ Greenhouse gas sequestration
→ Geothermal energy production
→ Seismicity / Induced seismicity
→ Computational medicine

Challenges:

→ Fully-coupled problem
→ Different space/time scales
→ Non-linearities
→ Geometrical complexity
→ Highly heterogeneous media
→ High computational costs



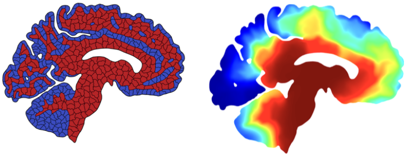
Goal

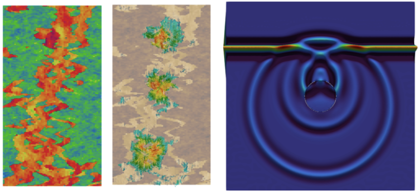
The goal of the project is to develop a software library for the finite element discretization of **coupled multiphysics problems** by employing:

discontinuous Galerkin finite element method on **polygonal** meshes,

ad-hoc models and methods to address **multiphysics interactions**,

with **several applications**, from geophysics to brain physiology.

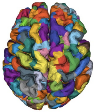
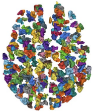
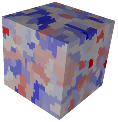
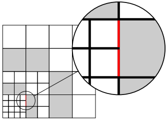




PolyDG discretization

Advantages of the PolyDG discretization [1-4]

- Support of **general polytopal** meshes (arbitrary number of edges/faces)
- Support local **mesh refinement** & **coarsening**
- **(Arbitrary) High-order** polynomials
- Robustness with respect to **heterogeneous media**
- **Scalable** and **parallel** algorithms

Use of classical **discrete spaces**: $V_h^\ell = \mathbb{P}^\ell(\mathcal{T}_h)$ and $\mathbf{V}_h^\ell = \left[\mathbb{P}^\ell(\mathcal{T}_h)\right]^d$,
where:

$$\mathbb{P}^\ell(\mathcal{T}_h) = \left\{ v_h \in L^2(\Omega) : v_h|_\kappa \in \mathbb{P}^{\ell_\kappa}(\kappa) \quad \forall \kappa \in \mathcal{T}_h \right\}$$

[1] P.F. Antonietti, S. Giani, P. Houston; *SISC* (2013)

[2] A. Cangiani, E. H. Georgoulis, P. Houston; *M3AS* (2014)

[3] A. Cangiani, Z. Dong, E. H. Georgoulis, et al.; *Springer International Publishing* (2017)

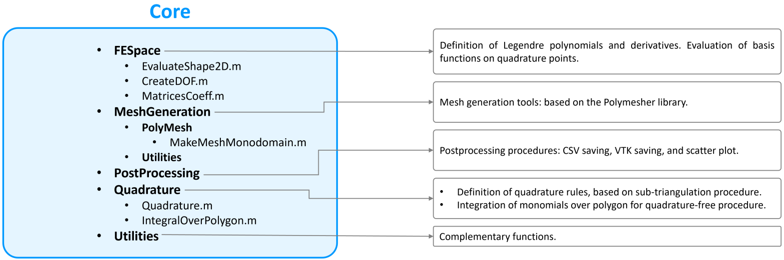
[4] P.F. Antonietti, C. Facciolà, P. Houston, et al.; *SEMA SIMAI Springer Series* (2021)

lymph

The lymph library – core

lymph: discontinuous po**LY**topal methods for **M**ulti-**PH**ysics [1]
https://lymph.bitbucket.io/.

Written in **MATLAB** for **2D** problems.
Parallelization with the **MATLAB Parallel Toolbox**.
Polymesher [2] integrated in the package for **mesh generation**.



[1] P.F. Antonietti, S. Bonetti, M. Botti, M. Corti, I.Fumagalli, I.Mazzieri.; *ACM Trans. Math. Softw.* (2025)

[2] Talischi, C., Paulino, G.H., Pereira, A. et al. *Struct Multidisc Optim* (2012)

The lymph library

lymph

Core

- **FESpace** — Definition of Legendre polynomials and derivatives. Evaluation of basis functions on quadrature points.
  - EvaluateShape2D.m
  - CreateDOF.m
  - MatricesCoeff.m
- **MeshGeneration** — Mesh generation tools: based on the Polymesher library.
  - **PolyMesh**
    - MakeMeshMonodomain.m
  - **Utilities**
- **PostProcessing** — Postprocessing procedures: CSV saving, VTK saving, and scatter plot.
- **Quadrature**
  - Definition of quadrature rules, based on sub-triangulation procedure.
  - Integration of monomials over polygon for quadrature-free procedure.
  - Quadrature.m
  - IntegralOverPolygon.m
- **Utilities** — Complementary functions.

Physics

Laplacian

$$-\nabla\cdot(\mu\nabla u) = f$$

- **Assembly**
- **InputData**
- **MainFunctions**
- **PostProcessing**
- RunSetup.m
- RunMainLaplacian.m
- ...

Heat

$$\frac{\partial u}{\partial t} - \nabla\cdot(\mu\nabla u) + \sigma u = f$$

- **Assembly**
- **InputData**
- **MainFunctions**
- **PostProcessing**
- **TimeIntegration**
- RunSetup.m
- RunMainParabolic.m
- ...

Elastodynamics

$$\frac{\partial^2 \boldsymbol{u}}{\partial t^2} - \nabla\cdot\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{f}$$

- **Assembly**
- **InputData**
- **MainFunctions**
- **PostProcessing**
- **TimeIntegration**
- RunSetup.m
- RunMainEla.m
- ...

Poroelastoacoustics

$$\begin{cases} \rho_p \dfrac{\partial^2 \boldsymbol{u}_p}{\partial t^2} + \rho_f \dfrac{\partial^2 \boldsymbol{u}_f}{\partial t^2} - \nabla\cdot\boldsymbol{\sigma}_p(\boldsymbol{u}_p,\boldsymbol{u}_f) = \boldsymbol{f}_p \\ \rho_f \dfrac{\partial^2 \boldsymbol{u}_p}{\partial t^2} + \rho_w \dfrac{\partial^2 \boldsymbol{u}_f}{\partial t^2} + \dfrac{\eta}{k}\dfrac{\partial \boldsymbol{u}_f}{\partial t} - \nabla\cdot\boldsymbol{\sigma}_f(\boldsymbol{u}_p,\boldsymbol{u}_f) = \boldsymbol{f}_f \end{cases}$$

$$c^{-2}\frac{\partial^2 \varphi_a}{\partial t^2} - \rho_a^{-1}\nabla\cdot(\rho_a\nabla\varphi_a) = f_a$$

- **Assembly**
- **InputData**
- **MainFunctions**
- **PostProcessing**
- **TimeIntegration**
- RunSetup.m
- RunMainPoroAcuEla.m
- ...

and many more physics to come in the next release!

The Poisson problem - model problem & discretization

We consider the Poisson problem in a polygonal domain $\Omega \subset \mathbb{R}^2$:

$$\begin{cases} -\nabla \cdot (\mu \nabla u)(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases}$$

PolyDG discretization

find $u_h \in V_h^\ell = \left\{ v_h \in L^2(\Omega) : v_h|_\kappa \in \mathcal{P}^{\ell_\kappa}(\kappa) \;\; \forall \kappa \in \mathcal{T}_h \right\}$ such that

$$a_{dG}(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h^\ell,$$

The Poisson problem - model problem & discretization

We consider the Poisson problem in a polygonal domain $\Omega \subset \mathbb{R}^2$:

$$\begin{cases} -\nabla \cdot (\mu \nabla u)(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases}$$

PolyDG discretization (SIPG)

find $u_h \in V_h^\ell = \left\{ v_h \in L^2(\Omega) : v_h|_\kappa \in \mathcal{P}^{\ell_\kappa}(\kappa) \;\; \forall \kappa \in \mathcal{T}_h \right\}$ such that

$$a_{dG}(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h^\ell,$$

where $\forall u, v \in V_h^\ell$ we consider:

$$\begin{aligned} a_{dG}(u,v) &= \sum_{\kappa \in \mathcal{T}_h} (\mu \nabla u, \nabla v)_\kappa - \sum_{e \in \mathcal{F}_h} \Big(\big(\{\!\{\mu \nabla u\}\!\}, [\![v]\!]\big)_e + \big([\![u]\!], \{\!\{\mu \nabla v\}\!\}\big)_e - \big(\alpha_e [\![u]\!], [\![v]\!]\big)_e \Big) \\ F(v) &= \sum_{\kappa \in \mathcal{T}_h} (f, v)_\kappa - \sum_{e \in \mathcal{F}_B} \Big((g, \mu \nabla v)_e - (\alpha_e g, v)_e \Big) \quad \forall v \in V_h^\ell. \end{aligned}$$

The Poisson problem - model problem & discretization

We consider the Poisson problem in a polygonal domain $\Omega \subset \mathbb{R}^2$:

$$\begin{cases} -\nabla \cdot (\mu \nabla u)(\mathbf{x}) = f(\mathbf{x}), & \mathbf{x} \in \Omega, \\ u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases}$$

PolyDG discretization (SIPG)

find $u_h \in V_h^\ell = \left\{v_h \in L^2(\Omega) : v_h|_\kappa \in \mathcal{P}^{\ell_\kappa}(\kappa) \;\; \forall \kappa \in \mathcal{T}_h\right\}$ such that

$$a_{dG}(u_h, v_h) = F(v_h) \quad \forall\, v_h \in V_h^\ell,$$

where $\forall u, v \in V_h^\ell$ we consider:

$$\begin{aligned} a_{dG}(u,v) &= \sum_{\kappa \in \mathcal{T}_h} (\mu \nabla u, \nabla v)_\kappa - \sum_{e \in \mathcal{F}_h} \Big((\{\!\{\mu \nabla u\}\!\}, [\![v]\!])_e + ([\![u]\!], \{\!\{\mu \nabla v\}\!\})_e - (\alpha_e [\![u]\!], [\![v]\!])_e \Big) \\ F(v) &= \sum_{\kappa \in \mathcal{T}_h} (f, v)_\kappa - \sum_{e \in \mathcal{F}_B} \Big((g, \mu \nabla v)_e - (\alpha_e g, v)_e \Big) \quad \forall v \in V_h^\ell. \end{aligned}$$

and the penalization parameter α_e (with $C_\alpha > 0$ denoting the penalty coefficient to be properly set) defined as:

$$\alpha_e(\mathbf{x}) = \begin{cases} C_\alpha \max_{\kappa \in \{\kappa^+, \kappa^-\}} \left(\mu_\kappa \frac{\ell_\kappa^2}{h_\kappa} \right), & \mathbf{x} \in e, e \in \mathcal{F}_I, e \subset \partial\kappa^+ \cap \partial\kappa^-, \\ C_\alpha \mu_\kappa \frac{\ell_\kappa^2}{h_\kappa}, & \mathbf{x} \in e, e \in \mathcal{F}_B, e \subset \partial\kappa^+ \cap \partial\Omega, \end{cases}$$

The Poisson problem in the unit square - verification test

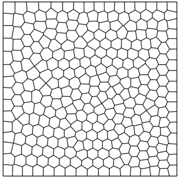


Figure: $2D$ Voronoi polygonal mesh of dimension $N = 300$.

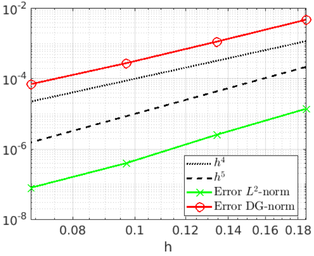
Problem data

$$u_{\text{ex}}(\mathbf{x}) = \sin(2\pi x)\cos(2\pi y)$$

$$\mu(\mathbf{x}) = 1$$

$$f(\mathbf{x}) = 8\pi^2 \sin(2\pi x)\cos(2\pi y)$$

$$g(\mathbf{x}) = \sin(2\pi x)\cos(2\pi y)$$



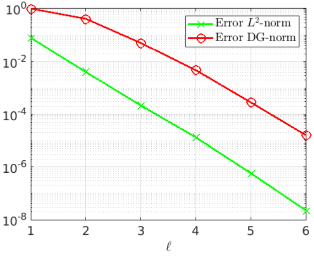


Figure: Left: computed errors dG- and L^2- errors as a function of the mesh size h by fixing the polynomial degree $\ell = 4$. Right: computed dG- and L^2- errors as a function of the polynomial degree $\ell = 4$ by fixing the number of mesh element $N_{el} = 100$.

The Poisson problem - performances

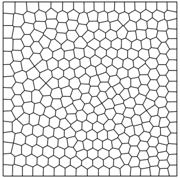


Figure: $2D$ Voronoi polygonal mesh of dimension $N = 300$.

Problem data

$$u_{\text{ex}}(\mathbf{x}) = \sin(2\pi x)\cos(2\pi y)$$

$$\mu(\mathbf{x}) = 1$$

$$f(\mathbf{x}) = 8\pi^2 \sin(2\pi x)\cos(2\pi y)$$

$$g(\mathbf{x}) = \sin(2\pi x)\cos(2\pi y)$$

To assess the performances of the method, we consider the Poisson problem on a Cartesian grid with $N = 19600$, $\ell = 5$ leading to 411.600 degrees of freedom.

Mat. assembly (QF)	Mat. assembly (ST)	RHS assembly	Linear system	`.csv`-file saving	`.vtk`-file saving
25.6500 s	31.7024 s	5.8172 s	24.7120 s	9.3620 s	13.2213 s

Table: Computational times for a test case with 411600 `dofs`. We compare the quadrature-free (QF) and subtriangulation (ST) strategies for the numerical evaluation of integrals during matrix assembly.

The numerical simulations have been performed as a *serial* job, using the `Kami` cluster (40 computing nodes configured as follows: **CPU** 2x AMD EPYC 7413 24-Core Processor, **RAM** 512Gb) at the Department of Mathematics, Politecnico di Milano.

The Poisson problem - advantages of polygonal meshes

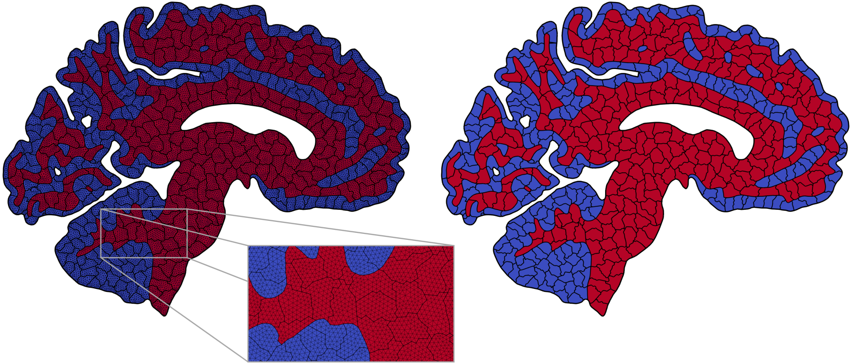


Figure: Triangular mesh of brain section made of 42891 triangles (left), agglomerated polygonal mesh made of 534 polygons (right) with a reduction factor of ≈ 80. The cells belonging to the white zone are colored red, while those belonging to the gray zone are colored blue.

P.F. Antonietti, M. Corti.; *Proc. ENUMATH 2023* (2025)

The Poisson problem - advantages of polygonal meshes

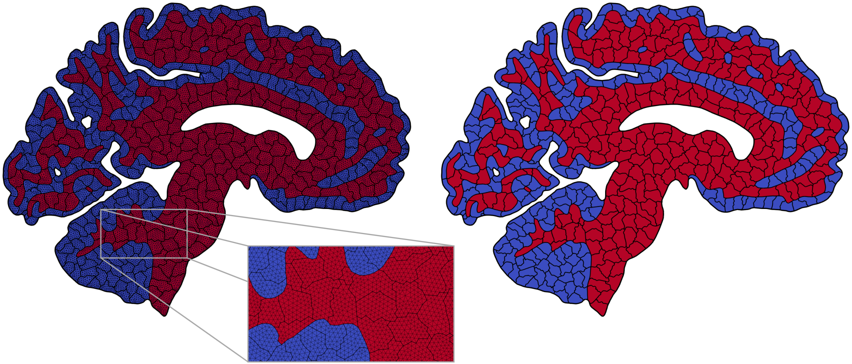


Figure: Triangular mesh of brain section made of 42891 triangles (left), agglomerated polygonal mesh made of 534 polygons (right). The cells belonging to the white zone are colored red, while those belonging to the gray zone are colored blue.

ADV: `MAGNET` [1] is a python package **intregrated with `lymph`** for mesh agglomeration. It includes state of the art methods (e.g. Metis and k-means), and methods exploiting DL and GNNs.

[1] Antonietti, P.F., Caldana, M., Mazzieri, I. et al. *Engineering with Computers* (2025).

The Poisson problem - advantages of polygonal meshes

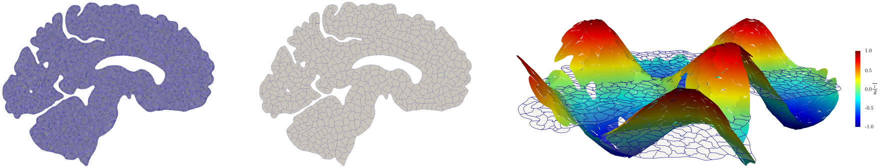


Figure: Triangular mesh of brain section made of 42891 triangles (left), agglomerated polygonal mesh made of 534 polygons (center), and computed PolyDG solution on the polygonal grid with $\ell = 1$ (right).

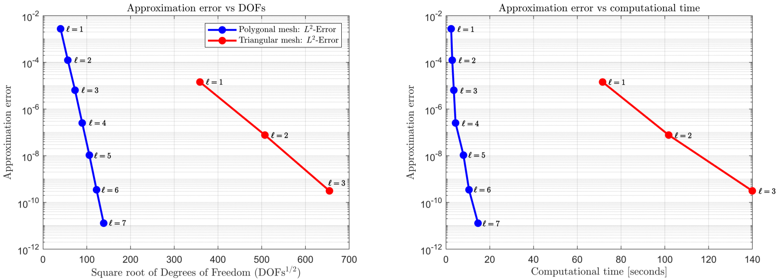
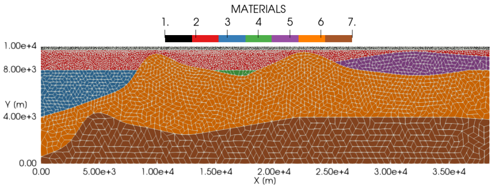


Figure: Computed approximation errors in the L^2-norm on the triangular (red) and agglomerated (blue) meshes. Computed errors versus the total number of degrees of freedom (left), and versus the computational time (right).

Applications

Elastodynamics



$$\rho \frac{\partial^2 \boldsymbol{u}}{\partial t^2} - \nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{f}, \quad \text{in } \Omega \times (0, T],$$

Figure: Unstructured polygonal grid with mesh spacing of about $h \approx 160$ m for material 1 to $h \approx 1500$ m for material 7.

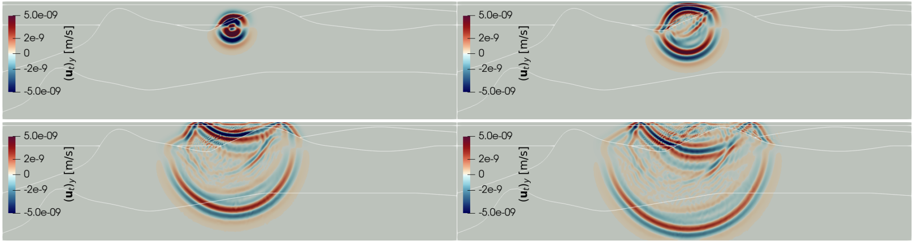


Figure: Snapshots of the computed vertical velocity $(\mathbf{u}_t)_y$ at different times $t = 0.75$ (top-left), $t = 1.25$ (top-right), $t = 2.25$ (bottom-left), $t = 2.75$ (bottom-right).

Thermo-poroelasticity

$$\begin{cases} \rho\ddot{\mathbf{u}} + \rho_f\ddot{\mathbf{w}} - \nabla\cdot\boldsymbol{\sigma} = \tilde{\mathbf{f}} & \text{in } \Omega \times (0, T_f], \\ \rho_f\ddot{\mathbf{u}} + \rho_w\ddot{\mathbf{w}} + \mathbf{K}^{-1}\dot{\mathbf{w}} + \nabla p = \tilde{\mathbf{g}} & \text{in } \Omega \times (0, T_f], \\ c_0\dot{p} - b_0\dot{T} + \alpha\nabla\cdot\dot{\mathbf{u}} + \nabla\cdot\dot{\mathbf{w}} = 0 & \text{in } \Omega \times (0, T_f], \\ a_0\left(\dot{T} + \tau\ddot{T}\right) - b_0\left(\dot{p} + \tau\ddot{p}\right) + \beta\left(\nabla\cdot\dot{\mathbf{u}} + \tau\nabla\cdot\ddot{\mathbf{u}}\right) - \nabla\cdot(\boldsymbol{\Theta}\nabla T) = H & \text{in } \Omega \times (0, T_f], \end{cases}$$

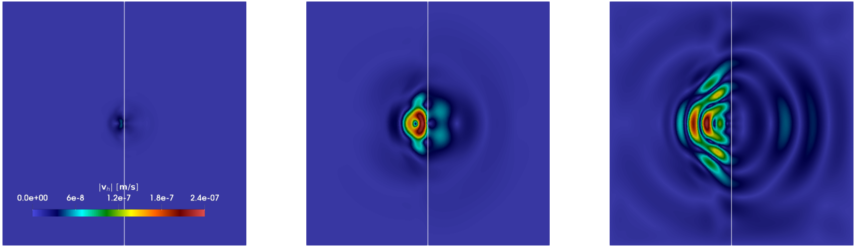


Figure: Computed magnitude of the velocity field $\|\dot{\mathbf{u}}_h\|$ at the time instants $t = 0.1s$ (left), $t = 0.3s$ (center), $t = 0.5s$ (right)

S. Bonetti, M. Botti, I. Mazzieri, et al; *JCP* (2023)

Poro-viscoelasticity

$$\begin{cases} \rho\ddot{\mathbf{u}} - \nabla\cdot(2\mu(\boldsymbol{\epsilon}(\mathbf{u}) + \delta_1\boldsymbol{\epsilon}(\dot{\mathbf{u}}))) - \nabla(\lambda(\nabla\cdot\mathbf{u} + \delta_2\nabla\cdot\dot{\mathbf{u}})) + \nabla(\gamma p) = \mathbf{f} & \text{in } \Omega\times(0,T_f], \\ d_0\dot{p} + \gamma\nabla\cdot\dot{\mathbf{u}} - \nabla\cdot(\mathbf{D}\nabla p) = g & \text{in } \Omega\times(0,T_f]. \end{cases}$$

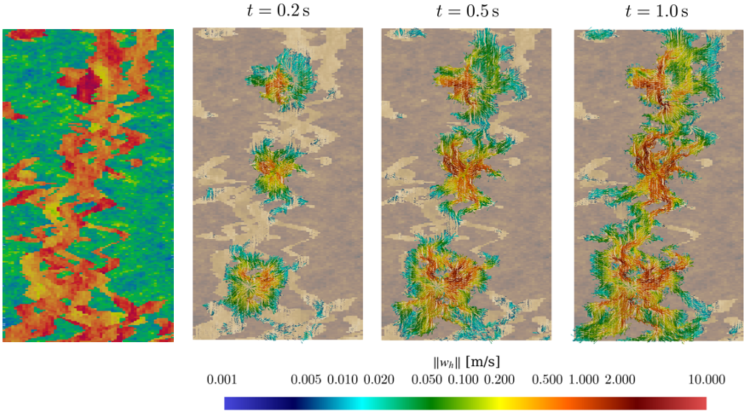
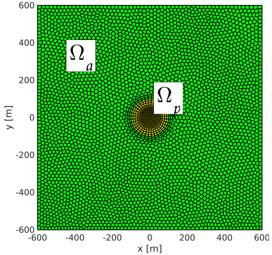


Figure: Permeability field (SPE10 dataset) and computed filtration velocity field $\|D\nabla p_h\|$ at the time instants $t = 0.2$ s, $t = 0.5$ s, and $t = 1.0$ s. Glyphs are not present wherever the flow is absent.

Coupled poroelasto-acoustic wave propagation



$$
\begin{aligned}
\rho\ddot{\boldsymbol{u}} + \rho_f\ddot{\boldsymbol{w}} - \nabla\cdot(\mathbb{C}:\boldsymbol{\epsilon}(\boldsymbol{u})) - \beta m\nabla(\beta\nabla\cdot\boldsymbol{u} + \nabla\cdot\boldsymbol{w}) &= \boldsymbol{f}_p && \text{in } \Omega_p,\\
\rho_f\ddot{\boldsymbol{u}} + \rho_w\ddot{\boldsymbol{w}} + \frac{\eta}{k}\dot{\boldsymbol{w}} - m\nabla(\beta\nabla\cdot\boldsymbol{u} + \nabla\cdot\boldsymbol{w}) &= \boldsymbol{g}_p && \text{in } \Omega_p,\\
\rho_a c^{-2}\ddot{\varphi} - \nabla\cdot(\rho_a\nabla\varphi) &= \rho_a f_a && \text{in } \Omega_a,\\
-(\mathbb{C}:\boldsymbol{\epsilon}(\boldsymbol{u}) + \beta m(\beta\nabla\cdot\boldsymbol{u} + \nabla\cdot\boldsymbol{w})\boldsymbol{I})\boldsymbol{n}_p &= \rho_a\dot{\varphi}\boldsymbol{n}_p && \text{on } \Gamma_I,\\
(\dot{\boldsymbol{u}} + \dot{\boldsymbol{w}})\cdot\boldsymbol{n}_p &= -\nabla\varphi\cdot\boldsymbol{n}_p && \text{on } \Gamma_I,\\
-m(\beta\nabla\cdot\boldsymbol{u} + \nabla\cdot\boldsymbol{w}) - \tau^{-1}(1-\tau)\dot{\boldsymbol{w}}\cdot\boldsymbol{n}_p &= \rho_a\dot{\varphi} && \text{on } \Gamma_I,
\end{aligned}
$$

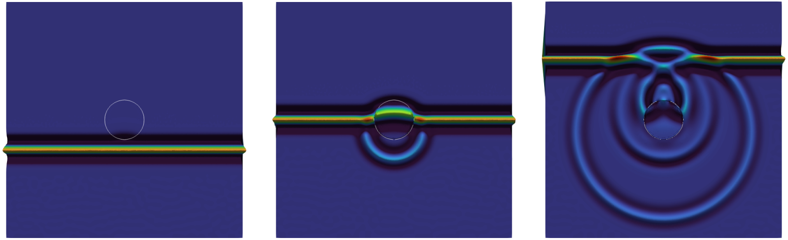


Figure: Circular porous domain Ω_p (yellow) surrounded by an acoustic medium Ω_a (green) (top-left). Snapshots of the computed pressure field at different time instants: $t = 0.4\ s$ (top-right), $t = 0.5\ s$ (bottom-left), and $t = 0.7\ s$ (bottom-right).

Neurodegenerative diseases (Fisher-Kolmogorov equation)

$$
\begin{aligned}
\partial_t c - \nabla \cdot (\mathbf{D}\nabla c) = \alpha\, c(1-c) =: f(c) \quad & \text{in } Q_T, \\
(\mathbf{D}\nabla c) \cdot \boldsymbol{n}_\Omega = \mathbf{0} \quad & \text{on } \Gamma \times (0,T),
\end{aligned}
$$

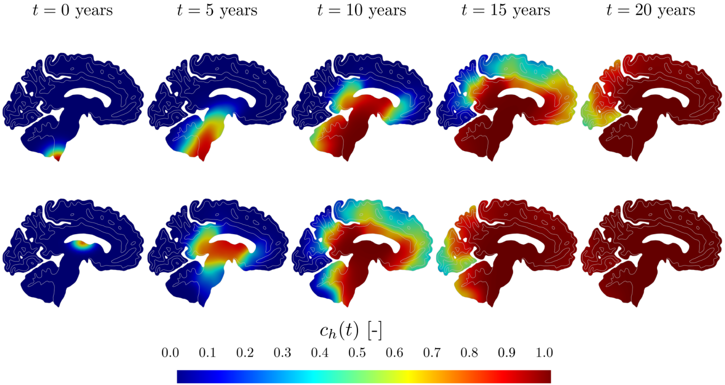


Figure: Patterns of α-synuclein concentration at different stages of the pathology with brainstem predominancy (top) and limbic predominancy (bottom)

P.F. Antonietti, M. Corti, S. Gomez, et al.; *MATCOM* (2026)

Coupled Stokes-multinetwork poroelasticity

$$\begin{cases} \rho_{\mathrm{el}} \partial_{tt}^2 \boldsymbol{d} - \nabla \cdot \sigma_{\mathrm{el}}(\boldsymbol{d}) + \sum_{\mathrm{k} \in J} \alpha_{\mathrm{k}} \nabla p_{\mathrm{k}} = \boldsymbol{f}_{\mathrm{el}}, & \text{in } \Omega_{\mathrm{el}} \times (0,T], \\ c_{\mathrm{j}} \partial_t p_{\mathrm{j}} + \nabla \cdot \left(\alpha_{\mathrm{j}} \partial_t \boldsymbol{d} - \frac{1}{\mu_{\mathrm{j}}} K_{\mathrm{j}} \nabla p_{\mathrm{j}} \right) & \\ \qquad + \sum_{\mathrm{k} \in J} \beta_{\mathrm{jk}} (p_{\mathrm{j}} - p_{\mathrm{k}}) + \beta_{\mathrm{j}}^{\mathrm{e}} p_{\mathrm{j}} = g_{\mathrm{j}}, & \text{in } \Omega_{\mathrm{el}} \times (0,T], \\ \rho_{\mathrm{f}} \partial_t \boldsymbol{u} - \nabla \cdot \sigma_{\mathrm{f}}(\boldsymbol{u}) + \nabla p = \boldsymbol{f}_{\mathrm{f}}, & \text{in } \Omega_{\mathrm{f}} \times (0,T], \\ \nabla \cdot \boldsymbol{u} = 0, & \text{in } \Omega_{\mathrm{f}} \times (0,T], \end{cases}$$

$$\begin{cases} \sigma_{\mathrm{el}}(\boldsymbol{d}) \boldsymbol{n}_{\mathrm{el}} - \sum_{\mathrm{k} \in J} \alpha_{\mathrm{k}} p_{\mathrm{k}} \boldsymbol{n}_{\mathrm{el}} + \sigma_{\mathrm{f}}(\boldsymbol{u}) \boldsymbol{n}_{\mathrm{f}} - p \boldsymbol{n}_{\mathrm{f}} = \boldsymbol{0}, & \text{on } \Sigma \times (0,T], \\ \frac{1}{\mu_{\mathrm{j}}} K_{\mathrm{j}} \nabla p_{\mathrm{j}} \cdot \boldsymbol{n}_{\mathrm{el}} = 0, & \text{on } \Sigma \times (0,T], \\ \boldsymbol{u} \cdot \boldsymbol{n}_{\mathrm{f}} + \left(\partial_t \boldsymbol{d} - \frac{1}{\mu_{\mathrm{E}}} K_{\mathrm{E}} \nabla p_{\mathrm{E}} \right) \cdot \boldsymbol{n}_{\mathrm{el}} = 0, & \text{on } \Sigma \times (0,T], \\ p_{\mathrm{E}} = p - \sigma_{\mathrm{f}}(\boldsymbol{u}) \boldsymbol{n}_{\mathrm{f}} \cdot \boldsymbol{n}_{\mathrm{f}}, & \text{on } \Sigma \times (0,T], \\ (\sigma_{\mathrm{f}}(\boldsymbol{u}) \boldsymbol{n}_{\mathrm{f}} - p \boldsymbol{n}_{\mathrm{f}}) \wedge \boldsymbol{n}_{\mathrm{f}} = \boldsymbol{0}, & \text{on } \Sigma \times (0,T]. \end{cases}$$

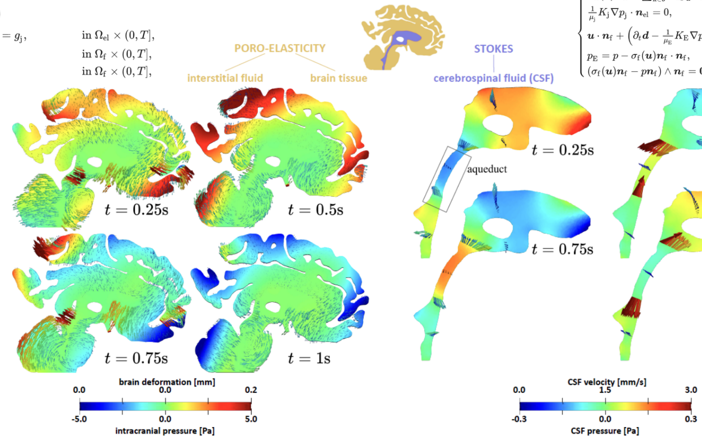


Figure: Discrete tissue displacements and pressures in the whole domain (left), discrete velocity and pressure in the brain ventricles (right).

I. Fumagalli, M. Corti, N. Parolini, et al.; *JCP* (2024)

Coupled fluid-structure interaction

Biot equation

$$
\begin{cases}
\rho\ddot{\boldsymbol{u}}_p + \rho_f\ddot{\boldsymbol{w}}_p - \nabla\cdot\boldsymbol{\sigma}_p = \boldsymbol{f}_p, & in\ \Omega_p \times (0,T],\\
\rho_f\ddot{\boldsymbol{u}}_p + \rho_w\ddot{\boldsymbol{w}}_p + \frac{\eta}{\kappa}\dot{\boldsymbol{w}}_p + \nabla p_p = \boldsymbol{g}_p, & in\ \Omega_p \times (0,T],\\
\boldsymbol{u}_p = \boldsymbol{0}, & on\ \Gamma_p^D \times (0,T],\\
\boldsymbol{w}_p\cdot\boldsymbol{n}_p = 0, & on\ \Gamma_p^D \times (0,T],\\
\boldsymbol{\sigma}_p\boldsymbol{n}_p = \boldsymbol{g}_p^N, & on\ \Gamma_p^N \times (0,T],\\
p_p = q_p^N, & on\ \Gamma_p^N \times (0,T],\\
\boldsymbol{u}_p = \boldsymbol{u}_{p0}, \quad \dot{\boldsymbol{u}}_p = \boldsymbol{v}_{p0}, & in\ \Omega_p \times \{0\},\\
\boldsymbol{w}_p = \boldsymbol{w}_{p0}, \quad \dot{\boldsymbol{w}}_p = \boldsymbol{z}_{p0}, & in\ \Omega_p \times \{0\},
\end{cases}
$$

Stokes equation

$$
\begin{cases}
(2\mu_f)^{-1}\mathrm{dev}(\dot{\boldsymbol{\Sigma}}_f) - \nabla(\rho_f^{-1}\nabla\cdot\boldsymbol{\Sigma}_f) + \boldsymbol{r}_f = \boldsymbol{F}_f, & in\ \Omega_f \times (0,T],\\
\mathrm{skew}(\dot{\boldsymbol{\Sigma}}_f) = \boldsymbol{0}, & in\ \Omega_f \times (0,T],\\
\rho_f^{-1}\nabla\cdot\boldsymbol{\Sigma}_f = \boldsymbol{G}_f, & on\ \Gamma_f^D \times (0,T],\\
\boldsymbol{\Sigma}_f\boldsymbol{n}_f = \int_0^t \boldsymbol{g}_f^N(s)\,ds, & on\ \Gamma_f^N \times (0,T],\\
\boldsymbol{\Sigma}_f = \boldsymbol{0}, & in\ \Omega_f \times \{0\},
\end{cases}
$$

Coupling conditions

$$
\begin{cases}
(\alpha\dot{\boldsymbol{u}}_p + \dot{\boldsymbol{w}}_p)\cdot\boldsymbol{n}_p = \boldsymbol{u}_f\cdot\boldsymbol{n}_p, \text{(flux conservation)},\\
\boldsymbol{\Sigma}_f\boldsymbol{n}_p\cdot\boldsymbol{n}_p = \gamma\dot{\boldsymbol{w}}_p\cdot\boldsymbol{n}_p - p_p, \text{(Robin condition)},\\
\alpha\dot{\boldsymbol{\Sigma}}_f\boldsymbol{n}_p\cdot\boldsymbol{n}_p = \boldsymbol{\sigma}_p\boldsymbol{n}_p\cdot\boldsymbol{n}_p, \text{(normal stress conservation)},\\
\dot{\boldsymbol{\Sigma}}_f\boldsymbol{n}_p\wedge\boldsymbol{n}_p = \boldsymbol{\sigma}_p\boldsymbol{n}_p\wedge\boldsymbol{n}_p = \delta(\boldsymbol{u}_f - \dot{\boldsymbol{u}}_p)\wedge\boldsymbol{n}_p, \text{(BJS condition)}.
\end{cases}
$$

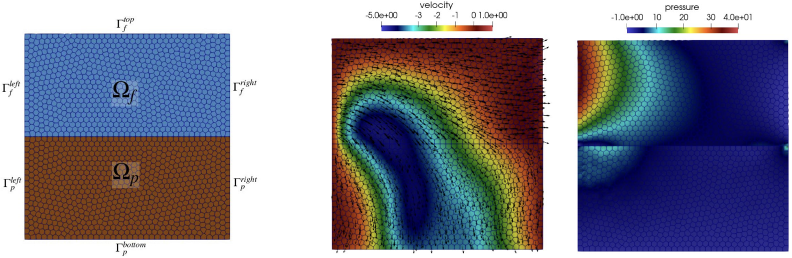
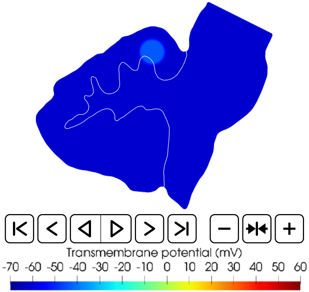


Figure: Computed solutions at final time $T = 1.5s$. Left: velocities $\dot{\mathbf{u}}_f$ and $\dot{\mathbf{u}}_p + \dot{\mathbf{w}}_p$ (arrows), $\mathbf{u}_{f,2}$ and $\dot{\mathbf{u}}_{p,2} + \dot{\mathbf{w}}_{p,2}$ (color). Right: computed pressures p_f and p_p

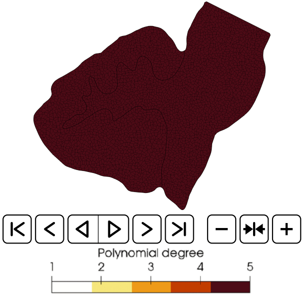
M. Botti , I. Fumagalli, I. Mazzieri.; *Engineering with Computers.* (2025)

Brain electrophysiology

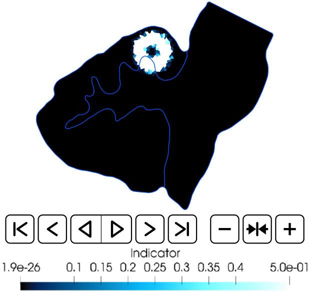
$$
\begin{cases}
\chi_m C_m \dfrac{\partial u}{\partial t} - \nabla \cdot (\boldsymbol{\Sigma} \nabla u) + \chi_m f(u, \boldsymbol{y}) = I_{\text{ext}} & \text{in } \Omega \times (0, T], \\
\dfrac{\partial \boldsymbol{y}}{\partial t} + \boldsymbol{m}(u, \boldsymbol{y}) = \boldsymbol{0} & \text{in } \Omega \times (0, T], \\
\boldsymbol{\Sigma} \nabla u \cdot \boldsymbol{n} = 0 & \text{on } \partial\Omega \times (0, T], \\
u(0) = u^0, \ \boldsymbol{y}(0) = \boldsymbol{y}^0 & \text{in } \Omega.
\end{cases}
$$



Transmembrane potential.



Polynomial degree.



Local error indicator.

Exploiting **p-adaptivity**, soon to be released!

C.B. Leimer Saglio, S. Pagani, P.F. Antonietti; *CMAME* (2025)

Vulpes

The vulpes library

vulpes: Discontinuous and **V**irt**U**aL **P**olytopal **E**lement**S** method
https://vulpeslib.github.io/.

Written in **C++** for **2D** and **3D** problems.
Parallelization with the **MPI**
Polymesher and **MAGNET** [1] integrated in the package for **mesh generation**

- **Mesh Generation**: Triangle, TetGen, Voronoi
- **Linear Algebra**: Eigen, Petsc
- **Postprocessing**: VTK
- **Testing**: Googletest

[1] Antonietti, P.F., Caldana, M., Mazzieri, I. et al. Engineering with Computers (2025).

Vulpes: linear elasticity problem

Find the displacement field $\boldsymbol{u} : \Omega \to \mathbb{R}^d$ such that

$$-\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{f} \quad \text{in } \Omega, \quad \boldsymbol{u} = \boldsymbol{0} \quad \text{on } \partial\Omega,$$

where $\boldsymbol{f} : \Omega \to \mathbb{R}^d$ is the body force, $\boldsymbol{\sigma}$ is the stress tensor (Hooke's law) and $\boldsymbol{\epsilon}$ is the strain tensor (symmetric gradient).

Weak form: find $\boldsymbol{u}_h \in \boldsymbol{V}_h^\ell$ s.t. for any $\boldsymbol{v} \in \boldsymbol{V}_h^\ell$ it holds

$$\sum_{K \in \mathcal{T}_h} (\boldsymbol{\sigma}(\boldsymbol{u}), \boldsymbol{\epsilon}(\boldsymbol{v}))_K - \sum_{F \in \mathcal{F}_h} \langle \{\boldsymbol{\sigma}(\boldsymbol{u})\}, [\boldsymbol{v}] \rangle_F - \sum_{F \in \mathcal{F}_h} \langle \{\boldsymbol{\sigma}(\boldsymbol{v})\}, [\boldsymbol{u}] \rangle_F + \sum_{F \in \mathcal{F}_h} \langle \eta [\boldsymbol{u}], [\boldsymbol{v}] \rangle_F = \sum_{K \in \mathcal{T}_h} (\boldsymbol{f}, \boldsymbol{v})_K.$$

Vulpes: linear elasticity problem (implementation)

Weak form: find $\boldsymbol{u}_h \in \boldsymbol{V}_h^\ell$ s.t. for any $\boldsymbol{v} \in \boldsymbol{V}_h^\ell$ it holds

$$\sum_{K\in\mathcal{T}_h} (\boldsymbol{\sigma}(\boldsymbol{u}), \boldsymbol{\epsilon}(\boldsymbol{v}))_K - \sum_{F\in\mathcal{F}_h} \langle\{\boldsymbol{\sigma}(\boldsymbol{u})\}, [\boldsymbol{v}]\rangle_F - \sum_{F\in\mathcal{F}_h} \langle\{\boldsymbol{\sigma}(\boldsymbol{v})\}, [\boldsymbol{u}]\rangle_F + \sum_{F\in\mathcal{F}_h} \langle\eta[\boldsymbol{u}], [\boldsymbol{v}]\rangle_F = \sum_{K\in\mathcal{T}_h} (\boldsymbol{f}, \boldsymbol{v})_K.$$

```
auto wf_stiff = twice_mu * fe::inner(fe::symgrad(u), fe::symgrad(v)) +
                lambda * (fe::trsymgrad(u) * fe::trsymgrad(v));
```

```
auto n = fe::normal(fe_space);
auto wf_IT = bc * (twice_mu * fe::inner(fe::symgrad(u), fe::outer(v, n)) +
                   lambda * (fe::trsymgrad(u) * fe::inner(v, n)));
auto wf_ITn =
    half * (twice_mu * fe::inner(fe::symgrad(u), fe::outer(fe::neigh(v), n)) +
            lambda * (fe::trsymgrad(u) * fe::inner(fe::neigh(v), n)));
```

```
auto wf_S = gamma * (lambda_plus_twice_mu * fe::inner(u, v));
auto wf_Sn = -(gamma * (lambda_plus_twice_mu * fe::inner(u, fe::neigh(v))));
```

```
auto wf_F = fe::inner(force, v);
```

Vulpes: linear elasticity problem (implementation)

Weak form: find $\boldsymbol{u}_h \in \boldsymbol{V}_h^\ell$ s.t. for any $\boldsymbol{v} \in \boldsymbol{V}_h^\ell$ it holds

$$\sum_{K\in\mathcal{T}_h} (\boldsymbol{\sigma}(\boldsymbol{u}), \boldsymbol{\epsilon}(\boldsymbol{v}))_K - \sum_{F\in\mathcal{F}_h} \langle\{\boldsymbol{\sigma}(\boldsymbol{u})\}, [\boldsymbol{v}]\rangle_F - \sum_{F\in\mathcal{F}_h} \langle\{\boldsymbol{\sigma}(\boldsymbol{v})\}, [\boldsymbol{u}]\rangle_F + \sum_{F\in\mathcal{F}_h} \langle\eta[\boldsymbol{u}], [\boldsymbol{v}]\rangle_F = \sum_{K\in\mathcal{T}_h} (\boldsymbol{f}, \boldsymbol{v})_K.$$

```
auto wf_stiff = twice_mu * fe::inner(fe::symgrad(u), fe::symgrad(v)) +
                lambda * (fe::trsymgrad(u) * fe::trsymgrad(v));
// Local computation over K for subtessellation
const auto Aloc = int_simplexify.getIntegral(&wf_stiff);
// Local computation over K for quadrature free
const auto Alocq = int_qfree.getIntegral(&wf_stiff);
...
// Assembly
system_matrix.insertValuesBlocked(Aloc.data(), ii, ii, ndof_comp);
...
// Solution -- wrapper for petsc
{ la::LinearSolver solver(
    la::LinearSolver::PrecondOnly, la::LinearSolver::LU);
 solver.setOperators(system_matrix);
 solver.solve(force_vec, solution);}
```

sub-tessellation

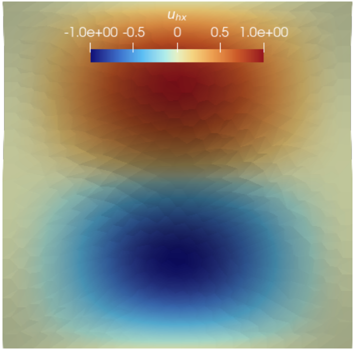
quadrature free

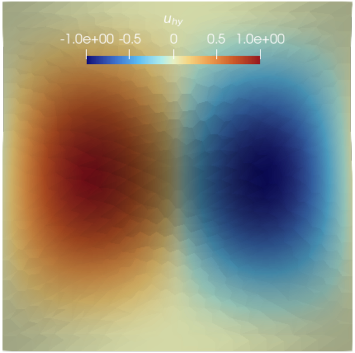
Vulpes: linear elasticity problem (output)

Test case in $\Omega = (0,1)^2$ with $\lambda = 2$ and μ and exact solution

$$\boldsymbol{u}(\boldsymbol{x},t) = [-\sin(\pi x)^2 \sin(2\pi y), \sin(\pi y)^2 \sin(2\pi x)]^T.$$

Polynomial degree $p = 5$ and $N_{el} = 512$.





Other **physics** already implemented: Poisson's problem, heat equation, Fisher-Kolmogorov equation.

Conclusions & further developments

Main contributions:

- we developed two libraries that implement DG discretization on polytopal meshes;
- we **documented** them in a very structured manner with a focus on **user-friendliness** (tutorials, guides, etc.);
- we included different quadrature strategies and post-processing routines
- we proposed models and **Poly-DG discretizations** of different multi-physics problems motivated by the fields of application.

Conclusions & further developments

Main contributions:

- we developed two libraries that implement DG discretization on polytopal meshes;
- we **documented** them in a very structured manner with a focus on **user-friendliness** (tutorials, guides, etc.);
- we included different quadrature strategies and post-processing routines
- we proposed models and **Poly-DG discretizations** of different multi-physics problems motivated by the fields of application.

Ongoing works:

- Add/Enhancement of the coupling between different physics
- Optimization/Improvement of parallel computation;
- Development of effective solution strategies and preconditioning techniques.

Thank you for the attention!

References

P.F. Antonietti, **S. Bonetti**, M. Botti, M. Corti, I. Fumagalli, and I. Mazzieri. `lymph`: discontinuous poLYtopal methods for Multi-PHysics differential problems. *ACM Transactions on Mathematical Software*, 51 (1), 2025.

P.F. Antonietti, M. Caldana, I. Mazzieri, and A.Re Fraschini. MAGNET: an open-source library for mesh agglomeration by Graph Neural Networks. *arXiv*, 2025.

This research and the partecipation to this conference have been funded by the **European Union (ERC, NEMESIS, project number 101115663)**. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Research Council Executive Agency.

The present research is part of the activities of the project **Dipartimento di Eccellenza 2023-2027**, Dipartimento di Matematica, Politecnico di Milano.

NEMESIS
New generation methods for numerical simulations

Funded by the European Union

European Research Council
Established by the European Commission