Numerical Methods for Partial Differential Equations
A.Y. 2025/2026

Project topics and general instructions

The present document contains the instructions for delivering the project results and a list of possible project topics. Students who want to propose alternative topics are welcome, but these topics should be discussed with the course teachers.

General instructions

1. Students should work in groups of 2 to 4 people. Groups of at least 3 students are strongly recommended due to the complexity of the projects. Students are free to form the groups as they choose.
2. Students should communicate via e-mail to `michele.bucelli@polimi.it` their choice of group and project topic, as soon as possible.
3. All the projects require the implementation of finite element solvers for problems that were not discussed during the course.
4. Project descriptions are deliberately lacking in detail. Students are encouraged to look for additional information in the scientific literature (including, but not limited to, the works cited in the project descriptions).
5. Students are encouraged to take parallel computing aspects into account when working on the project. Upon request, the teachers will grant access to the HPC resources of the mathematics department for parallel scalability tests.
6. The source code of a project should be uploaded to a GitHub repository obtained by forking the repository at the following URL: `https://github.com/michelebucelli/nmpde-projects`. Please read carefully the `README.md` file of that repository.
7. The delivery of the code should be done by creating a pull request into the original repository. The pull request page will be used by the course teachers to review the code and suggest changes or improvements.
8. The project report and presentations should demonstrate a clear understanding of the mathematical problem, of its weak and discrete formulation, and of the algorithmic and computational aspects of the solver.
9. The report should be delivered via e-mail to `michele.bucelli@polimi.it`. Students are not required to deliver the slides prior to the day of the presentation.

Project 1. Preconditioning heterogeneous diffusion problems

Consider the following 3D Poisson problem (1), in which the diffusion coefficient varies significantly (of orders of magnitude) over the domain:

$$\begin{cases} -\boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} u) = f & \text{in } \Omega \; , \\ u = 0 & \text{on } \partial\Omega \; , \end{cases} \tag{1}$$

$$\mu(\mathbf{x}) = \begin{cases} 10^p & \text{if } \mathbf{x} \in B \; , \\ 1 & \text{otherwise} \; , \end{cases} \tag{2}$$

where $p > 0$ and $B = \bigcup_{i=1}^{N} B_i$, with B_i a sphere of center $\mathbf{x}_i$ and radius r_i.

Implement a solver for the problem, and discuss the effectiveness of different preconditioning strategies depending on the heterogeneity. Consider the preconditioners that are offered by `deal.II` within the `TrilinosWrappers` namespace.

Discuss the computational efficiency, optimality and parallel scalability of the different preconditioners under consideration.

Project 2. Solving the wave equation

Consider the wave equation in 2D [8, 10]:

$$\begin{cases} \dfrac{\partial^2 u}{\partial t^2} - \Delta u = f & \text{in } \Omega \,, \\ u = g & \text{on } \partial\Omega \,, \\ u(t=0) = u_0 & \text{in } \Omega \,, \\ \dfrac{\partial u}{\partial t} = u_1 & \text{in } \Omega \,. \end{cases} \tag{3}$$

Implement a finite element solver for problem (3). Discuss the choice of the time and space discretization methods, the properties of the chosen method (especially in terms of numerical dissipation and dispersion, see [8, 10]) and the computational and algorithmic aspects of the solver.

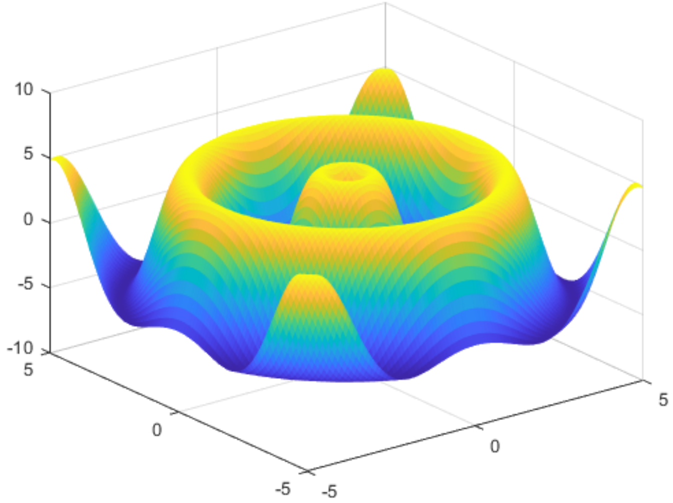

Figure 1: Example solution to the wave equation.

Project 3. Navier-Stokes equations

$$\begin{cases} \dfrac{\partial \mathbf{u}}{\partial t} + (\mathbf{u}\cdot\boldsymbol{\nabla})\mathbf{u} - \nu\Delta\mathbf{u} + \boldsymbol{\nabla} p = f & \text{in } \Omega\,, \\ \boldsymbol{\nabla}\cdot\mathbf{u} = 0 & \text{in } \Omega\,, \\ \mathbf{u} = \mathbf{g} & \text{on } \Gamma_{\mathrm{D}} \subset \partial\Omega\,, \\ \nu\boldsymbol{\nabla}\mathbf{u}\,\mathbf{n} - p\mathbf{n} = \mathbf{h} & \text{on } \Gamma_{\mathrm{N}} = \partial\Omega\backslash\Gamma_{\mathrm{D}}\,, \\ \mathbf{u}(t=0) = \mathbf{u}_0 & \text{in } \Omega\,. \end{cases} \tag{4}$$

Solve, by means of the finite element method, the unsteady, incompressible Navier-Stokes equations (4) to simulate the 2D or 3D benchmark problem “flow past a cylinder” (described in detail in [11]) for different values of the Reynolds number $\mathbb{R}\mathrm{e} \leq 200$.

Compute and plot the lift and drag coefficients over time for different values of $\mathbb{R}\mathrm{e}$. The coefficients are defined as follows:

$$C_{\mathrm{D}} = \frac{F_{\mathrm{D}}}{U^2 L} \qquad C_{\mathrm{L}} = \frac{F_{\mathrm{L}}}{U^2 L}\,, \tag{5}$$

where F_{D} and F_{L} denote the force acting on the cylindrical obstacle in the directions parallel and perpendicular to the fluid, respectively, U is the reference flow velocity and L is the cross-sectional length of the obstacle.

Discuss the results, the methods employed, their stability and accuracy, and their algorithmic and computational aspects. Refer to [4] for a discussion on possible preconditioners for this problem.

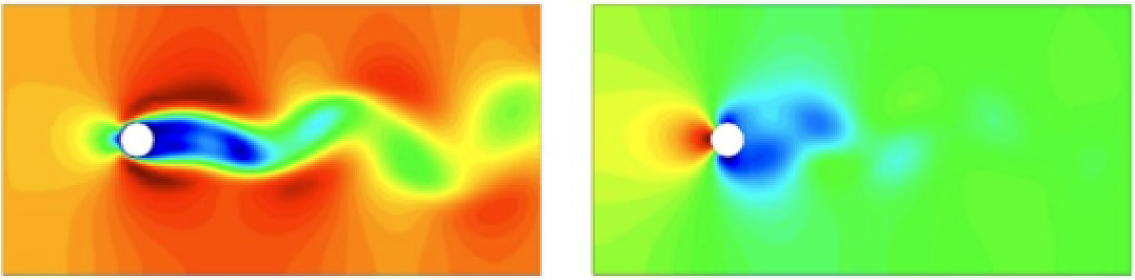

Figure 2: Example solution to Navier-Stokes equations. Image taken from [1].

Project 4. Cardiac electrophysiology

$$\begin{cases} \dfrac{\partial v}{\partial t} - \boldsymbol{\nabla} \cdot (\Sigma \boldsymbol{\nabla} v) + I_{\text{ion}}(v, \mathbf{w}) = I_{\text{app}} & \text{in } \Omega \, , \\ \Sigma \boldsymbol{\nabla} v \, \mathbf{n} = 0 & \text{on } \partial\Omega \, , \\ v(t=0) = v_0 & \text{in } \Omega \, , \\ \dfrac{\mathrm{d}\mathbf{w}}{\mathrm{d}t} = \mathbf{F}_{\text{BO}}(v, \mathbf{w}) & \text{in } \Omega \, , \\ \mathbf{w}(t=0) = \mathbf{w}_0 & \text{in } \Omega \, . \end{cases} \tag{6}$$

Solve, by means of the finite element method, the system (6), consisting of the monodomain equation [6] coupled to the Bueno-Orovio ionic model (refer to [3] for the definition of $\mathbf{F}_{\text{BO}}$). Notice that, since the ionic model is expressed as a system of ODEs (i.e. it involves no spatial derivatives), the coupling of the monodomain and ionic models can be tackled in several ways.

Discuss the results, the methods employed, their stability and accuracy, and their algorithmic and computational aspects.

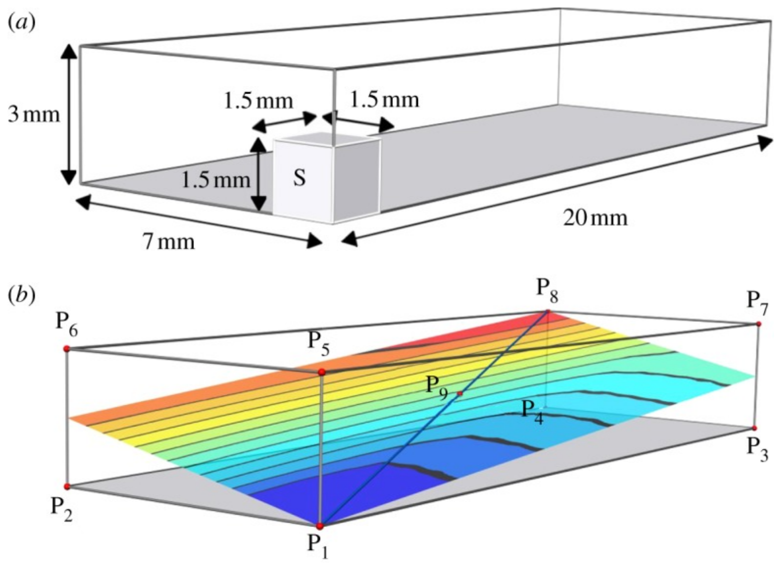

Figure 3: Example solution to the monodomain equation. Image taken from [6].

Project 5. Space and time adaptivity

The accuracy requirements of a problem (that is, how fine the discretization needs to be in order for the numerical solution to be accurate) can vary significantly both in space and in time. For example, a problem with sharp solution gradients only in some regions of the domain will require fine elements in those regions, but may allow for coarser elements elsewhere without compromising the accuracy. These situations are effectively handled by *adaptive methods*, wherein the space and time resolutions are dynamically adjusted based on an estimate of the solution error.

Consider the heat equation:

$$\begin{cases} \dfrac{\partial u}{\partial t} - \boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} u) = f & \text{in } \Omega \times (0, T) \ , \\ \mu \boldsymbol{\nabla} u \cdot \mathbf{n} = 0 & \text{on } \partial\Omega \times (0, T) \ , \\ u = 0 & \text{in } \Omega \times \{0\} \ , \end{cases}$$

where $\mu = 1$ and the forcing term f is a sequence of impulses, localized in space and time, and defined by the following expressions (Figure 4):

$$\begin{aligned} f(\mathbf{x}, t) &= g(t) h(\mathbf{x}) \ , \\ g(t) &= \frac{\exp(-a \cos(2N\pi t))}{\exp(a)} \ , \\ h(\mathbf{x}) &= \exp\left(-\frac{(\mathbf{x} - \mathbf{x}_0)^2}{\sigma^2} \right) \ , \end{aligned}$$

where $a > 0$, $N \in \mathbb{N}$, $\mathbf{x}_0 \in \Omega$ and $\sigma > 0$.

Based on `deal.II` tutorials on space adaptivity and on [7], implement a time- and space-adaptive solver for problem (5). Discuss the results in terms of accuracy and computational costs compared to a solver with homogeneous discretization.

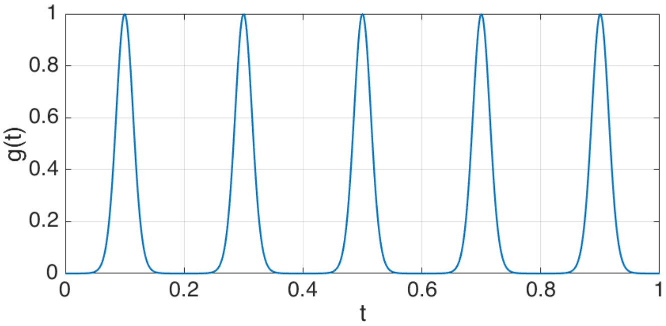

Figure 4: Plot of the function $g(t)$ for $a = 5$, $N = 5$.

Project 6. Non-linear cardiac mechanics

$$\begin{cases} -\boldsymbol{\nabla} \cdot P(\mathbf{d}) = 0 & \text{in } \Omega \,, \\ \mathbf{d} = \mathbf{g} & \text{on } \Gamma_{\mathrm{D}} \subset \partial\Omega \,, \\ P(\mathbf{d})\mathbf{n} = \mathbf{h} & \text{on } \Gamma_{\mathrm{N}} = \partial\Omega \backslash \Gamma_{\mathrm{N}} \,. \end{cases} \tag{7}$$

Consider problem (7), modeling the quasi-static passive mechanics of the cardiac muscle [9]. The first Piola-Kirchhoff stress tensor P (function of the displacement $\mathbf{d}$) characterizes the constitutive properties of the material. Implement a solver that computes the deformation of a slab of cardiac tissue subject to prescribed loads. Consider both isotropic materials (such as the neo-Hooke law) and anisotropic materials (such as the Guccione law, see e.g. [9]). Optionally, include active contraction effects [5]. Consider an idealized cardiac geometry such as the one described in [2].

Discuss the results, the methods employed, their stability and accuracy, and their algorithmic and computational aspects, with particular focus on the strategy for solving the non-linear problem.

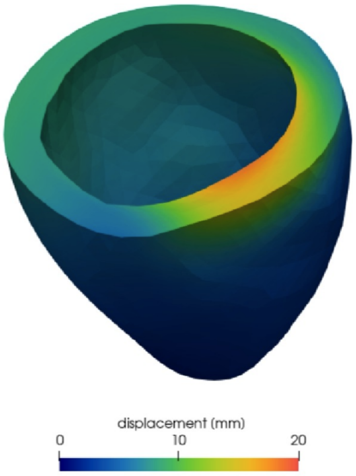

Figure 5: Snapshot of an example solution to a cardiac mechanics problem. Image taken from [9].

Project 7. Matrix-free solvers

$$\begin{cases} -\boldsymbol{\nabla} \cdot (\mu \boldsymbol{\nabla} u) + \boldsymbol{\nabla} \cdot (\boldsymbol{\beta} u) + \gamma u = f & \text{in } \Omega \;, \\ u = g & \text{on } \Gamma_{\mathrm{D}} \subset \partial\Omega \;, \\ \boldsymbol{\nabla} u \mathbf{n} = h & \text{on } \Gamma_{\mathrm{N}} = \partial\Omega \backslash \Gamma_{\mathrm{D}} \;. \end{cases} \tag{8}$$

Matrix-free solvers are PDE solvers based on avoiding the assembly of the system matrix, replacing it with the definition of the matrix-vector product operation. This approach takes advantage of modern computing architectures, where memory access is much slower than computations, and can effectively leverage vectorized CPU instructions.

Relying on `deal.II`'s documentation and tutorials, implement a matrix-free solver for the advection-diffusion-reaction problem (8). Compare the performance of the matrix-free solver with that of a matrix based solver, in terms of computational efficiency, parallel scalability and complexity of the implementation.

Project 8. Hybrid thread-MPI parallelization

Distributed memory parallelization is convenient when the different computing cores are connected to different memory units. Conversely, when multiple cores have access to the same memory, shared memory parallelization (multithreading) can be more convenient, to minimize memory redundancies. In real life situations, one typically has both: scientific computing clusters are usually composed of multiple nodes, each with its own memory, and each node has multiple cores. It is therefore especially convenient to use a hybrid parallelization approach, where the memory is shared within a node and distributed across nodes.

Considering problem (8), implement a solver that uses hybrid multithreading-MPI for parallelization, relying on `deal.II`'s documentation and tutorials (with reference in particular to tutorial steps 48 and 69).

Discuss the performance and parallel scalability of the solver, comparing to a fully distributed MPI parallelization.

Project 9. Fluid-structure interaction

$$\begin{cases} -\nu\Delta\mathbf{u} + \boldsymbol{\nabla} p = \mathbf{0} & \text{in } \Omega_{\text{fluid}} \ , \\ \boldsymbol{\nabla}\cdot\mathbf{u} = 0 & \text{in } \Omega_{\text{fluid}} \ , \\ -\boldsymbol{\nabla}\cdot\sigma(\mathbf{d}) = 0 & \text{in } \Omega_{\text{solid}} \ , \\ \mathbf{u} = \mathbf{0} & \text{on } \Sigma = \partial\Omega_{\text{fluid}} \cap \partial\Omega_{\text{solid}} \ , \\ \sigma(\mathbf{d})\mathbf{n} = \nu\boldsymbol{\nabla}\mathbf{u}\,\mathbf{n} - p\mathbf{n} & \text{on } \Sigma \ . \end{cases} \tag{9}$$

Implement a monolithic solver for problem (9), modeling 2D steady linear fluid-structure interaction (FSI). The problem couples the Stokes problem and the linear elasticity equations across a fluid-solid interface Σ, under the small displacements assumption (that is: the deformation of the solid does not affect the domain in which the fluid is solved, and the fluid velocity should be approximated to zero on the fluid-solid interface). Consider the domain $\Omega = \Omega_{\text{fluid}} \cup \Omega_{\text{solid}}$ depicted in Figure 6. Please notice that $\mathbf{n}$ denotes the unit vector on Σ, outgoing from Ω_{fluid} and incoming into Ω_{solid}. The problem should be endowed with suitable boundary conditions.

Discuss the results, the methods employed, their stability and accuracy, and their algorithmic and computational aspects. We advise referring to `deal.II` documentation and tutorials on multiphysics problems.

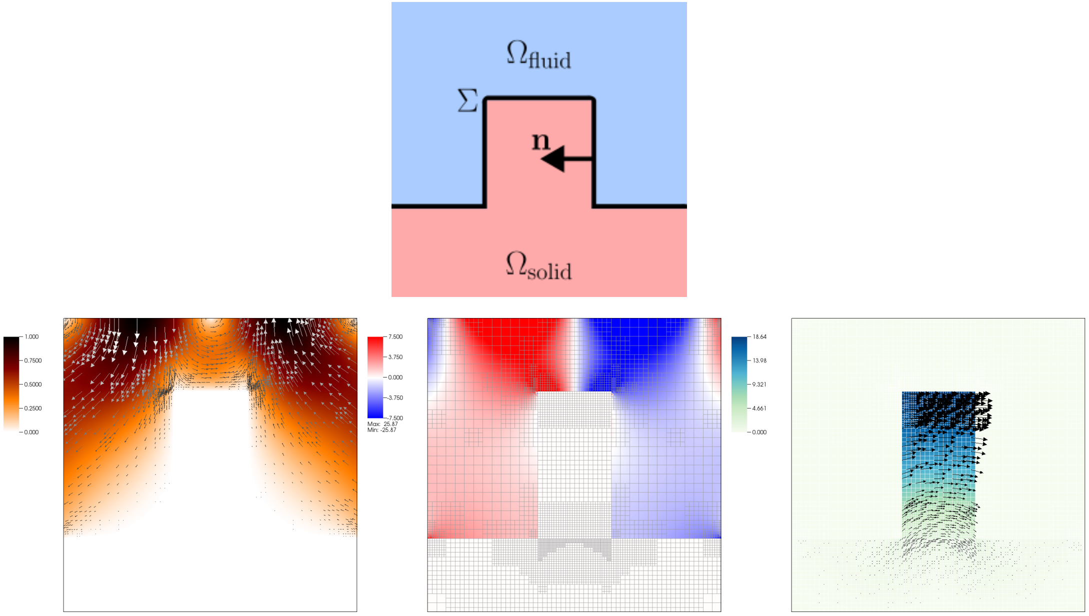

Figure 6: Top: domain for the FSI problem. Bottom: example solutions to an FSI problem. Image taken from https://dealii.org/9.3.3/doxygen/deal.II/step_46.html.

References

[1] J. Baiges, R. Codina, and S. Idelsohn. A domain decomposition strategy for reduced order models. application to the incompressible Navier–Stokes equations. *Computer Methods in Applied Mechanics and Engineering*, 267:23–42, 2013.

[2] L. Barbarotta, S. Rossi, L. Dedè, and A. Quarteroni. A transmurally heterogeneous orthotropic activation model for ventricular contraction and its numerical validation. *International journal for numerical methods in biomedical engineering*, 34(12):e3137, 2018.

[3] A. Bueno-Orovio, E. M. Cherry, and F. H. Fenton. Minimal model for human ventricular action potentials in tissue. *Journal of Theoretical Biology*, 253(3):544–560, 2008.

[4] S. Deparis, G. Grandperrin, and A. Quarteroni. Parallel preconditioners for the unsteady Navier–Stokes equations and applications to hemodynamics simulations. *Computers & Fluids*, 92:253–273, 2014.

[5] S. Land, V. Gurev, S. Arens, C. M. Augustin, L. Baron, R. Blake, C. Bradley, S. Castro, A. Crozier, M. Favino, et al. Verification of cardiac mechanics software: benchmark problems and solutions for testing active and passive material behaviour. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 471(2184):20150641, 2015.

[6] S. A. Niederer, E. Kerfoot, A. P. Benson, M. O. Bernabeu, O. Bernus, C. Bradley, E. M. Cherry, R. Clayton, F. H. Fenton, A. Garny, et al. Verification of cardiac tissue electrophysiology simulators using an N-version benchmark. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 369(1954):4331–4351, 2011.

[7] M. Picasso. Adaptive finite elements for a linear parabolic problem. *Computer Methods in Applied Mechanics and Engineering*, 167(3-4):223–237, 1998.

[8] A. Quarteroni. *Numerical models for differential problems*, volume 2. Springer, 2017.

[9] F. Regazzoni, M. Salvador, P. C. Africa, M. Fedele, L. Dede', and A. Quarteroni. A cardiac electromechanical model coupled with a lumped-parameter model for closed-loop blood circulation. *Journal of Computational Physics*, 457:111083, 2022.

[10] S. Salsa and G. Verzini. *Partial differential equations in action: from modelling to theory*, volume 147. Springer Nature, 2022.

[11] M. Schäfer, S. Turek, F. Durst, E. Krause, and R. Rannacher. *Benchmark Computations of Laminar Flow Around a Cylinder*, pages 547–566. Vieweg+Teubner Verlag, Wiesbaden, 1996.