Model Order Reduction techniques

Assignments - Part 1

Nicola R. Franco, Andrea Manzoni

Master's course in Mechanical Engineering

A.Y. 2024-2025

Students are required to pick (and elaborate upon) any of the three problems reported in the following pages.

All projects have a common skeleton. In particular, all case studies come with a Python code implementing a suitable Full Order Model (FOM) in FEniCS. Starting from there, students are required to

- Generate a dataset;
- Implement a Reduced Order Model (ROM) based on POD-Galerkin;
- Exploit the ROM to answer a given question.

Students are expected to present their work by synthesizing the main steps in a suitable IPython notebook. To foster readability, technical pieces of code should be placed outside of the notebook (i.e. within external ".py" files). Similarly, the data collected for training and testing should be stored within suitable ".npy" or ".npz" files, so that the notebook can be executed in its entirety within a reasonable amount of time.

The IPython notebook should report/contain:

i) details about the data collected (sample size, FOM dimension, etc.);

ii) a plot of the singular values, motivating the choice of the POD dimension;

iii) an RB solver that can be queried in real-time, together with a short summary about its performances (computational cost, accuracy);

iv) any additional results concerning project-specific tasks.

Designing a chemical dispenser (convection-diffusion, time-dependent)

A company is developing a chemical dispenser that uses fluidodynamics to control the flow and mixing of various chemicals with high precision. The dispenser must ensure accurate dosing, minimize waste, and maintain safety standards while handling potentially hazardous substances.

Figure 1 shows a 2D section of a prototype that is currently under study. The dispenser consists of a tank in which a liquid flows from left to right. The fluid flow is regulated by three parameters

- c_1: horizontal velocity of the fluid at the top inflow gate, Γ_1^{in};
- c_2 and c_3: horizontal and vertical velocities, respectively, of the fluid at the inflow gate Γ_2^{in}, located at the bottom-left of the tank.

After configuring the dispenser by setting c_1, c_2, and c_3, the chemical can be introduced through the top gate, Γ_1^{in}: the fluid flow will then naturally transport the chemical to the exit gates, Γ_1^{out} and Γ_2^{out}.

Before testing the prototype with actual experiments, the company wants to assess its potential by running some numerical simulations. For this reason, they introduce the following Full Order Model, which, for simplicity, describes the problem in 2D.

To start, given c_1, c_2, c_3, Stokes' equation is used to predict the fluid flow within the dispenser tank,

$$\begin{cases} -\Delta \boldsymbol{b} - \nabla p = 0 & \text{in } \Omega \\ \nabla \cdot \boldsymbol{b} = 0 & \text{in } \Omega \\ \boldsymbol{b} = [c_1, 0]^\top & \text{on } \Gamma_1^{in} \\ \boldsymbol{b} = [c_2, c_3]^\top & \text{on } \Gamma_2^{in} \\ \boldsymbol{b} = 0 & \text{on } \partial\Omega \setminus \left(\Gamma_1^{in} \cup \Gamma_2^{in}\right). \end{cases} \tag{1}$$

where $\boldsymbol{b} : \Omega \to \mathbb{R}^2$ and $p : \Omega \to \mathbb{R}$ are the velocity and the pressure of the fluid, respectively. After solving Stokes' equation, the velocity field $\boldsymbol{b}$ is inserted within an advection-diffusion model, which is used to study the spreading of the chemical $u : \Omega \times [0, T) \to \mathbb{R}$ over time,

$$\begin{cases} \dfrac{\partial u}{\partial t} - \dfrac{1}{2}\Delta u + \boldsymbol{b} \cdot \nabla u = 0 & \text{in } \Omega \times (0, T) \\ \nabla u \cdot \boldsymbol{n} = 0 & \text{on } (\partial\Omega \setminus \Gamma_1^{in}) \times (0, T) \\ u = 1 & \text{on } \Gamma_1^{in} \times (0, T) \\ u(\cdot, 0) = \mathbf{1}_{\Gamma_1^{in}} & \text{in } \Omega. \end{cases} \tag{2}$$

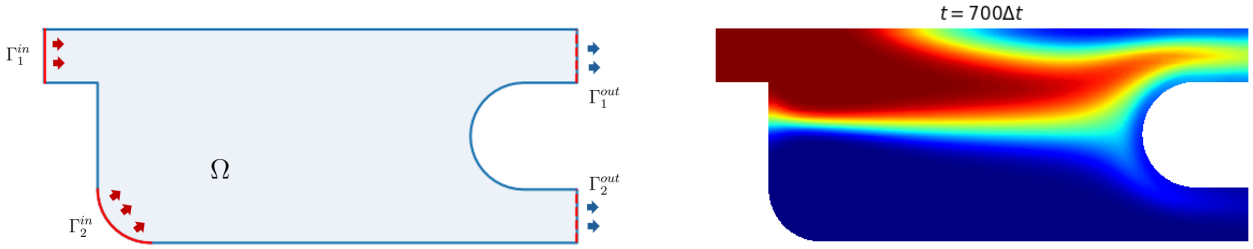

Figure 1. Left: spatial domain, inflow and outflow boundaries. Right: time stamp of a FOM simulation (chemical concentration u) with $c_1 = 40$, $c_2 = 0$ and $c_3 = 50$.

Set the final time to $T = 0.3505$. The script dispenser.py contains a FEniCS code implementing both the solution to Eq. (1) and Eq. (2) using a discrete time-step $\Delta t = 5 \cdot 10^{-4}$.

Tasks

i) For $i = 1, 2, 3$, let $\boldsymbol{b}_i$ be the solution to (1) with $c_i = 1$ and $c_j = 0$ for all $j \neq i$. Define $u_{\text{hom}} := u - 1$. Manipulate the equations to show that the solution to the advection-diffusion equation can be written as

$$u = u_{\text{hom}} + 1,$$

where, given c_1, c_2, c_3, u_{hom} is the solution to

$$\begin{cases} \dfrac{\partial u_{\text{hom}}}{\partial t} - \dfrac{1}{2}\Delta u_{\text{hom}} + (c_1\boldsymbol{b}_1 + c_2\boldsymbol{b}_2 + c_3\boldsymbol{b}_3) \cdot \nabla u_{\text{hom}} = 0 & \text{in } \Omega \times (0,T) \\ \nabla u_{\text{hom}} \cdot \boldsymbol{n} = 0 & \text{on } (\partial\Omega \setminus \Gamma_1^{in}) \times (0,T) \\ u_{\text{hom}} = 0 & \text{on } \Gamma_1^{in} \times (0,T) \\ u_{\text{hom}}(\cdot, 0) = \mathbf{1}_{\Gamma_1^{in}} - 1 & \text{in } \Omega. \end{cases} \tag{3}$$

In particular, up to pre-computing $\boldsymbol{b}_1, \boldsymbol{b}_2$ and $\boldsymbol{b}_3$, one can fully focus on the advection-diffusion problem, ignoring Stokes' equation.

ii) Let $\boldsymbol{\mu} = [c_1, c_2, c_3]$. Sample $N = 20$ random solutions by letting

$$0 \leq c_1, c_2, c_3 \leq 50.$$

Split the data in half, leaving 10 simulations for training and 10 for testing. Assemble the snapshot matrix and study the decay of the singular values: how many basis functions would you use for a POD-Galerkin ROM?

iii) Implement the reduced basis method through the POD-Galerkin scheme. Be sure to pre-compute all relevant operators as to make the assembling stage as efficient as possible. Quantify the performances of the RB solver in terms of efficiency (computational time) and accuracy (average relative error over the test set): to this end, use the same time-step of the FOM, $\Delta t = 5 \cdot 10^{-4}$. Inspect the distribution of test errors: how does the accuracy vary over time? How spread out are the errors across the 10 simulations in the test set?

The function bottomOutflow in the script dispenser.py allows you to evaluate the functional

$$J(u) := \frac{1}{T} \int_0^T \int_{\Gamma_2^{out}} u(s,t) ds dt,$$

for any given u. The latter can be used to quantify the average amount of chemical that is being delivered through the bottom gate Γ_2^{out}.

iv) Let $c_1 = 50$. For any given $\theta \in [0, \pi/2]$, set $c_2 = \cos\theta$ and $c_3 = \sin\theta$. Denote by u_θ the corresponding solution to (2). Exploit the ROM to draw a plot of the map $\theta \mapsto J(u_\theta)$. From there, estimate the optimal value of $\theta_* \in [0, \pi/2]$ minimizing J (that is: which angle should one use to ensure that most of the chemical is delivered to the top gate?).

Ensuring safety at a kids' playground (linear elasticity, stationary)

The town council of Paderno Dugnano has recently authorized the construction of a new playground for children. To ensure a safe environment and reduce the risk of accidental injuries, the council has requested the use of a soft flooring material capable of absorbing impact and cushioning falls. However, the council has also insisted that the floor be firm enough to support the entry of an ambulance in the unfortunate event of an emergency.

To meet all the requirements, the company hired for the construction is testing various solutions through numerical simulations. Figure 2 shows a 2D section (vertical slice) of a prototype that is currently under study. The latter represents a portion of the playground floor, 4.2 meters wide and 1 meter deep, $\Omega \subset [0, 4.2] \times [0, 1]$. Rectangular holes have been included to improve impact absorption. Concerning the materials, instead, the engineers at the company have proposed the use of polyurethane foam, which, depending on the manufactuing process, can present variable mechano-elastic properties. Specifically,

- density ρ ranging from 10 to 50 kg/m^3;
- first Lamé coefficient λ varying from 0.1 to 1 MPa;
- shear modulus μ going from 0.05 to 0.5 MPa.

To model the deformation of the playground floor under the weight of an ambulance, the company is relying on the following equations of linear elasticity:

$$\begin{cases} -\nabla \cdot \left[\lambda \nabla \cdot \boldsymbol{u} + \mu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^\top\right)\right] = \boldsymbol{f} & \text{in } \Omega \\ \boldsymbol{u} \equiv \boldsymbol{0} & \text{on } \Gamma_{\text{bottom}} \\ \lambda(\nabla \cdot \boldsymbol{u}) \cdot \boldsymbol{n} + \nu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^\top\right) \cdot \boldsymbol{n} \equiv [0, -mg/4.2]^\top & \text{on } \Gamma_{\text{top}} \\ \lambda(\nabla \cdot \boldsymbol{u}) \cdot \boldsymbol{n} + \nu \left(\nabla \boldsymbol{u} + \nabla \boldsymbol{u}^\top\right) \cdot \boldsymbol{n} \equiv \boldsymbol{0} & \text{on } \partial\Omega \setminus (\Gamma_{\text{bottom}} \cup \Gamma_{\text{top}}) \end{cases} \tag{4}$$

Here, $\boldsymbol{u} : \Omega \to \mathbb{R}^2$ is the displacement vector field, m is the mass of the ambulance, $g \approx 9.81\text{m/s}^2$ is the gravitational constant, $\boldsymbol{f} = [0, -\rho g]^\top$ is the body force, whereas $\Gamma_{\text{bottom}} := [0, 4.2] \times \{0\}$ and $\Gamma_{\text{up}} := [0, 4.2] \times \{1\}$.

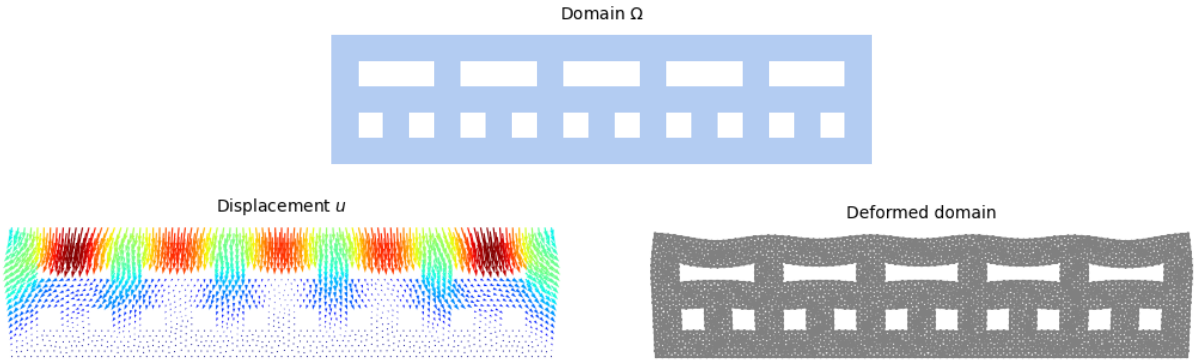

Figure 2. Top: spatial domain. Bottom left: FOM solution to Eq.(4) for $\rho = 50\text{kg/m}^3$, $\lambda = 0.5\text{MPa}$, $\mu = 0.15\text{MPa}$, $m = 4500\text{kg}$. Bottom right: deformed domain.

The script `playground.py` contains a FEniCS code that, given ρ, λ, μ and m, assembles and solves (4) using Finite Elements.

Tasks

i) Let $\boldsymbol{\mu} := [\rho, \lambda, \mu, m]$. Collect 30 different FOM simulations by sampling random values of the model parameters. To this end, use the physical ranges provided for ρ, λ and μ. For the ambulance mass, instead, let 3200kg $\leq m \leq$ 5400kg. Split the data into a training and a test set using a 2:1 ratio. Assemble the snapshot matrix and study the decay of the singular values: how many basis functions would you use for a POD-Galerkin ROM?

ii) Implement the reduced basis method through the POD-Galerkin scheme. Be sure to pre-compute all relevant operators as to make the assembling stage as efficient as possible. Quantify the performances of the RB solver in terms of efficiency (computational time) and accuracy (average relative error).

In order to be compliant with the safety measures, the company is seeking a material design that keeps the playground as soft as possible, while simultaneously ensuring that the floor does not exceed 20 cm of deformation under the weight of an ambulance.

iii) Fix $\rho = 50\text{kg/m}^3$, $\lambda = 0.5$MPa and $m = 4500$kg. For each μ in the physical range, let $\boldsymbol{u}_\mu$ denote the corresponding solution to the elasticity equations. Leveraging the ROM, draw a plot of the map

$$\mu \mapsto \max_{\boldsymbol{x} \in \Omega} \|\boldsymbol{u}_\mu(\boldsymbol{x})\|.$$

Considering that lower values of μ lead to softer materials, which is the smallest value of μ for which the maximum deformation does not exceed the limit threshold of 20cm?

Catching electrons in a bi-crystal (Schrödinger equation, time-dependent)

Two crystal plates are connected together to form a bi-crystal disk, $\Omega = \{\boldsymbol{x} \in \mathbb{R}^2 \ : \ \|\boldsymbol{x}\| < 1/\pi\}$, see Figure 3, which is later immersed in a uniform electric field $\boldsymbol{E}$.

An electron is confined within the bi-crystal, unable to escape due to a significant energy barrier. At time $t = 0$, the wave function of the electron reads

$$\Psi(\boldsymbol{x}) := e^{-100\|\boldsymbol{x}\|^2},$$

corresponding to a strong localization of the particle nearby the center of the disk. As time starts to flow, the dynamics of the electron is influenced by:

- $E > 0$, the magnitude of the electric field $\boldsymbol{E} = [E, 0]^\top$, here assumed to be pointing right (thus pushing the electron to the left);
- $\Delta V_0 > 0$, the difference in the electric potential between the two crystal plates.

According to the time-dependent Schrödinger equation, the wave function of the electron, $\Psi : \Omega \times [0, T) \to \mathbb{C}$, evolves over time via the formula

$$i\hbar \frac{\partial \Psi}{\partial t} = -\frac{\hbar^2}{2m} \Delta \Psi + V_{\boldsymbol{\mu}} \Psi, \tag{5}$$

where $i \in \mathbb{C}$ is the immaginary unit and

$$V_{\boldsymbol{\mu}}(\boldsymbol{x}) = E x_1 + \Delta V_0 \mathbf{1}_{[0,+\infty)}(x_2),$$

is the parametrized potential, with $\boldsymbol{\mu} := [E, \Delta V_0]$ being the problem parameters. Due to the energy barrier outside of the bi-crystal, Eq. (5) should also be complemented with a homogeneous Dirichlet boundary condition, namely

$$\Psi(\boldsymbol{x}) \equiv 0 \quad \forall \boldsymbol{x} \in \Omega.$$

At every time instant $t \in [0, T)$ the wave function can be used to compute the probability density function

$$p_t(\boldsymbol{x}) = \frac{1}{\int_\Omega |\Psi(\boldsymbol{x}, t)|^2 d\boldsymbol{x}} \cdot |\Psi(\boldsymbol{x}, t)|^2.$$

The latter is a function that, given $\boldsymbol{x} \in \Omega$, returns a the likelihood of finding the electron at that specific position.

NB: hereon, we trade physical rigor for mathematical simplicity and simply set $\hbar = 1$ and $m = 1/2$ in the Schrödinger equation.

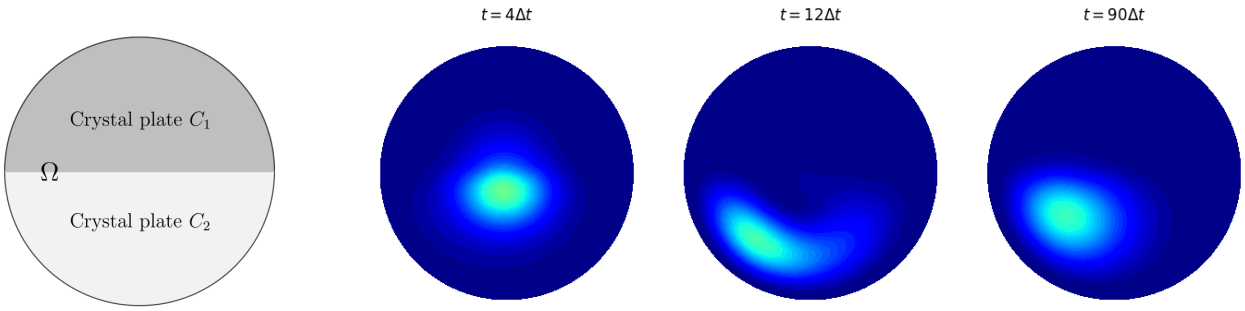

Figure 3. Far left: bi-crystal decomposition. Center-to-right: probability distribution of the electron position over time, computed for $E = 600$ and $\Delta V_0 = 400$.

The script `electron.py` contains a FEniCS code that, given E and ΔV_0, approximates the solution to the Schrödinger equation using Finite Elements. To address the fact that the wave function is complex valued, the script uses a decomposition into a real and an imaginary part. Simply put, if V_h is the space of (real-valued) piecewise linear polynomials arising from the usual Finite Element simulation, and $N_h = \dim(V_h)$, then each quantum state is represented as a vector $\boldsymbol{\Psi} \in \mathbb{R}^{2N_h}$, with

$$\boldsymbol{\Psi} = \begin{bmatrix} \boldsymbol{\psi}_{\text{real}} \\ \\ \boldsymbol{\psi}_{\text{imag}} \end{bmatrix}, \quad \text{where} \quad \boldsymbol{\psi}_{\text{real}}, \boldsymbol{\psi}_{\text{imag}} \in \mathbb{R}^{N_h}.$$

From a discrete / algebraic perspective, it is then easy to see that (5) becomes

$$\begin{bmatrix} \mathbf{M} & \\ & \mathbf{M} \end{bmatrix} \frac{\partial \boldsymbol{\Psi}}{\partial t} = \begin{bmatrix} & -\mathbf{S} - \mathbf{V}_{\boldsymbol{\mu}} \\ \mathbf{S} + \mathbf{V}_{\boldsymbol{\mu}} & \end{bmatrix} \boldsymbol{\Psi}$$

which, in the script `electron.py`, is solved numerically using the Backward Euler scheme.

Here, $\mathbf{M}$, $\mathbf{S}$ and $\mathbf{V}_{\boldsymbol{\mu}}$ are matrices assembled in the standard Finite Element space, V_h. More precisely, they are obtained by discretizing the following bilinear forms

$$(u, v) \mapsto \int_\Omega uv, \qquad (u, v) \mapsto \int_\Omega \nabla u \cdot \nabla v, \qquad (u, v) \mapsto \int_\Omega V_{\boldsymbol{\mu}} uv,$$

respectively.

Tasks

i) Using the terminal time T and the time-step Δt provided in the code, sample 30 FOM simulations. To this end, pick random values for the model parameters by letting

$$0 \leq E, \Delta V_0 \leq 1000,$$

uniformly. Split the data in half, leaving 15 simulations for training and 15 for testing. Assemble the snapshot matrix and study the decay of the singular values: how many basis functions would you use for a POD-Galerkin ROM?

ii) Implement the reduced basis method through the POD-Galerkin scheme. Be sure to pre-compute all relevant operators as to make the assembling stage as efficient as possible. Quantify the performances of the RB solver in terms of efficiency (computational time) and accuracy (average relative error).

The functions probabilities and caught, available in the script electron.py, allow you to estimate the average probability of finding the electron in the crystal plate at the bottom, $C_{\text{bottom}} := \{(x_1, x_2) \in \Omega \ : \ x_2 \leq 0\}$, that is,

$$P = \frac{1}{T} \int_0^T \int_{C_{\text{bottom}}} p_t(\boldsymbol{x}) d\boldsymbol{x} dt. \tag{6}$$

iii) Let $E = 500$. For each $0 \leq \Delta V_0 \leq 1000$, let $P_{\Delta V_0}$ be the corresponding value of P in Eq. (6). Exploit the ROM to draw a plot of the map $\Delta V_0 \mapsto P_{\Delta V_0}$. Which is the smallest value of ΔV_0 for which the average probability of finding the electron in the bottom plate is larger than 90%?