PINNs vs Greedy in Reduced Basis Method for parametric Advection-Diffusion problem

Exam:
MODEL ORDER REDUCTION AND MACHINE LEARNING
(Politecnico di Torino)

Davide Fassino and Gioana Teora

July 18, 2023

Abstract

The aim of this project is to compare two different strategies for solving parametric boundary problems: the Physics Informed Neural Network (PINN) and the Reduced Order Model (ROM) via the greedy algorithm. We compare these two strategies by solving an advection-diffusion problem in terms of online accuracy and speed-up with respect to Finite Element Methods (FEM) besides the offline computational cost.

1 The model problem

Let $\Omega = (0,1)^2$, we consider the following advection-diffusion problem

$$\begin{cases} -\nabla \cdot (\mu_1 \nabla u) + \boldsymbol{\beta} \cdot \nabla u = 0 & \text{in } \Omega, \\ \mu_1 \nabla u \cdot \mathbf{n} = \mu_2 & \text{on } \Gamma_b, \\ \nabla u \cdot \mathbf{n} = 0 & \text{on } \Gamma_r, \\ u = 0 & \text{on } \Gamma_{tl}, \end{cases} \quad \forall \boldsymbol{\mu} = (\mu_1, \mu_2) \in \mathcal{P}, \tag{1.1}$$

where

- $\boldsymbol{n}$ denotes the outward unit normal on the boundary $\partial\Omega$ of Ω;
- $\Gamma_{tl} = ([0,1] \times \{1\}) \cup (\{0\} \times [0,1])$ represents the Dirichlet boundary;
- $\Gamma_b = [0,1] \times \{0\}$ and $\Gamma_r = \{1\} \times [0,1]$ represent the Neumann boundary;
- the advection field $\boldsymbol{\beta} = \begin{bmatrix} y(1-y) \\ 0 \end{bmatrix}$ is such that $\nabla \cdot \boldsymbol{\beta} = 0$ and $\boldsymbol{\beta} \cdot \boldsymbol{n} \geq 0$ on $\Gamma_b \cup \Gamma_r$;
- the parameters $\boldsymbol{\mu} = (\mu_1, \mu_2)$, which are defined in the parametric space $\mathcal{P} = [0.1, 10] \times [-1, 1]$, represent the diffusion coefficient and the Neumann boundary condition on Γ_b, respectively.

Let us introduce the space

$$\mathbb{V} = H^1_{0,\Gamma_{tl}}(\Omega) = \left\{ v \in H^1(\Omega) : v = 0 \text{ on } \Gamma_{tl} \right\},$$

which is endowed with

$$(u,v)_{\mathbb{V}} = (\nabla u, \nabla v)_0\,, \quad \|u\|_{\mathbb{V}} = \|\nabla u\|_0, \quad \forall u, v \in \mathbb{V},$$

where $(\cdot,\cdot)_0$ and $\|\cdot\|_0$ are the scalar product and the norm in $L^2(\Omega)$ (and, for the ease of the notation, also in $(L^2(\Omega))^2$), respectively. The variational formulation of the problem (1.1) reads as

$$\begin{cases} \text{Find } u(\boldsymbol{\mu}) \in \mathbb{V} \text{ such that} \\ \mathcal{B}(u, v; \boldsymbol{\mu}) = \mathcal{F}(v; \boldsymbol{\mu}) \qquad \forall\, v \in \mathbb{V} \end{cases}, \quad \forall \boldsymbol{\mu} \in \mathcal{P}, \tag{1.2}$$

where

$$\mathcal{B}(u, v; \boldsymbol{\mu}) = \int_\Omega \mu_1 \nabla u \cdot \nabla v + \int_\Omega (\boldsymbol{\beta} \cdot \nabla u)\, v, \quad \mathcal{F}(v; \boldsymbol{\mu}) = \int_{\Gamma_b} \mu_2\, v, \quad \forall u, v \in \mathbb{V}.$$

More precisely,

- $\mathcal{F} : \mathbb{V} \times \mathcal{P} \to \mathbb{R}$ is linear and uniformly continuous, i.e.

$$\exists \rho < +\infty, \text{ independent of } \boldsymbol{\mu}, \text{ such that } \sup_{v \in \mathbb{V}} \frac{\mathcal{F}(v; \boldsymbol{\mu})}{\|v\|_{\mathbb{V}}} = \rho(\boldsymbol{\mu}) \le \rho, \quad \forall \boldsymbol{\mu} \in \mathcal{P};$$

- $\mathcal{B} : \mathbb{V} \times \mathbb{V} \times \mathcal{P} \to \mathbb{R}$ is bilinear, uniformly continuous, i.e.

$$\exists \gamma < +\infty, \text{ independent of } \boldsymbol{\mu}, \text{ such that } \sup_{u,v \in \mathbb{V}} \frac{\mathcal{B}(u, v; \boldsymbol{\mu})}{\|v\|_{\mathbb{V}} \|u\|_{\mathbb{V}}} = \gamma(\boldsymbol{\mu}) \le \gamma, \quad \forall \boldsymbol{\mu} \in \mathcal{P},$$

  and uniformly coercive, that is

$$\exists \alpha > 0, \text{ independent of } \boldsymbol{\mu}, \text{ such that } \inf_{v \in \mathbb{V}} \frac{\mathcal{B}(v, v; \boldsymbol{\mu})}{\|v\|^2_{\mathbb{V}}} = \alpha(\boldsymbol{\mu}) \ge \alpha, \quad \forall \boldsymbol{\mu} \in \mathcal{P}.$$

We observe that the coercivity constant $\alpha(\boldsymbol{\mu}) = \mu_1$ for each $\boldsymbol{\mu} \in \mathcal{P}$. Indeed, we first notice that, $\forall u \in \mathbb{V}$,

$$\int_\Omega (\boldsymbol{\beta} \cdot \nabla u)\, u = \int_\Omega \boldsymbol{\beta} \cdot \nabla \left(\frac{u^2}{2}\right) = -\int_\Omega \frac{1}{2} (\nabla \cdot \boldsymbol{\beta})\, u^2 + \int_{\partial\Omega} \frac{1}{2} (\boldsymbol{\beta} \cdot \boldsymbol{n})\, u^2 = \int_{\Gamma_b \cup \Gamma_r} \frac{1}{2} (\beta \cdot \boldsymbol{n})\, u^2,$$

since $\nabla \cdot \boldsymbol{\beta} = 0$. Furthermore,

$$\boldsymbol{\beta} \cdot \boldsymbol{n}_{\Gamma_b} = 0 \text{ on } \Gamma_b \text{ and } \boldsymbol{\beta} \cdot \boldsymbol{n}_{\Gamma_r} = y(1-y) \ge 0 \text{ on } \Gamma_r,$$

where $\boldsymbol{n}_{\Gamma_b} = \begin{bmatrix} 0 \\ -1 \end{bmatrix}$ and $\boldsymbol{n}_{\Gamma_r} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$. Finally,

$$\mathcal{B}(u, u; \boldsymbol{\mu}) = \int_\Omega \mu_1 \nabla u \cdot \nabla u + \int_\Omega (\boldsymbol{\beta} \cdot \nabla u)\, u \ge \mu_1 \|u\|^2_{\mathbb{V}}, \tag{1.3}$$

which brings us to the coercivity constant $\alpha(\boldsymbol{\mu}) = \mu_1$ for each $\boldsymbol{\mu} \in \mathcal{P}$. Using the Lax-Milgram theorem, we can state that problem (1.2) admits a unique solution in $\mathbb{V}$.

Remark 1. From (1.1), we notice that the solution of the problem becomes the trivial solution $u = 0$ on Ω when $\mu_2 = 0$. Moreover, we want to highlight that the infinity norm of the solution depends on the value of the ratio $\frac{\mu_2}{\mu_1}$, which gives a constraint for the normal derivative on the boundary Γ_b.

In order to numerically solve the problem (1.2), we introduce a triangulation $\mathcal{T}_\delta$ of Ω, where we define as usual δ as the maximum diameter of triangles in $\mathcal{T}_\delta$ and the finite element space $\mathbb{V}_\delta \subset \mathbb{V}$ of dimension N_δ and order 1, which is defined as

$$\mathbb{V}_\delta = \{v \in \mathbb{V} \cup C^0(\overline{\Omega}) : v_{|T} \in \mathbb{P}_1(T) \ \forall T \in \mathcal{T}_\delta\}.$$

In order to uniquely define functions in $\mathbb{V}_\delta$, we define the degrees of freedom as the values of such functions at the vertices of triangle which do not belong to the Dirichlet boundary. We further denote by $\{\varphi_i\}_{i=1}^{N_\delta}$ the lagrangian basis for $\mathbb{V}_\delta$ related to such degrees of freedom. The finite element discretization of the variational problem (1.2) can be written as

$$\begin{cases} \text{Find } u_\delta(\boldsymbol{\mu}) \in \mathbb{V}_\delta \text{ such that} \\ \mathcal{B}(u_\delta(\boldsymbol{\mu}), v_\delta; \boldsymbol{\mu}) = \mathcal{F}(v_\delta; \boldsymbol{\mu}) \quad \forall\, v_\delta \in \mathbb{V}_\delta. \end{cases}, \quad \forall \boldsymbol{\mu} \in \mathcal{P}, \tag{1.4}$$

or, algebraically, as find $\boldsymbol{u}_\delta(\boldsymbol{\mu}) \in \mathbb{R}^{N_\delta}$ such that

$$\mathbf{A}_\delta^{\boldsymbol{\mu}} \boldsymbol{u}_\delta(\boldsymbol{\mu}) = \boldsymbol{f}_\delta^{\boldsymbol{\mu}}, \tag{1.5}$$

where

$$\boldsymbol{f}_\delta^{\boldsymbol{\mu}} = \mu_2 \boldsymbol{f}_\delta \in \mathbb{R}^{N_\delta}, \quad \text{and } (\boldsymbol{f}_\delta)_i = \int_{\Gamma_b} \varphi_i, \ \forall i = 1, \dots, N_\delta, \tag{1.6}$$

$$\mathbf{A}_\delta^{\boldsymbol{\mu}} = \mu_1 \mathbf{X}_\delta + \mathbf{A}_\delta^{\boldsymbol{\beta}} \in \mathbb{R}^{N_\delta \times N_\delta}, \quad \left(\mathbf{A}_\delta^{\boldsymbol{\beta}}\right)_{ij} = \int_\Omega (\boldsymbol{\beta} \cdot \nabla \varphi_j)\, \varphi_i, \ \forall i, j = 1, \dots, N_\delta, \tag{1.7}$$

and $\mathbf{X}_\delta$ s.t. $(\mathbf{X}_\delta)_{ij} = \int_\Omega \nabla \varphi_j \cdot \nabla \varphi_i \ \forall i, j = 1, \dots, N_\delta$ is the matrix of the inner product in $\mathbb{V}$ of functions belonging to $\mathbb{V}_\delta$.

We observe that we must solve the linear system (1.5) for each value $\boldsymbol{\mu} \in \mathcal{P}$ of interest. We observe that, since the involved continuous forms are affine with respect to the parameter, it is possible to assemble $\mathbf{X}_\delta$, $\mathbf{A}_\delta^{\boldsymbol{\beta}}$ and $\boldsymbol{f}_\delta$ just once at all. However, solving the linear system (1.5) could be very computationally expensive if the dimension N_δ of $\mathbb{V}_\delta$ is quite large and it could require a lot of time. Finally, if $u(\boldsymbol{\mu})$ is sufficiently smooth, we recall that the following a priori error estimate holds

$$\|u(\boldsymbol{\mu}) - u_\delta(\boldsymbol{\mu})\|_{\mathbb{V}} = O(\delta) = O(N_\delta^{-1/2}). \tag{1.8}$$

The purpose of this work is to investigate techniques that represent a better trade-off between accuracy and speed, which represents a very important quality for real-time applications.

2 Reduced Order Model via greedy algorithm

The main idea of the ROM approach consists in extracting a set of $M \ll N_\delta$ basis functions $\{w_m = u_\delta(\boldsymbol{\mu}_m)\}_{m=1}^M$ from a collection of solutions obtained throughout a high fidelity method (FEM in our specific case) related to specific values of the parameters to be used to cheaply compute a solution related to a new instance of the parameter. Indeed, given the basis $\{w_m = u_\delta(\boldsymbol{\mu}_m)\}_{m=1}^M$, we introduce the reduced order space $\mathbb{V}_{\text{RB}} := \text{span}\{w_1, \dots, w_M\} \subset \mathbb{V}_\delta$ and define the ROM problem as

$$\begin{cases} \text{Find } u_{\text{RB}}(\boldsymbol{\mu}) \in \mathbb{V}_{\text{RB}} \text{ such that} \\ \mathcal{B}(u_{\text{RB}}(\boldsymbol{\mu}), v_{\text{RB}}; \boldsymbol{\mu}) = \mathcal{F}(v_{\text{RB}}; \boldsymbol{\mu}) \quad \forall\, v_{\text{RB}} \in \mathbb{V}_{\text{RB}}. \end{cases}, \quad \forall \boldsymbol{\mu} \in \mathcal{P}, \tag{2.1}$$

or, algebraically,

$$\mathbf{A}_{\text{RB}}^{\boldsymbol{\mu}} \boldsymbol{u}_{\text{RB}}^{\boldsymbol{\mu}} = \boldsymbol{f}_{\text{RB}}^{\boldsymbol{\mu}}$$

where $(\mathbf{A}^{\boldsymbol{\mu}}_{\mathrm{RB}})_{mn} = \mathcal{B}(w_n, w_m; \boldsymbol{\mu})$, $(\boldsymbol{f}^{\boldsymbol{\mu}}_{\mathrm{RB}})_m = \mathcal{F}(w_m, \boldsymbol{\mu})$, $\forall n, m = 1, \dots, M$. Now, since $w_m \in \mathbb{V}_\delta$, it can be written as $w_m = \sum_{i=1}^{N_\delta} (\mathbf{B})_{im}\, \varphi_i$, $\forall m = 1, \dots, M$. Thus,

$$\mathbf{A}^{\boldsymbol{\mu}}_{\mathrm{RB}} = \mathbf{B}^T\, \mathbf{A}^{\boldsymbol{\mu}}_{\delta}\, \mathbf{B} = \mu_1 \mathbf{B}^T \mathbf{X}_\delta \mathbf{B} + \mathbf{B}^T \mathbf{A}^{\boldsymbol{\beta}}_{\delta} \mathbf{B},$$

$$\boldsymbol{f}^{\boldsymbol{\mu}}_{RB} = \mathbf{B}^T \boldsymbol{f}^{\boldsymbol{\mu}}_{\delta} = \mu_2 \mathbf{B}^T \boldsymbol{f}_\delta.$$

We note that, given the matrices $\mathbf{X}_\delta$, $\mathbf{A}^{\boldsymbol{\beta}}_{\delta} \in \mathbb{R}^{N_\delta \times N_\delta}$, we can project them in the reduced space only once and then we can solve the reduced problem for different instances of the parameters with a computational cost that depends on $M \ll N_\delta$. Furthermore, it holds:

$$\|u_\delta(\boldsymbol{\mu}) - u_{RB}(\boldsymbol{\mu})\|_{\mathbb{V}} \leq \frac{||\hat{r}_\delta(u_{\mathrm{RB}}(\boldsymbol{\mu}); \boldsymbol{\mu}))||_{\mathbb{V}}}{\alpha(\boldsymbol{\mu})} := \eta_{\mathbb{V}}(\boldsymbol{\mu})$$

where $\alpha(\boldsymbol{\mu})$ is defined in (1.3) and $\hat{r}_\delta(u_{\mathrm{RB}}(\boldsymbol{\mu}); \boldsymbol{\mu})) \in \mathbb{V}_\delta$ is such that

$$(\hat{r}_\delta(u_{\mathrm{RB}}(\boldsymbol{\mu})), v_\delta)_{\mathbb{V}} = \mathcal{F}(v_\delta, \boldsymbol{\mu}) - \mathcal{B}(u_{\mathrm{RB}}(\boldsymbol{\mu}), v_\delta, \boldsymbol{\mu}),$$

or, algebraically,

$$\hat{\boldsymbol{r}}_\delta(\boldsymbol{\mu}) = \boldsymbol{f}^{\boldsymbol{\mu}}_{\delta} - \mathbf{A}^{\boldsymbol{\mu}}_{\delta} \mathbf{B} \boldsymbol{u}_{RB}(\boldsymbol{\mu}).$$

We observe that, from a computational point of view, the norm of a function $v_\delta = \sum_{i=1}^{N_\delta} (\boldsymbol{v}_\delta)_i \varphi_i \in \mathbb{V}_\delta$ can be computed as $\|v_\delta\|^2_{\mathbb{V}} = \boldsymbol{v}_\delta^T \mathbf{X}_\delta \boldsymbol{v}_\delta := \|\boldsymbol{v}_\delta\|^2_{\mathbf{X}_\delta}$. Thus,

$$\mathbf{A}^{\boldsymbol{\mu}}_{\delta}(\boldsymbol{u}_\delta(\boldsymbol{\mu}) - \boldsymbol{u}_{RB}(\boldsymbol{\mu})) = \hat{\boldsymbol{r}}_\delta(\boldsymbol{\mu}) \quad \Rightarrow \quad \boldsymbol{u}_\delta(\boldsymbol{\mu}) - \boldsymbol{u}_{RB}(\boldsymbol{\mu}) = (\mathbf{A}^{\boldsymbol{\mu}}_{\delta})^{-1} \hat{\boldsymbol{r}}_\delta(\boldsymbol{\mu})$$

$$\|\boldsymbol{u}_\delta(\boldsymbol{\mu}) - \boldsymbol{u}_{RB}(\boldsymbol{\mu})\|_{\mathbf{X}_\delta} \leq \|\mathbf{X}_\delta^{1/2} (\mathbf{A}^{\boldsymbol{\mu}}_{\delta})^{-1} \mathbf{X}_\delta^{1/2}\|_2 \|\hat{\boldsymbol{r}}_\delta(\boldsymbol{\mu})\|_{\mathbf{X}_\delta^{-1}} = \frac{1}{\sigma_{min}(\mathbf{X}_\delta^{-1/2}\, \mathbf{A}^{\boldsymbol{\mu}}_{\delta}\, \mathbf{X}_\delta^{-1/2})} \|\hat{\boldsymbol{r}}_\delta(\boldsymbol{\mu})\|_{\mathbf{X}_\delta^{-1}},$$

where $\sigma_{min}(\mathbf{X}_\delta^{-1/2}\, \mathbf{A}^{\boldsymbol{\mu}}_{\delta}\, \mathbf{X}_\delta^{-1/2}) = \alpha(\boldsymbol{\mu})$ is the smallest singular value of the matrix $\mathbf{X}_\delta^{-1/2}\, \mathbf{A}^{\boldsymbol{\mu}}_{\delta}\, \mathbf{X}_\delta^{-1/2}$. Indeed,

$$\alpha(\boldsymbol{\mu}) = \inf_{v_\delta \in \mathbb{V}_\delta} \frac{a(v_\delta, v_\delta; \boldsymbol{\mu})}{\|v_\delta\|^2_V} = \inf_{\boldsymbol{v}_\delta \in \mathbb{R}^{N_\delta}} \frac{\boldsymbol{v}_\delta^T \mathbf{A}^{\boldsymbol{\mu}}_{\delta} \boldsymbol{v}_\delta}{\boldsymbol{v}_\delta^T \mathbf{X}_\delta \boldsymbol{v}_\delta} = \inf_{\boldsymbol{v}_\delta \in \mathbb{R}^{N_\delta}} \frac{\boldsymbol{v}_\delta^T \mathbf{A}^{S,\boldsymbol{\mu}}_{\delta} \boldsymbol{v}_\delta}{\boldsymbol{v}_\delta^T \mathbf{X}_\delta \boldsymbol{v}_\delta},$$

where the matrix $\mathbf{A}^{S,\boldsymbol{\mu}}_{\delta} = \frac{1}{2}(\mathbf{A}^{\boldsymbol{\mu}}_{\delta} + (\mathbf{A}^{\boldsymbol{\mu}}_{\delta})^T)$. Finally, $\alpha(\boldsymbol{\mu})$ is the smallest eigenvalue which satisfies the following generalized problem

$$\mathbf{A}^{S,\boldsymbol{\mu}}_{\delta} \boldsymbol{v} = \lambda \mathbf{X}_\delta \boldsymbol{v}$$

or, by performing the change of variable $\boldsymbol{w} = \mathbf{X}_\delta^{1/2} \boldsymbol{v}$,

$$\mathbf{X}_\delta^{-1/2} \mathbf{A}^{S,\boldsymbol{\mu}}_{\delta} \mathbf{X}_\delta^{-1/2} \boldsymbol{w} = \lambda \boldsymbol{w}.$$

Now, it remains to discuss how to build the space $\mathbb{V}_{\mathrm{RB}}$. An efficient way to build a set of basis functions for $\mathbb{V}_{\mathrm{RB}}$ is via the greedy algorithm. More precisely, given a finite-dimensional subspace $\mathcal{P}_{\mathrm{RB}} \subset \mathcal{P}$, the greedy strategy is an iterative algorithm in which, at each step, we select a value $\overline{\mu} \in \mathcal{P}_{\mathrm{RB}}$ which maximizes the relative a posteriori error estimator $\eta_{\mathbb{V}}(\boldsymbol{\mu})$, i.e.

$$\overline{\boldsymbol{\mu}} \in \arg \max_{\mu \in \mathcal{P}_{\mathrm{RB}}} \eta_{\mathbb{V}}(\boldsymbol{\mu}).$$

Then, the high-fidelity solution $\overline{w} = u_\delta(\overline{\boldsymbol{\mu}})$ corresponding to the maximizer $\overline{\boldsymbol{\mu}}$ is computed and it is added to $\mathbb{V}_{\mathrm{RB}}$. The iterative procedure continues until the relative error estimator is below a user-specified tolerance *tol*, i.e.

$$\eta_{\mathbb{V}}(\boldsymbol{\mu}) \leq tol.$$

Remark 2. We note that the functions $\{w_m\}_{m=1}^M$ obtained through the greedy algorithm could be linearly dependent, then a Gram-Schmidt procedure is applied at each step.

3 The Physics Informed Neural Network for parametric boundary problem

The Physics Informed Neural Network is a tool exploiting the Deep Neural Network for numerically computing the solution of a Partial Differential Equation (PDE). More precisely, the solution $u(x, y, \mu_1, \mu_2)$ of the boundary parametric problem (1.1) is approximated by means of a neural network $u^{\mathcal{NN}}(x, y, \mu_1, \mu_2; \boldsymbol{w})$ whose weights $\boldsymbol{w}$ are chosen by minimizing a loss function which incorporate the residuals related to both the PDE and the boundary conditions. Thus, the loss function

$$\mathcal{L}_{PINN}(\boldsymbol{w}) = MSE_{PDE}(\boldsymbol{w}) + MSE_{NB}(\boldsymbol{w}) \tag{3.1}$$

is defined as the sum of

- the Mean Squared Error (MSE) induced by the residual related to the PDE, i.e.

$$MSE_{PDE}(\boldsymbol{w}) := \frac{1}{N_P}\sum_{k=1}^{N_P}\left[-\mu_1\Delta u^{\mathcal{NN}}(x_P^k, y_P^k, \mu_1^k, \mu_2^k; \boldsymbol{w}) + \boldsymbol{\beta}(x_P^k, y_P^k)\cdot\nabla u^{\mathcal{NN}}(x_P^k, y_P^k, \mu_1^k, \mu_2^k; \boldsymbol{w})\right]^2,$$

  given N_P input $(x_P^k, y_P^k, \mu_1^k, \mu_2^k) \in \Omega \times \mathcal{P}$, with $k = 1, \dots, N_p$;

- the MSE induced by the residual related to the Neumann Boundary conditions, i.e.

$$MSE_{NB}(\boldsymbol{w}) := \frac{1}{N_b}\sum_{k=1}^{N_b}\left[\mu_1^k\frac{\partial u^{\mathcal{NN}}}{\partial y}(x_b^k, y_b^k, \mu_1^k, \mu_2^k; \boldsymbol{w}) + \mu_2^k\right]^2 + \frac{1}{N_r}\sum_{k=1}^{N_r}\left[\frac{\partial u^{\mathcal{NN}}}{\partial x}(x_r^k, y_r^k, \mu_1^k, \mu_2^k; \boldsymbol{w})\right]^2,$$

  given N_b input $(x_b^k, y_b^k, \mu_1^k, \mu_2^k) \in \Gamma_b \times \mathcal{P}$, with $k = 1, \dots, N_b$, and N_r input $(x_r^k, y_r^k, \mu_1^k, \mu_2^k) \in \Gamma_r \times \mathcal{P}$, with $k = 1, \dots, N_r$.

The Dirichlet boundary conditions, instead, are imposed exactly by modifying the *output* of the network as $x(1 - y) * output$. The optimization is carried out through the first-order optimizer Adam. The architecture of the Network consists of a feed-forward fully-connected neural network made of four hidden layers with 30 nodes each. The activation function is set to tanh (no activation function is applied to the output), while the weights are initialized with the Xavier normal weights. The whole architecture is shown in Figure 1.

4 Numerical experiments

In this numerical experiment, we make a comparison between the PINN and the ROM approaches in terms of accuracy and speed-up with respect to a Finite Element method assumed here as the *exact* method.

The Finite Element Method is built over a triangular mesh made up of triangular with maximum size 1.0*e*-04 and the derived space $\mathbb{V}_\delta$ has dimension $N_\delta = 7757$.

Firstly, we generate a discrete parameter space $\mathcal{P}_\delta$ that we split into a training set $\mathcal{P}_{\delta,train}$ (80% of the original space), which is used to train both the PINN and the ROM, and a test set $\mathcal{P}_{\delta,test}$, which is used to carry out the comparison between such different approaches. In particular, we pick 80 values for μ_2 accordingly to a random uniform distribution in $[-1, 1]$ and 47 values for μ_1 accordingly to an exponential distribution with mean 1 truncated in the interval $[0.1, 10]$ and then we build $\mathcal{P}_\delta$ making a *meshgrid* of such values. The corresponding space is shown in Figure 2b. This choice is justified by the dependence of the solution on the parameters:

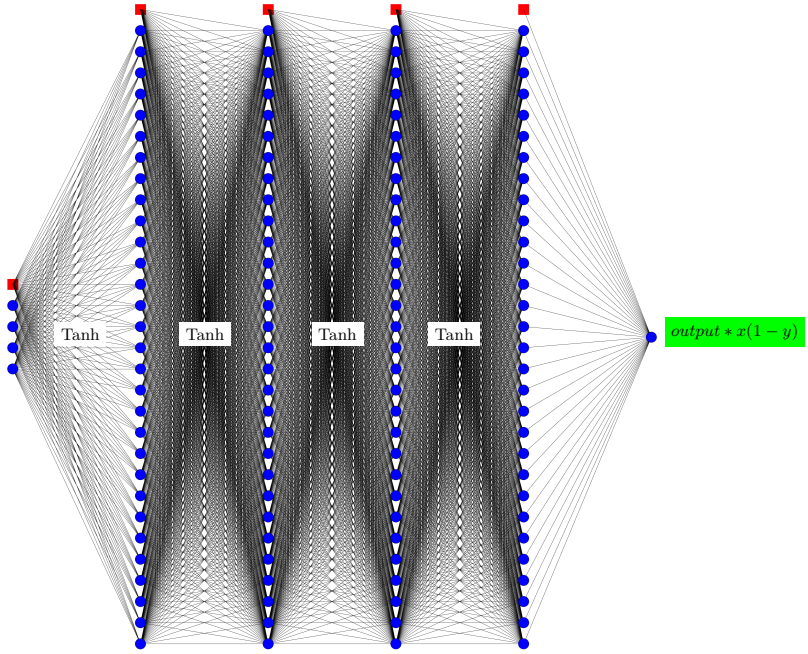


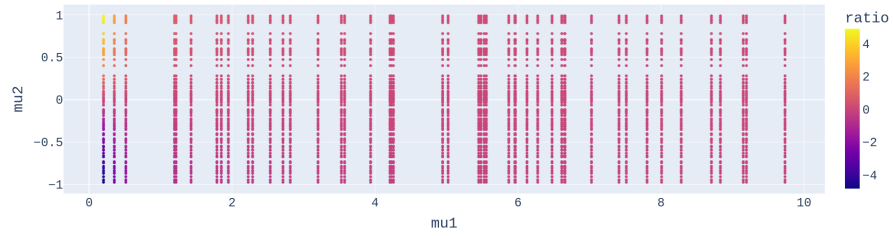
Figure 1: PINN architecture

when the ratio $\frac{\mu_2}{\mu 1}$ tends to zero the solution vanishes, while the bigger the module of the ratio the bigger the infinity norm of the solution. However, if we pick both μ_1 and μ_2 accordingly to a uniform distribution, we generate an imbalanced set with more than 95% of values of pair (μ_1, μ_2) with an absolute ratio smaller than 1, as highlighted from Figure 2a, while the other choice leads us to a more representative parameter space with just 70% of values of pair (μ_1, μ_2) with an absolute ratio smaller than 1.

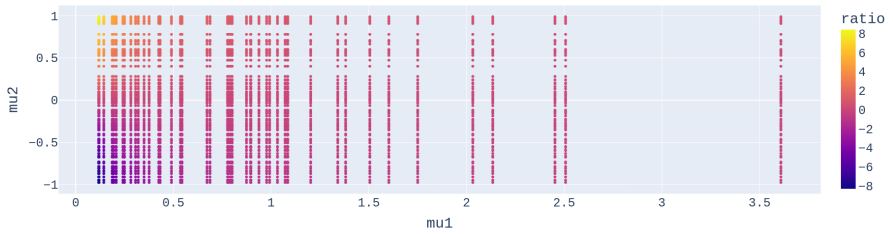
Concerning the Reduced Order Method, we define the set $\mathcal{P}_{\text{RB}}$ used in the greedy algorithm by randomly picking 100 values of the parameters $\boldsymbol{\mu}$ in $\mathcal{P}_{\delta,train}$ and we set $tol = 1.0e\text{-}05$ in the stopping criteria. We highlight that the high-fidelity method used to compute the basis functions of $\mathbb{V}_{\text{RB}}$ coincides with the model used as the exact model. The generated space $\mathbb{V}_{\text{RB}}$ has dimension $M = 4$.

Concerning the PINN approach, we train the NN for 30000 epochs and, at each epoch, we pick at random $N_p = 4000$ values of both $(x, y) \in (0, 1)^2$ and $\boldsymbol{\mu} \in \mathcal{P}_{\delta,train}$ for computing MSE_{PDE}, while we randomly select $N_b = N_r = 2000$ values of $\boldsymbol{\mu}$ for computing MSE_{BN}. Furthermore, we fix once at all $N_b = N_r = 2000$ equi-spaced values of (x, y) on both Γ_b and Γ_r to determine MSE_{BN}.

The accuracy and the speed-up of the resulting ROM and PINN approaches with respect to FEM are reported in Table 1. In particular, the accuracy is measured by the infinity norm of the difference between the FEM solution $u_\delta(\boldsymbol{\mu})$ and the solution obtained by PINN or ROM, at varying the parameter $\boldsymbol{\mu} \in \mathcal{P}_{\delta,test}$. We observe that the error is computed in the points (x, y) which are the vertices of the triangulation $\mathcal{T}_\delta$ used for computing the solutions of both FEM and ROM. Moreover, the speed-up is measured as the ratio between the time needed from the FEM to solve the discrete system (1.5) and the time needed from PINN or ROM for making predictions.



(a) $\mu_1 \sim \mathcal{U}(0.1, 10)$, $\mu_2 \sim \mathcal{U}(-1, 1)$



(b) $\mu_1 \sim Exp_{|[0.1,10]}(1)$, $\mu_2 \sim \mathcal{U}(-1, 1)$

Figure 2: Scatter plots of possible choice of the discrete parameter space. Colours denote the magnitude of the ratio $\frac{\mu_2}{\mu_1}$.

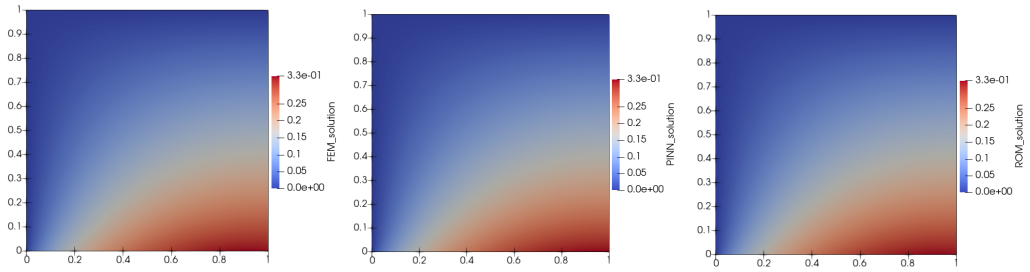
	$\|u_\delta(\boldsymbol{\mu}) - u^{\mathcal{NN}}(\boldsymbol{\mu})\|_\infty$	**Speed-up PINN**	$\|u_\delta(\boldsymbol{\mu}) - u_{RB}(\boldsymbol{\mu})\|_\infty$	**Speed-up ROM**
min	7.78e-04	1.77e+00	1.19e-19	2.19e+00
mean	4.59e-02	1.26e+01	1.27e-07	5.72e+00
max	7.88e-01	4.69e+01	6.59e-07	2.05e+01

Table 1: Statistics of the accuracy and of the speed-up related to both the PINN and the RB with respect to the FEM solution.

From this table 1, we observe that the infinity error norms are significantly low for the ROM strategy (the mean of the infinity error norms at varying $\boldsymbol{\mu} \in \mathcal{P}_{\delta,test}$ is about 10^{-7}), while the PINN solutions are on average much less accurate with respect to the FEM solutions (the mean error is $\approx 5e$-2). However, the PINN online time needed to make predictions is on average more than twice smaller than the time needed from ROM and about 47 times smaller than the full-order time.

We further observe that in each case (FEM, PINN and ROM) the boundary homogeneous Dirichlet conditions are imposed exactly by looking at Figure 3. This figure is related to the value $(\mu_1, \mu_2) = (1, 0.5)$ whose related ratio is $\mu_2/\mu_1 = 0.5 < 1$. We observe that, in this case, the PINN solution is very similar to the others. Indeed, despite the use of a more balanced parameter set, we observe that the greater PINN errors corresponding to the great values of the absolute ratio $\frac{\mu_2}{\mu_1}$, as shown in Figure 4, which reports the behaviour of the maximum and of the minimum values of the solution at varying of $\frac{\mu_2}{\mu_1}$ such that $(\mu_1, \mu_2) \in \mathcal{P}_{\delta,test}$.

As a further analysis, we test how our models predict the solutions at varying $\boldsymbol{\mu} \in \mathcal{P}_{\delta,test}$ when we restrict the values of parameters of the training set $\mathcal{P}_{\delta,test}$ in such a way $-1 \leq \mu_2/\mu_1 \leq 1$. From Figure 5, we note that the PINN is no longer able to predict solutions whose related values of the absolute ratio μ_2/μ_1 are higher than 1. Instead, the solutions obtained with the ROM



(a) FEM (b) PINN (c) ROM

Figure 3: Plot of the solution corresponding to the choice $\mu_1 = 1$ and $\mu_2 = 0.5$ (ratio $\frac{\mu_2}{\mu_1} = 0.5$). Left: FEM. Center: PINN. Right: ROM.

approach are again quite accurate.

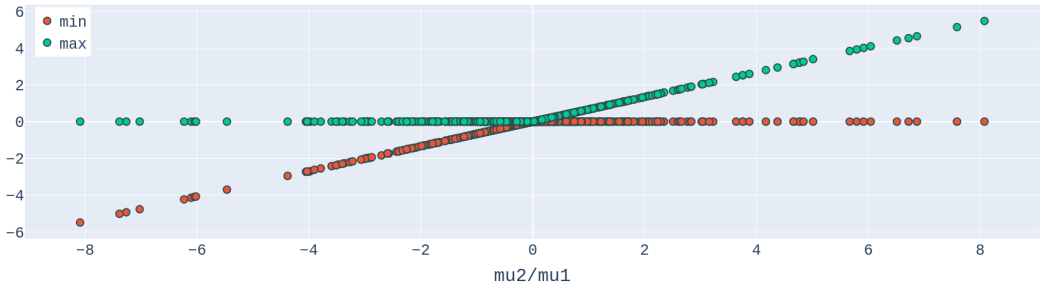
5 Conclusions

In this project, we compare the performances of the PINN and of the ROM approach built via a greedy algorithm when solving a parametric advection-diffusion problem in terms of accuracy and speed-up with respect to the FEM approach.

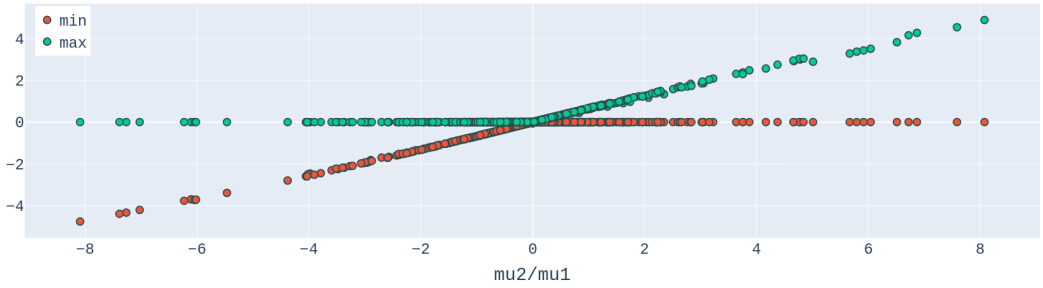
Since the problem is linear with respect to the parameters, the ROM solutions are the most accurate and the ROM offline stage requires a little computational time with respect to the PINN offline stage. Furthermore, the ROM approach is not dependent on the training set used in the greedy algorithm.

On the other hand, the time needed by PINN to make predictions is highly competitive, but its offline stage requires a greater computational time than the ROM. Furthermore, PINN requires a training set which is representative of the whole parameter space in order to be as much as possible accurate.

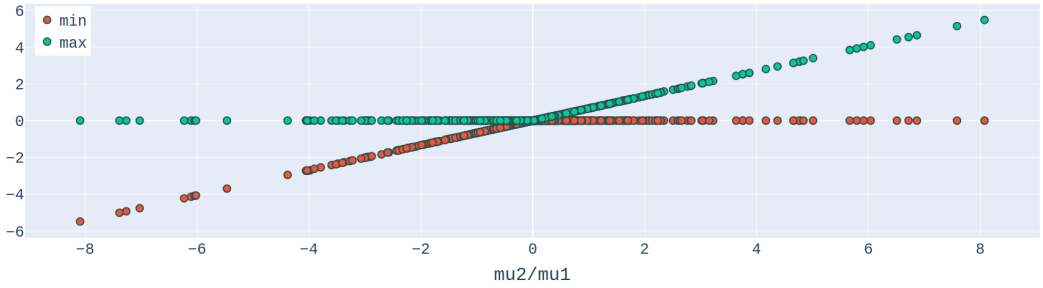
Moreover, we recall that the error is measured with respect to the high-fidelity model used to train the ROM. In this situation, the ROM approach is obviously the best one. In our opinion, in order to carry out a comparison between PINN and ROM, it is important to get out from the *comfort-zone* of the ROM, maybe by solving a highly non-linear problem with respect to the parameter and by comparing the results with respect to the exact solution or a solution obtained through a high order method. Furthermore, we recall that, differently from the ROM approach, the PINN is a meshless method and it requires the same computational cost to predict solutions in any point of the domain Ω independently of the points used to train it.



(a) FEM

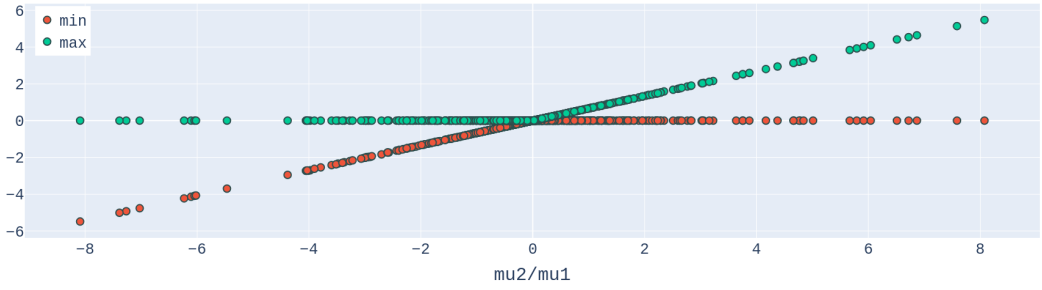


(b) PINN

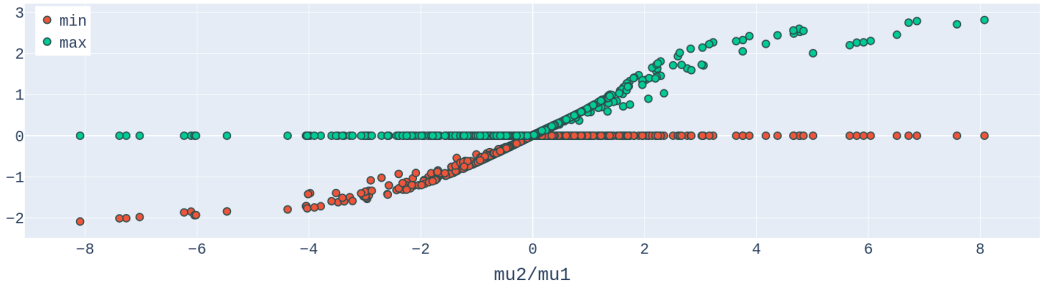


(c) ROM

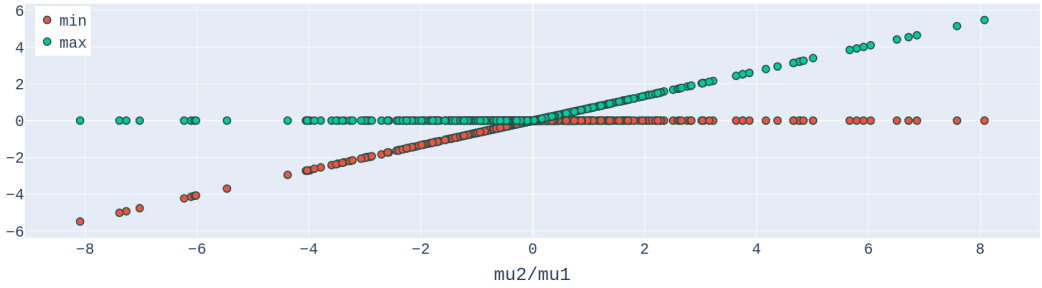
Figure 4: Minimum and maximum values of the solution at varying of the ratio μ_2/μ_1 for $(\mu_1, \mu_2) \in \mathcal{P}_{\delta,test}$. First row: FEM. Second row: PINN. Third row: ROM.



(a) FEM



(b) PINN



(c) ROM

Figure 5: Minimum and maximum values of the solution at varying of the ratio μ_2/μ_1 for $(\mu_1, \mu_2) \in \mathcal{P}_{\delta,test}$ in the case of the training set $\mathcal{P}_{\delta,train}$ is reduced to values (μ_1, μ_2) such that $\mu_2/\mu_1 \leq 1$. First row: FEM. Second row: PINN. Third row: ROM.