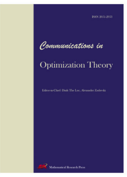
Commun. Optim. Theory 2022 (2022) 26 https://doi.org/10.23952/cot.2022.26

MATHRES

Communications in Optimization Theory

Available online at http://cot.mathres.org



A DATA-DRIVEN REDUCED ORDER METHOD FOR PARAMETRIC OPTIMAL BLOOD FLOW CONTROL: APPLICATION TO CORONARY BYPASS GRAFT

CATERINA BALZOTTI[1], PIERFRANCESCO SIENA[1], MICHELE GIRFOGLIO[1],
ANNALISA QUAINI[2,*], GIANLUIGI ROZZA[1]

[1]SISSA, International School for Advanced Studies, Mathematics Area, mathLab, via Bonomea 265, I-34136 Trieste, Italy

[2]University of Houston, Department of Mathematics, 3551 Cullen Blvd, 77204, Houston TX, USA

Dedicated to the memory of Roland Glowinski

Abstract. We consider an optimal flow control problem in a patient-specific coronary artery bypass graft with the aim of matching the blood flow velocity with given measurements as the Reynolds number varies in a physiological range. Blood flow is modelled with the steady incompressible Navier-Stokes equations. The geometry consists in a stenosed left anterior descending artery where a single bypass is performed with the right internal thoracic artery. The control variable is the unknown value of the normal stress at the outlet boundary, which is need for a correct set-up of the outlet boundary condition. For the numerical solution of the parametric optimal flow control problem, we develop a data-driven reduced order method that combines proper orthogonal decomposition (POD) with neural networks. We present numerical results showing that our data-driven approach leads to a substantial speed-up with respect to a more classical POD-Galerkin strategy proposed in [62], while having comparable accuracy.

Keywords. Coronary artery bypass; Neural networks; Optimal control; Reduced order model.

1. Introduction

It is well-known that cardiovascular disease is one of the leading causes of death in the world. This has motivated a large body of literature on mathematical and numerical modeling for blood flow problems since computational simulations in patient-specific geometries can assist medical doctors in clinical and surgical practice. This paper focuses on blood flow in coronary arteries when poor blood perfusion requires a surgical treatment known as coronary artery bypass graft (CABG). For recent computational work on this topic, we refer to, e.g., [2, 3, 8, 9, 50, 56, 57].

Computational hemodynamics in realistic geometries is challenging for several reasons. Here, we tackle two main challenges: (i) the high computational cost and (ii) the enforcement of

[*]Corresponding author.

E-mail address: cbalzott@sissa.it (C. Balzotti), psiena@sissa.it (P. Siena), mgirfogl@sissa.it (M. Girfoglio), quaini@math.uh.edu (A. Quaini), grozza@sissa.it (G. Rozza).

Received July 23, 2022; Accepted September 26, 2022.

boundary conditions that would lead to realistic flow. To deal with point (i), we develop a computationally efficient data-driven reduced order method (ROM). Our goal is not only to reduce drastically the computational time required by full order simulations, but also to achieve a speed-up with respect to a traditional reduced method (POD-Galerkin) used in [62]. As for point (ii), we set up an optimal control framework where the control variable is the unknown normal stress at the outlet boundary. Typically, one directly imposes boundary conditions with specified flow waveform at the inlets and constant values of pressure or velocity gradient at the outlets. See, e.g., [56, 57]. However, while this approach allows to take into account patient specific flow profiles, the non-physical outlet boundary conditions lead to inaccurate numerical approximations. An alternative is to employ surrogate models to enforce boundary conditions [22, 26, 27]. This results in increased accuracy, but requires a very large number of parameters which increases the model complexity too.

Our optimal control flow problem involves the minimization of an objective functional measuring the mismatch between computed and measured velocity fields and solves for the unknown control variable (i.e., the normal stress at the outlet boundary) with the constraint of the governing equations (i.e., the parameterized Navier-Stokes equations) [20,29,33,34,44,46]. The formulation of this optimal control problem was introduced in [62], with the assumption that the velocity data comes from magnetic resonance imaging 4D flow. In particular, we aim to minimize in a least-square sense the difference between numerical and measured velocity [35,48,60] in a patient specific geometry reconstructed from a computed tomography scan. In order to consider various hemodynamics scenarios, we let the Reynolds number vary in a given range. A finite element approach for the problem under consideration features high computational cost. To contain such cost, we adopt a data-driven ROM that combines proper orthogonal decomposition with neural networks (NNs) known as POD-NN [32]. The POD algorithm extracts the dominant modes that form a basis for the reduced space and NNs are used to interpolate the modal coefficients for each variable. During the so-called offline stage of the ROM, the POD algorithm is executed and the feedforward neural networks are trained. This is an expensive stage due to the high dimension of the full order model (i.e., the finite element method), which however needs to be performed only once. The so-called online phase requires only the evaluation of the modal coefficients through the trained NNs for each new Reynolds number of interest. The complete decoupling between offline/online phases together with a data-driven approach leads to a considerable computational efficiency, thereby allowing to solve the optimization problem for several hemodynamics scenarios at a greatly reduced cost.

The main reason for choosing NNs to interpolate the modal coefficients is that NNs are universal approximators [63], i.e., they are able to learn strongly non-linear trends with a simple trial and error process. This is a great advantage in the development of a ROM with several physical parameters (such as those associated with boundary conditions treated in this paper) and geometrical parameters (such as the degree of stenosis treated in [56]). In this paper, we do not consider geometrical parameters, and refer the reader interested in a thorough analysis of the performance of the POD-NN method for parametrized stenoses [56]. Other interpolation methods, such radial basis functions [38], are not universal like NNs and have certain regularity requirements to ensure convergence. The downsides of NNs are mainly two: the computational time needed to find and train the network optimal configuration and the (unknown) amount of data required to obtain satisfactory results.

A possible alternative to our approach is to skip the POD step and use NN surrogate models. This alternative has been explored in several recent papers. Particularly worthy of notice are references [25,39]. In [39], deep NNs are used to obtain pressure and velocity in patient-specific geometries of the thoracic aorta. The only input is the patient-specific geometry and the NN surrogate model produces the results in a few seconds. In [25], convolutional NNs are used to predict the time-averaged wall shear stress in a small tract of the left main coronary artery. In this case, various geometrical features (e.g., vessel radii and curvature) are provided as inputs. The advantages of these methods are that they are computationally cheap (once the NNs are trained) and they generalize well. The disadvantages are that they require a large number of patient-specific geometries (729 in [39] and 127 in [25]) and they have a high computational cost for the NN training. Since we currently do not have a large number of patient-specific geometries, NN surrogate models are not a viable choice for us.

The remainder of this paper is structured as follows. In Section 2, we report the mathematical details about the formulation of the continuous optimal control problem, together with the patient-specific geometry of the CABG. Section 3 describes the finite element discretization of our problem and introduces the data-driven reduced order framework. Numerical results are shown in Section 4. Conclusions and future perspectives are drawn in Section 5.

2. Optimal control problem

In this section, we introduce the mathematical details for optimal control flow problems in the context of a patient-specific CABG geometry.

In general, an optimal control flow problem is composed of three elements:

- an *objective functional* to optimize;
- a *control variable* to choose in order to minimize the objective functional;
- the *fluid flow model*, which represents a set of constraints for the optimization step.

More details can be found in, e.g., [14,24,29]. In our specific constrained optimization problem, the aim is to minimize an objective functional representing the distance between computational results and clinical data.

This section is organized as follows. First, we provide some information about the spatial domain. Then, we briefly describe the governing equations for blood flow in large vessels, i.e., the incompressible Navier-Stokes equations. Finally, we cast the mathematical problem into a saddle-point formulation, which we will argue is easier to handle.

2.1. **The geometry.** The CABG geometry used for this work is shown in Figure 1. A post-surgery computed tomography (CT) scan was provided by the *Sunnybrook Health Science Centre* in *Toronto*. The patient exhibited at least three stenoses in the coronary system ranging from mild to severe. We will consider only one of such stenoses to test our methodology. The bypass of interest is performed with the right internal thoracic artery (RITA) on the left anterior descending artery (LAD). See Figure 1. The approach used to extract the domain is based on *Visualization Toolkit* (VTK) and *Vascular Modeling Toolkit* (VMTK) Python libraries [6, 52] and it is accurately described in [8, 9].

2.2. **Problem formulation.** Let us consider the computational domain $\Omega \subset \mathbb{R}^3$ in Figure 1 and denote by $\partial\Omega$ its boundary. More precisely, the boundary consists of:

- two inlets denoted by Γ_{inlet}, where the flux enters the RITA and the LAD;

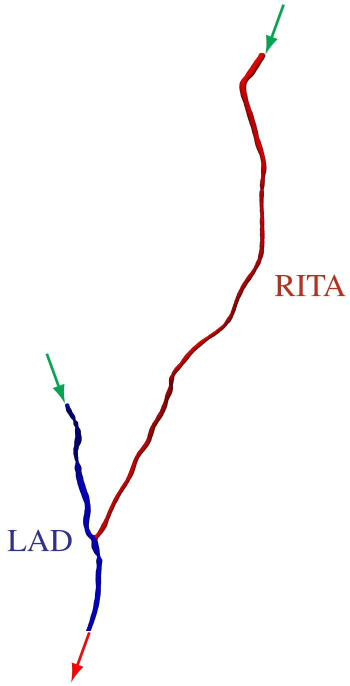


FIGURE 1. The spatial domain: a coronary artery bypass graft (CABG) performed with the right internal thoracic artery (RITA) on the left anterior descending artery (LAD). The arrows indicate the flow direction.

- one outlet Γ_{outlet}, where the blood leaves the LAD;
- the rigid vessel wall Γ_{wall}.

Due to the radii of the coronary arteries, the apparent viscosity of the blood can be considered constant. Therefore, we adopt a Newtonian model (see also [1, 61]).

Let $\boldsymbol{\mu} \in \mathscr{P}$ be a set of physical parameters belonging to a given parameter space $\mathscr{P}$. The parameterized, steady-state incompressible Navier-Stokes equations describing the blood flow are given by:

$$\begin{cases} -\nu\Delta\boldsymbol{v}(\boldsymbol{\mu}) + (\boldsymbol{v}(\boldsymbol{\mu})\cdot\nabla)\boldsymbol{v}(\boldsymbol{\mu}) + \nabla p(\boldsymbol{\mu}) = 0, & \text{in} \quad \Omega, \\ \nabla\cdot\boldsymbol{v} = 0, & \text{in} \quad \Omega, \end{cases} \tag{2.1}$$

where $\boldsymbol{v}(\boldsymbol{\mu})$ is the velocity field, $p(\boldsymbol{\mu})$ the blood pressure, and ν the constant viscosity. Problem (2.1) is endowed with a non-homogeneous Dirichlet condition at the inlet, a non-homogeneous Neumann condition at the outlet, and a no-slip condition on the wall:

$$\begin{cases} \boldsymbol{v}(\boldsymbol{\mu}) = \boldsymbol{v}_{\text{inlet}}(\boldsymbol{\mu}), & \text{on} \quad \Gamma_{\text{inlet}}, \\ -\nu(\nabla\boldsymbol{v}(\boldsymbol{\mu}))\boldsymbol{n} + p(\boldsymbol{\mu})\boldsymbol{n} = \boldsymbol{u}(\boldsymbol{\mu}), & \text{on} \quad \Gamma_{\text{outlet}}, \\ \boldsymbol{v}(\boldsymbol{\mu}) = \boldsymbol{0}, & \text{on} \quad \Gamma_{\text{wall}}, \end{cases} \tag{2.2}$$

where $\boldsymbol{v}_{\text{inlet}}(\boldsymbol{\mu})$ is a given inlet velocity profile, $\boldsymbol{n}$ is the outward unit normal vector, and $\boldsymbol{u}(\boldsymbol{\mu})$ is the unknown control variable.

Let us introduce the Hilbert spaces $V = V(\Omega)$ and $Q = Q(\Omega)$, with the corresponding dual spaces V^* and Q^*. Let $S = V \times Q$ be the self-adjoint space (i.e., $S = S^*$) of the state variables $(\boldsymbol{v}(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$. Moreover, let $U = U(\Gamma_{\text{outlet}})$ be the Hilbert space the control $\boldsymbol{u}$ belongs to. In order to state the weak formulation of problem (2.1)-(2.2), we need to introduce forms a :

$V \times V^* \to \mathbb{R}$, $e : V \times V \times V^* \to \mathbb{R}$, $b : Q^* \times V \to \mathbb{R}$ and $c : U \times V^* \to \mathbb{R}$:

$$a(\boldsymbol{v},\boldsymbol{w};\boldsymbol{\mu}) = \nu \int_\Omega \nabla \boldsymbol{v}(\boldsymbol{\mu}) \cdot \nabla \boldsymbol{w} d\Omega, \quad e(\boldsymbol{v},\boldsymbol{v},\boldsymbol{w};\boldsymbol{\mu}) = \int_\Omega (\boldsymbol{v}(\boldsymbol{\mu}) \cdot \nabla)\boldsymbol{v}(\boldsymbol{\mu}) \cdot \boldsymbol{w} d\Omega,$$
$$b(q,\boldsymbol{v};\boldsymbol{\mu}) = -\int_\Omega q(\nabla \cdot \boldsymbol{v}(\boldsymbol{\mu}))d\Omega, \quad c(\boldsymbol{u},\boldsymbol{w};\boldsymbol{\mu}) = -\int_{\Gamma_{outlet}} \boldsymbol{u}(\boldsymbol{\mu}) \cdot \boldsymbol{w} d\Gamma.$$

Then, the weak formulation can be written as: Given $\boldsymbol{\mu} \in \mathscr{P}$, find the solution $\boldsymbol{s}(\boldsymbol{\mu}) = (\boldsymbol{v}(\boldsymbol{\mu}), p(\boldsymbol{\mu}))$ $\in S$ such that:

$$\begin{cases} a(\boldsymbol{v},\boldsymbol{w};\boldsymbol{\mu}) + e(\boldsymbol{v},\boldsymbol{v},\boldsymbol{w};\boldsymbol{\mu}) + b(p,\boldsymbol{w};\boldsymbol{\mu}) + c(\boldsymbol{u},\boldsymbol{w};\boldsymbol{\mu}) = 0, & \forall \boldsymbol{w} \in V^*, \\ b(q,\boldsymbol{v};\boldsymbol{\mu}) = 0, & \forall \boldsymbol{q} \in Q^*. \end{cases} \tag{2.3}$$

The objective functional chosen for this work measures the distance between $\boldsymbol{v}$ and the clinically measured velocity $\boldsymbol{v}_{\mathrm{m}}$ as:

$$\mathscr{I}(\boldsymbol{v},\boldsymbol{u};\boldsymbol{\mu}) = \frac{1}{2}\int_\Omega |\boldsymbol{v}(\boldsymbol{\mu}) - \boldsymbol{v}_{\mathrm{m}}|^2 d\Omega + \frac{\alpha}{2}\int_{\Gamma_{outlet}} |\boldsymbol{u}(\boldsymbol{\mu})|^2 d\Gamma. \tag{2.4}$$

where $\alpha > 0$ is a constant penalization parameter. Let us introduce the Hilbert space $G \supseteq S$, bilinear forms $m : G \times G \to \mathbb{R}$ and $n : U \times U \to \mathbb{R}$ such that the cost functional (2.4) can be written as:

$$\mathscr{I}(\boldsymbol{v},\boldsymbol{u};\boldsymbol{\mu}) = \frac{1}{2} m(\boldsymbol{v}(\boldsymbol{\mu}) - \boldsymbol{v}_{\mathrm{m}}, \boldsymbol{v}(\boldsymbol{\mu}) - \boldsymbol{v}_{\mathrm{m}}) + \frac{\alpha}{2} n(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{u}(\boldsymbol{\mu})). \tag{2.5}$$

The optimal control flow problem reads:

> Given $\boldsymbol{\mu} \in \mathscr{P}$, find $(\boldsymbol{v}(\boldsymbol{\mu}), p(\boldsymbol{\mu}), \boldsymbol{u}(\boldsymbol{\mu})) \in V \times Q \times U$ such that the objective functional (2.5) is minimized under the constrain (2.3).

2.3. **Saddle-point structure.** A classical approach to treat non-linear optimal control problems is the adjoint-based Lagrangian method. Here, we briefly describe the procedure adopted to cast the constrained optimization problem with the well-known saddle-point theory. The reader interested in more details is referred to, e.g., [15, 16, 44].

Let us introduce $X = V \times Q \times U$ and denote by $\boldsymbol{x} = (\boldsymbol{v}, p, \boldsymbol{u}) \in X$ the state and the control variables of the optimal control problem. In addition, let $\boldsymbol{y} = (\boldsymbol{y}_{\boldsymbol{v}}, \boldsymbol{y}_p, \boldsymbol{y}_{\boldsymbol{u}}) \in X$. We define $\mathscr{A} : X \times X \to \mathbb{R}$ as:

$$\mathscr{A}(\boldsymbol{x},\boldsymbol{y};\boldsymbol{\mu}) = m(\boldsymbol{v}(\boldsymbol{\mu}), \boldsymbol{y}_{\boldsymbol{v}}) + \alpha n(\boldsymbol{u}(\boldsymbol{\mu}), \boldsymbol{y}_{\boldsymbol{u}}), \quad \forall \boldsymbol{x}, \boldsymbol{y} \in X.$$

Moreover, let $\mathscr{B} : X \times S^* \to \mathbb{R}$ be the operator related to the linear part of (2.3), such that we can rewrite (2.3) as:

$$\mathscr{B}(\boldsymbol{x}, \boldsymbol{z};\boldsymbol{\mu}) + e(\boldsymbol{v},\boldsymbol{v},\boldsymbol{w};\boldsymbol{\mu}) = 0, \quad \forall \boldsymbol{z} \in S^*, \tag{2.6}$$

with $\boldsymbol{z} = (\boldsymbol{w}, q)$. Finally, let $\boldsymbol{h}(\boldsymbol{\mu}) \in X^*$ be such that

$$\langle \boldsymbol{h}(\boldsymbol{\mu}), \boldsymbol{y} \rangle = m(\boldsymbol{v}, \boldsymbol{v}_{\mathrm{m}}). \quad \forall \boldsymbol{y} \in X,$$

Therefore, a new functional can be introduced:

$$J(\boldsymbol{x};\boldsymbol{\mu}) = \frac{1}{2}\mathscr{A}(\boldsymbol{x},\boldsymbol{x};\boldsymbol{\mu}) - \langle \boldsymbol{h}(\boldsymbol{\mu}), \boldsymbol{x} \rangle = \mathscr{J}(\boldsymbol{v},\boldsymbol{u};\boldsymbol{\mu}) - \frac{1}{2} m(\boldsymbol{v}_{\mathrm{m}}, \boldsymbol{v}_{\mathrm{m}}). \tag{2.7}$$

Since $m(\boldsymbol{v}_{\mathrm{m}}, \boldsymbol{v}_{\mathrm{m}})$ is constant, an equivalent form of the optimal control flow problem is:

Given $\boldsymbol{\mu} \in \mathscr{P}$, find $\boldsymbol{x}(\boldsymbol{\mu}) = (\boldsymbol{v}(\boldsymbol{\mu}), p(\boldsymbol{\mu}), \boldsymbol{u}(\boldsymbol{\mu})) \in X$ such that the objective functional (2.7) is minimized under the constrain (2.6).

In order to obtain an unconstrained formulation, let us define the Lagrangian $\mathscr{L} : X \times S^* \to \mathbb{R}$:

$$\mathscr{L}(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\mu}) = J(\boldsymbol{x}; \boldsymbol{\mu}) + \mathscr{B}(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\mu}) + e(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{w}; \boldsymbol{\mu}).$$

The saddle-point structure is obtained with the optimality condition: $\nabla \mathscr{L}(\boldsymbol{x}, \boldsymbol{z}; \boldsymbol{\mu})[\boldsymbol{y}, \boldsymbol{k}] = 0, \forall \boldsymbol{y} \in X, \forall \boldsymbol{k} = (\boldsymbol{k}_{\boldsymbol{w}}, \boldsymbol{k}_q) \in S^*$. This leads to the following formulation:

Given $\boldsymbol{\mu} \in \mathscr{P}$, find $(\boldsymbol{x}(\boldsymbol{\mu}), \boldsymbol{z}(\boldsymbol{\mu})) \in X \times S^*$ such that:

$$\begin{cases} \mathscr{A}(\boldsymbol{x}, \boldsymbol{y}; \boldsymbol{\mu}) + \mathscr{B}(\boldsymbol{y}, \boldsymbol{z}; \boldsymbol{\mu}) + e(\boldsymbol{y}_{\boldsymbol{v}}, \boldsymbol{v}, \boldsymbol{w}; \boldsymbol{\mu}) + e(\boldsymbol{v}, \boldsymbol{y}_{\boldsymbol{v}}, \boldsymbol{w}; \boldsymbol{\mu}) = \langle \boldsymbol{h}(\boldsymbol{\mu}), \boldsymbol{y} \rangle, \quad \forall \boldsymbol{y} \in X, \\ \mathscr{B}(\boldsymbol{x}, \boldsymbol{k}; \boldsymbol{\mu}) + e(\boldsymbol{v}, \boldsymbol{v}, \boldsymbol{k}_{\boldsymbol{w}}; \boldsymbol{\mu}) = 0, \quad \forall \boldsymbol{k} \in S^*. \end{cases} \tag{2.8}$$

Existence and uniqueness of the solution to the above problem is ensured by Brezzi's theorem (see [16] for details).

3. Numerical discretization

In this section, we introduce the numerical methods to find the fluid state variables, the control variable and the adjoint variables. To find the full order solution of problem (2.8), we adopt an *optimize-then-discretize* approach. This procedure formalizes the continuous problem and then discretizes the optimal control system. We choose a finite element method for the discretization and use a *one-shot* method to solve directly the resulting equations. See, e.g., [29,53] for details.

For the reduction step, we opt for a *data-driven*, *non-intrusive* ROM. More precisely, we employ the POD-NN method [32]. Our aim is to compare our results with those obtained with an intrusive POD-Galekin procedure in [62].

3.1. **The full order model.** We consider a mesh for domain Ω with size h. Let us consider spaces $S_h = V_h \times Q_h \subset S$ and $X_h = S_h \times U_h \subset X$. The full order model (FOM) is:

Given $\boldsymbol{\mu} \in \mathscr{P}$, find $(\boldsymbol{x}_h(\boldsymbol{\mu}), \boldsymbol{z}_h(\boldsymbol{\mu})) \in X_h \times S_h^*$ such that:

$$\begin{cases} \mathscr{A}(\boldsymbol{x}_h, \boldsymbol{y}_h; \boldsymbol{\mu}) + \mathscr{B}(\boldsymbol{y}_h, \boldsymbol{z}_h; \boldsymbol{\mu}) + e(\boldsymbol{y}_{\boldsymbol{v}_h}, \boldsymbol{v}_h, \boldsymbol{w}_h; \boldsymbol{\mu}) + e(\boldsymbol{v}_h, \boldsymbol{y}_{\boldsymbol{v}_h}, \boldsymbol{w}_h; \boldsymbol{\mu}) = \langle \boldsymbol{h}(\boldsymbol{\mu}), \boldsymbol{y}_h \rangle, \quad \forall \boldsymbol{y}_h \in X_h, \\ \mathscr{B}(\boldsymbol{x}_h, \boldsymbol{k}_h; \boldsymbol{\mu}) + e(\boldsymbol{v}_h, \boldsymbol{v}_h, \boldsymbol{k}_{\boldsymbol{w}_h}; \boldsymbol{\mu}) = 0, \quad \forall \boldsymbol{k}_h \in S_h^*. \end{cases} \tag{3.1}$$

We refer to eq. (3.1) as the *truth problem*, which has dimension $N = \sum_\delta N_\delta$, with N_δ the dimension of the solution space for variable $\delta = \boldsymbol{v}, p, \boldsymbol{u}, \boldsymbol{w}, q$.

In order to introduce the matrix formulation of the truth problem, let $\{\phi_i\}_{i=1}^{N_{\boldsymbol{v}}}$, $\{\psi_k\}_{k=1}^{N_p}$ and $\{\sigma_l\}_{l=1}^{N_{\boldsymbol{u}}}$ be bases for V_h, Q_h and U_h, respectively. The state and control variables can be written as linear combination of the basis functions:

$$\boldsymbol{v}_h(\boldsymbol{\mu}) = \sum_{i=1}^{N_{\boldsymbol{v}}} \bar{v}_i(\boldsymbol{\mu}) \phi_i, \quad p_h(\boldsymbol{\mu}) = \sum_{k=1}^{N_p} \bar{p}_k(\boldsymbol{\mu}) \psi_k, \quad \boldsymbol{u}_h(\boldsymbol{\mu}) = \sum_{l=1}^{N_{\boldsymbol{u}}} \bar{u}_l(\boldsymbol{\mu}) \sigma_l,$$

where $\bar{\boldsymbol{v}}(\mu) = [\bar{v}_1, \dots, \bar{v}_{N_{\boldsymbol{v}}}]^T$, $\bar{\boldsymbol{p}}(\mu) = [\bar{p}_1, \dots, \bar{p}_{N_p}]^T$ and $\bar{\boldsymbol{u}}(\mu) = [\bar{u}_1, \dots, \bar{u}_{N_{\boldsymbol{u}}}]^T$ are the vectors of the coefficients. Taking the basis functions as test functions for problem (3.1), we introduce the matrices associated with the forms in (2.3):

$$(A(\boldsymbol{\mu}))_{ij} = a(\phi_i, \phi_j; \boldsymbol{\mu}), \quad (B(\boldsymbol{\mu}))_{ik} = b(\psi_k, \phi_i; \boldsymbol{\mu}), \quad (C(\boldsymbol{\mu}))_{il} = c(\sigma_l, \phi_i; \boldsymbol{\mu}),$$

$$(M(\boldsymbol{\mu}))_{ij} = m(\phi_i, \phi_j; \boldsymbol{\mu}), \quad N(\boldsymbol{\mu})_{lr} = n(\sigma_r, \sigma_l; \boldsymbol{\mu}),$$

with $1 \le i, j \le N_{\boldsymbol{v}}$, $1 \le k \le N_p$ and $1 \le l, r \le N_{\boldsymbol{u}}$. Furthermore, we define:

$$(E(\bar{\boldsymbol{v}}(\boldsymbol{\mu}); \boldsymbol{\mu}))_{ij} = \sum_{k=1}^{N_v} \bar{v}_k(\boldsymbol{\mu}) e(\phi_k, \phi_j, \phi_i; \boldsymbol{\mu}), \quad (\tilde{E}(\bar{\boldsymbol{w}}(\boldsymbol{\mu}); \boldsymbol{\mu}))_{ij} = \sum_{k=1}^{N_v} \bar{w}_k(\boldsymbol{\mu}) e(\phi_k, \phi_j, \phi_i; \boldsymbol{\mu}),$$

where $\bar{\boldsymbol{w}}(\mu) = [\bar{w}_1, \dots, \bar{w}_{N_v}]^T$ is the vector of the coefficients of the adjoint velocity and $\bar{\boldsymbol{q}}(\mu) = [\bar{q}_1, \dots, \bar{q}_{N_p}]^T$ is the vector of the coefficients for the adjoint pressure.

The algebraic formulation of the truth problem can be written as:

$$\left\{\begin{matrix} M(\boldsymbol{\mu}) + \tilde{E}(\bar{\boldsymbol{w}}(\boldsymbol{\mu}); \boldsymbol{\mu}) & 0 & 0 & A(\boldsymbol{\mu}) + E(\bar{\boldsymbol{v}}(\boldsymbol{\mu}); \boldsymbol{\mu}) & B(\boldsymbol{\mu}) \\ 0 & 0 & 0 & B^T(\boldsymbol{\mu}) & 0 \\ 0 & 0 & \alpha N(\boldsymbol{\mu}) & C(\boldsymbol{\mu}) & 0 \\ A(\boldsymbol{\mu}) + E(\bar{\boldsymbol{v}}(\boldsymbol{\mu}); \boldsymbol{\mu}) & B^T(\boldsymbol{\mu}) & C(\boldsymbol{\mu}) & 0 & 0 \\ B(\boldsymbol{\mu}) & 0 & 0 & 0 & 0 \end{matrix}\right\} \left\{\begin{matrix} \bar{\boldsymbol{v}}(\boldsymbol{\mu}) \\ \bar{\boldsymbol{p}}(\boldsymbol{\mu}) \\ \bar{\boldsymbol{u}}(\boldsymbol{\mu}) \\ \bar{\boldsymbol{w}}(\boldsymbol{\mu}) \\ \bar{\boldsymbol{q}}(\boldsymbol{\mu}) \end{matrix}\right\} = \left\{\begin{matrix} \boldsymbol{h}(\boldsymbol{\mu}) \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \\ \boldsymbol{0} \end{matrix}\right\}. \tag{3.2}$$

One way to ensure uniqueness of the solution for the above system is to employ inf-sup stable finite element pairs for velocity and pressure. See, e.g., [4, 10] for more details.

3.2. **The reduced order model.** Like many other ROM techniques, the POD-NN method relies on the *offline-online* paradigm, i.e., computationally intensive simulations are carried out on powerful computational facilities during the offline phase and computationally cheap calculations for every new parameter of interest are run on any device during the online phase. More specifically:

- during the *offline* phase: a set of high-fidelity solutions is collected for a wide range of parameter values and a reduced basis for the space of the reduced solutions is extracted via POD. In this stage, the training of the neural networks is performed in order to establish the relationship between parameters and coefficients of the reduced solutions (modal coefficients). Due to the large number of degree of freedom, the offline phase is computationally expensive. However, it is carried out only once.
- during the *online* phase: the modal coefficients for every new parameter are quickly obtained from the trained neural networks. The reduced solution is given by the linear combination of the reduced basis functions with the modal coefficients as weights. The computational cost of the online phase is much smaller than the cost of the offline phase.

Next, we are going to describe the building blocks of the above algorithm.

3.2.1. *Proper orthogonal decomposition.* The POD procedure is widely used to compress data [7, 12, 13, 17, 31]. We adopt it to extract an orthonormal basis with a least squares approach [37, 45].

Let us consider the discrete finite dimensional set $\{\boldsymbol{\mu}_1, \dots, \boldsymbol{\mu}_L\} \subset \mathscr{P}$. The FOM is solved for each parameter value $\boldsymbol{\mu}_l, l = 1, \dots, L$, and the corresponding high fidelity solutions $\bar{\Phi} =$

$\{\bar{\boldsymbol{v}}, \bar{\boldsymbol{p}}, \bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \bar{\boldsymbol{q}}\}$ called snapshots, are arranged in the snapshot matrix S_Φ as follows:

$$S_\Phi = \begin{Bmatrix} \bar{\Phi}_1(\boldsymbol{\mu}_1) & \cdots & \bar{\Phi}_1(\boldsymbol{\mu}_L) \\ \vdots & \vdots & \vdots \\ \bar{\Phi}_{N_\delta}(\boldsymbol{\mu}_1) & \cdots & \bar{\Phi}_{N_\delta}(\boldsymbol{\mu}_L) \end{Bmatrix}.$$

The POD space can be extracted by solving the following eigenvalue problem [37]:

$$C_\Phi \boldsymbol{c}_l = \lambda_l \boldsymbol{c}_l, \quad l = 1, \ldots, L,$$

where $C_\Phi = \frac{1}{L} S_\Phi^T S_\Phi \in \mathbb{R}^{L \times L}$ is the correlation matrix associated with the snapshots. For fixed $R \ll L$, the reduced orthonormal basis is constructed as follows:

$$\boldsymbol{w}_r = \frac{1}{\sqrt{\lambda_r}} S_\Phi \boldsymbol{c}_r, \quad r = 1, \ldots, R.$$

The value of R is commonly chosen to reach a user-provided threshold ε for the cumulative energy of the eigenvalues:

$$\frac{\sum_{i=1}^R \lambda_i}{\sum_{i=1}^L \lambda_i} \geq \varepsilon. \tag{3.3}$$

Let us store the vectors of the reduced basis functions (also called modes) in matrix V_δ:

$$V_\delta = [\boldsymbol{w}_1 | \ldots | \boldsymbol{w}_R] \in \mathbb{R}^{N_\delta \times R}. \tag{3.4}$$

Then, the reduced solution $\Phi_\mathrm{r}(\boldsymbol{\mu}_l)$ can be written as:

$$\Phi_\mathrm{r}(\boldsymbol{\mu}_l) = \sum_{r=1}^R ({V_\delta}^T \bar{\Phi}(\boldsymbol{\mu}_l))_r \boldsymbol{w}_r, \quad l = 0, \ldots, L, \tag{3.5}$$

where $({V_\delta}^T \bar{\Phi}(\boldsymbol{\mu}_l))_r$ is the modal coefficient associated with the r-th mode. Note that the POD space coincides with the resolution of the following minimization problem [37, 45]:

$$\min_{V_\delta} \| S_\Phi - V_\delta {V_\delta}^T S_\Phi \| \quad s.t. \quad {V_\delta}^T V_\delta = I, \tag{3.6}$$

which optimizes the distance between the snapshots and their projection onto the reduced space.

3.2.2. *Neural network interpolation.* We choose a fully connected feedforward neural network [18, 28, 36], also called perceptron. Such network is a set of layered neurons (or nodes), where each neuron of a layer is connected with all the neurons of the next layer through oriented edges (or synapses) [23,41,49]. The input layer takes the set of parameters and the output layer returns the modal coefficients. Others neurons form the hidden layers (see Figure 2).

During the training process in the offline phase, we optimize the weights of the synapses with the backpropagation algorithm [47, 51]. A loss function, which measures the distance between the actual output and the required output, is minimized by computing (backward) the gradient with respect to the weights. Hyperparameters such as the activation function, the number of layers, the number of neurons per layer and the learning rate are tuned to improve and accelerate the learning process (see [42, 55] for further details). The values resulting from this tuning procedure are reported in Sec. 4.

The neural network offers an approximation $\boldsymbol{\pi}^{\mathrm{NN}}$ of the function:

$$\boldsymbol{\pi} : \boldsymbol{\mu}_l \mapsto [({V_\delta}^T \bar{\Phi}(\boldsymbol{\mu}_l))_r]_{r=1}^R, \quad l = 1, \ldots, L. \tag{3.7}$$

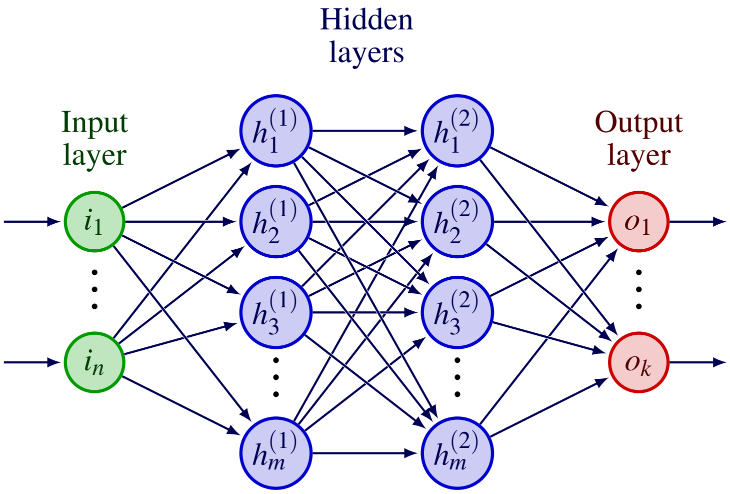


FIGURE 2. Feedforward neural network.

Once the network is trained, the reduced solution can be computed for every new parameter value $\boldsymbol{\mu}_{\text{new}}$ during the online phase as [19, 32, 43, 54]:

$$\Phi_{\text{r}}(\boldsymbol{\mu}_{\text{new}}) = \sum_{r=1}^{R} \pi_r^{\text{NN}}(\boldsymbol{\mu}_{\text{new}})\boldsymbol{w}_r, \tag{3.8}$$

where π_r^{NN} is the r-th component of $\boldsymbol{\pi}^{\text{NN}}$.

Next, we briefly discuss backpropagation and hyperparameters, which are fundamental concepts for every algorithm of NN optimization. Let us denote by $\mathfrak{L} = \mathfrak{L}(\boldsymbol{\pi}^{\text{NN}}(\boldsymbol{\mu}_l), \boldsymbol{\pi}(\boldsymbol{\mu}_l))$ the loss function of choice. For regression tasks, one typically chooses the mean squared error:

$$\mathfrak{L} = \sum_{l=1}^{L} \frac{1}{R} \sum_{r=1}^{R} (\pi_r(\boldsymbol{\mu}_l) - \pi_r^{\text{NN}}(\boldsymbol{\mu}_l))^2. \tag{3.9}$$

In order to understand the training process, let us recall three functions that characterize a generic neuron j, i.e., the propagation function, the activation function, and the output function. The propagation function converts the arriving input y_k (i.e., the output of the sending neuron k) into a scalar output u_j through a weighted sum:

$$u_j = \sum_{k=1}^{m} w_{k,j} y_k + b_j,$$

where b_j is the bias, $w_{k,j}$ are the weights of the synapses, and m is the number of neurons connected with the neuron j. The weights can be stored in a matrix W and the biases in a vector $\boldsymbol{b}$. The activation function decides how u_j is converted into an output:

$$a_j = f_{\text{act}}(u_j).$$

Finally, the output function f_{out} can modify again the product of a neuron. Often time though, it coincides with the identity:

$$\pi_j^{\text{NN}} = f_{\text{out}}(a_j) = a_j.$$

The minimization of the distance between the actual output and the correct one (3.9) is carried on thought the computation of the gradient of $\mathfrak{L}$ using the chain rule. Indeed, thanks to the functions characterizing a neuron, the gradient of the loss function can be computed with respect to the weights $w_{k,j}$ and the biases b_j as follows:

$$\begin{aligned} \frac{\partial \mathfrak{L}}{\partial w_{k,j}} &= \frac{\partial \mathfrak{L}}{\partial a_j}\frac{\partial a_j}{\partial u_j}\frac{\partial u_j}{\partial w_{k,j}}, \\ \frac{\partial \mathfrak{L}}{\partial b_j} &= \frac{\partial \mathfrak{L}}{\partial a_j}\frac{\partial a_j}{\partial u_j}\frac{\partial u_j}{\partial b_j}. \end{aligned} \tag{3.10}$$

During the training process, the forward pass computes the values of the output layers from the inputs data (i.e, the u_j) and the value of the loss function. After each forward pass, *back-propagation* measures the gradient of $\mathfrak{L}$ "backwards" as shown in (3.10). Finally, W and $\boldsymbol{b}$ are updated as follows:

$$\begin{aligned} W &\longleftarrow W - \eta\frac{\partial \mathfrak{L}}{\partial W}, \\ \boldsymbol{b} &\longleftarrow \boldsymbol{b} - \eta\frac{\partial \mathfrak{L}}{\partial \boldsymbol{b}}, \end{aligned} \tag{3.11}$$

where η is the learning rate. The specific algorithm of NN optimization we use in this paper is the Adam optimizer, which typically performs well,

In Sec. 4, we will carry out the tuning of the learning rate and number of neurons per layer during the learning process.

4. Numerical results

In order to compare our methodology with the one proposed in [62], we consider the same numerical setting, which is briefly recalled in the following. Using TetGen [30], we build a triangular mesh with a given size over the boundary $\partial\Omega$ and a tetrahedral mesh of the same size inside the volume Ω. The total number of elements for the generated mesh is 42354. The inf-sup stable $\mathbb{P}2-\mathbb{P}1$ elements are employed for velocity and pressure.

Let $\boldsymbol{t}_c$ be the vector tangent to the vessel centerline (obtained using VMTK as explained in [62]) in axial direction, R the maximum vessel radius (with respect to the centerline) and r the distance between the mesh nodes and the centerline. We defined the desired blood flow velocity in the entire domain Ω as

$$\boldsymbol{v}_{\mathrm{m}} = v_{\mathrm{const}}\left(1-\frac{r^2}{R^2}\right)\boldsymbol{t}_c,$$

where $v_{\mathrm{const}} = 350\,\mathrm{mm/s}$ is the desired velocity magnitude. At the inlets, we prescribe the following velocity:

$$\boldsymbol{v}_{\mathrm{inlet}} = \frac{\nu Re}{R_{\mathrm{inlet}}}\left(1-\frac{r^2}{R^2_{\mathrm{inlet}}}\right)\boldsymbol{n}_{\mathrm{inlet}}, \tag{4.1}$$

where $\nu = 3.6\,\mathrm{mm^2/s}$ is the constant kinematic viscosity, Re is the Reynolds number, R_{inlet} is the maximum radius of the given inlet surface, and $\boldsymbol{n}_{\mathrm{inlet}}$ denotes the outward unit normal to the inlet surface. Notice the inlet velocity profiles are parameterized by the Reynolds number in (4.1), which is the variable parameter for the study presented in this Section, while the "measured" velocity $\boldsymbol{v}_{\mathrm{m}}$ is independent of Re. In fact, we want to address the case when $\boldsymbol{v}_{\mathrm{m}}$ comes from in-vivo flow measurements and the associated Reynolds number is expected to vary in a certain range. Our optimal control problem will identify the normal stress that must be imposed at the outlet so that the computed velocity field is the best approximation of $\boldsymbol{v}_{\mathrm{m}}$ for each chosen Re in the expected range. We set the regularization parameter α to 1e-02.

Recalling that $\bar{\Phi} = \{\bar{\boldsymbol{v}}, \bar{\boldsymbol{p}}, \bar{\boldsymbol{u}}, \bar{\boldsymbol{w}}, \bar{\boldsymbol{q}}\}$ is the *truth solution* and $\Phi_{\mathrm{r}} = \{\boldsymbol{v}_{\mathrm{r}}, \boldsymbol{p}_{\mathrm{r}}, \boldsymbol{u}_{\mathrm{r}}, \boldsymbol{w}_{\mathrm{r}}, \boldsymbol{q}_{\mathrm{r}}\}$ is the *reduced solution*, we introduce the absolute and relative L^2 errors defined as:

$$\varepsilon_{\Phi}^{\mathrm{abs}} = \left\|\bar{\Phi} - \Phi_{\mathrm{r}}\right\|_{L^2}, \qquad \varepsilon_{\Phi}^{\mathrm{rel}} = \frac{\varepsilon_{\Phi}^{\mathrm{abs}}}{\left\|\bar{\Phi}\right\|_{L^2}}, \tag{4.2}$$

respectively. The total relative error is defined as:

$$\varepsilon^{\mathrm{rel}} = \frac{\left((\varepsilon_{\boldsymbol{v}}^{\mathrm{abs}})^2 + (\varepsilon_{p}^{\mathrm{abs}})^2 + (\varepsilon_{\boldsymbol{u}}^{\mathrm{abs}})^2 + (\varepsilon_{\boldsymbol{w}}^{\mathrm{abs}})^2 + (\varepsilon_{q}^{\mathrm{abs}})^2\right)^{1/2}}{(\|\bar{\boldsymbol{v}}\|_{L^2}^2 + \|\bar{\boldsymbol{p}}\|_{L^2}^2 + \|\bar{\boldsymbol{u}}\|_{L^2}^2 + \|\bar{\boldsymbol{w}}\|_{L^2}^2 + \|\bar{\boldsymbol{q}}\|_{L^2}^2)^{1/2}}. \tag{4.3}$$

Following [62], we let the Reynolds number vary in the interval $[70, 80]$. The choice of this interval is to ensure that the flow remains laminar everywhere in the domain, to demonstrate the feasibility of our approach, and to compare its performance with the method in [62]. We consider equispaced values $\mu_l = Re$, for $l = 1, \dots, L$ with $L = 100$ as in [62]. We split the finite dimensional set of parameters into a train set $\boldsymbol{\mu}^{\mathrm{train}}$ and a test set $\boldsymbol{\mu}^{\mathrm{test}}$. The former is used during the offline phase to build the reduced basis space and to train the feedforward neural networks, while the latter is used during the online phase to validate the method by reconstructing the solutions and comparing them with the corresponding snapshots. We analyze several sets of training parameters, in order to find the one that minimizes the average relative error and use that in our tests. Specifically, we set $\boldsymbol{\mu}^{\mathrm{train}} = \{\mu_{(k-1)m}\}$, for $k \in \{2, 4, 20, 50\}$ and $m = 1, \dots, L/k$, and $\boldsymbol{\mu}^{\mathrm{test}} = \{\mu_l \,|\, \mu_l \notin \boldsymbol{\mu}^{\mathrm{train}}\}$, for $l = 1, \dots, L$. Let us set the number of modes to 6. Figure 3 shows the variation of the mean relative error for each variable for the different $\boldsymbol{\mu}^{\mathrm{train}}$. When the number of snapshots is greater than 2, each relative error oscillates between 10^{-3} and 10^{-4}, with small variations. Since the optimal result for most variables is for 50 snapshots, for the tests in Sec. 4, we choose $\boldsymbol{\mu}^{\mathrm{train}} = \{\mu_{2m-1}\}$ and $\boldsymbol{\mu}^{\mathrm{test}} = \{\mu_{2m}\}$, with $m = 1, \dots, L/2$.

The solutions associated with all the parameters μ_l, with $l = 1, \dots, L$, are computed during the offline phase using Python libraries FEniCS [5, 40] and *multiphenics* [11]. This represents the snapshots set. Then, we consider subset $\boldsymbol{\mu}^{\mathrm{train}}$ to build the snapshots matrix S_{Φ} and the corresponding reduced space through the POD algorithm as explained in Sec. 3.2. In Figure 4, we show the eigenvalues associated with the POD on the snapshots matrix S_{Φ} for all variables using $n = 8$ reduced basis functions. We see that for every n the largest eigenvalue is associated with the pressure, while the smallest eigenvalue is associated with the control variable. In addition, the eigenvalues for the velocity and pressure snapshots matrices are larger than the eigenvalues of the respective adjoint counterpart. Table 1 reports the cumulative energy (3.3) of the first three eigenvalues. We observe that more than the 99.99% of the energy of the eigenvalues is retained with the first two eigenvalue for all the variables.

Once the POD step is completed, the offline phase is concluded by training the neural networks to compute the modal coefficients. The entire ROM has been implemented within Python package EZyRB [21], developed and maintained by the mathLab group at SISSA. For the neural networks, we choose the Sigmoid activation function $f(x) = (1 + e^{-x})^{-1}$ and we consider two

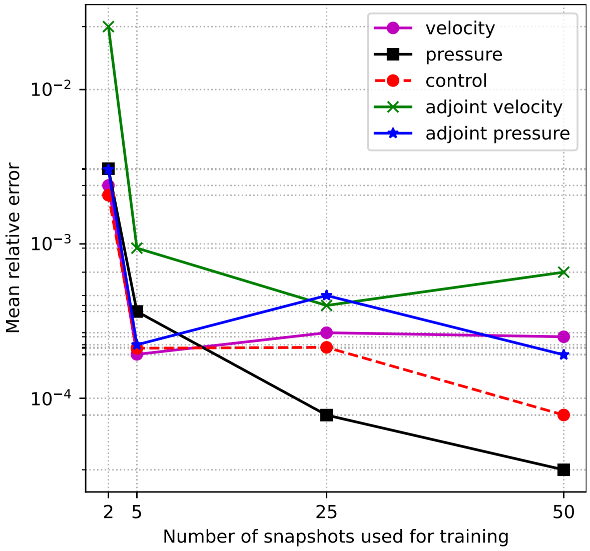


FIGURE 3. Variation of the mean relative error for each variable as the number of snapshots used for training changes.

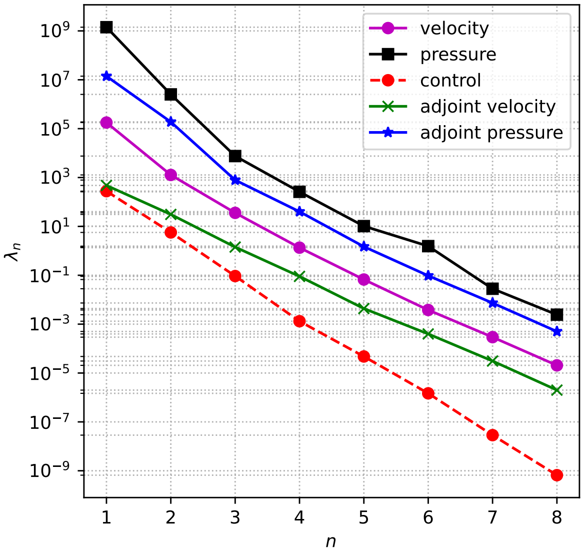


FIGURE 4. Eigenvalues of the snapshots matrix for all the variables in our problem.

fully connected hidden layers with the same number of neurons. Since we have obtained satisfactory results with two hidden layers, we have not increased the number of layers. We have estimated the optimal number of neurons per layer and the optimal learning rate of the neural network through an exhaustive search. We varied the number of neurons in $\{20, 30, \ldots, 100\}$ and the learning rate in $\{1\text{e-}01, 1\text{e-}02, \ldots, 1\text{e-}05\}$ for each variable and for a different number of reduced basis functions. The results of this search are shown in Table 2. As we can see, the optimal number of neurons per layer and the optimal learning rate vary widely from one variable to the other and for different numbers of reduced basis functions. There seems to be no way to predict the optimal network configuration, which makes the exhaustive search by trial

Variable	λ_1	λ_2	λ_3
Velocity	99.99482%	100%	-
Pressure	99.99968%	100%	-
Control	99.95826%	99.99999%	100%
Adjoint velocity	99.57621%	99.99910%	100%
Adjoint pressure	99.98195%	100%	-

TABLE 1. Cumulative energy associated with the first three eigenvalues.

and error unavoidable. The results reported next have been obtained with the optimal neural networks.

Variable	Parameter	Number of basis functions							
		1	2	3	4	5	6	7	8
Velocity	Neurons/layer	90	50	100	40	60	70	20	90
	Learning rate	1e-05	1e-05	1e-04	1e-05	1e-04	1e-03	1e-05	1e-03
Pressure	Neurons/layer	20	50	100	30	30	60	80	20
	Learning rate	1e-05	1e-05	1e-05	1e-05	1e-04	1e-05	1e-03	1e-04
Control	Neurons/layer	40	30	40	70	70	70	20	20
	Learning rate	1e-05	1e-05	1e-05	1e-03	1e-04	1e-05	1e-02	1e-05
Adjoint velocity	Neurons/layer	50	90	50	30	70	30	40	30
	Learning rate	1e-05	1e-05	1e-04	1e-04	1e-05	1e-05	1e-03	1e-03
Adjoint pressure	Neurons/layer	20	40	70	60	60	40	90	70
	Learning rate	1e-05	1e-05	1e-05	1e-05	1e-02	1e-02	1e-05	1e-05

TABLE 2. Optimal values of neurons per layer and learning rate for each variable and for different numbers of reduced basis functions.

In the online phase, we use the trained neural networks to construct the approximated solutions corresponding to the parameters in $\boldsymbol{\mu}^{\text{test}}$. Figure 5 (left) shows the average relative error (4.2) for each variable as the number of reduced basis functions n is increased and Figure 5 (right) displays the maximum and average relative error (4.3) as n varies. We note that the errors reach their minimum for $n = 3$ or $n = 4$, depending on the variable. For almost every n, larger average relative errors are attained for the adjoint variables. The average order of accuracy is about $\mathcal{O}(10^{-4})$ when using more than one reduced basis function, which compares well with [62].

We conclude this section by fixing $n = 6$ in order to compare our results with those in [62]. We start by reporting the computational times required by our methodology in Table 3. During the offline phase, about 540 s are need to compute each snapshot with the full order model, while the POD and the training of a neural network require about 18 s. The average computational time required by the online phase for all the cases in $\boldsymbol{\mu}^{\text{test}}$ is $\mathcal{O}(10^{-4})$ leading to an average speed-up of $\mathcal{O}(10^{6})$, which is 4 times larger than the speed-up obtained using the POD-Galerkin approach in [62]. This is a remarkable improvement.

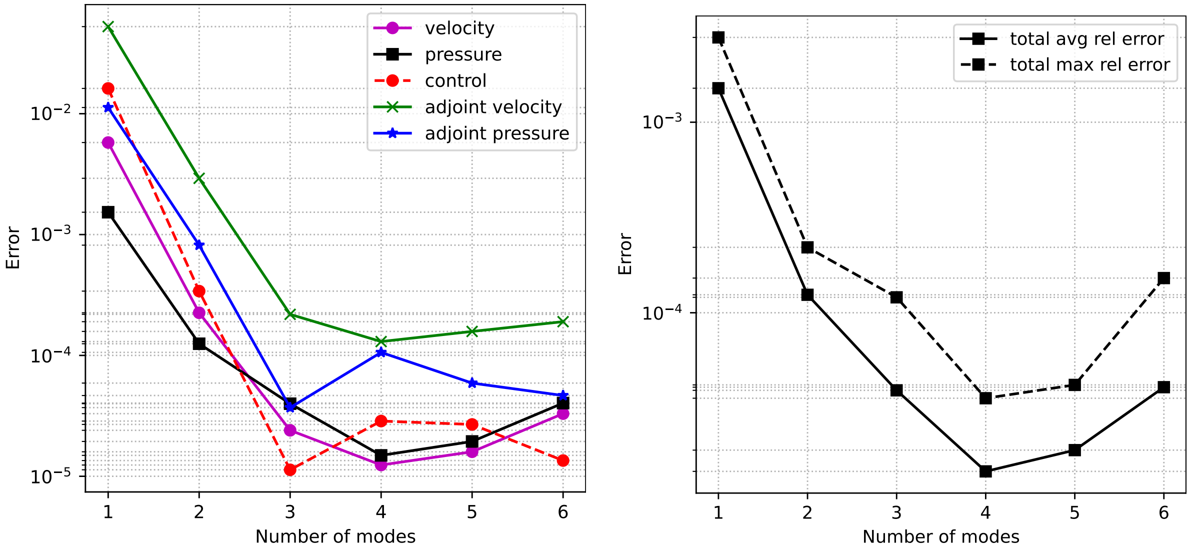


FIGURE 5. Left: average relative error (4.2) for each variable as the number of reduced basis functions is increased. Right: total maximum and average relative error (4.3) as the number of reduced basis functions is increased.

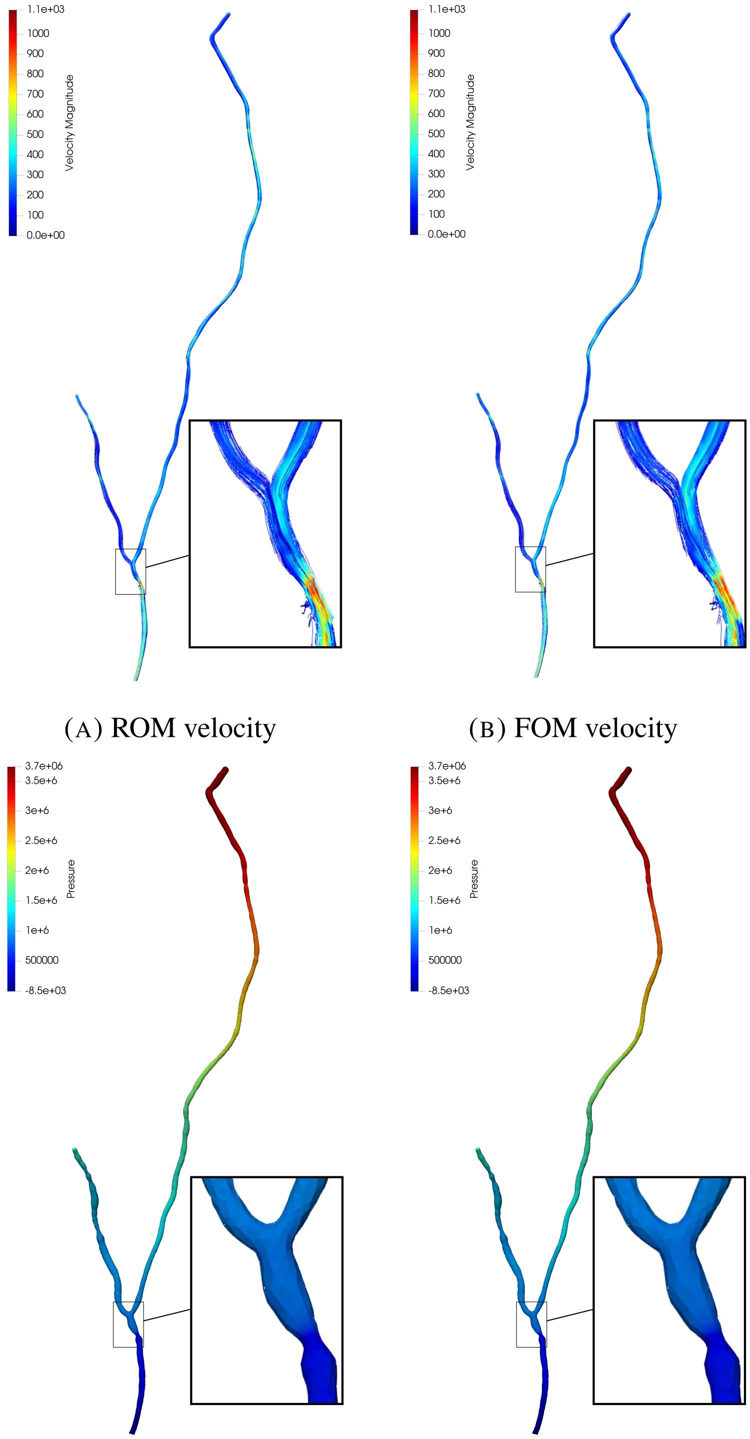
Offline phase			Online phase
FOM	POD	NN train	
540	0.06	17	9.2e-04

TABLE 3. Computational times in s required by the offline phase (broken into sub-phases) and the online phase of our ROM.

Figure 6 shows a qualitative comparison between the velocity and pressure fields for $Re = 80$ computed by ROM and FOM. We see a substantial qualitative agreement over the entire geometry for both flow variables. In particular, notice that our ROM approach is able to provide a good approximation for the velocity also in the region downstream the graft, where the magnitude is higher due to the flow rate introduced by the RITA (compare the zoomed-in views in Figure 6 (A) and 6 (B)). Figure 7 reports the ROM-FOM comparison for the control variable, again for $Re = 80$. We observe very good agreement also between ROM-computed and FOM-computed control variable over the entire outlet surface.

5. CONCLUSIONS AND FUTURE PERSPECTIVES

We focused on a parameterized optimal blood flow control problem in a patient-specific coronary artery bypass graft geometry. The bypass was performed with the right internal thoracic artery on the left anterior descending artery. The optimal control variable is the normal stress at the outlet needed to set the outlet boundary condition in order to minimize in the least squares sense the difference between computed and measured velocity. The blood flow is modeled by the steady-state incompressible Navier-Stokes equations and parameterized by the Reynolds number. We restricted our attention to a physiological range of Reynolds numbers that generate laminar flow.



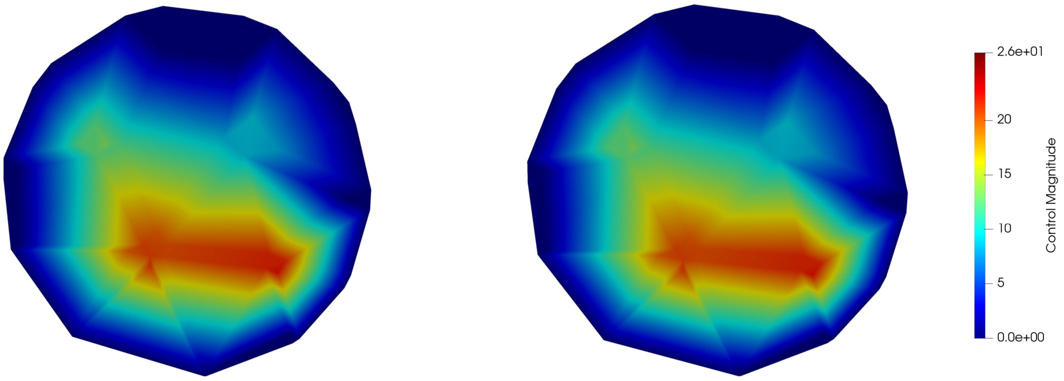
(A) ROM velocity

(B) FOM velocity

(C) ROM pressure

(D) FOM pressure

FIGURE 6. Comparison of the velocity (mm/s) field computed by (A) ROM and (B) FOM and pressure ($\mathrm{mm}^2/\mathrm{s}^2$) field given by the (C) ROM and (D) FOM for $Re = 80$. The number of POD modes used for the ROM is $n = 6$.



(A) ROM control (B) FOM control

FIGURE 7. Control variable (mm^2/s^2) at the outflow boundary for $Re = 80$ computed by the (A) ROM and (B) FOM. The number of POD modes used for the ROM is $n = 6$.

To reduce the high computational costs of a finite element method (FEM) for our parameterized optimization problem, we introduced a non-intrusive reduced order method based on the so-called POD-NN approach. During the offline phase of our ROM, three steps are performed: (i) the high fidelity solutions are computed via FEM and collected into the snapshots matrix, (ii) proper orthogonal decomposition is performed on the snapshots matrix to build a basis for the reduced order space, and (iii) a neural network is trained to estimate the modal coefficients for each variable. After this expensive phase, we were able to simulate the hemodynamics corresponding to the desired Reynolds number at a considerably reduced time during the online phase. Our numerical tests show that this data-driven methodology is as accurate as the POD-Galerkin approach proposed in [62], but it is considerably faster. In particular, we managed to achieve a speed-up of about 4 orders of magnitude, i.e., the simulation is almost real-time.

The proposed framework is a preliminary step in the direction of building a user-friendly virtual platform that could be used for clinical studies [27]. The quantification of the normal stress at the outlet can be improved with the introduction of surrogate lumped parameter network models. This would introduce additional parameters that could be found through an automated quantification via the control variables as done in [26]. This work is limited to one physical parameter (i.e., the Reynolds number). The introduction of geometrical features, such as the degree of the stenosis, would expand the set of possible configurations and generalize our approach to a wider range of clinical cases as shown in [56]. Finally, following [58, 59] one could increase the realism and complexity of the problem by making it time-dependent and accounting for the compliance of the arterial wall.

Acknowledgements

We acknowledge the support provided by the European Research Council Executive Agency by the Consolidator Grant project AROMA-CFD "Advanced Reduced Order Methods with Applications in Computational Fluid Dynamics" - GA 681447, H2020-ERC CoG 2015 AROMA-CFD, PI G. Rozza, and INdAM-GNCS 2019-2020 projects. This work was also partially supported by US National Science Foundation through grant DMS-1953535 (PI A. Quaini).

A. Quaini acknowledges support from the Radcliffe Institute for Advanced Study at Harvard University where she has been the 2021-2022 William and Flora Hewlett Foundation Fellow.

REFERENCES

[1] M. Abbasian, M. Shams, Z. Valizadeh, A. Moshfegh, A. Javadzadegan, Effect of non-newtonian blood flow on coronary artery hemodynamics in a cohort of patients with stenosed artery, In: Fluids Engineering Division Summer Meeting, volume 51555, page V001T02A006. American Society of Mechanical Engineers, 2018.

[2] V. Agoshkov, A. Quarteroni, G. Rozza, A mathematical approach in the de-sign of arterial bypass using unsteady stokes equations, J. Sci. Computing, 28 (2006) 139-165.

[3] V. Agoshkov, A. Quarteroni, G. Rozza, Shape design in aorto-coronaric bypass anastomoses using perturbation theory, SIAM J. Numer. Anal. 44 (2006) 367-384.

[4] S. Ali, F. Ballarin, G. Rozza, Stabilized reduced basis methods for parametrized steady Stokes and Navier-Stokes equations, Comput. Math. Appl. 80 (2020) 2399-2416.

[5] M. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, G. N. Wells, The fenics project version 1.5, Archive Numer. Softw. 3 (100) 2015.

[6] L. Antiga, B. Ene-Iordache, A. Remuzzi, Computational geometry for patient-specific reconstruction and meshing of blood vessels from mr and ct angiography, IEEE Trans. Medical Imaging 22 (2003) 674-684.

[7] J. Baiges, R. Codina, S. R. Idelsohn, Reduced-order modelling strategies for the finite element approximation of the incompressible Navier-Stokes equations, In: Numerical simulations of coupled problems in engineering, volume 33 of Computational Methods in Applied Sciences, pp. 189-216. Springer, Cham, 2014.

[8] F. Ballarin, E. Faggiano, S. Ippolito, A. Manzoni, A. Quarteroni, G. Rozza, R. Scrofani, Fast simulations of patient-specific haemodynamics of coronary artery bypass grafts based on a POD-Galerkin method and a vascular shape parametrization, J. Comput. Phys. 315 (2016) 609-628.

[9] F. Ballarin, E. Faggiano, A. Manzoni, A. Quarteroni, G. Rozza, S. Ippolito, C. Antona, R. Scrofani, Numerical modeling of hemodynamics scenarios of patient-specific coronary artery bypass grafts, Biomechanics and Modeling in Mechanobiology, 16 (2017) 1373-1399.

[10] F. Ballarin, A. Manzoni, A. Quarteroni, G. Rozza, Supremizer stabilization of POD-Galerkin approximation of parametrized steady incompressible Navier-Stokes equations, Int. J. Numer. Meth. Eng. 102 (2015) 1136-1161.

[11] F. Ballarin, G. Rozza, Multiphenics-easy prototyping of multiphysics problems in fenics, https://mathlab. sissa. it/multiphenics, 2019.

[12] P. Benner, W. Schilders, S. Grivet-Talocia, A. Quarteroni, G. Rozza, L. Miguel Silveira, Model Order Reduction: Volume 2: Snapshot-Based Methods and Algorithms, De Gruyter, 2020.

[13] M. Bergmann, C.-H. Bruneau, A. Iollo, Enablers for robust POD models, J. Comput. Phys. 228 (2009) 516-538.

[14] T. R. Bewley, Flow control: new challenges for a new renaissance, Progress in Aerospace Sci. 37 (2001) 21-58.

[15] P. B. Bochev, M. D. Gunzburger, Least-squares finite element methods, volume 166 of Applied Mathematical Sciences, Springer, New York, 2009.

[16] D. Boffi, F. Brezzi, M. Fortin, Mixed finite element methods and applications, vol 44 of Springer Series in Computational Mathematics, Springer, Heidelberg, 2013.

[17] J. Burkardt, M. Gunzburger, H.-C. Lee, POD and CVT-based reduced-order modeling of Navier-Stokes flows, Comput. Meth. Appl. Mech. Eng. 196 (2006) 337-355.

[18] O. Calin, Deep Learning Architectures, Springer, 2020.

[19] W. Chen, Q. Wang, J. S. Hesthaven, C. Zhang, Physics-informed machine learning for reduced-order modeling of nonlinear problems, J. Comput. Phys. 446 (2021) 110666.

[20] L. Dedè, Optimal flow control for Navier–Stokes equations: drag minimization, Int. J. Numer. Meth. Fluids 55 (2007) 347-366.

[21] N. Demo, M. Tezzele, G. Rozza, Ezyrb: Easy reduced basis method, J. Open Source Softw. 3 (2018) 661.

[22] E. Fevola, F. Ballarin, L. Jiménez-Juan, S. Fremes, S. Grivet-Talocia, G. Rozza, P. Triverio, An optimal control approach to determine resistance-type boundary conditions from in-vivo data for cardiovascular simulations, Int. J. Numer. Meth. Biomedical Eng. 37 (2021) e3516.

[23] T. L. Fine, Feedforward neural network methodology, Statistics for Engineering and Physical Science. Springer-Verlag, New York, 1999.

[24] M. Gad-el Hak, A. Pollard, J.-P. Bonnet, Flow control: fundamentals and practices, volume 53. Springer Science & Business Media, 2003.

[25] R. Gharleghi, G. Samarasinghe, A. Sowmya, S. Beier, Deep learning for time averaged wall shear stress prediction in left main coronary bifurcations, IEEE 17th International Symposium on Biomedical Imaging (ISBI), 1-4, 2020.

[26] M. Girfoglio, F. Ballarin, G. Infantino, F. Nicolò, A. Montalto, G. Rozza, R. Scrofani, M. Comisso, F. Musumeci, Non-intrusive PODI-ROM for patient-specific aortic blood flow in presence of a LVAD device, Medical Engineering & Physics 107 (2022) 103849.

[27] M. Girfoglio, L. Scandurra, F. Ballarin, G. Infantino, F. Nicolo, A. Montalto, G. Rozza, R. Scrofani, M. Comisso, F. Musumeci, Non-intrusive data-driven rom framework for hemodynamics problems, Acta Mech. Sinica 37 (2021) 1183-1191.

[28] I. Goodfellow, Y. Bengio, A. Courville, Deep learning, Adaptive Computation and Machine Learning. MIT Press, Cambridge, MA, 2016.

[29] M. D. Gunzburger, Perspectives in flow control and optimization, volume 5 of Advances in Design and Control, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA, 2003.

[30] S. Hang, Tetgen, a Delaunay-based quality tetrahedral mesh generator, ACM Trans. Math. Softw. 41 (2015) 11.

[31] J. S. Hesthaven, G. Rozza, B. Stamm, Certified reduced basis methods for parametrized partial differential equations, volume 590. Springer, 2016.

[32] J. S. Hesthaven S. Ubbiali, Non-intrusive reduced order modeling of nonlinear problems using neural networks, J. Comput. Phys. 363 (2018) 55-78.

[33] M. Hinze, R. Pinnau, M. Ulbrich, S. Ulbrich, Optimization with PDE constraints, volume 23. Springer Science & Business Media, 2008.

[34] L. Hou, S. Ravindran, Numerical approximation of optimal flow control problems by a penalty method: Error estimates and numerical results, SIAM J. Sci. Computing 20 (1999) 1753-1777.

[35] T. S. Koltukluoğlu, P. J. Blanco, Boundary control in computational haemodynamics, J. Fluid Mech. 847 (2018) 329-364.

[36] D. Kriesel, A brief introduction on neural networks, 2007.

[37] K. Kunisch, S. Volkwein, Galerkin proper orthogonal decomposition methods for a general equation in fluid dynamics, SIAM J. Numer. Anal. 40 (2002) 492-515.

[38] D. Lazzaro, L. Montefusco, Radial basis functions for the multivariate interpolation of large scattered data sets, J. Comput. Appl. Math. 140 (2002) 521-536.

[39] L. Liang, W. Mao, W. Sun, A feasibility study of deep learning for predicting hemodynamics of human thoracic aorta, J. Biomechanics 99 (2020) 109544.

[40] A. Logg, K.-A. Mardal, G. N. Wells, editors. Automated solution of differential equations by the finite element method, volume 84 of Lecture Notes in Computational Science and Engineering, Springer, Heidelberg, 2012. The FEniCS book.

[41] M. Minsky, S. Papert, An introduction to computational geometry, Cambridge tiass. HIT, 479 (1969) 480.

[42] D. J. Montana, L. Davis, et al. Training feedforward neural networks using genetic algorithms, IJCAI 89 (1989) 762-767.

[43] F. Pichi, F. Ballarin, G. Rozza, J. S. Hesthaven, Artificial neural network for bifurcating phenomena modelled by nonlinear parametrized PDEs, PAMM, 20(S1):e202000350, 2021.

[44] A. Quarteroni, Numerical models for differential problems, volume 8 of MS&A. Modeling, Simulation and Applications, Springer, Milan, 2014.

[45] A. Quarteroni, A. Manzoni, F. Negri, Reduced basis methods for partial differential equations, volume 92, Springer, Cham, 2016.

[46] A. Quarteroni, G. Rozza, Optimal control and shape optimization of aorto-coronaric bypass anastomoses, Math, Models Meth. Appl. Sci. 13 (2003) 1801-1823.

[47] R. Rojas, The backpropagation algorithm, Neural Networks, pp. 149-182, Springer, 1996.

[48] R. M. Romarowski, A. Lefieux, S. Morganti, A. Veneziani, F. Auricchio, Patient-specific cfd modelling in the thoracic aorta with PC-MRI–based boundary conditions: A least-square three-element Windkessel approach, Int. J. Numer. Meth. Biomedical Eng. 34 (2018) e3134.

[49] F. Rosenblatt, The perceptron: a probabilistic model for information storage and organization in the brain, volume 65, American Psychological Association, 1958.

[50] G. Rozza, Real time reduced basis techniques for arterial bypass geometries, In: Computational Fluid and Solid Mechanics-Third MIT Conference on Computational Fluid and Solid Mechanics, number CONF, pp. 1283–1287. Elsevier, 2005.

[51] D. E. Rumelhart, G. E. Hinton, R. J. Williams. Learning representations by back-propagating errors Nature, 323 (1986) 533-536.

[52] W. Schroeder, K. M. Martin, W. E. Lorensen, The visualization toolkit an object-oriented approach to 3D graphics, Prentice-Hall, Inc., 1998.

[53] V. Schulz, I. Gherman, One-shot methods for aerodynamic shape optimization, In: MEGADESIGN and MegaOpt-German Initiatives for Aerodynamic Simulation and Optimization in Aircraft Design, pp. 207-220, Springer, 2009.

[54] N. V. Shah, M. Girfoglio, P. Quintela, G. Rozza, A. Lengomin, F. Ballarin, P. Barral, Finite element based model order reduction for parametrized one-way coupled steady state linear thermomechanical problems, Finite Elements in Analysis and Design 2012 (2022) 103837.

[55] S. Sharma, S. Sharma, A. Athaiya, Activation functions in neural networks, Towards Data Sci. 6 (2017) 310-316.

[56] P. Siena, M. Girfoglio, F. Ballarin, G. Rozza, Data-driven reduced order modelling for patient-specific hemodynamics of coronary artery bypass grafts with physical and geometrical parameters, arXiv preprint arXiv:2203.13682, 2022.

[57] P. Siena, M. Girfoglio, G. Rozza, Fast and accurate numerical simulations for the study of coronary artery bypass grafts by artificial neural network, arXiv preprint arXiv:2201.01804, 2022.

[58] M. Stoll, A. Wathen, All-at-once solution of time-dependent stokes control,J. Comput. Phys. 232 (2013) 498-515.

[59] M. Strazzullo, Z. Zainib, F. Ballarin, G. Rozza, Reduced order methods for parametrized non-linear and time dependent optimal flow control problems, towards applications in biomedical and environmental sciences, Numerical Mathematics and Advanced Applications ENUMATH 2019, pp. 841-850, Springer, 2021.

[60] J. Tiago, T. Guerra, A. Sequeira, A velocity tracking approach for the data assimilation problem in blood flow simulations, Int. J. Numer. Meth. Biomedical Eng. 33 (2017) e2856.

[61] R. Torii, M. Oshima, T. Kobayashi, K. Takagi, T. E. Tezduyar, Computer modeling of cardiovascular fluid-structure interactions with the deforming-spatial-domain/stabilized space-time formulation, Comput. Meth. Appl. Mech. Eng. 195 (2006) 1885-1895.

[62] Z. Zainib, F. Ballarin, S. Fremes, P. Triverio, L. Jiménez-Juan, G. Rozza, Reduced order methods for parametric optimal flow control in coronary bypass grafts, toward patient-specific data assimilation, Int. J. Numer. Meth. Biomedical Eng. 37 (2021) e3367.

[63] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of Control, Signals and Systems 2 (1989) 303-314.