# A machine-learning minimal-residual (ML-MRes) framework for goal-oriented finite element discretizations

Ignacio Brevis[*], Ignacio Muga[†], and Kristoffer G. van der Zee[‡]

12th August, 2020

**Abstract**

We introduce the concept of machine-learning minimal-residual (ML-MRes) finite element discretizations of partial differential equations (PDEs), which resolve quantities of interest with striking accuracy, regardless of the underlying mesh size. The methods are obtained within a machine-learning framework during which the parameters defining the method are tuned against available training data. In particular, we use a stable parametric Petrov–Galerkin method that is equivalent to a minimal-residual formulation using a weighted norm. While the trial space is a standard finite element space, the test space has parameters that are tuned in an off-line stage. Finding the optimal test space therefore amounts to obtaining a goal-oriented discretization that is completely tailored towards the quantity of interest. We use an artificial neural network to define the parametric family of test spaces. Using numerical examples for the Laplacian and advection equation in one and two dimensions, we demonstrate that the ML-MRes finite element method has superior approximation of quantities of interest even on very coarse meshes.

**Keywords** Goal-oriented finite elements · Machine-Learning acceleration · Residual Minimization · Petrov-Galerkin method · Weighted inner-products · Data-driven algorithms.
**MSC 2020** 41A65 · 65J05 · 65N15 · 65N30 · 65L60 · 68T07

[*]Pontificia Universidad Católica de Valparaíso, Instituto de Matemáticas.
ignacio.brevis.v@gmail.com
[†]Pontificia Universidad Católica de Valparaíso, Instituto de Matemáticas.
ignacio.muga@pucv.cl
[‡]University of Nottingham, School of Mathematical Sciences.
kg.vanderzee@nottingham.ac.uk

# Contents



# 1 Introduction

In this paper we consider the machine-learning acceleration of Galerkin-based discretizations, in particular the finite element method, for the approximation of partial differential equations (PDEs). The aim is to obtain approximations on meshes that are *very* coarse, but nevertheless resolve quantities of interest with *striking* accuracy.

We follow the machine-learning framework of Mishra [27], who considered the data-driven acceleration of *finite-difference* schemes for ordinary differential equations (ODEs) and PDEs. In Mishra's machine learning framework, one starts with a *parametric family* of a stable and consistent numerical method on a fixed mesh (think of, for example, the $\theta$-method for ODEs). Then, a training set is prepared, typically by offline computations of the PDE subject to

a varying set of data values (initial conditions, boundary conditions, etc), using a standard method on a (very) fine mesh. Accordingly, an optimal numerical method on the coarse grid is found amongst the general family, by minimizing a loss function consisting of the errors in quantities of interest with respect to the training data.

The objective of this paper is to extend Mishra's machine-learning framework to finite element methods. Since a key idea in our framework is the principle of residual minimization (see below), we refer to our discretization technique as a Machine-Learning Minimal-Residual (ML-MRes) method. The main contribution of our work lies in the identification of a proper stable and consistent general family of finite element methods for a given mesh that allows for a robust optimization. In particular, we consider a parametric Petrov–Galerkin method, where the trial space is fixed on the given mesh, but the test space has trainable parameters that are to be determined in the offline training process. Finding this optimized test space therefore amounts to obtaining a coarse-mesh discretization that is completely tailored for the quantity of interest.

A crucial aspect for the stability analysis is the equivalent formulation of the parametric Petrov–Galerkin method as a *minimal-residual* formulation using discrete dual norms. Such techniques have been studied in the context of discontinuous Petrov–Galerkin (DPG) and optimal Petrov–Galerkin methods; see for example the overview by Demkowicz & Gopalakrishnan [8] (and also [29] for the recent Banach-space extension). A key insight is that we can define a suitable test-space parametrization, by using a (discrete) trial-to-test operator for a test-space norm based on a parametric weight function. This allows us to prove the stability of the parametric minimal-residual method, and thus, by equivalence, proves stability for the parametric Petrov–Galerkin method.

As is natural in *deep learning*, we furthermore propose to use an *artificial neural network* for the weight function defining the test space in the Petrov–Galerkin method. The training of the tuning parameters in this neural network is achieved minimizing a user-defined loss function, which contains the neural network implicitly.

## 1.1 Motivating example

To briefly illustrate our idea, let us consider a simple motivating example, driven by the following 1-D elliptic boundary-value problem:

$$\begin{cases} -u_\lambda'' = \delta_\lambda & \text{in } (0,1), \\ u_\lambda(0) = u_\lambda'(1) = 0, \end{cases} \tag{1}$$

where $\delta_\lambda$ denotes the usual Dirac's delta distribution centered at the point $\lambda \in (0,1)$. The quantity of interest (QoI) is the value $u_\lambda(x_0)$ of the solution at some fixed point $x_0 \in (0,1)$.

The standard variational formulation of problem (1) reads:

$$\begin{cases} \text{Find } u_\lambda \in H^1_{(0}(0,1) \text{ such that:} \\ \displaystyle\int_0^1 u'_\lambda v' = v(\lambda), \qquad \forall\, v \in H^1_{(0}(0,1), \end{cases} \tag{2}$$

where $H^1_{(0}(0,1) := \{v \in L^2(0,1) : v' \in L^2(0,1) \wedge v(0) = 0\}$. For the very coarse discrete subspace $\mathbb{U}_h := \text{Span}\{\psi\} \subset H^1_{(0}(0,1)$ consisting of the single linear trial function $\psi(x) = x$, the usual Galerkin method approximating (2) delivers the discrete solution $u_h(x) = \lambda x$. However, the exact solution to (1) is:

$$u_\lambda(x) = \begin{cases} x & \text{if } x \leq \lambda, \\ \lambda & \text{if } x \geq \lambda. \end{cases} \tag{3}$$

Hence, the relative error in the QoI for this case becomes:

$$\frac{|u_\lambda(x_0) - u_h(x_0)|}{|u_\lambda(x_0)|} = \begin{cases} 1-\lambda & \text{if } x_0 \leq \lambda, \\ 1-x_0 & \text{if } x_0 \geq \lambda, \end{cases} \tag{4}$$

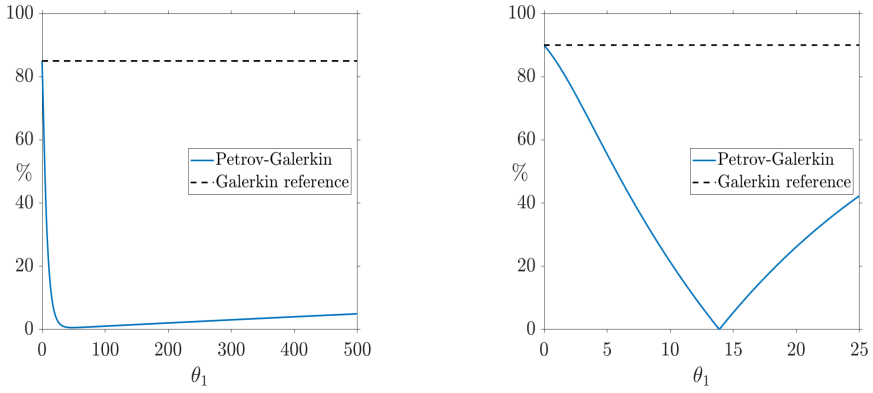
As may be expected for this very coarse approximation, the relative errors are large (and actually never vanish except in limiting cases).

Let us instead consider a Petrov–Galerkin method for (2), with the same trial space $\mathbb{U}_h$, but a special test space $\mathbb{V}_h$, i.e., $u_h \in \mathbb{U}_h := \text{Span}\{\psi\}$ such that $\int_0^1 u'_h v'_h = v_h(\lambda)$, for all $v_h \in \mathbb{V}_h := \text{Span}\{\varphi\}$. We use the parametrized test function $\varphi(x) = \theta_1 x + e^{-\theta_2}(1 - e^{-\theta_1 x})$, which is motivated by our simplest artificial neural network; see Section 4.1 for details. By *varying* the parameters $\theta_1, \theta_2 \in \mathbb{R}$, the errors in the quantity of interest can be significantly reduced. Indeed, Figure 1 shows the relative error in the QoI, plotted as a function of the $\theta_1$-parameter, with the other parameter set to $\theta_2 = -9$, in the case of $x_0 = 0.1$ and two values of $\lambda$. When $\lambda = 0.15 > 0.1 = x_0$ (left plot in Figure 1), the optimal value $\theta_1 \approx 48.5$ delivers a relative error of 0.575% in the quantity of interest. Notice that the Galerkin method has a relative error $> 80\%$. For $\lambda = 0.05 < 0.1 = x_0$ (right plot in Figure 1), the value $\theta_1 \approx 13.9$ actually delivers an *exact* approximation of the QoI, while the Galerkin method has a relative error $\approx 90\%$.

This example illustrates a general trend that we have observed in our numerical test (see Section 4): Striking improvements in quantities of interest are achieved using well-tuned test spaces.

## 1.2 Related literature

Let us note that deep learning, in the form of artificial neural networks, has become extremely popular in scientific computation in the past few years, a crucial feature being the capacity of neural networks to approximate any continuous function [6]. While classical applications concern classification and prediction for image and speech recognition [14, 24, 18], there have




(a) Relative error for $\lambda = 0.15$ (b) Relative error for $\lambda = 0.05$.

Figure 1: Relative error in the quantity of interest $x_0 = 0.1$, for different values of $\theta_1$.

been several new advances related to differential equations, either focussing on the data-driven discovery of governing equations [34, 3, 31] or the numerical approximation of (parametric) differential equations.

On the one hand, artificial neural networks can be directly employed to approximate a single PDE solution, see e.g. [2, 23, 25], and in particular the recent high-dimensional Ritz method [10]. On the other hand, in the area of model order reduction of differential equations, there have been tremendous recent developments in utilizing machine learning to obtain the reduced-order model for parametric models [19, 17, 33, 36, 22]. These developments are very closely related to recent works that use neural networks to optimize numerical methods, e.g., tuning the turbulence model [26], slope limiter [32] or artificial viscosity [9].

The idea of goal-oriented adaptive (finite element) methods date back to the late 1990s, see e.g., [1, 30, 28] for early works and analysis, and [13, 21, 38, 11, 16] for some recent new developments. These methods are based on a different idea than the machine-learning framework that we propose. Indeed, the classical goal-oriented methods aim to adaptively refine the underlying meshes (or spaces) so as to control the error in the quantity of interest, thereby adding more degrees of freedom at each adaptive step. In our framework, we train a finite element method so as to control the error in the quantity of interest based on training data for a parametric model. In particular, we do not change the number of degrees of freedom.

## 1.3 Outline

The contents of this paper are arranged as follows. Section 2 presents the machine-learning methodology for constructing minimal-residual finite element methods driven by a training dataset. It also presents the stability analysis of the discrete method as well as equivalent discrete formulations. Section 3 presents several implementational details related to artificial

neural networks and the training procedure. Section 4 present numerical experiments for 1-D and 2-D elliptic and hyperbolic PDEs. Finally, Section 5 contains our conclusions.

# 2 Methodology

## 2.1 Abstract problem

Let $\mathbb{U}$ and $\mathbb{V}$ be infinite dimensional Hilbert spaces, with respective dual spaces $\mathbb{U}^*$ and $\mathbb{V}^*$. Consider a boundedly invertible linear operator $B : \mathbb{U} \to \mathbb{V}^*$, a family of right-hand-side functionals $\{\ell_\lambda\}_{\lambda\in\Lambda} \subset \mathbb{V}^*$ that may depend non-affinely on $\lambda$, and a quantity of interest functional $q \in \mathbb{U}^*$. Given $\lambda \in \Lambda$, the continuous (or infinite-dimensional) problem will be to find $u_\lambda \in \mathbb{U}$ such that:

$$Bu_\lambda = \ell_\lambda, \quad \text{in } \mathbb{V}^*, \tag{5}$$

where the interest is put in the quantity $q(u_\lambda)$. In particular, we consider the case when $\langle Bu, v\rangle_{\mathbb{V}^*,\mathbb{V}} := b(u, v)$, for a given bilinear form $b : \mathbb{U} \times \mathbb{V} \to \mathbb{R}$. If so, problem (5) translates into:

$$\begin{cases} \text{Find } u_\lambda \in \mathbb{U} \text{ such that:} \\ b(u_\lambda, v) = \ell_\lambda(v), \qquad \forall\, v \in \mathbb{V}, \end{cases} \tag{6}$$

which is a type of problem that naturally arises in the context of variational formulations of partial differential equations with multiple right-hand-sides or *parametrized PDEs*.[1]

## 2.2 Main idea of the accelerated methods

We assume that the space $\mathbb{V}$ can be endowed with a family of equivalent *weighted* inner products $\{(\cdot,\cdot)_{\mathbb{V},\omega}\}_{\omega\in\mathcal{W}}$ and inherited norms $\{\|\cdot\|_{\mathbb{V},\omega}\}_{\omega\in\mathcal{W}}$, without affecting the topology given by the original norm $\|\cdot\|_{\mathbb{V}}$ on $\mathbb{V}$. That is, for each $\omega \in \mathcal{W}$, there exist equivalence constants $C_{1,\omega} > 0$ and $C_{2,\omega} > 0$ such that:

$$C_{1,\omega}\|v\|_{\mathbb{V},\omega} \le \|v\|_{\mathbb{V}} \le C_{2,\omega}\|v\|_{\mathbb{V},\omega}\,, \qquad \forall\, v \in \mathbb{V}. \tag{7}$$

Consider a *coarse* finite dimensional subspace $\mathbb{U}_h \subset \mathbb{U}$ where we want to approximate the solution of (6), and let $\mathbb{V}_h \subset \mathbb{V}$ be a discrete test space such that $\dim \mathbb{V}_h \ge \dim \mathbb{U}_h$. The discrete method that we want to use to approach the solution $u_\lambda \in \mathbb{U}$ of problem (6), is to find $(r_{h,\lambda,\omega}, u_{h,\lambda,\omega}) \in \mathbb{V}_h \times \mathbb{U}_h$ such that:

$$\begin{cases} (r_{h,\lambda,\omega}, v_h)_{\mathbb{V},\omega} + b(u_{h,\lambda,\omega}, v_h) = \ell_\lambda(v_h) & \forall\, v_h \in \mathbb{V}_h\,, \quad (8a) \\ b(w_h, r_{h,\lambda,\omega}) = 0 & \forall\, w_h \in \mathbb{U}_h\,. \quad (8b) \end{cases}$$

[1] While parametrized bilinear forms $b_\lambda(\cdot,\cdot)$ are also possible, they lead to quite distinct algorithmic details. We therefore focus on parametrized right-hand sides and leave parametrized bilinear forms for future work.

System (8) corresponds to a residual minimization in a discrete dual norm that is equivalent to a Petrov–Galerkin method. See Section 2.3 for equivalent formulations and analysis of this discrete approach. In particular, the counterpart $r_{h,\lambda,\omega} \in \mathbb{V}_h$ of the solution of (8) is interpreted as a minimal residual representative, while $u_{h,\lambda,\omega} \in \mathbb{U}_h$ is the coarse approximation of $u_\lambda \in \mathbb{U}$ that we are looking for.

Assume now that one has a reliable sample set of $N_s \in \mathbb{N}$ (precomputed) data $\{(\lambda_i, q(u_{\lambda_i}))\}_{i=1}^{N_s}$, where $q(u_{\lambda_i})$ is either the quantity of interest of the exact solution of (6) with $\lambda = \lambda_i \in \Lambda$, or else, a high-precision approximation of it. The main goal of this paper is to find a particular weight $\omega^* \in \mathcal{W}$, such that for the finite sample of parameters $\{\lambda_i\}_{i=1}^{N_s} \subset \Lambda$, the discrete solutions $\{u_{h,\lambda_i,\omega^*}\}_{i=1}^{N_s} \subset \mathbb{U}_h$ of problem (8) with $\omega = \omega^*$, makes the errors in the quantity of interest as small as possible, i.e.,

$$\frac{1}{2}\sum_{i=1}^{N_s} |q(u_{\lambda_i}) - q(u_{h,\lambda_i,\omega^*})|^2 \to \min . \tag{9}$$

To achieve this goal we will work with a particular family of weights described by artificial neural networks (ANN). The particular *optimal* weight $\omega^*$ will be *trained* using machine-learning algorithms that we describe in the following. Our methodology will be divided into an expensive *offline procedure* (see Section 3.2) and an unexpensive *online procedure* (see Section 3.3).

In the offline procedure:

- A weight function $\omega^* \in \mathcal{W}$ that minimizes (9) for a sample set of training data $\{(\lambda_i, q(u_{\lambda_i}))\}_{i=1}^{N_s}$ is obtained.
- From the matrix related with the discrete mixed formulation (8) using $\omega = \omega^*$, a *static condensation* procedure is applied to condense-out the residual variable $r_{h,\lambda,\omega^*}$. The condensed matrices are stored for the online procedure.

In the online procedure:

- The condensed mixed system (8) with $\omega = \omega^*$ is solved for multiple right-hand-sides in $\{\ell_\lambda\}_{\lambda\in\Lambda}$, and the quantities of interest $\{q(u_{h,\lambda,\omega^*})\}_{\lambda\in\Lambda}$ are directly computed as reliable approximations of $\{q(u_\lambda)\}_{\lambda\in\Lambda}$.

## 2.3 Analysis of the discrete method

In this section we analyze the well-posedness of the discrete system (8), as well as equivalent interpretations of it. The starting point will be always to assume well-posedness of the continuous (or infinite-dimensional) problem (6), which we will establish below.

**Theorem 2.A** *Let $(\mathbb{U}, \|\cdot\|_{\mathbb{U}})$ and $(\mathbb{V}, \|\cdot\|_{\mathbb{V}})$ be Hilbert spaces, and let $\|\cdot\|_{\mathbb{V},\omega}$ be the norm inherited from the weighted inner-product $(\cdot,\cdot)_{\mathbb{V},\omega}$, which satisfies the equivalence* (7)*. Consider the problem* (6) *and assume the existence of constants $M_\omega > 0$ and $\gamma_\omega > 0$ such that:*

$$\gamma_\omega \|u\|_{\mathbb{U}} \leq \sup_{v\in\mathbb{V}} \frac{|b(u,v)|}{\|v\|_{\mathbb{V},\omega}} \leq M_\omega \|u\|_{\mathbb{U}}\,, \qquad \forall\, u \in \mathbb{U}. \tag{10}$$

*Furthermore, assume that for any $\lambda \in \Lambda$:*

$$\langle \ell_\lambda, v\rangle_{\mathbb{V}^*,\mathbb{V}} = 0, \quad \forall\, v \in \mathbb{V} \text{ such that } b(\cdot,v) = 0 \in \mathbb{U}^*\,. \tag{11}$$

*Then, for any $\lambda \in \Lambda$, there exists a unique $u_\lambda \in \mathbb{U}$ solution of problem* (6). □

**Proof** This result is classical. Using operator notation (see eq. (5)), condition (10) says that the operator $B : \mathbb{U} \to \mathbb{V}^*$ such that $\langle Bu, v\rangle_{\mathbb{V}^*,\mathbb{V}} = b(u,v)$ is continuous, injective and has a closed range. In particular, if $u_\lambda \in \mathbb{U}$ exists, then it must be unique. The existence of $u_\lambda$ is guaranteed by condition (11), since $\ell_\lambda$ is orthogonal to the kernel of $B^*$, which means that $\ell_\lambda$ is in the range of $B$ by the Banach closed range theorem. ■

**Remark 2.1** Owing to the equivalence of norms (7), if (10) holds true for a particular weight $\omega \in \mathcal{W}$, then it also holds true for the original norm $\|\cdot\|_{\mathbb{V}}$ of $\mathbb{V}$, and for any other weighted norm linked to the family of weights $\mathcal{W}$. □

The next Theorem 2.B establishes the well-posedness of the discrete mixed scheme (8).

**Theorem 2.B** *Under the same assumptions of Theorem 2.A, let $\mathbb{U}_h \subset \mathbb{U}$ and $\mathbb{V}_h \subset \mathbb{V}$ be finite dimensional subspaces such that $\dim \mathbb{V}_h \geq \dim \mathbb{U}_h$, and such that the following discrete inf-sup conditon is satisfied:*

$$\sup_{v_h\in\mathbb{V}_h} \frac{|b(u_h,v_h)|}{\|v_h\|_{\mathbb{V},\omega}} \geq \gamma_{h,\omega} \|u_h\|_{\mathbb{U}}\,, \qquad \forall\, u_h \in \mathbb{U}_h\,, \tag{12}$$

*where $\gamma_{h,\omega} > 0$ is the associated discrete inf-sup constant. Then, the mixed system* (8) *has a unique solution $(r_{h,\lambda,\omega}, u_{h,\lambda,\omega}) \in \mathbb{V}_h \times \mathbb{U}_h$. Moreover, $u_{h,\lambda,\omega}$ satifies the a priori estimates:*

$$\|u_{h,\lambda,\omega}\|_{\mathbb{U}} \leq \frac{M_\omega}{\gamma_{h,\omega}} \|u_\lambda\|_{\mathbb{U}} \quad \text{and} \quad \|u_\lambda - u_{h,\lambda,\omega}\|_{\mathbb{U}} \leq \frac{M_\omega}{\gamma_{h,\omega}} \inf_{u_h\in\mathbb{U}_h} \|u_\lambda - u_h\|_{\mathbb{U}}. \tag{13}$$

□

**Proof** See Appendix A. ■

**Remark 2.2** It is straightforward to see, using the equivalences of norms (7), that having the discrete inf-sup condition in one weighted norm $\|\cdot\|_{\mathbb{V},\omega}$ is fully equivalent to have the discrete inf-sup condition in the original norm of $\mathbb{V}$, and also to have the discrete inf-sup condition in another weighted norm linked to the family of weights $\mathcal{W}$. If (12) holds true for any weight of the family $\mathcal{W}$ (or for the original norm of $\mathbb{V}$) we say that the the discrete pairing $\mathbb{U}_h$-$\mathbb{V}_h$ is compatible. □

**Remark 2.3 (Influence of the weight)** In general, to make the weight $\omega \in \mathcal{W}$ influence the mixed system (8), we need $\dim \mathbb{V}_h > \dim \mathbb{U}_h$. In fact, the case $\dim \mathbb{V}_h = \dim \mathbb{U}_h$ is not interesting because equation (8b) becomes a square system and one would obtain $r_{h,\lambda,\omega} = 0$ from it, thus recovering a standard Petrov-Galerkin method without any influence of $\omega$. □

### 2.3.1 Equivalent Petrov-Galerkin formulation

For any weight $\omega \in \mathcal{W}$, consider the trial-to-test operator $T_\omega : \mathbb{U} \to \mathbb{V}$ such that:

$$(T_\omega u, v)_{\mathbb{V},\omega} = b(u, v)\,, \qquad \forall\, u \in \mathbb{U},\, \forall\, v \in \mathbb{V}. \tag{14}$$

Observe that for any $u \in \mathbb{U}$, the vector $T_\omega u \in \mathbb{V}$ is nothing but the Riesz representative of the functional $b(u, \cdot) \in \mathbb{V}^*$ under the weighted inner-product $(\cdot, \cdot)_{\mathbb{V},\omega}$.

Given a discrete subspace $\mathbb{U}_h \subset \mathbb{U}$, the optimal test space paired with $\mathbb{U}_h$, is defined as $T_\omega \mathbb{U}_h \subset \mathbb{V}$. The concept of optimal test space was introduced by [7] and its main advantage is that the discrete pairing $\mathbb{U}_h$-$T_\omega \mathbb{U}_h$ (with equal dimensions) satisfies automatically the inf-sup condition (12), with inf-sup constant $\gamma_\omega > 0$, inherited from the stability at the continuous level (see eq. (10)).

Of course, equation (14) is infinite dimensional and thus not solvable in the general case. Instead, having the discrete finite-dimensional subspace $\mathbb{V}_h \subset \mathbb{V}$, we can define the discrete trial-to-test operator $T_{h,\omega} : \mathbb{U} \to \mathbb{V}_h$ such that:

$$(T_{h,\omega} u, v_h)_{\mathbb{V},\omega} = b(u, v_h)\,, \quad \forall\, u \in \mathbb{U},\, \forall\, v_h \in \mathbb{V}_h. \tag{15}$$

Observe now that the vector $T_{h,\omega} u \in \mathbb{V}_h$ corresponds to the orthogonal projection of $T_\omega u$ into the space $\mathbb{V}_h$, by means of the weighted inner-product $(\cdot, \cdot)_{\mathbb{V},\omega}$. This motivates the definition of the *projected optimal test space* (of the same dimension of $\mathbb{U}_h$) as $\mathbb{V}_{h,\omega} := T_{h,\omega} \mathbb{U}_h$ (cf. [4]). It can be proven that if the discrete pairing $\mathbb{U}_h$-$\mathbb{V}_h$ satisfies the inf-sup condition (12), then the discrete pairing $\mathbb{U}_h$-$\mathbb{V}_{h,\omega}$ also satisfies the inf-sup conditon (12), with the same inf-sup constant $\gamma_{h,\omega} > 0$. Moreover, the solution $u_{h,\lambda,\omega} \in \mathbb{U}_h$ of the mixed system (8) is also the unique solution of the well-posed Petrov-Galerkin scheme:

$$b(u_{h,\lambda,\omega}, v_h) = \ell_\lambda(v_h), \qquad \forall\, v_h \in \mathbb{V}_{h,\omega}\,. \tag{16}$$

Indeed, from equation (8b), for any $v_h = T_{h,\omega} w_h \in \mathbb{V}_{h,\omega} \subset \mathbb{V}_h$, we obtain that

$$(r_{h,\lambda,\omega}, v_h)_{\mathbb{V},\omega} = (r_{h,\lambda,\omega}, T_{h,\omega} w_h)_{\mathbb{V},\omega} = b(w_h, r_{h,\lambda,\omega}) = 0,$$

which upon being replaced in equation (8a) of the mixed system gives (16). We refer to [4, Proposition 2.2] for further details.

### 2.3.2 Equivalent Minimal Residual formulation

Let $\mathbb{U}_h \subset \mathbb{U}$ and $\mathbb{V}_h \subset \mathbb{V}$ as in Theorem 2.B, and consider the following discrete-dual residual minimization:

$$u_{h,\lambda,\omega} = \underset{w_h \in \mathbb{U}_h}{\operatorname{argmin}} \|\ell_\lambda(\cdot) - b(w_h,\cdot)\|_{(\mathbb{V}_h)^*_\omega} \quad , \quad \text{where } \|\cdot\|_{(\mathbb{V}_h)^*_\omega} := \sup_{v_h \in \mathbb{V}_h} \frac{|\langle \cdot, v_h\rangle_{\mathbb{V}^*,\mathbb{V}}|}{\|v_h\|_{\mathbb{V},\omega}}.$$

Let $R_{\omega,\mathbb{V}_h} : \mathbb{V}_h \to (\mathbb{V}_h)^*$ be the Riesz map (isometry) linked to the weighted inner-product $(\cdot,\cdot)_{\mathbb{V},\omega}$, that is:

$$\langle R_{\omega,\mathbb{V}_h} v_h \, , \, \cdot \rangle_{(\mathbb{V}_h)^*,\mathbb{V}_h} = (v_h \, , \, \cdot)_{\mathbb{V},\omega} \, , \quad \forall\, v_h \in \mathbb{V}_h.$$

Defining the minimal residual representative $r_{h,\lambda,\omega} := R^{-1}_{\omega,\mathbb{V}_h}(\ell_\lambda(\cdot) - b(u_{h,\lambda,\omega},\cdot))$, we observe that the couple $(r_{h,\lambda,\omega}, u_{h,\lambda,\omega}) \in \mathbb{V}_h \times \mathbb{U}_h$ is the solution of the mixed system (8). Indeed, $r_{h,\lambda,\omega} \in \mathbb{V}_h$ satisfies:

$$(r_{h,\lambda,\omega}, v_h)_{\mathbb{V},\omega} = \ell_\lambda(v_h) - b(u_{h,\lambda,\omega}, v_h) \, , \quad \forall\, v_h \in \mathbb{V}_h \, ,$$

which is nothing but equation (8a) of the mixed system. On the other hand, using the isometric property of $R_{\omega,\mathbb{V}_h}$ we have:

$$u_{h,\lambda,\omega} = \underset{w_h \in \mathbb{U}_h}{\operatorname{argmin}} \|\ell_\lambda(\cdot) - b(w_h,\cdot)\|^2_{(\mathbb{V}_h)^*_\omega} = \underset{w_h \in \mathbb{U}_h}{\operatorname{argmin}} \left\|R^{-1}_{\omega,\mathbb{V}_h}(\ell_\lambda(\cdot) - b(w_h,\cdot))\right\|^2_{\mathbb{V},\omega} \, .$$

Differentiating the norm $\|\cdot\|_{\mathbb{V},\omega}$ and using first-order optimality conditions we obtain:

$$0 = \left(R^{-1}_{\omega,\mathbb{V}_h}(\ell_\lambda(\cdot) - b(u_{h,\lambda,\omega},\cdot)), R^{-1}_{\omega,\mathbb{V}_h} b(w_h,\cdot)\right)_{\mathbb{V},\omega} = b(w_h, r_{h,\lambda,\omega}), \quad \forall\, w_h \in \mathbb{U}_h \, ,$$

which gives equation (8b).

# 3 Implementational details

## 3.1 Artificial Neural Networks

Roughly speaking, an artificial neural network is obtained from compositions and superpositions of a single, simple nonlinear activation or response function [6]. Namely, given an input $x_{\text{in}} \in \mathbb{R}^d$ and an activation function $\sigma$, an artificial neural network looks like:

$$\text{ANN}(x_{\text{in}}) = \Theta_n \sigma(\cdots \sigma(\Theta_2 \sigma(\Theta_1 x_{\text{in}} + \phi_1) + \phi_2) \cdots) + \phi_n, \tag{17}$$

where $\{\Theta_j\}_{j=1}^n$ are weight matrices (of different size) and $\{\phi_j\}_{j=1}^n$ are bias vectors (of different length) of coefficients to be determined by a "training" procedure. Depending on the application, an extra activation function can be added at the end. A classical activation function is the logistic sigmoid function:

$$\sigma(x) = \frac{1}{1+e^{-x}} \, . \tag{18}$$

Other common activation functions used in artificial neural network applications are the rectified linear unit (ReLU), the leaky ReLU, and the hyperbolic tangent (see, e.g.[5, 37]).

The process of training an artificial neural network as (17) is performed by the minimization of a given functional $J(\Theta_1, \phi_1, \Theta_2, \phi_2, \ldots, \Theta_n, \phi_n)$. We search for optimal sets of parameters $\{\Theta_j^*\}_{j=1}^n$ and $\{\phi_j^*\}_{j=1}^n$ minimizing the cost functional $J$. For simplicity, in what follows we will denote all the parameters (weights and bias) of an artificial neural network by $\theta \in \Phi$, for a given set $\Phi$ of admissible parameters. A standard cost functional is constructed with a sample training set of known values $\{x_1, x_2, \ldots, x_{N_s}\}$ and its corresponding labels $\{y_1, y_2, \ldots, y_{N_s}\}$ as follows:

$$J(\theta) = \frac{1}{2}\sum_{i=1}^{N_s} \left(y_i - F(\mathrm{ANN}(x_i;\theta))\right)^2,$$

(for some real function $F$) which is known as supervised learning [14]. Training an artificial neural network means to solve the following minimization problem:

$$\theta^* = \underset{\theta\in\Phi}{\operatorname{argmin}}\, J(\theta). \tag{19}$$

Thus, the artificial neural network evaluated in the optimal $\theta^*$ (i.e., $\mathrm{ANN}(x;\theta^*)$) is the trained network. There are many sophisticated tailor-made procedures to perform the minimization in (19) efficiently. The reader may refer to [35] for inquiring into this topic, which is out of the scope of this paper.

## 3.2 Offline procedures

The first step is to choose an artificial neural network $\mathrm{ANN}(\cdot\,;\theta)$ that will define a family $\mathcal{W}$ of positive weight-functions to be used in the weighted inner products $\{(\cdot,\cdot)_{\mathbb{V},\omega}\}_{\omega\in\mathcal{W}}$. Typically we have:

$$\mathcal{W} = \{\omega(\cdot) = g(\mathrm{ANN}(\cdot\,;\theta)) : \theta \in \Phi\},$$

where $g$ is a suitable positive continuous function.

Next, given a discrete trial-test pairing $\mathbb{U}_h$-$\mathbb{V}_h$ satisfying (12), we construct the map $\mathcal{W}\times\Lambda \ni (\omega,\lambda) \mapsto q(u_{h,\lambda,\omega}) \in \mathbb{R}$, where $u_{h,\lambda,\omega} \in \mathbb{U}_h$ is the second component of the solution the mixed system (8). Having coded this map, we proceed to *train* the ANN by computing:

$$\begin{cases} \theta^* = \underset{\theta\in\Phi}{\operatorname{argmin}} \dfrac{1}{2}\displaystyle\sum_{i=1}^{n} \left|q(u_{h,\lambda_i,\omega}) - q(u_{\lambda_i})\right|^2, \\ \text{s.t.} \begin{cases} \omega(\cdot) = g(\mathrm{ANN}(\cdot\,;\theta)) \\ (r_{h,\lambda_i,\omega}, v_h)_{\mathbb{V},\omega} + b(u_{h,\lambda_i,\omega}, v_h) = \ell_{\lambda_i}(v_h), \quad \forall\, v_h \in \mathbb{V}_h, \\ b(w_h, r_{h,\lambda_i,\omega}) = 0, \qquad\qquad \forall\, w_h \in \mathbb{U}_h. \end{cases} \end{cases} \tag{20}$$

The last step is to build the matrices of the linear system needed for the online phase. Denote the basis of $\mathbb{U}_h$ by $\{\psi_1, ..., \psi_n\}$, and the basis of $\mathbb{V}_h$ by $\{\varphi_1, ..., \varphi_m\}$ (recall that $m > n$).

Having $\theta^* \in \Phi$ approaching (20), we extract from the mixed system (8) the matrices $A \in \mathbb{R}^{m\times m}$ and $B \in \mathbb{R}^{m\times n}$ such that:

$$A_{ij} = (\varphi_j, \varphi_i)_{\mathbb{V},\omega^*} \qquad \text{and} \qquad B_{ij} = b(\psi_j, \varphi_i),$$

where $\omega^*(\cdot) = g(\mathrm{ANN}(\cdot\,;\theta^*))$. Finally, we store the matrices $B^TA^{-1}B \in \mathbb{R}^{n\times n}$ and $B^TA^{-1} \in \mathbb{R}^{n\times m}$ to be used in the online phase to compute directly $u_{h,\lambda,\omega^*} \in \mathbb{U}_h$ for any right hand side $\ell_\lambda \in \mathbb{V}^*$. Basically, we have condensed-out the residual variable of the mixed system (8), since it is useless for the application of the quantity of interest $q \in \mathbb{U}^*$. In addition, it will be also important to store the vector $Q \in \mathbb{R}^n$ such that:

$$Q_j := q(\psi_j)\,, \quad j = 1, ..., n.$$

### 3.3 Online procedures

For each $\lambda \in \Lambda$ for which we want to obtain the quantity of interest $q(u_{h,\lambda,\omega^*})$, we first compute the vector $L_\lambda \in \mathbb{R}^m$ such that its $i$-th component is given by:

$$(L_\lambda)_i = \langle \ell_\lambda, \varphi_i \rangle_{\mathbb{V}^*,\mathbb{V}}\ ,$$

where $\varphi_i$ is the $i$-th vector of in the basis of $\mathbb{V}_h$. Next, we compute

$$q(u_{h,\lambda,\omega^*}) = Q^T(B^TA^{-1}B)^{-1}B^TA^{-1}L_\lambda.$$

Observe that the matrix $Q^T(B^TA^{-1}B)^{-1}B^TA^{-1}$ can be fully obtained and stored from the previous offline phase (see Section 3.2).

## 4 Numerical tests

In this section, we show some numerical examples in 1D and 2D to investigate the main features of the proposed ML-MRes finite element method. In particular, we consider in the following order: 1D diffusion, 1D advection, 1D advection with multiple QoIs, and finally 2D diffusion.

### 4.1 1D diffusion with one QoI

We recover here the motivational example from the introduction (see Section 1.1). Consider the variational formulation (2), with trial and test spaces $\mathbb{U} = \mathbb{V} = H^1_{(0}(0,1)$. We endowed $\mathbb{V}$ with the weighted inner product:

$$(v_1, v_2)_{\mathbb{V},\omega} := \int_0^1 \omega\, v_1' v_2'\,, \qquad \forall\, v_1, v_2 \in \mathbb{V}.$$

As in the introduction, we consider the simplest coarse discrete trial space $\mathbb{U}_h := \text{Span}\{\psi\} \subset \mathbb{U}$, where $\psi(x) = x$. The optimal test function (see Section 2.3.1), paired with the trial function $\psi$, is given by $\varphi := T_\omega \psi \in \mathbb{V}$, which is the solution of (14) with $u = \psi$. Hence,

$$\varphi(x) = \int_0^x \frac{1}{\omega(s)} ds. \tag{21}$$

Let us consider the Petrov-Galerkin formulation with *optimal* test functions, which is equivalent to the mixed system (8) in the optimal case $\mathbb{V}_h = \mathbb{V}$. Consequently, the Petrov-Galerkin scheme with trial function $\psi$ and optimal test function $\varphi$, delivers the discrete solution $u_{h,\lambda,\omega}(x) = x\varphi(\lambda)/\varphi(1)$ (notice that the trivial weight $\omega \equiv 1$ recovers the test function $\varphi = \psi$, and therefore the standard Galerkin approach).

Recalling the exact solution (3), we observe that the relative error in the quantity of interest for our Petrov-Galerkin approach is:

$$\text{Err} = \begin{cases} \left|1 - \frac{\varphi(\lambda)}{\varphi(1)}\right| & \text{if } x_0 \leq \lambda, \\ \left|1 - \frac{x_0}{\lambda}\frac{\varphi(\lambda)}{\varphi(1)}\right| & \text{if } x_0 \geq \lambda. \end{cases} \tag{22}$$

Of course, any function such that $\varphi(\lambda) = \varphi(x_0) \neq 0$ for $\lambda \geq x_0$, and $\varphi(\lambda) = \lambda\varphi(x_0)/x_0$ for $\lambda \leq x_0$, will produce *zero error for all* $\lambda \in (0,1)$. Notice that such a function indeed exists, and in this one-dimensional setting it solves the adjoint problem:

$$\begin{cases} \text{Find } z \in H^1_{(0}(0,1) \text{ such that:} \\ \displaystyle\int_0^1 w'z' = w(x_0), \qquad \forall\, w \in H^1_{(0}(0,1). \end{cases}$$

This optimal test function is also obtained in our framework via (21), by using a limiting weight of the form:

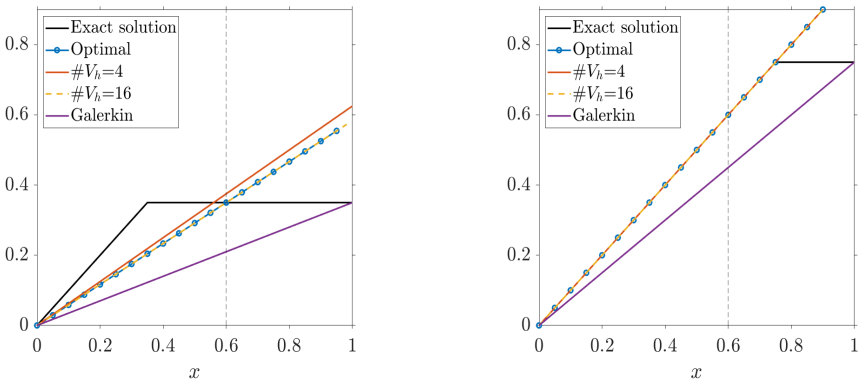
$$\omega(x) \to \begin{cases} c & \text{if } x < x_0, \\ +\infty & \text{if } x > x_0, \end{cases} \tag{23}$$

for some constant $c > 0$. Hence, the Petrov–Galerkin method using a test function of the form (21) has sufficient variability to eliminate any errors for any $\lambda$!

We now restrict the variability by parametrizing $\omega$. In the motivating example given in Section 1.1, for illustration reasons we chose a weight of the form $\omega(x) = \sigma(\theta_1 x + \theta_2)$, which corresponds to the simplest artificial neural network, having only one layer with one neuron. We now select a slightly more complex family of weights having the form $\omega(x) = \exp(\text{ANN}(x;\theta))$, where

$$\text{ANN}(x;\theta) = \sum_{j=1}^{5} \theta_{j3}\sigma(\theta_{j1}x + \theta_{j2}). \tag{24}$$

Notice that the weight function $\omega(\cdot)$ is positive and the artificial neural network $\text{ANN}(x;\theta)$ corresponds to a network with one hidden layer (five neurons in the hidden layer and no bias parameter in the output layer).




(a) $\lambda = 0.35$ (b) $\lambda = 0.75$

Figure 2: Discrete solutions computed using the optimal test function approach (blue line), and discrete mixed form approach (8) with different discrete test spaces $\mathbb{V}_h$ (red and yellow lines). Dotted line shows the QoI location.

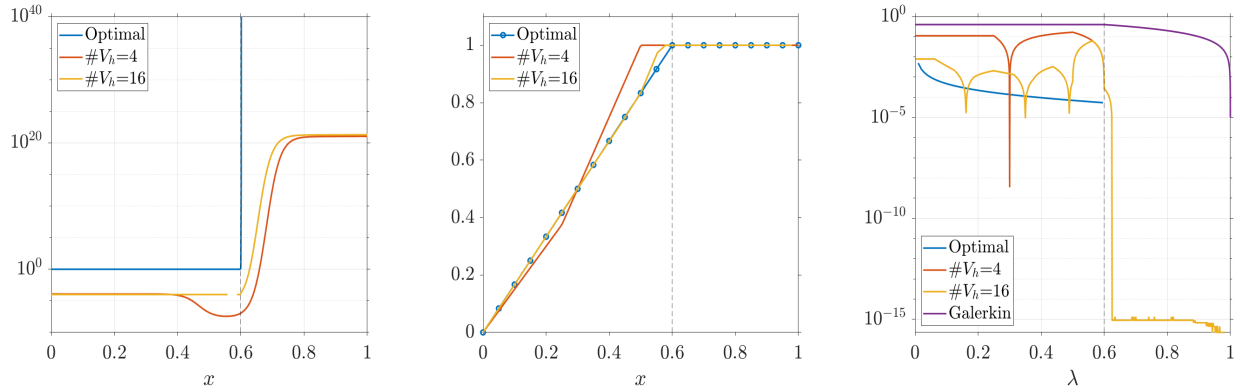
The training set of parameters has been chosen as $\lambda_i = 0.1i$, with $i = 1, ..., 9$. For comparisons, we perform three different experiments. The first experiment trains the network (24) based on a cost functional that uses the relative error formula (22), where the optimal test function $\varphi$ is computed using eq. (21). The other two experiments use the training approach (20), with discrete spaces $\mathbb{V}_h$ consisting of conforming piecewise linear functions over uniform meshes of 4 and 16 elements respectively. The quantity of interest has been set to $x_0 = 0.6$, which does not coincide with a node of the discrete test spaces. Figure 2 shows the obtained discrete solutions $u_{h,\lambda,\omega^*}$ for each experiment, and for two different values of $\lambda$. Figure 3a shows the trained weight obtained for each experiment (cf. eq. (23)), while Figure 3b depicted the associated optimal and projected-optimal test functions linked to those trained weights. Finally, Figure 3c shows the relative errors in the quantity of interest for each discrete solution in terms of the $\lambda$ parameter.

It can be observed that the trained method using a parametrized weight function based on (24), while consisting of only one degree of freedom, gives quite accurate quantities of interest for the entire range of $\lambda$. This should be compared to the $O(1)$ error for standard Galerkin given by (4). We note that some variation can be observed depending on whether the optimal or a projected optimal test function is used (with a richer $\mathbb{V}_h$ being better).

## 4.2 1D advection with one QoI

Consider the family of ODEs:

$$\begin{cases} u' = f_\lambda & \text{in } (0,1), \\ u(0) = 0, \end{cases} \tag{25}$$




(a) Trained weights (b) Optimal test functions (c) Relative errors in QoI

Figure 3: Trained weights, optimal (and projected-optimal) test functions, and relative errors computed with three different approaches. Dotted line shows the QoI location.

for a family of continuous functions $\{f_\lambda\}_{\lambda\in[0,1]}$ given by $f_\lambda(x) := (x-\lambda)\mathbb{1}_{[\lambda,1]}(x)$, where $\mathbb{1}_{[\lambda,1]}$ denotes the characteristic function of the interval $[\lambda, 1]$. The exact solution of (25) will be used as a reference solution and is given by $u_\lambda(x) = \frac{1}{2}(x-\lambda)^2\mathbb{1}_{[\lambda,1]}(x)$. The quantity of interest considered for this example will be $q_{x_0}(u_\lambda) := u_\lambda(x_0)$, where $x_0$ could be any value in $[0, 1]$.

Let us consider the following variational formulation of problem (25):

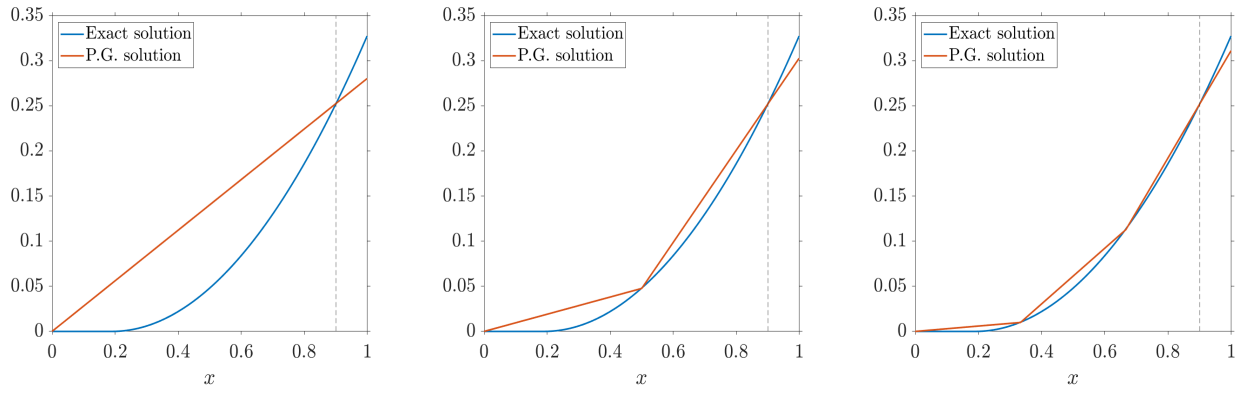
$$\begin{cases} \text{Find } u_\lambda \in \mathbb{U} \text{ such that:} \\ b(u_\lambda, v) := \displaystyle\int_0^1 u_\lambda' v = \int_0^1 f_\lambda v =: \ell_\lambda(v), \quad \forall\, v \in \mathbb{V}, \end{cases}$$

where $\mathbb{U} := H^1_{(0}(0,1) := \{u \in L^2(0,1) : u' \in L^2(0,1) \wedge u(0) = 0\}$, and $\mathbb{V} := L^2(0,1)$ is endowed with the weighted inner-product:

$$(v_1, v_2)_{\mathbb{V},\omega} := \int_0^1 \omega\, v_1 v_2\,, \quad \forall\, v_1, v_2 \in \mathbb{V}.$$

We want to approach this problem using coarse discrete trial spaces $\mathbb{U}_h \subset \mathbb{U}$ of piecewise linear polynomials on a partition of one, two and three elements.

We describe the weight $\omega(x)$ by the sigmoid of an artificial neural network that depends on parameters $\theta$, i.e., $\omega(x) = \sigma(\text{ANN}(x;\theta)) > 0$ (see Section 3.1). In particular, we use the artificial neural network given in (24). To train such a network, we consider a training set $\{\lambda_i\}_{i=1}^9$, where $\lambda_i = 0.125(i-1)$, together with the set of exact quantities of interest $\{q_{x_0}(u_{\lambda_i})\}_{i=1}^9$, computed using the reference exact solution with $x_0 = 0.9$. The training procedure uses the constrained minimization problem (20), where for each low-resolution trial space $\mathbb{U}_h$ (based on one, two and three elements), the same discrete test space $\mathbb{V}_h$ has been used: a high-resolution




(a) One element (b) Two elements (c) Three elements

Figure 4: Petrov-Galerkin solution with projected optimal test functions with trained weight. Dotted line shows the QoI location (0.9) and parameter value is $\lambda = 0.19$.

space of piecewise linear and continuous functions linked to a uniform partition of 128 elements. The minimization algorithm has been stopped once the cost functional reaches the tolerance $\texttt{tol} = 9 \cdot 10^{-7}$.

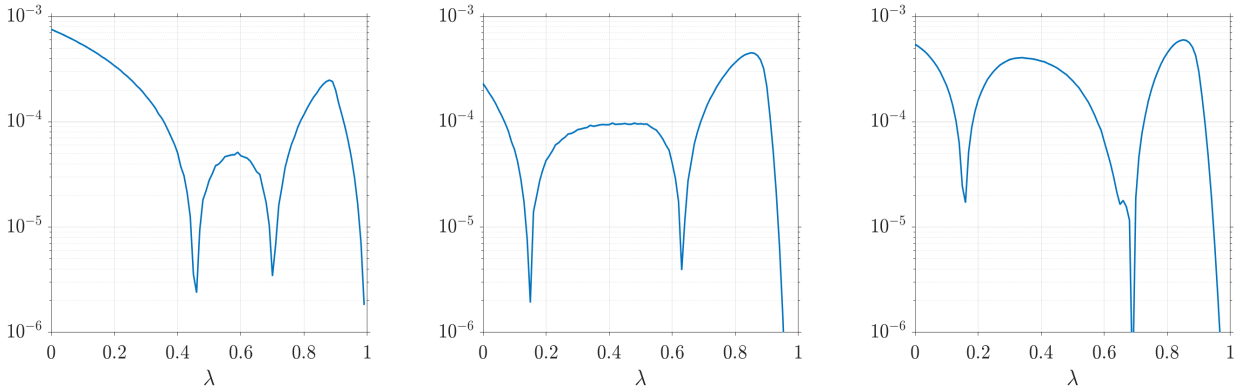
After an optimal parameter $\theta^*$ has been found (see (20)), we follow the matrix procedures described in Section 3.2 and Section 3.3 to approximate the quantity of interest of the discrete solution for any $\lambda \in [0, 1]$.

Figures 4 and 5 show numerical experiments considering model problem (25) in three different trial spaces. Figure 4 shows, for $\lambda = 0.19$, the exact solution and the Petrov-Galerkin solution computed with projected optimal test functions given by the trained weighted inner-product. Notice that for the three cases (with one, two, and three elements) the Petrov-Galerkin solution intends to approximate the quantity of interest (dotted line).

Figure 5 displays the QoI error $|q_{x_0}(u_\lambda) - q_{x_0}(u_{\lambda,h,\omega^*})|$ for different values of $\lambda \in [0, 1]$. When the ANN-training stops at a cost functional smaller than $\texttt{tol} = 9 \cdot 10^{-7}$, the QoI error remains smaller than $10^{-3}$ for all $\lambda \in [0.1]$. In particular, Figure 5a shows that even in the simplest case of one-degree of freedom, it is possible to get reasonable approximations of the QoI for the entire range of $\lambda$.

## 4.3 1D advection with multiple QoIs

This example is based on the same model problem of Section 4.2, but now we intend to approach two quantities of interest simultaneously: $q_1(u_\lambda) := u_\lambda(x_1)$ and $q_2(u_\lambda) := u_\lambda(x_2)$, where $x_1, x_2 \in [0, 1]$ are two different values. We also have considered now discrete trial spaces based on three, four and five elements. The training routine has been modified accordingly,




(a) One DoF (b) Two DoF (c) Three DoF

Figure 5: Absolute error between QoI of exact and approximate solutions for different $\lambda$ values.

and is driven now by the following minimization problem:

$$\begin{cases} \theta^* = \underset{\theta\in\Phi}{\operatorname{argmin}} \dfrac{1}{2}\displaystyle\sum_{i=1}^{N_s} \left|q_1(u_{h,\lambda_i,\omega}) - q_1(u_{\lambda_i})\right|^2 + \left|q_2(u_{h,\lambda_i,\omega}) - q_2(u_{\lambda_i})\right|^2, \\ \text{subject to: } \begin{cases} \omega(\cdot) = \sigma(\mathrm{ANN}(\cdot\,;\theta)). \\ (r_{h,\lambda_i,\omega}, v_h)_{\mathbb{V},\omega} + b(u_{h,\lambda_i,\omega}, v_h) &= \ell_{\lambda_i}(v_h), \quad \forall\, v_h \in \mathbb{V}_h, \\ b(w_h, r_{h,\lambda_i,\omega}) &= 0, \quad \forall\, w_h \in \mathbb{U}_h. \end{cases} \end{cases}$$

For this example, we consider a training set of size $N_s = 12$ with $\lambda_i = (i-1)/11$, for all $i = 1, \ldots, 12$. The weight $\omega(x)$ will be described by the sigmoid of an artificial neural network that depends on one single hidden layer and $N_n = 6$ hidden neurons. Numerical results are depicted in Figures 6 and 7 for $x_1 = 0.3$ and $x_2 = 0.7$. Accurate values of both QoIs are obtained for the entire range of $\lambda$. These results are roughly independent of the size of the trial space.

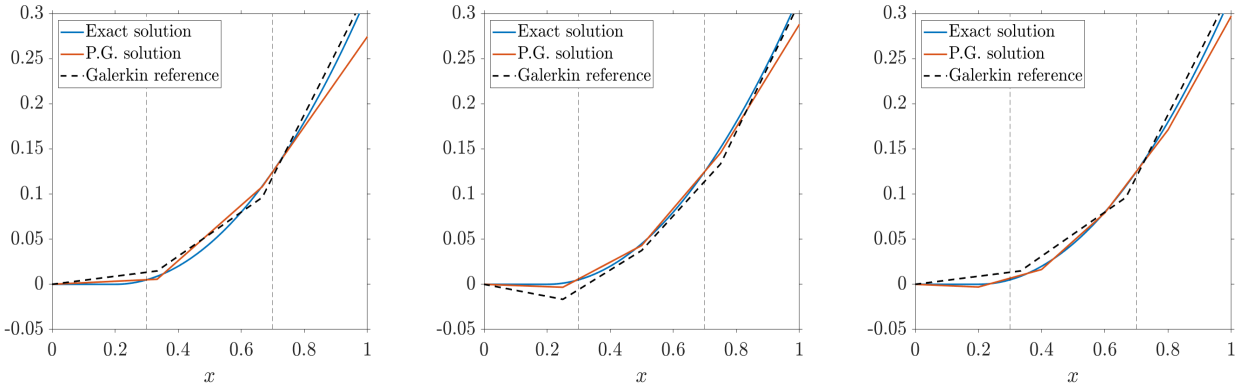
## 4.4 2D diffusion with one QoI

Consider the two-dimensional unit square $\Omega = [0,1] \times [0,1]$ and the family of PDEs:

$$\begin{cases} -\Delta u = f_\lambda & \text{in } \Omega, \\ u = 0 & \text{over } \partial\Omega, \end{cases} \tag{26}$$

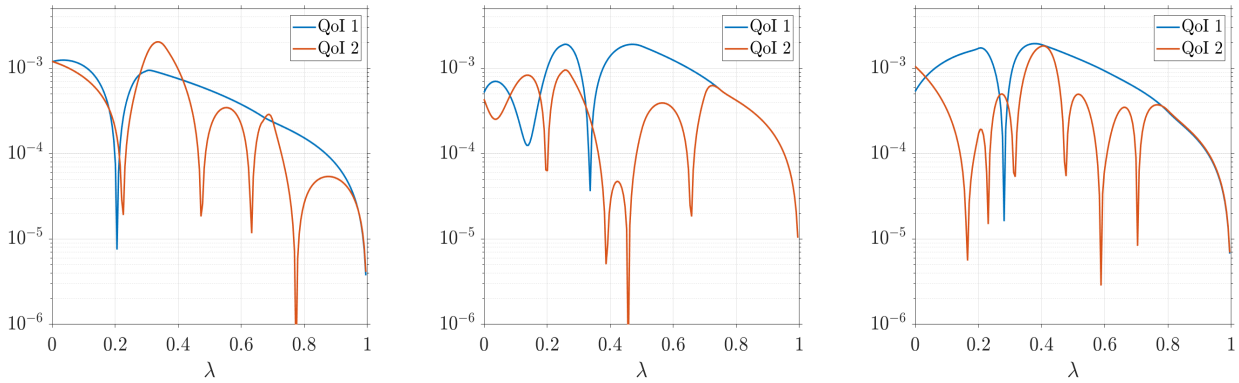
where the family of functions $\{f_\lambda\}_{\lambda\in(0,1)}$ is described by the formula:

$$\begin{aligned} f_\lambda(x_1, x_2) = 2\pi^2(1+\lambda^2)\sin(\pi x_1)\sin(\lambda\pi x_1)\sin(\pi x_2)\sin(\lambda\pi x_2) - \\ 2\lambda\pi^2[\cos(\pi x_1)\cos(\lambda\pi x_1)\sin(\pi x_2)\sin(\lambda\pi x_2) + \\ \sin(\pi x_1)\sin(\lambda\pi x_1)\cos(\pi x_2)\cos(\lambda\pi x_2)]. \end{aligned}$$




(a) Three elements (b) Four elements (c) Five elements

Figure 6: Petrov-Galerkin solution with projected optimal test functions with trained weight. Dotted lines show the QoI locations (0.3 and 0.7) and parameter value is $\lambda = 0.2$.




(a) Three DoF (b) Four DoF (c) Five DoF

Figure 7: Absolute error between QoI of exact and approximate solutions for different $\lambda$ values, for each QoI $q_1(u) = u(0.3)$ and $q_2(u) = u(0.7)$.

Accordingly, the reference exact solution of (26) is:

$$u_\lambda(x) = \sin(\pi x_1)\sin(\lambda\pi x_1)\sin(\pi x_2)\sin(\lambda\pi x_2).$$

The quantity of interest chosen for this example will be the average

$$q(u_\lambda) := \frac{1}{|\Omega_0|}\int_{\Omega_0} u_\lambda(x)\,dx, \tag{27}$$

with $\Omega_0 := [0.79\,,\,0.81] \times [0.39\,,\,0.41] \subset \Omega$ (see Figure 8).

The variational formulation of problem (26) will be:

$$\begin{cases} \text{Find } u_\lambda \in \mathbb{U} \text{ such that:} \\ b(u_\lambda, v) := \displaystyle\int_\Omega \nabla u_\lambda \cdot \nabla v = \int_\Omega f_\lambda v =: \ell_\lambda(v), \qquad \forall\, v \in \mathbb{V}, \end{cases}$$

where $\mathbb{U} = \mathbb{V} = H_0^1(\Omega)$, and $\mathbb{V}$ is endowed with the weighted inner-product:

$$(v_1, v_2)_{\mathbb{V},\omega} := \int_\Omega \omega\, \nabla v_1 \cdot \nabla v_2\,, \quad \forall\, v_1, v_2 \in \mathbb{V}.$$

As in the previous example, the weight is going to be determined using an artificial neural network so that $\omega(x_1, x_2) = \sigma(\mathrm{ANN}(x_1, x_2; \theta))$. Such a network is composed by one hidden layer with $N_n = 5$ neurons (10 weights and 5 bias parameters) and one output layer (5 weights and no bias term). Hence, $\theta$ contains 20 parameters to estimate, i.e.,

$$\mathrm{ANN}(x_1, x_2; \theta) = \sum_{j=1}^{N_n} \theta_{j4} \sigma(\theta_{j1} x_1 + \theta_{j2} x_2 + \theta_{j3}).$$

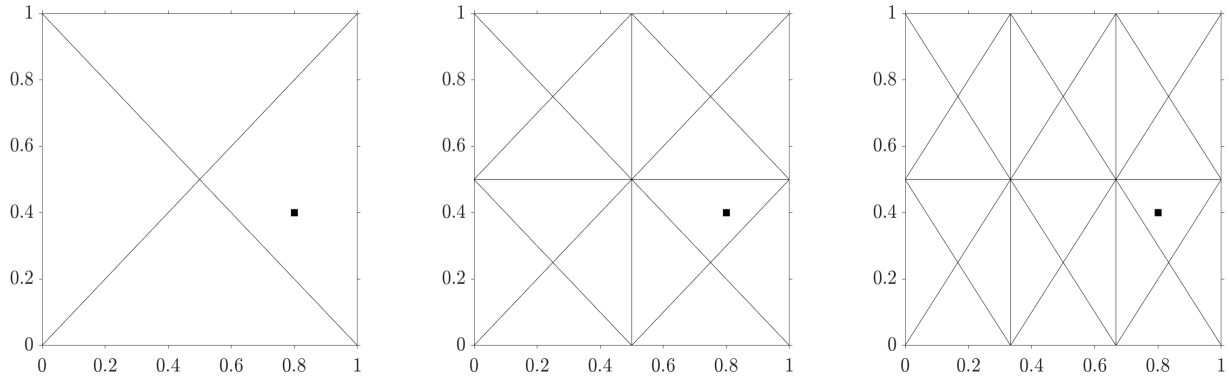
To train the ANN, we use the inputs $\{\lambda_i\}_{i=1}^9$, where $\lambda_i = 0.125(i-1)$, and its corresponding quantities of interest $\{q(u_{\lambda_i})\}_{i=1}^9$, by means of equation (27). Again, the training procedure is based on the constrained minimization (20). For the experiments, we use coarse discrete trial spaces $\mathbb{U}_h$ having one, five, and eight degrees of freedom respectively (see Figure 8). In each case, the test space $\mathbb{V}_h$ has been set to be a piecewise quadratric conforming polynomial space, over a uniform triangular mesh of 1024 elements. The minimization algorithm (20) stops when a tolerance $\texttt{tol} = 9 \cdot 10^{-7}$ is reached.

The errors on the QoI are depicted in Figure 9 for each trial space under consideration, and show relative errors below $10^{-3}$ for the entire range of $\lambda$.

# 5 Conclusions

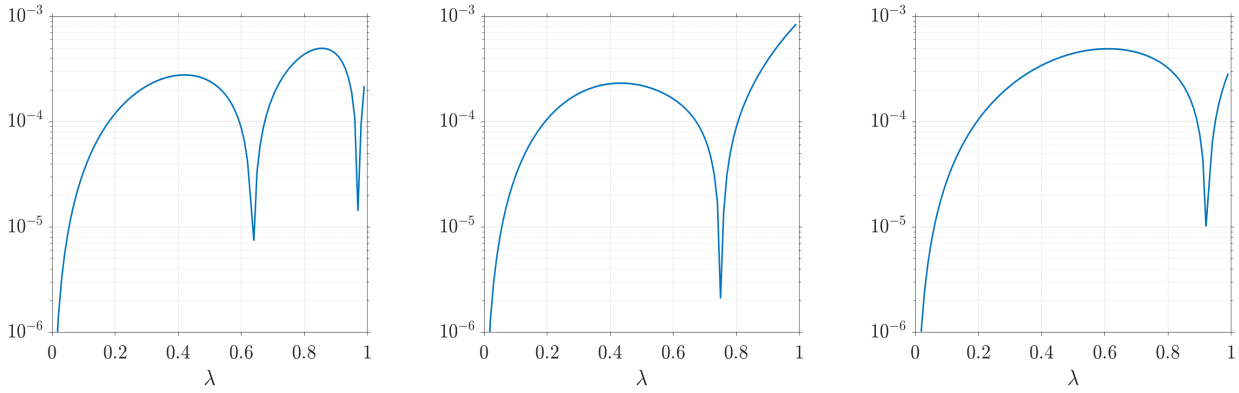
In this paper, we introduced the concept of machine-learning minimal-residual (ML-MRes) finite element methods. These methods are tuned within a machine-learning framework, and are tailored for the accurate computation of output quantities of interest, regardless of the underlying mesh size. In fact, on coarse meshes, they deliver significant improvements in the accuracy compared to standard methods on such meshes.

We presented a stability analysis for the discrete method, which can be regarded as a Petrov–Galerkin scheme with parametric test space that is equivalent to a minimal-residual formulation measuring the residual in a (discrete) dual weighted norm. Numerical examples were presented for elementary one- and two-dimensional elliptic and hyperbolic problems, in which weight functions are tuned that are represented by artificial neural networks with up to 20 parameters.



(a) One DoF (b) Five DoF (c) Eight DoF

Figure 8: Meshes considered for the discrete trial space $\mathbb{U}_h$. The black square represent the quantity of interest location $\Omega_0 = [0.79\,,\,0.81] \times [0.39\,,\,0.41]$.




(a) One DoF (b) Five DoF (c) Eight DoF

Figure 9: Absolute error between QoI of exact and approximate solutions for different $\lambda$ values.

Various extensions of our methodology are possible. While we only focussed on linear quantities of interest, nonlinear ones can be directly considered. Also, it is possible to consider a dependence of the bilinear form on $\lambda$, however, this deserves a completely separate treatment, because of the implied $\lambda$-dependence of the trial-to-test map and the $B$-matrix. An open problem of significant interest is the dependence of the performance of the trained method on the richness of the parametrized weight function. While we showed that in the simplest example of 1-D diffusion with one degree of freedom, the weight function allows for exact approximation of quantities of interest, it is not at all clear if this is valid in more general cases, and what the effect is of (the size of) parametrization.

Another subject that deserves further attention is the cost of obtaining synthetic training and validation datasets, versus the total cost of the methodology. Of course, this is not an

issue when datasets are given, e.g., by experimental measurements. However, reliable synthetic datasets can be expensive to produce, and this cost cannot be ignored when evaluating the overall cost of the method.

# A Proof of Theorem 2.B

The mixed scheme (8) has a classical *saddle point* structure, which is uniquely solvable since the *top left* bilinear form is an inner-product (therefore coercive) and $b(\cdot,\cdot)$ satisfies the discrete inf-sup condition (12) (see, e.g., [12, Proposition 2.42]).

The a priori estimates (13) are well-known in the residual minimization FEM literature (see, e.g., [15, 4, 29]). However, for the sake of completeness, we show here how to obtain them. Let $v_{h,\lambda,\omega} \in \mathbb{V}_h$ be such that (cf. (15)):

$$(v_{h,\lambda,\omega}, v_h)_{\mathbb{V},\omega} = b(u_{h,\lambda,\omega}, v_h), \quad \forall\, v_h \in \mathbb{V}_h\,. \tag{28}$$

In particular, combining eq. (28) with eq. (8b) of the mixed scheme, we get the orthogonality property:

$$(v_{h,\lambda,\omega}, r_{h,\lambda,\omega})_{\mathbb{V},\omega} = b(u_{h,\lambda,\omega}, r_{h,\lambda,\omega}) = 0\,. \tag{29}$$

To get the first estimate observe that:

$$\begin{aligned}
\|u_{h,\lambda,\omega}\|_{\mathbb{U}} \leq & \frac{1}{\gamma_{h,\omega}} \sup_{v_h\in\mathbb{V}_h} \frac{|b(u_{h,\lambda,\omega}, v_h)|}{\|v_h\|_{\mathbb{V},\omega}} && \text{(by (12))}\\
= & \frac{1}{\gamma_{h,\omega}} \sup_{v_h\in\mathbb{V}_h} \frac{|(v_{h,\lambda,\omega}, v_h)_{\mathbb{V},\omega}|}{\|v_h\|_{\mathbb{V},\omega}} && \text{(by (28))}\\
= & \frac{1}{\gamma_{h,\omega}} \frac{|(v_{h,\lambda,\omega}, v_{h,\lambda,\omega})_{\mathbb{V},\omega}|}{\|v_{h,\lambda,\omega}\|_{\mathbb{V},\omega}} && \text{(since } v_{h,\lambda,\omega} \text{ is the supremizer)}\\
= & \frac{1}{\gamma_{h,\omega}} \frac{|(r_{h,\lambda,\omega} + v_{h,\lambda,\omega}, v_{h,\lambda,\omega})_{\mathbb{V},\omega}|}{\|v_{h,\lambda,\omega}\|_{\mathbb{V},\omega}} && \text{(by (29))}\\
= & \frac{1}{\gamma_{h,\omega}} \frac{|\ell_\lambda(v_{h,\lambda,\omega})|}{\|v_{h,\lambda,\omega}\|_{\mathbb{V},\omega}} && \text{(by (28) and (8a))}\\
= & \frac{1}{\gamma_{h,\omega}} \frac{|b(u_\lambda, v_{h,\lambda,\omega})|}{\|v_{h,\lambda,\omega}\|_{\mathbb{V},\omega}} && \text{(by (6))}\\
\leq & \frac{M_\omega}{\gamma_{h,\omega}} \|u_\lambda\|_{\mathbb{U}}\,. && \text{(by (10))}
\end{aligned}$$

For the second estimate we define the projector $P : \mathbb{U} \to \mathbb{U}_h$, such that $Pu \in \mathbb{U}_h$ corresponds to the second component of the solution of the mixed system (8) with right hand side $b(u,\cdot) \in \mathbb{V}^*$ in (8a). We easily check that $P$ is a bounded linear proyector satisfying $P^2 = P \neq 0, I$, and $\|P\| \leq M_\omega/\gamma_{h,\omega}$. Hence, from Kato's identity $\|I - P\| = \|P\|$ for Hilbert space projectors [20], we get for any $w_n \in \mathbb{U}_n$:

$$\|u_\lambda - u_{h,\lambda,\omega}\|_{\mathbb{U}} = \|(I-P)u_\lambda\|_{\mathbb{U}} = \|(I-P)(u_\lambda - w_h)\|_{\mathbb{U}} \leq \|P\|\|u_\lambda - w_h\|_{\mathbb{U}}\,.$$

Thus, the a priori error estimate follows using the bound of $\|P\|$ and taking the infimum over all $w_h \in \mathbb{U}_h$.

## Acknowledgements

The authors want to thank Professor David Pardo for helpful discussions. The work by IM was done in the framework of Chilean FONDECYT research project #1160774. IM has also received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Sklodowska-Curie grant agreement No 777778 (MATHROCKS). The work by IB was partially supported by PUCV DI Postdoctorado 2019 and FONDECYT grant No 3200827. The research by KvdZ was supported by the Engineering and Physical Sciences Research Council (EPSRC) under grant EP/T005157/1.

## References

[1] R. Becker and R. Rannacher, *An optimal control approach to a posteriori error estimation in finite element methods*, Acta Numer., 10 (2001), pp. 1–102.

[2] J. Berg and K. Nyström, *A unified deep artificial neural network approach to partial differential equations in complex geometries*, Neurocomputing, 317 (2018), pp. 28–41.

[3] J. Berg and K. Nyström, *Data-driven discovery of PDEs in complex datasets*, Journal of Computational Physics, 384 (2019), pp. 239–252.

[4] D. Broersen and R. Stevenson, *A robust Petrov–Galerkin discretisation of convection–diffusion equations*, Comput. Math. Appl., 68 (2014), pp. 1605–1618.

[5] J.-T. Chien, *Source separation and machine learning*, Academic Press, 2018.

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

[7] L. Demkowicz and J. Gopalakrishnan, *A class of discontinuous Petrov-Galerkin methods. II. Optimal test functions*, Numerical Methods for Partial Differential Equations, 27 (2010), pp. 70–105.

[8] L. Demkowicz and J. Gopalakrishnan, *An overview of the discontinuous Petrov Galerkin method*, in Recent Developments in Discontinuous Galerkin Finite Element Methods for Partial Differential Equations: 2012 John H Barrett Memorial Lectures, X. Feng, O. Karakashian, and Y. Xing, eds., vol. 157 of The IMA Volumes in Mathematics and its Applications, Springer, Cham, 2014, pp. 149–180.

[9] N. Discacciati, J. S. Hesthaven, and D. Ray, *Controlling oscillations in high-order Discontinuous Galerkin schemes using artificial viscosity tuned by neural networks*, To appear in J. Comput. Phys (2020), (2020).

[10] W. E and B. Yu, *The deep Ritz method: A deep learning-based numerical algorithm for solving variational problems*, Commun. Math. Stat., 6 (2018), pp. 1–12.

[11] B. Endtmeyer and T. Wick, *A partition-of-unity dual-weighted residual approach for multi-objective goal functional error estimation applied to elliptic problems*, Comput. Methods Appl. Math., 17 (2017), pp. 575–600.

[12] A. Ern and J.-L. Guermond, *Theory and practice of finite elements*, Springer, New-York, 2004.

[13] M. Feischl, D. Praetorius, and K. G. van der Zee, *An abstract analysis of optimal goal-oriented adaptivity*, SIAM J. Numer. Anal., 54 (2016), pp. 1423–1448.

[14] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*, The MIT Press, 2019.

[15] J. Gopalakrishnan and W. Qiu, *An analysis of the practical DPG method*, Math. Comp., 83 (2014), pp. 537–552.

[16] B. Hayhurst, M. Keller, C. Rai, X. Sun, and C. R. Westphal, *Adaptively weighted least squares finite element methods for partial differential equations with singularities*, Comm. Applied Math. and Comput. Science, 13 (2018), pp. 1–25.

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

[18] C. F. Higham and D. J. Higham, *Deep learning: An introduction for applied mathematicians*, SIREV, 61 (2019), pp. 860–891.

[19] A. C. Ionita and A. C. Antoulas, *Data-driven parametrized model reduction in the Lowner framework*, SIAM Journal of Scientific Computing, 36 (2014), pp. A984–A1007.

[20] T. Kato, *Estimation of iterated matrices with application to von Neumann condition*, Numer. Math., 2 (1960), pp. 22–29.

[21] K. Kergrene, S. Prudhomme, L. Chamoin, and M. Laforest, *A new goal-oriented formulation of the finite element method*, Comput. Methods Appl. Mech. Engrg., 327 (2017), pp. 256–276.

[22] G. Kutyniok, P. Petersen, M. Raslan, and R. Schneider, *A theoretical analysis of deep neural networks and parametric PDEs*, arXiv preprint, arXiv:1904.00377v2 (2019).

[23] I. Lagaris, A. Likas, and D. Fotiadis, *Artificial neural networks for solving ordinary and partial differential equations*, IEEE Transactions on Neural Networks, 9 (1998), pp. 987–1000.

[24] Y. LeCun, Y. Bengio, and G. Hinton, *Deep learning*, Nature, 521 (2015), pp. 436–444.

[25] H. Lee and I. S. Kang, *Neural algorithm for solving differential equations*, Journal of Computational Physics, 91 (1990), pp. 110–131.

[26] J. Ling, A. Kurzawski, and J. Templeton, *Reynolds averaged turbulence modelling using deep neural networks with embedded invariance*, J. Fluid Mech., 807 (2016), pp. 155–166.

[27] S. Mishra, *A machine learning framework for data driven acceleration of computations of differential equations*, Mathematics in Engineering, 1 (2018), pp. 118–146.

[28] M. S. Mommer and R. Stevenson, *A goal-oriented adaptive finite element method with convergence rates*, SIAM J. Numer. Anal., 47 (2009), pp. 861–886.

[29] I. Muga and K. G. van der Zee, *Discretization of linear problems in Banach spaces: Residual minimization, nonlinear Petrov–Galerkin, and monotone mixed methods.* http://arxiv.org/abs/1511.04400arXiv:1511.04400v3 [math.NA], 2018.

[30] J. T. Oden and S. Prudhomme, *Goal-oriented error estimation and adaptivity for the finite element method*, Comput. Math. Appl., 41 (2001), pp. 735–756.

[31] M. Raissi and G. E. Karniadakis, *Hidden physics models: Machine learning of nonlinear partial differential equations*, J. Comput. Phys., 357 (2018), pp. 125–141.

[32] D. Ray and J. S. Hesthaven, *An artificial neural network as a troubled-cell indicator*, J. Comput. Phys., 367 (2018), pp. 166–191.

[33] F. Regazzoni, L. Dedè, and A. Quarteroni, *Machine learning for fast and reliable solution of time-dependent differential equations*, Journal of Computational Physics, 397 (2019), p. 108852.

[34] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, *Data-driven discovery of partial differential equations*, Science Advances, 3 (2017), p. e1602614.

[35] S. Sra, S. Nowozin, and S. J. Wright, *Optimization for machine learning*, The MIT Press, 2011.

[36] R. Swischuka, L. Maininia, B. Peherstorferb, and K. Willcox, *Projection-based model reduction: Formulations for physics-based machine learning*, Computers and Fluids, 179 (2019), pp. 704–717.

[37] G. Tsihrintzis, D. N. Sotiropoulos, and L. C. Jain, *Machine learning paradigms: Advances in data analytics*, Springer, 2019.

[38] E. H. van Brummelen, S. Zhuk, and G. J. van Zwieten, *Worst-case multi-objective error estimation and adaptivity*, Comput. Methods Appl. Mech. Engrg., 313 (2017), pp. 723–743.