# A DEEP FIRST-ORDER SYSTEM LEAST SQUARES METHOD FOR SOLVING ELLIPTIC PDES

FRANCISCO M. BERSETCHE AND JUAN PABLO BORTHAGARAY

Abstract. We propose a First-Order System Least Squares (FOSLS) method based on deep-learning for numerically solving second-order elliptic PDEs. The method we propose is capable of dealing with either variational and non-variational problems, and because of its meshless nature, it can also deal with problems posed in high-dimensional domains. We prove the $\Gamma$-convergence of the neural network approximation towards the solution of the continuous problem and extend the convergence proof to some well-known related methods. Finally, we present several numerical examples illustrating the performance of our discretization.

## 1. Introduction

Approximate solution of PDEs using machine learning techniques has been considered in various forms in the past thirty years. For instance, [1, 2, 3, 4] propose to use neural networks to solve PDEs and ODEs. These articles compute neural network solutions by using an a priori fixed mesh. In recent years, there has been an incipient development of mesh-free numerical methods to solve PDEs by using neural networks. Although the approaches have been diverse, most of these algorithms aim to train a neural network to approximate the unknown function, forcing the fulfillment of the PDE and its boundary conditions through a suitable loss functional. In this regard, among other works, let us mention [5, 6, 7, 8, 9, 10, 11, 12, 13].

Deep neural networks are not necessarily suitable for solving PDEs in low dimensions, where they may be outperformed by classical methods specifically tailored for the problems under consideration. However, neural network methods have proven to be effective in some circumstances where the application of classical methods becomes impractical. Such is the case of high-dimensional PDEs. We refer to [14, 15] for discussion about the suitability of shallow neural networks for solving high-dimensional PDEs.

The method we propose in this work aims to overcome some disadvantages of the algorithms available in the literature. Using a first-order formulation we are able to avoid the computation of second-order derivatives in cost functionals, thereby saving a significant computational cost in high dimensions. Avoiding second-order derivatives also allows us to use linear activation functions and, a priori, gives us the possibility of approximating weak solutions. On the other hand, counting on explicit representations of the gradients simplifies the strong imposition of Neumann-type boundary conditions. Namely, we can impose boundary conditions without adding penalty terms in the loss function. This results in a reduction in training time. First order formulations have also been used in [12, 13].

Let $\Omega \subset \mathbb{R}^d$ be an open domain. In this work, we shall make use of the spaces

$$\begin{aligned} H^1(\Omega) &= \{v \in L^2(\Omega) \colon \nabla v \in L^2(\Omega)\}, \\ H(\mathrm{div}; \Omega) &= \{\boldsymbol{\psi} \in [L^2(\Omega)]^d \colon \operatorname{div} \boldsymbol{\psi} \in L^2(\Omega)\}. \end{aligned}$$

We assume there exists a disjoint partition $\partial\Omega = \Gamma_{\mathcal{D}} \cup \Gamma_{\mathcal{N}}$, with $|\Gamma_{\mathcal{D}}| > 0$, and let $\boldsymbol{\nu}$ denote the outward normal to $\Omega$. Given sufficiently regular functions $f, g_{\mathcal{D}}, g_{\mathcal{N}}$, we aim to solve the problem

$$\begin{cases} -\mathrm{div}(\boldsymbol{A}\nabla u) + Bu = f & \text{in } \Omega, \\ u = g_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \boldsymbol{A}\nabla u \cdot \boldsymbol{\nu} = g_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}, \end{cases} \tag{1.1}$$

where we assume $\boldsymbol{A} \in [L^\infty(\Omega)]^{d\times d}$ is a.e. symmetric and uniformly positive definite: there exist constants $\lambda, \Lambda$ such that

$$0 < \lambda \le \lambda_{\min}(\boldsymbol{A}(x)) \le \lambda_{\max}(\boldsymbol{A}(x)) \le \Lambda, \quad \text{for a.e. } x \in \Omega,$$

where $\lambda_{\min}(\boldsymbol{A}(\cdot))$ (resp. $\lambda_{\max}(\boldsymbol{A}(\cdot))$) denotes the minimum (resp. maximum) eigenvalue of $\boldsymbol{A}(\cdot)$.

We assume the linear operator $B \colon H^1(\Omega) \to L^2(\Omega)$ in (1.1) satisfies

$$\|Bv\|_{L^2(\Omega)} \le C\|\nabla v\|_{L^2(\Omega)} \quad \forall v \in H^1(\Omega) \text{ such that } v = 0 \text{ on } \Gamma_{\mathcal{D}}.$$

Examples satisfying this condition include $Bv = \mathrm{div}(\boldsymbol{\beta} v)$, with $\boldsymbol{\beta} \in [W^{1,\infty}(\Omega)]^d$, and $Bv = \boldsymbol{\beta} \cdot \nabla v + \gamma v$ for some $\boldsymbol{\beta} \in [L^\infty(\Omega)]^d$, $\gamma \in L^\infty(\Omega)$. We thus remark that (1.1) can accommodate, for example, stationary convection-reaction-diffusion problems. Following [16], we require problem (1.1) to be invertible in $H^1(\Omega)$, namely, that for every $f \in H^{-1}(\Omega)$ there exists a weak solution $u \in H^1(\Omega)$ with $u = 0$ on $\Gamma_{\mathcal{D}}$.

We introduce the flux variable $\boldsymbol{\phi} = \boldsymbol{A}\nabla u$ and rewrite (1.1) as a first-order system:

$$\begin{cases} \boldsymbol{\phi} - \boldsymbol{A}\nabla u = 0 & \text{in } \Omega, \\ -\operatorname{div} \boldsymbol{\phi} + Bu - f = 0 & \text{in } \Omega, \\ u = g_{\mathcal{D}} & \text{on } \Gamma_{\mathcal{D}}, \\ \boldsymbol{\phi} \cdot \boldsymbol{\nu} = g_{\mathcal{N}} & \text{on } \Gamma_{\mathcal{N}}. \end{cases} \tag{1.2}$$

Our approach is based on seeking minimizers of the loss function

$$\mathcal{L}(u, \boldsymbol{\phi}) := \|\boldsymbol{\phi} - \boldsymbol{A}\nabla u\|^2_{L^2(\Omega)} + \|\operatorname{div} \boldsymbol{\phi} - Bu + f\|^2_{L^2(\Omega)} \tag{1.3}$$

on a suitable set of admissible functions

$$\mathcal{A} := \{\boldsymbol{q} = (u, \boldsymbol{\phi}) \in H^1(\Omega) \times H(\mathrm{div}; \Omega) \colon u = g_{\mathcal{D}} \text{ on } \Gamma_{\mathcal{D}},\ \boldsymbol{\phi} \cdot \boldsymbol{\nu} = g_{\mathcal{N}} \text{ on } \Gamma_{\mathcal{N}}\}.$$

Clearly, if (1.1) has a unique solution $u \in H^1(\Omega)$, then the unique minimizer of $\mathcal{L}$ in $\mathcal{A}$ is $\boldsymbol{q} := (u, \boldsymbol{A}\nabla u)$. Our goal is to compute approximations to such a minimizer within a suitable finite dimensional space $\mathcal{A}_m \subset \mathcal{A}$. Particularly, in our method we consider a space $\mathcal{A}_m$ composed of neural networks with a fixed architecture and parameters $\boldsymbol{\Theta} \in \mathbb{R}^m$. Some efforts in this direction include the deep FOSLS method from [17] and the deep mixed residual method proposed in [18]. Reference [17] proposes the use of a partition of $\Omega$ and a mid-point quadrature rule for the evaluation of the discrete loss functional; instead, our algorithm is meshfree and uses random quadrature points. In more recent work by three of the authors of that work [12], the use of Monte Carlo integration is discussed albeit not pursued in detail. Such an approach yields a significant advantage in high-dimensional problems. Our method can be understood in the setting of the mixed residual methods in [18]. However, a significant difference between our work and [18] is that here we propose a strong imposition of the boundary conditions instead of the inclusion of penalization terms in the loss functional. We pre-train neural networks to accommodate boundary data, which results in a reduction in the number of iterations required in the solution of the PDE [19, 20].

The error in the approximation of continuous functionals with their discrete counterparts is usually not taken into account in numerical methods based on neural networks available in the literature; some recent efforts in this direction include [21, 22], where convergence rates are proved for a certain class of elliptic functionals under strong regularity conditions on the solution of the continuous problem. In other words, the focus is generally on the convergence of the minimizers of functionals such as $\mathcal{L}$ in (1.3) over certain neural network spaces towards the minimizer of the same functional at the continuous level. However, in practice one does not compute $\mathcal{L}$ exactly but rather approximates it by means of quadrature rules. Let us call $\mathcal{L}_N$ such an approximation to the functional $\mathcal{L}$, where $N$ is, for example, the number of quadrature points. The computation of $\mathcal{L}_N$ instead of $\mathcal{L}$ can introduce important changes in the nature of the minimization problem, such as the loss of convexity of the associated functional [5]. A major contribution of this work is to present a convergence analysis that considers the discretization of the functional $\mathcal{L}$. Specifically, we prove the almost-sure $\Gamma$-convergence of the discrete loss functions towards the continuous one. As stated in Theorem 3.3, this implies the almost-sure convergence of the solutions computed numerically to the solution of the continuous problem.

The techniques we develop for this purpose are not only valid for the method we propose, and we generalize and apply them to the convergence analysis of a broad class of methods, including the Deep Ritz [5] and the Deep Galerkin [6] Methods (DRM and DGM, respectively; see Remarks 1 and 2).

**Organization of the paper.** The rest of the paper is organized as follows. Section 2 describes the method we propose for dealing with (1.2), including the treatment of Dirichlet and Neumann boundary conditions in strong form, and discusses some aspects pertaining to its implementation. We perform a convergence analysis for our method in Section 3. This analysis takes into account the approximation of the loss functional by means of Monte Carlo integration, and establishes the convergence of the discrete minimization problem towards the continuous one in the sense of almost sure $\Gamma$-convergence. Section 4 generalizes the analysis to include some other well-known methods, thereby establishing their convergence as well. We illustrate the performance of our method through computational examples in Section 5, and provide some concluding remarks in Section 6.

## 2. Description of the method

The goal of the method we propose is to approximate the unique minimizer $(u, \boldsymbol{\phi})$ of the functional in (1.3). A natural first approach would consist in seeking a set of parameters $\boldsymbol{\Theta}_0 \in \mathbb{R}^m$ such that

$$\mathcal{L}(u_{\boldsymbol{\Theta}_0}, \boldsymbol{\phi}_{\boldsymbol{\Theta}_0}) = \min_{\boldsymbol{\Theta} \in \mathbb{R}^m} \mathcal{L}(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}),$$

with the functions $(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}})$ belonging to a suitable neural network space. The use of neural networks in this setting has the advantage that one can easily implement meshfree methods by randomly sampling collocation points (see [6, 23, 24, 25], for example), and thereby be able to deal with high-dimensional problems, where most classical numerical PDE methods become unfeasible.

The enforcement of boundary conditions is a non-trivial aspect to take into account in this approach. A typical way to tackle this issue is to incorporate boundary conditions by adding a penalization term [5, 6, 9]. However, in practice it is observed that enforcing discrete functions to satisfy the boundary conditions gives rise to a faster training process [19, 20]. We shall first create suitable auxiliary functions with the purpose of imposing the boundary conditions in a strong fashion. In this way, we ensure $(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) \in \mathcal{A}_m \subset \mathcal{A}$, for all $\boldsymbol{\Theta} \in \mathbb{R}^m$. Then, the optimization procedure consists of sampling $N$ points $\{x_k\}_{k=1}^N \subset \Omega$ uniformly, and approximating $\mathcal{L}(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) \approx \mathcal{L}_N(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}})$, at every step of a gradient descent algorithm, with $\mathcal{L}_N$ defined as

$$\mathcal{L}_N(u, \boldsymbol{\phi}) := \frac{|\Omega|}{N} \sum_{k=1}^{N} \big(\boldsymbol{\phi}(x_k) - \boldsymbol{A}\nabla u(x_k)\big)^2 + \big(\operatorname{div} \boldsymbol{\phi}(x_k) - Bu(x_k) + f(x_k)\big)^2. \tag{2.1}$$

We expose the details below.

### 2.1. **Strong imposition of boundary conditions.**

We follow the ideas from [19] about the imposition of Dirichlet boundary conditions, and extend the approach to include Neumann boundary conditions. Instead of trying to compute either $u$ or $\boldsymbol{\phi}$ directly and incorporate the boundary conditions by a penalization term, we shall enforce them in the construction of the neural network approximations. For that purpose, we make use of the following notion.

**Definition 2.1** (smooth distance function)**.** *Let $\Gamma_* \subset \overline{\Omega}$ be a closed set. We say that a Lipschitz continuous function $d_* \colon \Omega \to \mathbb{R}$ is a* smooth distance function *if it satisfies $d_* \geq 0$ and $d_*(x) = 0$ if and only if $x \in \Gamma_*$.*

We briefly comment on the use of smooth distance functions in the strong imposition of Dirichlet and Neumann boundary conditions. In the computation of $u$ in (1.3), we restrict the class of functions to be

$$u(x) := G_{\mathcal{D}}(x) + d_{\mathcal{D}}(x)\, v(x), \tag{2.2}$$

where the unknown is the function $v \colon \Omega \to \mathbb{R}$, $G_{\mathcal{D}}$ is a lifting of the Dirichlet datum, and $d_{\mathcal{D}}$ is a smooth distance function to $\Gamma_{\mathcal{D}}$.

In a similar fashion, we can incorporate normal boundary conditions on the flux variable $\boldsymbol{\phi}$ in a strong way. We first construct a vector field $\boldsymbol{n} \colon \Omega \to \mathbb{R}^d$ such that $\boldsymbol{n}|_{\Gamma_{\mathcal{N}}} = \boldsymbol{\nu}$ and $|\boldsymbol{n}(x)| = 1$ for a.e. $x \in \Omega$, and consider

$$\boldsymbol{\phi}(x) := \boldsymbol{\psi}(x) + \left( G_{\mathcal{N}}(x) - \frac{\boldsymbol{\psi}(x) \cdot \boldsymbol{n}(x)}{1 + d_{\mathcal{N}}(x)} \right) \boldsymbol{n}(x). \tag{2.3}$$

Above, $G_{\mathcal{N}}$ is a lifting of the Neumann boundary condition, $d_{\mathcal{N}}$ is a smooth distance function to $\Gamma_{\mathcal{N}}$, and the unknown is the function $\boldsymbol{\psi} \colon \Omega \to \mathbb{R}^d$. By its definition, the function $\boldsymbol{\phi}$ satisfies the boundary condition $\boldsymbol{\phi} \cdot \boldsymbol{\nu} = g_{\mathcal{N}}$ at $\Gamma_{\mathcal{N}}$. We remark that we do not require any smoothness on $\boldsymbol{n}$: in particular this field may be discontinuous at some points in the domain.

Therefore, in the construction of approximate solutions we shall first compute the vector field $\boldsymbol{n}$ and the scalar functions $d_{\mathcal{D}}$, $d_{\mathcal{N}}$, $G_{\mathcal{D}}$, $G_{\mathcal{N}}$. Then, we seek $\boldsymbol{y} = (v, \boldsymbol{\psi})$ such that the corresponding pair $(u, \boldsymbol{\phi})$, given by (2.2) and (2.3), minimizes the loss function $\mathcal{L}$. The computation of the auxiliary functions $\boldsymbol{n}$, $d_{\mathcal{D}}$, $d_{\mathcal{N}}$, $G_{\mathcal{D}}$, $G_{\mathcal{N}}$ typically requires fewer degrees of freedom and iterations than the computation of $(v, \boldsymbol{\psi})$, depending on the complexity of the domain or the boundary data. Consequently, we shall frequently use a simpler architecture to represent them. Below, we give details on the computation of the auxiliary functions.

2.1.1. *Computation of smooth distance functions.* Loosely, for $* \in \{\mathcal{D}, \mathcal{N}\}$, a smooth distance function to $\Gamma_*$ is a function $d_* \colon \Omega \to [0,\infty)$ that approximates the distance to $\Gamma_*$, cf. Definition 2.1. To construct such functions, we first randomly choose $N_d$ points $\{x_i\}_{i=1}^{N_d} \subset \Omega$ (the same set of points can be used for either $* = \mathcal{D}$ and $* = \mathcal{N}$) and compute

$$d^{(*)}(x_i) \approx \operatorname{dist}(x_i, \Gamma_*).$$

This can be done by choosing points on $\Gamma_*$ and using efficient nearest-neighbor search strategies. Once we have computed the quantities $\{d^{(*)}(x_i)\}_{i=1}^{N_d}$, we train a neural network for $d_*$ by using the cost function

$$\mathcal{L}_*(d) = \frac{1}{N_d} \sum_{i=1}^{N} |d(x_i) - d^{(*)}(x_i)|^2 + \frac{1}{N_{d,*}} \sum_{i=1}^{N_{d,*}} |d(x_{*,i})|^2,$$

where $\{x_{*,i}\}_{i=1}^{N_{d,*}}$ is a random batch of points on $\Gamma_*$.

In the setting of $d_{\mathcal{D}}$ and $d_{\mathcal{N}}$, we use neural networks with a single hidden layer and significantly less parameters than the networks employed in the PDE resolution.

2.1.2. *Boundary data liftings and normal field.* We approximate liftings of the boundary data to $\Omega$ by *smooth liftings* [19]: in either (2.2) and (2.3), we require $G_{\mathcal{D}}$ and $G_{\mathcal{N}}$ to coincide with $g_{\mathcal{D}}$ on $\Gamma_{\mathcal{D}}$ and with $g_{\mathcal{N}}$ on $\Gamma_{\mathcal{N}}$, respectively, and to be smooth enough so that we can apply the differential operator to them pointwise. A natural way to enforce the former is to set the $L^2$-norms of the discrepancies on the corresponding boundary subsets as loss functions, namely

$$\mathcal{L}_{\mathcal{D}}(G) = \|G - g_{\mathcal{D}}\|^2_{L^2(\Gamma_{\mathcal{D}})}, \quad \mathcal{L}_{\mathcal{N}}(G) = \|G - g_{\mathcal{N}}\|^2_{L^2(\Gamma_{\mathcal{N}})}.$$

In practice, we consider sets of boundary nodes $\{z_i^{\mathcal{D}}\}_{i=1}^{M_{\mathcal{D}}} \subset \Gamma_{\mathcal{D}}$, $\{z_i^{\mathcal{N}}\}_{i=1}^{M_{\mathcal{N}}} \subset \Gamma_{\mathcal{N}}$ and define the quadratic cost functionals

$$\mathcal{L}_{\mathcal{D}}(G) = \frac{1}{M_{\mathcal{D}}} \sum_{i=1}^{M_{\mathcal{D}}} |G(z_i^{\mathcal{D}}) - g_{\mathcal{D}}(z_i^{\mathcal{D}})|^2, \qquad \mathcal{L}_{\mathcal{N}}(G) = \frac{1}{M_{\mathcal{N}}} \sum_{i=1}^{M_{\mathcal{N}}} |G(z_i^{\mathcal{N}}) - g_{\mathcal{N}}(z_i^{\mathcal{N}})|^2.$$

In the same fashion as for the smooth distance functions, we consider neural networks with a single hidden layer to compute the functions $d_{\mathcal{D}}$ and $d_{\mathcal{N}}$.

Analogously, for the computation of the vector field $\boldsymbol{n}$ we start from the loss function

$$\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{m}) = \|\boldsymbol{m} - \boldsymbol{\nu}\|^2_{L^2(\Gamma_{\mathcal{N}})} + \||\boldsymbol{m}|^2 - 1\|^2_{L^2(\Omega)},$$

consider a set of randomly selected points $\{z_i^{\mathcal{N},\boldsymbol{n}}\}_{i=1}^{M_{\mathcal{N},\boldsymbol{n}}} \subset \Gamma_{\mathcal{N}}$ and $\{z_i^{\boldsymbol{n}}\}_{i=1}^{M_{\boldsymbol{n}}} \subset \Omega$, and minimize the cost functional

$$\mathcal{L}_{\boldsymbol{n}}(\boldsymbol{m}) = \frac{1}{M_{\mathcal{N},\boldsymbol{n}}} \sum_{i=1}^{M_{\mathcal{N},\boldsymbol{n}}} |\boldsymbol{m}(z_i^{\mathcal{N},\boldsymbol{n}}) - \boldsymbol{\nu}|^2 + \frac{1}{M_{\boldsymbol{n}}} \sum_{i=1}^{M_{\boldsymbol{n}}} ||\boldsymbol{m}(z_i^{\boldsymbol{n}})|^2 - 1|^2.$$

We point out that, in practice, the set of auxiliary points $\{z_i^{\mathcal{N},\boldsymbol{n}}\}_{i=1}^{M_{\mathcal{N},\boldsymbol{n}}}$ can be the same as the set $\{z_i^{\mathcal{N}}\}_{i=1}^{M_{\mathcal{N}}}$ used in the approximation of $\mathcal{L}_{\mathcal{N}}$.

2.2. **Computational aspects.** Once we have built the auxiliary functions, we proceed to compute $u$ and $\boldsymbol{\phi}$. For this purpose, we consider a set of random points $\{x_k\}_{k=1}^N \subset \Omega$, and seek to minimize the cost functional

$$\mathcal{L}_N(u, \boldsymbol{\phi}) := \frac{|\Omega|}{N} \sum_{k=1}^{N} \big(\boldsymbol{\phi}(x_k) - \boldsymbol{A}\nabla u(x_k)\big)^2 + \big(\operatorname{div} \boldsymbol{\phi}(x_k) - Bu(x_k) + f(x_k)\big)^2. \tag{2.4}$$

From the construction of $u$ and $\boldsymbol{\phi}$ (see (2.2) and (2.3)), the trainable parameters $\boldsymbol{\Theta}$ arise in the computation of the auxiliary functions $v$ and $\boldsymbol{\psi}$.

In broad terms, the method we propose can be summarized as follows:

- **Stage 1:** Train auxiliary functions $d_{\mathcal{D}}$, $d_{\mathcal{N}}$, $G_{\mathcal{D}}$, $G_{\mathcal{N}}$, and $\boldsymbol{n}$.
- **Stage 2:** Until some stop criterion is reached, do:
  - Select random points $\{x_k\}_{k=1}^N \subset \Omega$.
  - For some learning rate $\ell$, do:
  $$\boldsymbol{\Theta} = \boldsymbol{\Theta} - \ell \nabla_{\boldsymbol{\Theta}} \mathcal{L}_N(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}).$$
  - Update learning rate.

The computation of $\mathcal{L}_N(u, \boldsymbol{\phi})$ requires computing the derivatives of $u$ and $\boldsymbol{\phi}$ with respect to the input variables, evaluated at $\{x_k\}_{k=1}^N$. Since we constructed our auxiliary functions as neural networks, it is possible to compute efficiently these derivatives by means of the Back-Propagation algorithm. Packages like TensorFlow allow this kind of computation.

Additionally, our least-squares loss function (2.4) only involves first-order derivatives in space. We discretize such derivatives by using finite-difference quotients. Namely, for any function $\boldsymbol{\varphi} \colon \mathbb{R}^d \to \mathbb{R}^n$ we let $h > 0$ be a fixed constant and consider the second-order (with respect to $h$) formula

$$\partial_i \boldsymbol{\varphi}(x_k) \simeq \frac{\boldsymbol{\varphi}(x_k + h\boldsymbol{e}_i) - \boldsymbol{\varphi}(x_k - h\boldsymbol{e}_i)}{2h},$$

where $\boldsymbol{e}_i \in \mathbb{R}^d$ is the $i$-th canonical basis vector in $\mathbb{R}^d$. We employ this formula for the approximation of $\nabla u$, $\operatorname{div} \boldsymbol{\phi}$ and the first-order derivatives involved in $B$.

For the numerical examples we implemented our algorithm by using PyTorch and discretizing the differential operators by means of finite differences. We typically use about 10,000 steps of gradient descent, sampling between 1,000 and 5,000 random points in $\Omega$ at each step. A step-type decrease in the learning rate showed good results in practice. In particular, we start from a learning rate $\ell = 10^{-2}$, which we halve every 1,000-2,500 gradient descent steps. No particular type of architecture was chosen for the functions involved. We use three-layer neural networks with linear activation function (ReLU) for the auxiliary functions, and five-layer networks for the main variables $v$ and $\boldsymbol{\psi}$. The ADAM [26] optimization algorithm showed good results in numerical experiments. Further details about the implementation of the method can be found in Section 5.

Regarding the training of auxiliary functions $d_{\mathcal{D}}$ and $d_{\mathcal{N}}$, the following procedure showed good results in practice:

- Select $N_d$ random points $\{x_k\}_{k=1}^{N_d} \subset \Omega$.
- Initialize a vector $\boldsymbol{D}$ as $\boldsymbol{D}_i = \infty$ for $i = 1, ..., N_d$.
- Until some stop criterion is reached, do:
  - Select $M_*$ random points $\{z_i^*\}_{i=1}^{M_*} \subset \Gamma_*$.
  - Update $\boldsymbol{D}$ as: $\boldsymbol{D}_k = \min\{\min_{i=1,\dots M_*} |x_k - z_i^*|, \boldsymbol{D}_k\}$

- Define the loss function:

$$\mathcal{L}_*(d_*) = \frac{1}{N_d}\sum_{k=1}^{N_d} |d_*(x_k) - \boldsymbol{D}_k|^2 + \frac{1}{M_*}\sum_{i=1}^{M_*} |d(z_i^*)|^2.$$

- For some learning rate $\ell$, do:

$$\boldsymbol{\Theta}_{d_*} = \boldsymbol{\Theta}_{d_*} - \ell\nabla_{\boldsymbol{\Theta}_{d_*}}\mathcal{L}_*(d_*).$$

- Update learning rate.

Here $* \in \{D, N\}$, and $\boldsymbol{\Theta}_{d_*}$ denotes the trainable parameters of $d_*$.

## 3. Analysis of the method

In this section, we prove the convergence of our method by using two main ingredients. First, we put the discretization in a $\Gamma$-convergence framework. More precisely, the sequence of functionals we consider is related to the use of meshfree methods in the computation of a regularized version of the discrete loss functional $\mathbb{R}^m \ni \boldsymbol{\Theta} \mapsto \mathcal{L}(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}})$; see Theorem 3.2 below. Second, we exploit the coercivity of the least-squares functional and approximation properties of neural networks to conclude that the sequence of minimizers of the regularized discrete loss functionals converges to the solution of (1.1) as the number of neural network parameters $m \to \infty$.

For the sake of simplicity, we consider problem (1.2) with $g_{\mathcal{D}} = g_{\mathcal{N}} = 0$. Otherwise, one could consider $G_{\mathcal{D}}$ and $G_{\mathcal{N}}$ such that $G_{\mathcal{D}} = g_{\mathcal{D}}$ on $\Gamma_{\mathcal{D}}$ and $G_{\mathcal{N}} = g_{\mathcal{N}}$ on $\Gamma_{\mathcal{N}}$, a smooth normal field $\boldsymbol{n}$ such that $\boldsymbol{n} = \boldsymbol{\nu}$ on $\Gamma_{\mathcal{N}}$, and then the auxiliary functions $u_0 = u - G_{\mathcal{D}}$ and $\boldsymbol{\phi}_0 = \boldsymbol{\phi} - G_{\mathcal{N}}\boldsymbol{n}$ would solve the first-order system

$$\begin{cases} \boldsymbol{\phi}_0 - \boldsymbol{A}\nabla u_0 = & \boldsymbol{A}\nabla G_{\mathcal{D}} - G_{\mathcal{N}}\boldsymbol{n} & \text{in } \Omega, \\ -\operatorname{div}(\boldsymbol{\phi}_0) + Bu_0 = & f + \operatorname{div}(G_{\mathcal{N}}\boldsymbol{n}) - BG_{\mathcal{D}} & \text{in } \Omega, \\ u_0 = & 0 & \text{on } \Gamma_{\mathcal{D}}, \\ \boldsymbol{\phi}_0 \cdot \boldsymbol{\nu} = & 0 & \text{on } \Gamma_{\mathcal{N}}. \end{cases}$$

Naturally, the solution to this system corresponds to the minimum of the least-squares functional

$$(u, \boldsymbol{\phi}) \mapsto \|\boldsymbol{\phi} - \boldsymbol{A}\nabla u + \widetilde{g}\|^2_{L^2(\Omega)} + \|\operatorname{div}(\boldsymbol{\phi}) - Bu + \widetilde{f}\|^2_{L^2(\Omega)},$$

with $\widetilde{g} = -\boldsymbol{A}\nabla G_{\mathcal{D}} + G_{\mathcal{N}}\boldsymbol{n}$ and $\widetilde{f} = f + \operatorname{div}(G_{\mathcal{N}}\boldsymbol{n}) - BG_{\mathcal{D}}$. This functional can be dealt with by using the same tools as for (1.3), the only difference being the presence of the zero-order correction term $\widetilde{g}$ in the first $L^2$-norm.

In the following proof of convergence, we restrict ourselves to one hidden layer neural networks with $n$ neurons. We define the set of discrete functions

$$\mathcal{C}_m := \Big\{(v_{\boldsymbol{\Theta}}, \boldsymbol{\psi}_{\boldsymbol{\Theta}}) : v_{\boldsymbol{\Theta}} = B_v\sigma(A_v x + c_v), \boldsymbol{\psi}_{\boldsymbol{\Theta}} = B_{\boldsymbol{\psi}}\sigma(A_{\boldsymbol{\psi}} x + c_{\boldsymbol{\psi}})\Big\},$$

with $A_v, A_{\boldsymbol{\psi}} \in \mathbb{R}^{n\times d}$, $c_v, c_{\boldsymbol{\psi}} \in \mathbb{R}^{n\times 1}$, $B_v \in \mathbb{R}^{1\times n}$, $B_{\boldsymbol{\psi}} \in \mathbb{R}^{d\times n}$, and $\sigma\colon \mathbb{R}^n \to \mathbb{R}^n$, and $\sigma$ a smooth and bounded non-constant activation function, applied elementwise. We collect all the parameters in $\boldsymbol{\Theta} \in \mathbb{R}^m$ with $m = 3n(d+1)$. We remark that, whenever we state that $m \to \infty$, we mean that the number of neurons $n$ is growing to infinity.

Assuming that we are able to construct smooth auxiliary functions $d_{\mathcal{D}}$, $d_{\mathcal{N}}$ and $\boldsymbol{n}$ as in Section 2.1, we define the set of discrete admissible functions

$$\mathcal{A}_m := \Big\{\boldsymbol{q}_{\boldsymbol{\Theta}} = (u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) : u_{\boldsymbol{\Theta}} = d_{\mathcal{D}}v_{\boldsymbol{\Theta}} \text{ and } \boldsymbol{\phi}_{\boldsymbol{\Theta}} = \boldsymbol{\psi}_{\boldsymbol{\Theta}} - \Big(\frac{\boldsymbol{\psi}_{\boldsymbol{\Theta}}\cdot\boldsymbol{n}}{1+d_{\mathcal{N}}}\Big)\boldsymbol{n},\ (v_{\boldsymbol{\Theta}}, \boldsymbol{\psi}_{\boldsymbol{\Theta}}) \in \mathcal{C}_m\Big\}. \tag{3.1}$$

We remark that the fulfillment of the boundary conditions is guaranteed within the set $\mathcal{A}_m$, in the sense that $u_{\boldsymbol{\Theta}} = 0$ if $d_{\mathcal{D}} = 0$ and $\boldsymbol{\phi}_{\boldsymbol{\Theta}} \cdot \boldsymbol{n} = 0$ if $d_{\mathcal{N}} = 0$.

3.1. **Approximation properties of neural networks.** Let $\boldsymbol{q}_0 = (u_0, \boldsymbol{\phi}_0) \in \mathcal{A}$ be the unique minimizer of (1.3). We shall make the assumption that $\boldsymbol{q}_0$ can be approximated by the neural network spaces. Namely, let us assume that

$$d(\boldsymbol{q}_0, \mathcal{A}_m) := \inf_{\boldsymbol{q}_{\boldsymbol{\Theta}} \in \mathcal{A}_m} \|\boldsymbol{q}_0 - \boldsymbol{q}_{\boldsymbol{\Theta}}\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} \to 0 \quad \text{as } m \to \infty. \tag{3.2}$$

We briefly comment on this hypothesis. In first place, there are several by now classical results [27, 28, 29] regarding the approximation properties of neural networks, although without the incorporation of boundary conditions. We additionally point out to [30, 31] for recent results regarding approximation capabilities of ReLU neural networks, including approximation rates. For deep ReLU neural networks (with at most $\lceil \log_2(d+1) \rceil$ hidden layers), references [31, 32] establish the capability of networks to represent simplicial linear finite element functions, which possess good approximation properties in the $H^1$-norm. Therefore, if we use a nonconstant activation function $\sigma$, then we expect $d(\boldsymbol{q}, \mathcal{C}_m) \to 0$ when $m \to \infty$ for any $\boldsymbol{q} \in H^1(\Omega) \times H(\mathrm{div};\Omega)$.

Condition (3.2) further assumes that the solution $q_0$ can be approximated through the admissible classes $\mathcal{A}_m$ that incorporate boundary conditions. This hypothesis holds, for example, if one assumes certain regularity of solutions to (1.2). For instance, if $u_0 \in C^1(\overline{\Omega})$, then it satisfies (recall $g_{\mathcal{D}} = 0$)

$$\left| \lim_{t \to 0^+} \frac{u_0(z - t\boldsymbol{\nu})}{t} \right| = \left| \frac{\partial u_0}{\partial \boldsymbol{\nu}}(z) \right| < \infty, \quad z \in \Gamma_{\mathcal{D}}.$$

If we write $x = z - t\boldsymbol{\nu}$, then $t \approx \mathrm{dist}(x, \Gamma_{\mathcal{D}}) \approx d_{\mathcal{D}}(x)$ and the finiteness of the limit above essentially means that $u_0/d_{\mathcal{D}}$ is a bounded function. Additionally, if we can construct auxiliary functions $d_{\mathcal{D}}$, $d_{\mathcal{N}}$, and $\boldsymbol{n}$ in such a way that

$$\frac{u_0}{d_{\mathcal{D}}} \in H^1(\Omega), \text{ and } \frac{(\boldsymbol{\phi}_0 \cdot \boldsymbol{n})\boldsymbol{n}}{d_{\mathcal{N}}} \in H(\mathrm{div};\Omega). \tag{3.3}$$

then there exists a sequence $\{(v_m, \boldsymbol{\psi}_m)\}_{m \in \mathbb{N}}$ with $(v_m, \boldsymbol{\psi}_m) \in \mathcal{C}_m$ for all $m$, such that

$$\left\| v_m - \frac{u_0}{d_{\mathcal{D}}} \right\|_{H^1(\Omega)} \to 0 \quad \text{and} \quad \left\| \boldsymbol{\psi}_m - \sum_{i=1}^{d-1} (\boldsymbol{\phi}_0 \cdot \boldsymbol{t})\boldsymbol{t} - \frac{1 + d_{\mathcal{N}}}{d_{\mathcal{N}}} (\boldsymbol{\phi}_0 \cdot \boldsymbol{n})\boldsymbol{n} \right\|_{H(\mathrm{div};\Omega)} \to 0$$

as $m \to \infty$. Defining the sequence $\{(u_m, \boldsymbol{\phi}_m)\}_{m \in \mathbb{N}}$ as $u_m = d_{\mathcal{D}} v_m$ and $\boldsymbol{\phi}_m = \boldsymbol{\psi}_m - \left(\frac{\boldsymbol{\psi}_m \cdot \boldsymbol{n}}{1 + d_{\mathcal{N}}}\right)\boldsymbol{n}$, we would have $(u_m, \boldsymbol{\phi}_m) \in \mathcal{A}_m$ for all $m$, and $(u_m, \boldsymbol{\phi}_m) \to (u_0, \boldsymbol{\phi}_0)$ in $\| \cdot \|_{H^1(\Omega)\times H(\mathrm{div};\Omega)}$ and therefore (3.2) would hold. Clearly, (3.3) is a regularity assumption on the solution of (1.2), and in turn it translates into its approximability by neural networks.

3.2. **Γ-convergence.** We aim to prove the convergence of the neural network approximations computed by our method towards minimizers of the least-squares functional $\mathcal{L}$ in (1.3). For this purpose, we shall make use of Γ-convergence theory, that provides a framework for the convergence of functionals. In particular, if one has proven the Γ-convergence of a sequence of functionals and has a converging sequence of minimizers, then one can guarantee the existence of solutions to the limit problem, as well as the convergence of either minimum values and minimizers. We next briefly review the definition and some basic results pertaining to Γ-convergence and refer to [33] for further details.

**Definition 3.1** (sequential Γ-convergence). *Let $X$ be a metric space and let $F_n$, $F : X \to \overline{\mathbb{R}}$, where $\overline{\mathbb{R}} := [-\infty, +\infty]$. We say that $F_n$ Γ-converges to $F$ (and write $F_n \xrightarrow{\Gamma} F$) if, for every $x \in X$ we have*

- (lim-inf inequality) *for every sequence $\{x_n\}_{n\in\mathbb{N}} \subset X$ converging to $x$,*
$$F(x) \leq \liminf_{n\to\infty} F_n(x_n);$$
- (lim-sup inequality) *there exists a sequence $\{x_n\}_{n\in\mathbb{N}}$ converging to $x$ such that*
$$F(x) \geq \limsup_{n\to\infty} F_n(x_n).$$

**Definition 3.2** (equi-coercivity). *Let $\{F_n\}_{n\in\mathbb{N}}$ be a sequence of functions $F_n : X \to \overline{\mathbb{R}}$. We say that $\{F_n\}$ is equi-coercive if for all $t \in \mathbb{R}$ there exists a compact set $K_t \subset X$ such that $\{F_n \leq t\} \subset K_t$.*

**Theorem 3.1** (fundamental theorem of Γ-convergence). *Let $(X, d)$ be a metric space, $\{F_n\}_{n\in\mathbb{N}}$ be an equi-coercive sequence of functions on $X$, and $F$ be such that $F_n \xrightarrow{\Gamma} F$. Then,*
$$\exists \min_X F = \lim_{n\to\infty} \inf_X F_n.$$
*Moreover, if $\{x_n\}_{n\in\mathbb{N}}$ is a precompact sequence in $X$ such that $\lim_{n\to\infty} F_n(x_n) = \lim_{n\to\infty} \inf_X F_n$, then every limit of a subsequence of $\{x_n\}$ is a minimum point for $F$.*

We emphasize that this result guarantees that the equi-coercivity of a family of functionals combined with their Γ-convergence yields the convergence of the minimizers towards the minimizers of the Γ-limit.

### 3.3. Convergence of the method.

We split the proof of convergence of our method into several steps. We start by proving the following auxiliary lemma, that shows the continuity of the neural network functions with respect to the parameters.

**Lemma 3.1** (continuity with respect to neural network parameters). *The map*
$$\boldsymbol{\Theta} \mapsto \boldsymbol{q}_{\boldsymbol{\Theta}} = (u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) \in (\mathcal{A}_m, \|\cdot\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)})$$
*is continuous. Moreover, defining the functions $G_1, G_2 : \mathbb{R}^m \times \Omega \to \mathbb{R}$,*
$$G_1(\boldsymbol{\Theta}, x) := |\boldsymbol{\phi}_{\boldsymbol{\Theta}}(x) - \boldsymbol{A}\nabla u_{\boldsymbol{\Theta}}(x)|^2, \quad G_2(\boldsymbol{\Theta}, x) := |\operatorname{div} \boldsymbol{\phi}_{\boldsymbol{\Theta}}(x) - B u_{\boldsymbol{\Theta}}(x) + f(x)|^2, \tag{3.4}$$
*for any $R > 0$ we have $G_1 \in L^\infty(B(0, R) \times \Omega)$ and, assuming $f \in L^2(\Omega)$, there exists a function $s \in L^1(B(0, R) \times \Omega))$, depending on $R$, such that $|G_2(\boldsymbol{\Theta}, x)| \leq s(\boldsymbol{\Theta}, x)$ for all $(\boldsymbol{\Theta}, x) \in B(0, R) \times \Omega$.*

*Proof.* Let us first focus on a generic neural network $v_{\boldsymbol{\Theta}} : \mathbb{R}^d \to \mathbb{R}$ with one hidden layer,
$$v_{\boldsymbol{\Theta}}(x) = B\sigma(Ax + c).$$
Above, we assume $\sigma$ is a Lipschitz continuous activation function, and the parameters $B \in \mathbb{R}^{1\times n}$, $A \in \mathbb{R}^{n\times d}$ and $c \in \mathbb{R}^{n\times 1}$ are collected in $\boldsymbol{\Theta} \in \mathbb{R}^m$, $m = n(d + 2)$. Using the fact that $v_{\boldsymbol{\Theta}}$ and its derivatives depend continuously on the parameters, one can verify easily that the map $\mathbb{R}^m \mapsto W^{1,\infty}(\Omega)$ such that $\boldsymbol{\Theta} \mapsto v_{\boldsymbol{\Theta}}$ is continuous. Moreover, the function $G : \mathbb{R}^m \times \Omega \to \mathbb{R}$, defined as $G(\boldsymbol{\Theta}, x) := v_{\boldsymbol{\Theta}}(x)$ is Lipschitz continuous, and therefore it is bounded on $B(0, R) \times \Omega$ and its (weak) derivatives are essentially bounded on the same set as well. Furthermore, if $f \in L^2(\Omega)$ then $|G(\boldsymbol{\Theta}, x) + f(x)|^2 \leq 2|G(\boldsymbol{\Theta}, x)|^2 + 2|f(x)|^2 \leq 2M + 2|f(x)|^2 =: s(\boldsymbol{\Theta}, x)$, with $s \in L^1(B(0, R) \times \Omega)$.

For arbitrary neural network functions $(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}})$ in the space $\mathcal{A}_m$, defined by (3.1), we exploit the idea above together with the fact that the auxiliary functions $d_{\mathcal{D}}, d_{\mathcal{N}}$ and $\boldsymbol{n}$ are smooth to conclude the desired result. □

The following lemma guarantees that, for the loss function $\mathcal{L}$ defined in (1.3), minimizers over $\mathcal{A}_m$ converge towards the minimizer $\boldsymbol{q}_0 \in \mathcal{A}$ as $m \to \infty$.

**Lemma 3.2** (approximation properties of $\mathcal{A}_m$)**.** *For every $m \in \mathbb{N}$ there exists $\boldsymbol{q}_m \in \mathcal{A}_m$ such that $\mathcal{L}(\boldsymbol{q}_m) \leq \mathcal{L}(\boldsymbol{q}_{\boldsymbol{\Theta}})$ for all $\boldsymbol{q}_{\boldsymbol{\Theta}} \in \mathcal{A}_m$. Moreover, if $\boldsymbol{q}_0$ is the unique minimizer of $\mathcal{L}$ in $\mathcal{A}$, defining the sequence $\{\boldsymbol{q}_m\}_{m\in\mathbb{N}}$ with $\boldsymbol{q}_m \in \mathcal{A}_m$ being a minimizer of $\mathcal{L}$ in $\mathcal{A}_m$, we have $\|\boldsymbol{q}_m - \boldsymbol{q}_0\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} \to 0$ with $m \to \infty$.*

*Proof.* From [16], we know that $\mathcal{L}$ is elliptic with respect to the $H^1(\Omega) \times H(\mathrm{div};\Omega)$ norm. Namely, there exist positive constants $\alpha$ and $\beta$ such that

$$\alpha\|(u,\boldsymbol{\phi})\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} \leq \|\boldsymbol{\phi} - \boldsymbol{A}\nabla u\|_0^2 + \|\operatorname{div}(\boldsymbol{\phi}) - Bu\|_0^2 \leq \beta\|(u,\boldsymbol{\phi})\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)}, \tag{3.5}$$

for all $(u, \boldsymbol{\phi}) \in H^1(\Omega) \times H(\mathrm{div};\Omega)$. Let $\{\boldsymbol{q}_{\boldsymbol{\Theta}_i}\}_{i\in\mathbb{N}} \subset \mathcal{A}_m$ be a minimizing sequence in $\mathcal{A}_m$, namely, $\mathcal{L}(\boldsymbol{q}_{\boldsymbol{\Theta}_i}) \to \inf_{\boldsymbol{q}\in\mathcal{A}_m} \mathcal{L}(\boldsymbol{q})$. It follows from (3.5) that $\{\boldsymbol{q}_{\boldsymbol{\Theta}_i}\}_{i\in\mathbb{N}}$ is bounded in $H^1(\Omega) \times H(\mathrm{div};\Omega)$. Lemma 3.1 implies that the map $\boldsymbol{\Theta} \mapsto (u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) \in (\mathcal{A}_m, \|\cdot\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)})$ is continuous and, because $\mathcal{A}_m$ is finite dimensional, we can extract a subsequence $\{\boldsymbol{q}_{\boldsymbol{\Theta}_j}\}_{j\in\mathbb{N}} \subset \{\boldsymbol{q}_{\boldsymbol{\Theta}_i}\}_{i\in\mathbb{N}}$ in such a way that $\boldsymbol{q}_j \to \boldsymbol{q}_m \in \mathcal{A}_m$, with $\boldsymbol{q}_m \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_m} \mathcal{L}(\boldsymbol{q})$.

Next, let $\varepsilon > 0$. For every $m \in \mathbb{N}$, we let $\boldsymbol{q}_m = (u_m, \boldsymbol{\phi}_m) \in \mathcal{A}_m$ be a minimizer of $\mathcal{L}$ in $\mathcal{A}_m$ and $\boldsymbol{q}_m^* = (u_m^*, \boldsymbol{\phi}_m^*) \in \mathcal{A}_m$ be such that $d(\boldsymbol{q}_0, \mathcal{A}_m) \geq \|\boldsymbol{q}_m^* - \boldsymbol{q}_0\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} - \varepsilon$. Then, using that the solution $\boldsymbol{q}_0 = (u_0, \boldsymbol{\phi}_0)$ of (1.2) satisfies the conditions $\boldsymbol{\phi}_0 = \boldsymbol{A}\nabla u_0$ and $-\operatorname{div}(\boldsymbol{\phi}_0) + Bu_0 = f$ a.e. in $\Omega$ and exploiting the upper bound in (3.5), we deduce

$$\begin{aligned} 0 \leq \mathcal{L}(\boldsymbol{q}_m) \leq \mathcal{L}(\boldsymbol{q}_m^*) &= \|\boldsymbol{\phi}_m^* - \boldsymbol{A}\nabla u_m^*\|_0^2 + \|\operatorname{div}(\boldsymbol{\phi}_m^*) - Bu_m^* + f\|_0^2 \\ &= \|\boldsymbol{\phi}_m^* - \boldsymbol{\phi}_0 - \boldsymbol{A}\nabla(u_m^* - u_0)\|_0^2 + \|\operatorname{div}(\boldsymbol{\phi}_m^* - \boldsymbol{\phi}_0) - B(u_m^* - u_0)\|_0^2 \\ &\leq \beta\|\boldsymbol{q}_m^* - \boldsymbol{q}_0\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} \leq \beta(d(\boldsymbol{q}_0, \mathcal{A}_m) + \varepsilon). \end{aligned}$$

This, together with (3.2), shows that $\mathcal{L}(\boldsymbol{q}_m) \to 0 = \mathcal{L}(\boldsymbol{q}_0)$ as $m \to \infty$. Finally, by combining the lower bound in (3.5) with the fact that $\boldsymbol{q}_0$ satisfies (1.2) a.e. in $\Omega$, we reach the estimate

$$\begin{aligned} \|\boldsymbol{q}_m - \boldsymbol{q}_0\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} &\leq \frac{1}{\alpha}\Big(\|\boldsymbol{\phi}_m - \boldsymbol{\phi}_0 - \boldsymbol{A}\nabla(u_m - u_0)\|_0^2 + \|\operatorname{div}(\boldsymbol{\phi}_m - \boldsymbol{\phi}_0) + B(u_m - u_0)\|_0^2\Big) \\ &\leq \frac{1}{\alpha}\Big(\|\boldsymbol{\phi}_m - \boldsymbol{A}\nabla u_m\|_0^2 + \|\operatorname{div}(\boldsymbol{\phi}_m) + Bu_m + f\|_0^2\Big) = \mathcal{L}(\boldsymbol{q}_m) \to 0. \end{aligned}$$

This concludes the proof. □

The result above assumes that, given $\boldsymbol{\Theta} \in \mathbb{R}^m$, one can compute $\mathcal{L}(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}})$ exactly. This is not the case in general, because we resort to Monte Carlo integration for the computation of the $L^2$ norms in (1.3); cf. the discrete loss functional (2.4). To deal with this issue, we consider a regularized version of the loss functions $\mathcal{L}$ and $\mathcal{L}_N : \mathcal{A}_m \to \overline{\mathbb{R}}$, using $\mathbb{R}^m$ as domain. Given $R > 0$, we define the regularized functional $L : \mathbb{R}^m \to \mathbb{R}$ as

$$L(\boldsymbol{\Theta}) := \begin{cases} \mathcal{L}(u_{\boldsymbol{\Theta}}, \boldsymbol{\phi}_{\boldsymbol{\Theta}}) & \text{if } |\boldsymbol{\Theta}| \leq R, \\ +\infty & \text{otherwise.} \end{cases} \tag{3.6}$$

Next, we let $\{X_i\}_{i\in\mathbb{N}}$ be an i.i.d. sequence of random variables, defined on a probability space $(\Lambda, \Sigma, P)$ with $X_i : \Lambda \to \Omega \quad \forall i \in \mathbb{N}$, with uniform probability density on $\Omega$. Given $\lambda \in \Lambda$, $R > 0$, and $N \in \mathbb{N}$ we set $V_N(\lambda) := \cup_{i \leq N}\{X_i(\lambda)\}$, and the regularized discrete functional $L_{\lambda,N} : \mathbb{R}^m \to \overline{\mathbb{R}}$ as

$$L_{\lambda,N}(\boldsymbol{\Theta}) := \begin{cases} \dfrac{|\Omega|}{N} \displaystyle\sum_{x \in V_N(\lambda)} G_1(\boldsymbol{\Theta}, x) + G_2(\boldsymbol{\Theta}, x) & \text{if } |\boldsymbol{\Theta}| \leq R, \\ +\infty & \text{otherwise,} \end{cases} \tag{3.7}$$

with $G_1$ and $G_2$ as in (3.4).

With these definitions, we can prove the pointwise $P$-almost sure convergence of the sequence $\{L_{\lambda,N}\}_{N\in\mathbb{N}}$ towards $L$.

**Lemma 3.3** (almost sure convergence of regularized discrete loss functions)**.** *Consider $R > 0$, $L$ as in* (3.6)*, $L_{\lambda,N}$ and $\{X_i\}_{i\in\mathbb{N}}$ an i.i.d. family of random variables defined in the probability space* $(\Lambda, \Sigma, P)$ *as in* (3.7)*. Then $L_{\lambda,N}(\boldsymbol{\Theta}) \to L(\boldsymbol{\Theta})$ as $N \to \infty$ $P$-almost surely, for all $\boldsymbol{\Theta} \in \mathbb{R}^m$.*

*Proof.* Since we are using the same parameter $R$ in the definitions of $L$ and $L_{\lambda,N}$, if $|\boldsymbol{\Theta}| > R$ we have $L(\boldsymbol{\Theta}) = L_{\lambda,N}(\boldsymbol{\Theta}) = +\infty$ and there is nothing to be proven. We therefore assume $|\boldsymbol{\Theta}| \leq R$. Recalling $V_N(\lambda) = \cup_{i\leq N}\{X_i(\lambda)\}$ with $\lambda \in \Lambda$ and the definition of $G_1$ and $G_2$ (3.4), an application of the strong law of large numbers yields

$$\frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} |\boldsymbol{\phi}(x) - \boldsymbol{A}\nabla u(x)|^2 \xrightarrow[N\to\infty]{a.s.} \int_\Omega |\boldsymbol{\phi} - \boldsymbol{A}\nabla u|^2,$$

and

$$\frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} |\operatorname{div} \boldsymbol{\phi}(x) - Bu(x) + f(x)|^2 \xrightarrow[N\to\infty]{a.s.} \int_\Omega |\operatorname{div} \boldsymbol{\phi} - Bu + f|^2$$

for all $(u, \boldsymbol{\phi}) \in \mathcal{A}_m$. It follows immdiately that $L_{\lambda,N}(\boldsymbol{\Theta}) \to L(\boldsymbol{\Theta})$ $P$-almost surely as $N \to \infty$. □

We are now in position to prove the almost sure $\Gamma$-convergence of $L_{\lambda,N}$ to $L$ as the number of quadrature points $N \to \infty$.

**Theorem 3.2** (almost sure $\Gamma$-convergence)**.** *Let $R > 0$, $L$ be as in* (3.6)*, and $L_{\lambda,N}$ and $\{X_i\}_{i\in\mathbb{N}}$ be an i.i.d. family of random variables defined in the probability space* $(\Lambda, \Sigma, P)$ *as in* (3.7)*. Then, assuming $f \in L^2(\Omega)$, it holds that $L_{\lambda,N} \xrightarrow{\Gamma} L$ as $N \to \infty$ $P$-almost surely.*

*Proof.* We first observe that the lim-sup inequality is a trivial corollary of Lemma 3.3. Indeed, it suffices to consider the recovery sequence $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}} \subset \mathbb{R}^m$, $\boldsymbol{\Theta}_N \equiv \boldsymbol{\Theta}$, and by Lemma 3.3 we have $L_{\lambda,N}(\boldsymbol{\Theta}_N) \to L(\boldsymbol{\Theta})$ with $N \to \infty$ $P$-almost surely.

We next prove the lim-inf inequality. Given $\boldsymbol{\Theta} \in \mathbb{R}^m$, let $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}} \subset \mathbb{R}^m$ be a sequence of parameters such that $\boldsymbol{\Theta}_N \to \boldsymbol{\Theta}$. We aim to prove that

$$L(\boldsymbol{\Theta}) \leq \liminf_{N\to\infty} L_{\lambda,N}(\boldsymbol{\Theta}_N). \tag{3.8}$$

We observe that, if $|\boldsymbol{\Theta}| > R$ then there exists $N_0 = N_0(\lambda)$ such that $L(\boldsymbol{\Theta}) = L_{\lambda,N}(\boldsymbol{\Theta}_N) = +\infty$ for all $N > N_0$, and (3.8) trivially holds. Therefore, without loss of generality we assume $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}} \subset \overline{B(0,R)}$. In that case, we extract a subsequence in such a way that $L_{\lambda,N}(\boldsymbol{\Theta}_N) \to \liminf_{N\to\infty} L_{\lambda,N}(\boldsymbol{\Theta}_N)$ and, for the sake of simplicity, we omit the relabeling. By Lemma 3.1, the map $\mathbf{\Theta} \mapsto (u_{\mathbf{\Theta}}, \boldsymbol{\phi}_{\mathbf{\Theta}}) \in (\mathcal{A}_m, \|\cdot\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)})$ is continuous and therefore $(u_{\mathbf{\Theta}_N}, \boldsymbol{\phi}_{\mathbf{\Theta}_N}) \to (u_{\mathbf{\Theta}}, \boldsymbol{\phi}_{\mathbf{\Theta}})$ in the $H^1(\Omega) \times H(\mathrm{div};\Omega)$ norm. Because $\Omega$ is bounded, this implies

$$\begin{aligned} &\|u_{\mathbf{\Theta}_N} - u_{\mathbf{\Theta}}\|_{L^1(\Omega)} \to 0, && \|\nabla u_{\mathbf{\Theta}_N} - \nabla u_{\mathbf{\Theta}}\|_{L^1(\Omega)} \to 0, \\ &\|\boldsymbol{\phi}_{\mathbf{\Theta}_N} - \boldsymbol{\phi}_{\mathbf{\Theta}}\|_{L^1(\Omega)} \to 0, \quad \text{and} && \|\operatorname{div} \boldsymbol{\phi}_{\mathbf{\Theta}_N} - \operatorname{div} \boldsymbol{\phi}_{\mathbf{\Theta}}\|_{L^1(\Omega)} \to 0. \end{aligned}$$

Then, defining $G_1$ and $G_2$ as in (3.4), we extract another subsequence in such a way that $G_1(\mathbf{\Theta}_N, x) + G_2(\mathbf{\Theta}_N, x) \to G_1(\mathbf{\Theta}, x) + G_2(\mathbf{\Theta}, x)$ almost everywhere in $\Omega$, and, as before, we omit the relabeling.

In order to prove (3.8), we are going to show that the latter subsequence satisfies $L_{\lambda,N}(\mathbf{\Theta}_N) \to L(\mathbf{\Theta})$ with $N \to \infty$ $P$-almost surely. Let $\varepsilon > 0$ be an arbitrary number, using the triangle inequality, we split

$$|L_{\lambda,N}(\mathbf{\Theta}_N) - L(\mathbf{\Theta})| \le |L_{\lambda,N}(\mathbf{\Theta}_N) - L_{\lambda,N}(\mathbf{\Theta})| + |L_{\lambda,N}(\mathbf{\Theta}) - L(\mathbf{\Theta})|. \tag{3.9}$$

From Lemma 3.3, it follows that $|L_{\lambda,N}(\mathbf{\Theta}) - L(\mathbf{\Theta})| \to 0$ $P$-almost surely. Thus, there exists $N_0 = N_0(\lambda)$ such that $|L_{\lambda,N}(\mathbf{\Theta}) - L(\mathbf{\Theta})| \le \varepsilon/4$ for all $N > N_0$.

In order to bound the first term in the right hand side in (3.9), we first observe that Lemma 3.1 shows that $G_1$ is uniformly bounded and $G_2$ is bounded above by some integrable function. Thus, there exists $s \in L^1(\Omega)$, depending on $R$, such that

$$\left|G_1(\mathbf{\Theta}_N, x) + G_2(\mathbf{\Theta}_N, x) - G_1(\mathbf{\Theta}, x) - G_2(\mathbf{\Theta}, x)\right| \le s(x), \tag{3.10}$$

for all $(\mathbf{\Theta}, x) \in B(0,R) \times \Omega$. Now we apply Egorov's Theorem to construct a set $\mathcal{K} \subset \Omega$ such that $\int_{\mathcal{K}} s(x)dx < \varepsilon/8$ and $G_1(\mathbf{\Theta}_N, \cdot) + G_2(\mathbf{\Theta}_N, \cdot) \to G_1(\mathbf{\Theta}, \cdot) + G_2(\mathbf{\Theta}, \cdot)$ uniformly in $\Omega \setminus \mathcal{K}$. We bound

$$|L_{\lambda,N}(\mathbf{\Theta}_N) - L_{\lambda,N}(\mathbf{\Theta})| \le A_1 + A_2,$$

where

$$\begin{aligned} A_1 &= \frac{|\Omega|}{N} \sum_{x \in V_N(\lambda) \cap (\Omega \setminus \mathcal{K})} \left|G_1(\mathbf{\Theta}_N, x) + G_2(\mathbf{\Theta}_N, x) - G_1(\mathbf{\Theta}, x) - G_2(\mathbf{\Theta}, x)\right|, \\ A_2 &= \frac{|\Omega|}{N} \sum_{x \in V_N(\lambda) \cap \mathcal{K}} \left|G_1(\mathbf{\Theta}_N, x) + G_2(\mathbf{\Theta}_N, x) - G_1(\mathbf{\Theta}, x) - G_2(\mathbf{\Theta}, x)\right|. \end{aligned}$$

Using the uniform convergence in $\Omega \setminus \mathcal{K}$, $P$-almost surely there exists $N_1 = N_1(\lambda)$ such that, if $N > N_1$, then $\left|G_1(\mathbf{\Theta}_N, x) + G_2(\mathbf{\Theta}_N, x) - G_1(\mathbf{\Theta}, x) - G_2(\mathbf{\Theta}, x)\right| < \frac{\varepsilon}{4|\Omega|}$ for all $x \in \Omega \setminus \mathcal{K}$. Then, it follows that $A_1 < \varepsilon/4$ if $N > N_1$.

On the other hand, we use (3.10) to derive

$$A_2 \le \frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} \chi_{\mathcal{K}}(x) s(x).$$

By the strong law of large numbers, we have

$$\frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} \chi_{\mathcal{K}}(x) s(x) \xrightarrow[N \to \infty]{a.s.} \int_{\mathcal{K}} s(x) < \frac{\varepsilon}{8}.$$

Therefore, $P$-almost surely there exists $N_2 = N_2(\lambda)$ such that, if $N > N_2$ then

$$\Big|\frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} \chi_{\mathcal{K}}(x) - \int_{\mathcal{K}} s(x)\Big| < \frac{\varepsilon}{8},$$

which implies that $\frac{|\Omega|}{N} \sum_{x \in V_N(\lambda)} \chi_{\mathcal{K}}(x) s(x) < \frac{\varepsilon}{4}$. Consequently, we have $A_2 < \frac{\varepsilon}{4}$.

Collecting the estimates above, it follows that $P$-almost surely we can choose $N' = N'(\lambda) = \max\{N_0, N_1, N_2\}$ such that

$$|L_{\lambda,N}(\mathbf{\Theta}_N) - L(\mathbf{\Theta})| \leq |L_{\lambda,N}(\mathbf{\Theta}_N) - L_{\lambda,N}(\mathbf{\Theta})| + |L_{\lambda,N}(\mathbf{\Theta}) - L(\mathbf{\Theta})| \leq \varepsilon,$$

for all $N > N'$. This shows that (3.8) holds, and concludes the proof. □

The following theorem is the main result of this section and it roughly states that, if we have a reasonable procedure for the minimization of $L_{\lambda,N}$ on $\mathcal{A}_m$, then we can expect convergence to the solution $\boldsymbol{q}_0$.

**Theorem 3.3** (convergence)**.** *Suppose that for any fixed $m \in \mathbb{N}$ and $R > 0$ we can construct a sequence $\{\mathbf{\Theta}_N\}_{N\in\mathbb{N}} \subset B(0,R) \subset \mathbb{R}^m$ such that $\lim_{N\to\infty} L_{\lambda,N}(\mathbf{\Theta}_N) = \lim_{N\to\infty} \inf_{\mathbf{\Theta}\in\mathbb{R}^m} L_{\lambda,N}(\mathbf{\Theta})$, with $L_{\lambda,N}$ defined as in (3.7). Let $(u_0, \boldsymbol{\phi}_0) = \boldsymbol{q}_0 = \arg\min_{\boldsymbol{q}\in\mathcal{A}} \mathcal{L}(\boldsymbol{q})$. Given $\varepsilon > 0$, there $P$-almost surely exist $m_0 = m_0(\varepsilon) \in \mathbb{N}$, $R = R(m_0) > 0$ and $N_0 = N_0(m_0) \in \mathbb{N}$ such that, if one constructs a sequence $\{\mathbf{\Theta}_N\}_{N\in\mathbb{N}}$ as above, then*

$$\|(u_0, \boldsymbol{\phi}_0) - (u_{\mathbf{\Theta}_N}, \boldsymbol{\phi}_{\mathbf{\Theta}_N})\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} \leq \varepsilon \quad \textit{for all } N > N_0,$$

*where $(u_{\mathbf{\Theta}_N}, \boldsymbol{\phi}_{\mathbf{\Theta}_N})$ is the neural network function defined by the parameters $\mathbf{\Theta}_N$.*

*Proof.* Let $\varepsilon > 0$. From Lemma 3.2, we know that there exists $m_0$ such that

$$\|\boldsymbol{q}_0 - \boldsymbol{q}_{m_0}\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} < \varepsilon/2, \tag{3.11}$$

where $\boldsymbol{q}_{m_0} \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_{m_0}} \mathcal{L}(\boldsymbol{q})$. We next fix $R_0 > 0$ large enough in such a way that there exists a $\mathbf{\Theta} \in B(0, R_0)$ with $(u_{\mathbf{\Theta}}, \boldsymbol{\phi}_{\mathbf{\Theta}}) \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_{m_0}} \mathcal{L}(\boldsymbol{q})$. For this choice of $m_0$ and $R_0$, from Theorem 3.2 we have $L_{\lambda,N} \xrightarrow{\Gamma} L$ P-almost surely. From the definition of $L_{\lambda,N}$ (3.7), it follows immediately that $\{L_{\lambda,N}\}_{N\in\mathbb{N}}$ is an equi-coercive sequence, according to Definition 3.2. Therefore, we deduce that $P$-almost surely there exists $N_0 > 0$ such that

$$\|(u_{\mathbf{\Theta}_N}, \boldsymbol{\phi}_{\mathbf{\Theta}_N}) - \boldsymbol{q}_{m_0}\|_{H^1(\Omega)\times H(\mathrm{div};\Omega)} < \varepsilon/2 \tag{3.12}$$

for all $N > N_0$. This bound follows by Theorem 3.1 because every cluster point of $\{\mathbf{\Theta}_N\}$ is a minimum point for $\mathcal{L}$, and because of the continuity of the map $\mathbf{\Theta} \mapsto (u_{\mathbf{\Theta}}, \boldsymbol{\phi}_{\mathbf{\Theta}})$.

The proof concludes upon combining (3.11) and (3.12). □

## 4. General framework

In this section, we extend the theoretical analysis we performed in Section 3 and put it into an abstract framework. Afterwards, we illustrate how such a framework applies to some well-established unstructured neural-network methods for the approximation of PDEs.

Let $\Omega \subset \mathbb{R}^d$ and $\gamma \in \mathbb{N}$. We assume our problem is posed in some admissible vector space

$$\mathcal{A} \subset W^{\gamma,1}_{loc}(\Omega; \mathbb{R}^n),$$

namely, that every function $\boldsymbol{q} \in \mathcal{A}$ has locally integrable weak derivatives of order up to $\gamma$. The space $\mathcal{A}$ may or may not include boundary conditions or constraints of any type. In the setting we described in Section 1, the target dimension is $n = 1 + d$, the differentiability index is $\gamma = 1$, and we identify $\mathcal{A} \ni \boldsymbol{q} = (u, \boldsymbol{\phi})$. Additionally, we assume the space $\mathcal{A}$ is furnished with some norm $\|\cdot\|_{\mathcal{A}}$, which in our setting corresponds to the $H^1(\Omega) \times H(\mathrm{div};\Omega)$-norm.

We consider $\omega_1, ..., \omega_K$ Borel subsets of $\overline{\Omega}$, each $\omega_i$ furnished with a finite Radon measure $\mu_i$, and some given functions $f_1, ..., f_{n_f}$ with $f_i : \Omega \to \mathbb{R}$. Given some integrable functions $F_i : \mathbb{R}^{n_\gamma n+n_f+d} \to \mathbb{R}$, $1 \le i \le K$, we define the loss functional

$$\mathcal{L}(\boldsymbol{q}) := \sum_{i=1}^{K} \int_{\omega_i} F_i(D^{\alpha_1}\boldsymbol{q}, \dots, D^{\alpha_{n_\gamma}}\boldsymbol{q}, f_1, \dots, f_{n_f}, x)\, d\mu_i,$$

with $F_i$ in such a way that all the integrals involved are well defined. Namely, we assume the loss functional consists of $K$ terms, each of which may be defined on different subdomains of $\overline{\Omega}$. Each of these terms involves certain partial derivatives of $\boldsymbol{q}$ of order up to $\gamma$. The subdomains $\omega_i$ need not be open; for example, we could allow for $\omega_i \subset \partial\Omega$ and the corresponding term would be able to accommodate boundary data. In such a case, the corresponding trace operator must be bounded on the space $\mathcal{A}$.

Consider now a space $\mathcal{A}_m \subset \mathcal{A}$ in such a way that we have a surjective map $\boldsymbol{\Theta} : \mathbb{R}^m \mapsto \mathcal{A}_m$. In the setting from Section 3, this space consists of the functions obtained through a neural network with a modification to account for boundary conditions, cf. (3.1). We denote by $\boldsymbol{q}_{\boldsymbol{\Theta}}$ a generic element of $\mathcal{A}_m$. For $1 \le i \le K$, we define $G_i(\boldsymbol{\Theta}, x) : \mathbb{R}^n \times \Omega \to \mathbb{R}$ as

$$G_i(\boldsymbol{\Theta}, x) = F_i(D^{\alpha_1}\boldsymbol{q}_{\boldsymbol{\Theta}}, \dots, D^{\alpha_{n_\gamma}}\boldsymbol{q}_{\boldsymbol{\Theta}}, f_1, \dots, f_{n_f}, x)$$

and, given $R > 0$, we define the regularized loss functional $L : \mathbb{R}^m \to \mathbb{R}$

$$L(\boldsymbol{\Theta}) := \begin{cases} \mathcal{L}(\boldsymbol{q}_{\boldsymbol{\Theta}}) & \text{if } |\boldsymbol{\Theta}| \le R, \\ +\infty & \text{otherwise.} \end{cases} \tag{4.1}$$

Let $\{X_j^1\}_{j\in\mathbb{N}}, ..., \{X_j^K\}_{j\in\mathbb{N}}$ i.i.d. sequences of random variables, defined in the probability space $(\Lambda, \Sigma, P)$ with $X_j^i : \Lambda \to \omega_i \quad \forall j \in \mathbb{N}$, $1 \le i \le K$, in such a way that the probability density $\overline{\mu}^i$ of $X_j^i$ is distributed as $\mu_i$ on $\omega_i$, that is

$$\overline{\mu}^i(E) = \frac{\mu_i(E)}{\mu_i(\omega_i)} \quad \text{for every Borel set } E \subset \omega_i.$$

Given $\lambda \in \Lambda$, $R > 0$, and $N \in \mathbb{N}$ we define the sampling nodes $V_N^i(\lambda) := \cup_{j\le N}\{X_j^i(\lambda)\}$, and the regularized discrete loss functional $L_{\lambda,N} : \mathbb{R}^m \to \mathbb{R}$,

$$L_{\lambda,N}(\boldsymbol{\Theta}) := \begin{cases} \displaystyle\sum_{i=1}^{K} \frac{\mu_i(\omega_i)}{N} \sum_{x \in V_N^i(\lambda)} G_i(\boldsymbol{\Theta}, x) & \text{if } |\boldsymbol{\Theta}| \le R, \\ +\infty & \text{otherwise.} \end{cases} \tag{4.2}$$

In order to extend our convergence estimates in Section 3 to a general framework, we consider the following hypotheses:

(H1) The map $\mathbb{R}^m \mapsto (\mathcal{A}_m, \|\cdot\|_{\mathcal{A}})$ with $\boldsymbol{\Theta} \mapsto \boldsymbol{q}_{\boldsymbol{\Theta}}$ is continuous.

(H2) For all $1 \le i \le K$ and every convergent sequence $\{\boldsymbol{q}_{\boldsymbol{\Theta}_n}\}_{n\in\mathbb{N}} \subset \mathcal{A}_m$, with $\boldsymbol{q}_{\boldsymbol{\Theta}_n} \to \boldsymbol{q}_{\boldsymbol{\Theta}} \in \mathcal{A}_m$ with respect to the $\mathcal{A}$-norm, there exists a subsequence $\{\boldsymbol{q}_{\boldsymbol{\Theta}_{n_j}}\}_{j\in\mathbb{N}}$ such that $G_i(\boldsymbol{\Theta}_{n_j}, x) \to G_i(\boldsymbol{\Theta}, x)$ $\mu_i$-almost everywhere.

(H3) For every $R > 0$, there exist functions $s_i \in L^1_{\mu_i}(\omega_i)$ such that $|G_i(\boldsymbol{\Theta}, x)| \le s_i(x)$ for all $1 \le i \le K$, for all $\boldsymbol{\Theta} \in B(0, R)$, and $\mu_i$-almost every $x \in \omega_i$.

(H4) The loss function $\mathcal{L}$ has a unique minimizer $\boldsymbol{q}_0 \in \mathcal{A}$, and at least one minimizer in $\mathcal{A}_m$ for all $m \in \mathbb{N}$.

(H5) Let $\{\boldsymbol{q}_m\}_{m\in\mathbb{N}} \subset \mathcal{A}$ be a sequence of minimizers of $\mathcal{L}$, namely, $\boldsymbol{q}_m \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_m} \mathcal{L}(\boldsymbol{q})$ for all $m \in \mathbb{N}$. Then, $\|\boldsymbol{q}_m - \boldsymbol{q}_0\|_{\mathcal{A}} \to 0$ as $m \to \infty$.

Let us comment on these assumptions and how they relate to our analysis in the previous section. Hypothesis (H1) corresponds to the first part in the conclusion of Lemma 3.1, and guarantees the stability of neural network functions with respect to the parameters. Hypothesis (H2) roughly states that, for neural network functions, one can pass from convergence in $\mathcal{A}$ to almost everywhere convergence (up to a subsequence). In our setting, we showed this condition to hold in the proof of Theorem 3.2. Our assumption (H3) requires the existence of an $L^1$-upper bound for the terms $G_i$. This condition appeared in the second part of Lemma 3.1. The ellipticity of the loss functional $\mathcal{L}$ guarantees that hypothesis (H4) is satisfied. Finally, hypothesis (H5) involves the approximability of the solution to the continuous problem by the neural network minimizers of $\mathcal{L}$. In our setting, this appeared in Lemma 3.2, and is a consequence of ellipticity and assumption (3.2).

The following two results extend Theorem 3.2 and Theorem 3.3, respectively; we outline the main steps of their proofs. We first address the Γ-convergence of the regularized discrete functionals.

**Theorem 4.1** (almost sure Γ-convergence, general case)**.** *Let $R > 0$, and $L$, $L_{\lambda,N}$ be as in* (4.1) *and* (4.2)*, respectively. Then, under assumptions (H1), (H2), and (H3), it holds that $L_{\lambda,N} \xrightarrow{\Gamma} L$ with $N \to \infty$ P-almost surely.*

*Proof.* The arguments used in the proof of Theorem 3.2 can be easily adapted to this case. Indeed, the lim-sup inequality follows trivially by taking the recovery sequence $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}}$, $\boldsymbol{\Theta}_N \equiv \boldsymbol{\Theta}$ and using a strong law of large numbers.

To prove the lim-inf inequality, we start from a bounded sequence of parameters $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}}$ and use (H1) to extract a converging subsequence $\{\boldsymbol{q}_{\boldsymbol{\Theta}_N}\}_{N\in\mathbb{N}}$ in the $\mathcal{A}$-norm. Then, by (H2) we can extract another subsequence such that $G_i(\boldsymbol{\Theta}_{n_j}, x) \to G_i(\boldsymbol{\Theta}, x)$ $\mu_i$-almost everywhere for all $1 \le i \le K$ and by (H3) we know that every function $G_i$ has an upper bound in $L^1_{\mu_i}(\Omega)$. The conclusion then follows by applying Egorov's Theorem on every subset $\omega_1, \dots \omega_K$. □

Once we have the almost sure Γ-convergence of the regularized discrete functionals, the convergence of the neural network minimizers can be proved by arguing as in Theorem 3.3.

**Theorem 4.2** (convergence, general case)**.** *Assume hypotheses (H1)–(H5) are satisfied, and suppose that for any fixed $m \in \mathbb{N}$ and $R > 0$ we can construct a sequence $\{\boldsymbol{\Theta}_N\}_{N\in\mathbb{N}} \subset B(0,R) \subset \mathbb{R}^m$ such that $\lim_{N\to\infty} L_{\lambda,N}(\boldsymbol{\Theta}_N) = \lim_{N\to\infty} \inf_{\boldsymbol{\Theta}\in\mathbb{R}^m} L_{\lambda,N}(\boldsymbol{\Theta})$, with $L_{\lambda,N}$ defined as in* (3.7)*. Let $\boldsymbol{q}_0 = \arg\min_{\boldsymbol{q}\in\mathcal{A}} \mathcal{L}(\boldsymbol{q})$. Given $\varepsilon > 0$ there exist $m_0 = m_0(\varepsilon) \in \mathbb{N}$, $R = R(m_0) > 0$ and $N_0 = N_0(m_0) \in \mathbb{N}$ P-almost surely, such that*

$$\|\boldsymbol{q}_0 - \boldsymbol{q}_{\boldsymbol{\Theta}_N}\|_{\mathcal{A}} \le \varepsilon \quad \textit{for all } N > N_0,$$

*where $\boldsymbol{q}_{\boldsymbol{\Theta}_N} \in \mathcal{A}_{m_0}$ is the neural network function defined by the parameters $\boldsymbol{\Theta}_N$.*

*Proof.* We first remark that hypothesis (H4) is needed to guarantee that $\boldsymbol{q}_0 \in \mathcal{A}$ is well defined and therefore that (H5) is meaningful. Given $\varepsilon > 0$, we use hypothesis (H5) to find $m_0 \in \mathbb{N}$ such that, if $\boldsymbol{q}_{m_0} \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_{m_0}} \mathcal{L}(\boldsymbol{q})$ then $\|\boldsymbol{q}_0 - \boldsymbol{q}_{m_0}\|_{\mathcal{A}} < \varepsilon/2$.

Next, we fix $R_0$ sufficiently large so that there exists $\boldsymbol{\Theta} \in B(0, R_0)$ with $\boldsymbol{q}_{\boldsymbol{\Theta}} \in \arg\min_{\boldsymbol{q}\in\mathcal{A}_{m_0}} \mathcal{L}(\boldsymbol{q})$, and we use this $R_0$ in Theorem 4.1 to deduce that $L_{\lambda,N} \xrightarrow{\Gamma} L$ with $N \to \infty$ P-almost surely. The result then follows by the equi-coercivity of the sequence $\{L_{\lambda,N}\}_{N\in\mathbb{N}}$ applying the fundamental theorem of Γ-convergence (Theorem 3.1). □

We next discuss how two well-known methods fit into the framework in hypotheses (H1)–(H5), and thus Theorem 4.2 establishes their convergence.

*Remark* 1 (Deep Ritz Method). The DRM was proposed by E and Yu in [5], and is tailored for numerically solving variational problems. A prototypical example is the homogeneous Dirichlet problem, that corresponds to the minimization of the energy $\mathcal{L}\colon H_0^1(\Omega) \to \mathbb{R}$,

$$\mathcal{L}(u) = \frac{1}{2}\int_\Omega |\nabla u|^2 - \int_\Omega fu.$$

We assume $\|f\|_{L^2(\Omega)} < \infty$, consider $\mathcal{A} = H_0^1(\Omega)$, and define the neural network spaces $\mathcal{A}_m$ as in (3.1). Arguing as in Section 3, it is possible to show that hypotheses (H1)–(H4) hold for this loss function. Indeed, (H1) and (H3) can be proved in the same fashion as Lemma 3.1, while (H2) follows because for every bounded sequence in $H_0^1(\Omega)$ we can extract an almost everywhere convergent subsequence, and (H4) is a standard PDE result. Finally, hypothesis (H5) can be obtained from classical approximation results [27, 28, 29, 30, 31].

*Remark* 2 (Deep Galerkin Method). The DGM was introduced by Sirignano and Spiliopoulos in [6], and uses as loss functional the $L^2$-norm of the PDE residual on the neural network functions. Within the convergence framework in [6, Section 7], and the conditions assumed there, we set $\mathcal{A} := \mathcal{C}^{0,\delta,\delta/2}(\overline{\Omega}_T) \cap L^2((0,T];W_0^{1,2}(\Omega)) \cap W_0^{(1,2),2}(\Omega'_T)$, where $\delta > 0$ and $\Omega'_T$ is any interior subdomain of $\Omega_T$, cf. Theorem 7.3. We furnish this space with the $\|\cdot\|_{H^2(\Omega_T)}$ norm, and define $\mathcal{A}_m$ according to (3.1).

Then, assumptions (H1) and (H3) can be verified by arguing as in Lemma 3.1 by requiring suitable regularity assumptions on the initial and boundary data and parameters of the equation; for example, these hold straightforwardly for these data and parameters are bounded. Hypothesis (H2) can be proved by using the boundedness of $\Omega_T$ and arguing as in the proof of Theorem 3.2 to exploit the convergence properties of the $H^2(\Omega)$-norm. Finally, hypotheses (H4) and (H5) are addressed in [6, Theorem 7.3]. We point out, however, that the convergence of discrete minimizers of $\mathcal{L}$ is proven in the weaker norm $\|\cdot\|_{L^\rho(\Omega_T)}$, with $\rho < 2$. Therefore, our conclusion in Theorem 4.2 is valid if we measure convergence in such a norm.

## 5. Numerical experiments

In this section, we present numerical results for the method we proposed in Section 2. We did not prioritize any particular neural network architecture, and used between one- and five-layer networks with sigmoidal activation functions to construct $u_{\boldsymbol{\Theta}}$ and $\boldsymbol{\phi}_{\boldsymbol{\Theta}}$. For the construction of the auxiliary functions $\boldsymbol{n}$, $d_{\mathcal{D}}$, $d_{\mathcal{N}}$, $G_{\mathcal{D}}$, $G_{\mathcal{N}}$, we used between one and three-layer networks with less neurons per layer. In the training process, we used the ADAM [26] algorithm to update the parameters, with a decaying learning rate schedule.

We observe an improvement in the method's performance when explicit approximations of the auxiliary functions $d_{\mathcal{D}}$ and $d_{\mathcal{N}}$ are used. These functions, which depend on the geometry of the domain, are many times explicitly available in practice.

We recall that, as explained in sections 3 and 4, the numerical solution depends on the number of degrees of freedom $m$ and the number of collocation points $N$. Both must go to infinity to guarantee convergence. In all the numerical examples we show below, these quantities remain fixed. Therefore, in these examples the convergence as a function of the iterations occurs towards the minimizer of the discrete loss functional $L_{\lambda,N}$ (cf. (3.7)) corresponding to the values of $m$ and $N$ we have set.

*Example* 5.1 (Laplace operator). We consider the following problem in arbitrary dimension. Let $\Omega = \{x \in \mathbb{R}^d : \; -1 < x_1, ..., x_d < 1\}$, $\Gamma_{\mathcal{N}} = [-1,1]^{d-1} \times \{1\}$, and $k \in \mathbb{N}$. We seek $u \colon \Omega \to \mathbb{R}$ such that

$$\begin{cases} -\Delta u = \displaystyle\prod_{i=1}^{d-1} \sin(k\pi x_i) \big((d-1)k^2\pi^2(1-x_d^2)+2\big) & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega \setminus \Gamma_{\mathcal{N}}, \\ \nabla u \cdot \boldsymbol{\nu} = -2 \displaystyle\prod_{i=1}^{d-1} \sin(k\pi x_i) & \text{on } \Gamma_{\mathcal{N}}. \end{cases} \tag{5.1}$$

Here, we have $\boldsymbol{\nu} = (0, ..., 0, 1)$ on $\Gamma_{\mathcal{N}}$, and the solution to (5.1) is

$$u = \prod_{i=1}^{d-1} \sin(k\pi x_i)(1-x_d^2).$$

We point out that the parameter $k$ is a frequency that allows us to choose how oscillatory the exact solution $u$ is. We first tested the method in a two-dimensional domain ($d = 2$). Figure 5.1 displays the results we obtained for $k = 1$ and by constructing $u_{\boldsymbol{\Theta}}$ and $\boldsymbol{\phi}_{\boldsymbol{\Theta}}$ using neural networks with 15 sigmoidal activation functions per layer. At the end of the stochastic gradient descent algorithm we computed the value $L_N(\boldsymbol{\Theta}) = 0.0450$. Taking into account the ellipticity of the loss function $\mathcal{L}$, arguing as in Lemma 3.2 we deduce

$$\mathcal{L}(\boldsymbol{q}_m) \simeq \|\boldsymbol{q}_m - \boldsymbol{q}_0\|_{H^1(\Omega)\times H(\text{div};\Omega)},$$

and therefore this quantity serves as an error estimator.

Figure 5.2 corresponds to $k = 2$, and we used a similar architecture, but with 18 sigmoidal activation functions per layer. We observed a fast convergence in the number of iterations, reaching $L_N(\boldsymbol{\Theta}) = 1.89$ by the end of the minimization algorithm. Finally, Figure 5.3 reports the results we obtained in case $d = 5$, $k = 1$. In this case, we used networks with 25 sigmoidal activation functions per layer and obtained $L_N(\boldsymbol{\Theta}) = 2.32$.

*Example* 5.2 (singularly perturbed problem). Let $\varepsilon > 0$, $\Omega = (0,1)^2$, $\boldsymbol{b} = (-1+2\varepsilon, -1+2\varepsilon)$, $c = 2(1-\varepsilon)$, and the function $f \colon \Omega \to \mathbb{R}$,

$$f(x,y) = -\left[x - \left(\frac{1-e^{-x/\varepsilon}}{1-e^{-1/\varepsilon}}\right) + y - \left(\frac{1-e^{-y/\varepsilon}}{1-e^{-1/\varepsilon}}\right)\right] e^{x+y}.$$

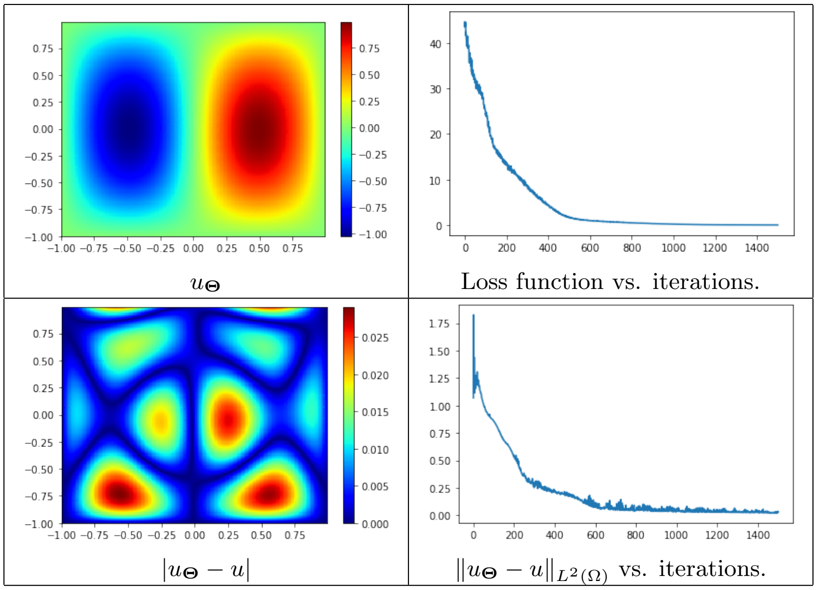
We consider the singularly perturbed problem: find $u \colon \Omega \to \mathbb{R}$ such that

$$\begin{cases} -\varepsilon\Delta u + \boldsymbol{b}\cdot\nabla u + cu = f & \text{in } \Omega, \\ u = 0 & \text{on } \partial\Omega. \end{cases} \tag{5.2}$$

The exact solution to (5.2) is

$$u(x,y) = \left(x - \frac{1-e^{-x/\varepsilon}}{1-e^{-1/\varepsilon}}\right)\left(y - \frac{1-e^{-y/\varepsilon}}{1-e^{-1/\varepsilon}}\right) e^{x+y}.$$

Figure 5.4 exhibits our computed solutions for this example with $\varepsilon = 0.05$. In that case, we observed a fast convergence towards the solution, reaching $L_N(\boldsymbol{\Theta}) = 0.0112$, as well as a good adaptation of the discrete solution to the boundary layers.



$u_{\mathbf{\Theta}}$

Loss function vs. iterations.

$|u_{\mathbf{\Theta}} - u|$

$\|u_{\mathbf{\Theta}} - u\|_{L^2(\Omega)}$ vs. iterations.

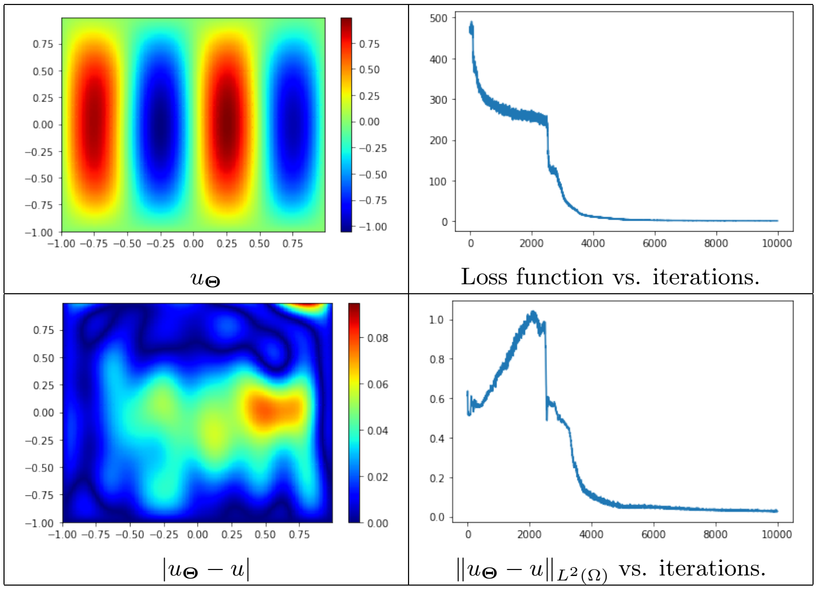
Figure 5.1. Top left: computational solution $u_{\mathbf{\Theta}}$ to (5.1) in case $k = 1$ and $d = 2$. In the computation, we used a learning rate $\ell = 0.005$, with 2,000 collocation points, 1,500 optimization steps, and 3603 degrees of freedom (including auxiliary functions). We used a one-layer networks both for the main and auxiliary functions. The panel in bottom left exhibits the pointwise discrepancy $|u - u_{\mathbf{\Theta}}|$. We also report on the evolution of the loss function (top right) and the $L^2$ error (bottom right).

## 6. Concluding remarks

In this work, we have proposed a First-Order System Least Squares (FOSLS) method based on deep learning for numerically solving second-order elliptic PDEs. This method is meshless, which is naturally advantageous for high-dimensional problems, but as a consequence implies that we cannot compute the loss functions exactly. Taking into account this practical issue, we proved the almost sure convergence of the neural network minimizers towards the PDE solutions. We furthermore extended the theoretical framework to incorporate other methods based on Monte Carlo quadrature.

*Remark* 3 (almost-everywhere solutions)*.* The convergence proofs in Sections 3 and 4 are based on the use of regularized versions of the cost functionals and their discretizations. Regularization consists in restricting the size of the parameters, namely, imposing that $|\mathbf{\Theta}| < R$ for certain $R < \infty$. This ensures that any neural network function with large derivatives is penalized, thereby preventing minimizers from approximating non-smooth functions.

Far from being an artificial condition of the proof, regularization mechanisms of this kind are necessary in the implementation to avoid convergence towards functions that satisfy the PDE almost everywhere but are not weak solutions of the target problem. To illustrate this point, consider the



$u_\Theta$

Loss function vs. iterations.

$|u_\Theta - u|$

$\|u_\Theta - u\|_{L^2(\Omega)}$ vs. iterations.

FIGURE 5.2. Computational solution $u_\Theta$ (top left), evolution of the loss function (top right), pointwise error (bottom left), and evolution of the $L^2$-error (bottom right) for (5.1) with $k = 2$ and $d = 2$. We employed five-layer networks for the main functions and three-layer networks for the auxiliary functions. We used an initial learning rate $\ell = 0.005$, with 5,000 collocation points, 10,000 optimization steps, and 2901 degrees of freedom. We halved the learning rate every 2,500 optimization steps.

following example, which is just (1.2) in a simplified setting: seek $u, \phi : (0, 1) \to \mathbb{R}$ such that

$$\begin{cases} \phi - u' = 0 & \text{in } (0,1), \\ \phi' = 0 & \text{in } (0,1), \\ u(0) = 0, & \\ u(1) = 1. & \end{cases} \tag{6.1}$$

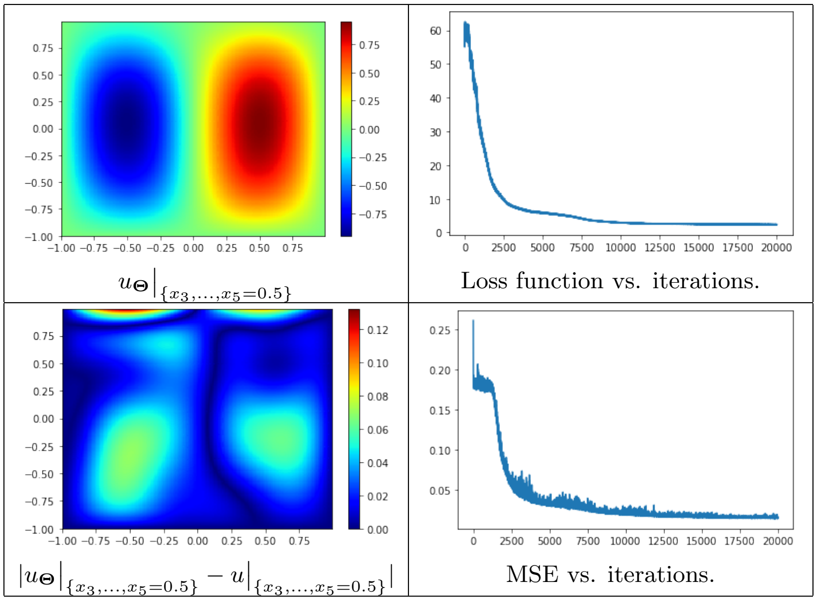
Naturally, the unique minimizer of the least-squares functional (cf. (1.3))

$$\mathcal{L}(u, \phi) := \|\phi - u'\|^2_{L^2(\Omega)} + \|\phi'\|^2_{L^2(\Omega)}$$

in the corresponding admissible set $\mathcal{A} = \{(u, \phi) \in [H^1(\Omega)]^2 \colon u(0) = 0,\ u(1) = 1\}$ is $u(x) = x$ and $\phi(x) = 1$. Let $\delta \in (0, 1/2)$ be a small number, and consider the functions

$$u_\delta(x) = \begin{cases} 0 & \text{in } (0, 1/2 - \delta) \\ \dfrac{x - 1/2 + \delta}{2\delta} & \text{in } (1/2 - \delta, 1/2 + \delta) \\ 1 & \text{in } (1/2 + \delta, 1) \end{cases}, \qquad \phi_\delta(x) = \begin{cases} 0 & \text{in } (0, 1/2 - \delta) \\ \dfrac{1}{2\delta} & \text{in } (1/2 - \delta, 1/2 + \delta) \\ 0 & \text{in } (1/2 + \delta, 1) \end{cases}. \tag{6.2}$$

We notice $\phi_\delta = u'_\delta$ a.e. in $(0, 1)$ and $\mathcal{L}(u_\delta, \phi_\delta) = 0$, although $(u_\delta, \phi_\delta) \notin \mathcal{A}$, because $\phi_\delta$ is not an $H^1$ function.



$u_{\mathbf{\Theta}}\big|_{\{x_3,\dots,x_5=0.5\}}$

Loss function vs. iterations.

$\big|u_{\mathbf{\Theta}}\big|_{\{x_3,\dots,x_5=0.5\}} - u\big|_{\{x_3,\dots,x_5=0.5\}}\big|$

MSE vs. iterations.

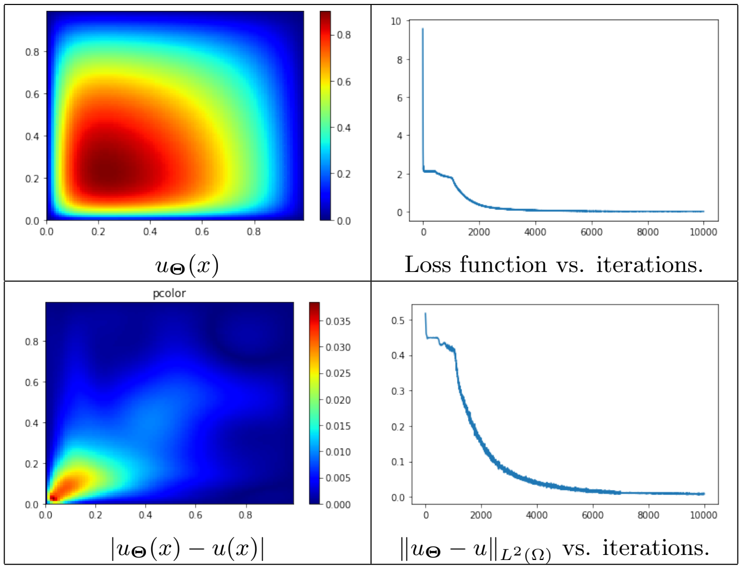
Figure 5.3. Slice of the solution $u_{\mathbf{\Theta}}$ (top left), evolution of the loss function (top right), pointwise error (bottom left), and evolution of the Mean Squared Error (MSE) (bottom right) for (5.1) with $k = 1$ and $d = 5$. We employed five-layer networks for the main functions and three-layer networks for the auxiliary functions. We used an initial learning rate $\ell = 0.005$, with 12,000 collocation points, 20,000 optimization steps, and 5656 degrees of freedom. We halved the learning rate every 4,000 optimization steps. We estimated the MSE by using 5,000 random points in $\Omega$ (re-sampled at every step).

If we utilize the discrete functional (2.1) with collocation points, and none of these points lies in the interval $(1/2 - \delta, 1/2 + \delta)$, then for these two functions we would have

$$\mathcal{L}_N(u_\delta, \phi_\delta) = 0.$$

We remark that, independently of the number of collocation points $N$, one can always take $\delta > 0$ sufficiently small such that the probability of none of the sampling points lies in $(1/2 - \delta, 1/2 + \delta)$ is significant. Therefore, if our neural network is capable of producing functions $(u_{\mathbf{\Theta}}, \phi_{\mathbf{\Theta}})$ approximating $(u_\delta, \phi_\delta)$ in (6.2) (cf. Figure 6.1), then during the optimization process the descent algorithm may choose to approximate the pair $(u, \phi) = (\chi_{(1/2,1)}, 0)$. This function satisfies the differential equations in (6.1) almost everywhere, but is not a significant solution. The issue of approximating bad solutions of this kind is mitigated by applying classic regularization techniques that penalize large parameters, because $|\mathbf{\Theta}|$ must be large in order to $u'_{\mathbf{\Theta}}$ be large at some portion of the domain.

This difficulty extends to all methods based on the minimization of cost functionals similar to (2.1), such as DGM [6] or DRM [5]. The issue stems from the fact that the functional (2.1) is unable to distinguish between regular solutions (belonging to a suitable Sobolev space) from any other functions that satisfy the equation almost everywhere. As far as we know, this problem has



$u_{\Theta}(x)$

Loss function vs. iterations.

$|u_{\Theta}(x) - u(x)|$

$\|u_{\Theta} - u\|_{L^2(\Omega)}$ vs. iterations.

FIGURE 5.4. Computational solution $u_{\Theta}$ (top left), evolution of the loss function (top right), pointwise error (bottom left), and evolution of the $L^2$-error (bottom right) for (5.2) with $\varepsilon = 0.05$. We used an initial learning rate $\ell = 0.005$, with 5,000 collocation points, 10,000 optimization steps, and 3543 degrees of freedom. Auxiliary functions have been approximated exactly.

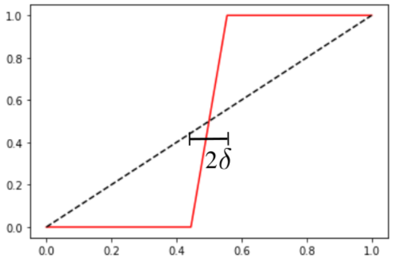


FIGURE 6.1. In red the function $u_{\delta}$ defined in 6.2. On dashed lines the solution of problem (6.1).

not been addressed in the literature, and the question of how to develop suitable regularization techniques for these approaches remains open.

*Remark* 4 (approximation of non-smooth solutions). There are, however, problems in which the solution presents large gradients in regions of the domain. One can typically think of singularly perturbed problems, such as (5.2), or singularities arising due to poor boundary regularity, such as for the Poisson problem on an $L$-shaped domain. In those problems, regularization can limit the approximation capabilities of the algorithm.

For algebraic boundary singularities, if the boundary conditions are imposed in a strong fashion, as discussed in Section 2.1.2, one could aim to modify the rate at which the corresponding auxiliary function $d_{\mathcal{D}}$ or $d_{\mathcal{N}}$ decreases to zero near the singularity. This could potentially avoid $v_{\mathbf{\Theta}}$ having to approximate a singular function and lead to a faster convergence. Nevertheless, this requires an a priori knowledge about the location and behavior of the singularities of the solution, that is not available in general. We emphasize that the theory we developed in Section 4 does not make any regularity assumption on the PDE, and therefore includes the case of non-smooth solutions.

## Acknowledgements

FMB has been supported in part by a PEDECIBA postdoctoral fellowship.

## References

[1] I. E. Lagaris, A. Likas, D. I. Fotiadis, Artificial neural networks for solving ordinary and partial differential equations, IEEE transactions on neural networks 9 (5) (1998) 987–1000.

[2] I. E. Lagaris, A. C. Likas, D. G. Papageorgiou, Neural-network methods for boundary value problems with irregular boundaries, IEEE Transactions on Neural Networks 11 (5) (2000) 1041–1049.

[3] H. Lee, I. S. Kang, Neural algorithm for solving differential equations, Journal of Computational Physics 91 (1) (1990) 110–131.

[4] A. Malek, R. S. Beidokhti, Numerical solution for high order differential equations using a hybrid neural network—optimization method, Applied Mathematics and Computation 183 (1) (2006) 260–271.

[5] W. E, B. Yu, The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems, Communications in Mathematics and Statistics 6 (1) (2018) 1–12.

[6] J. Sirignano, K. Spiliopoulos, DGM: A deep learning algorithm for solving partial differential equations, Journal of Computational Physics 375 (2018) 1339–1364.

[7] C. He, X. Hu, L. Mu, A mesh-free method using piecewise deep neural network for elliptic interface problems, arXiv preprint arXiv:2005.04847.

[8] Z. Wang, Z. Zhang, A mesh-free method for interface problems using the deep learning approach, Journal of Computational Physics 400 (2020) 108963.

[9] Y. Zang, G. Bao, X. Ye, H. Zhou, Weak adversarial networks for high-dimensional partial differential equations, Journal of Computational Physics (2020) 109409.

[10] M. Raissi, P. Perdikaris, G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, Journal of Computational physics 378 (2019) 686–707.

[11] J. Xu, Finite neuron method and convergence analysis, Communications in Computational Physics 28 (2020) 1707–1745.

[12] M. Liu, Z. Cai, J. Chen, Adaptive two-layer ReLU neural network: I. best least-squares approximation, Computers & Mathematics with Applications 113 (2022) 34–44.

[13] M. Liu, Z. Cai, Adaptive two-layer ReLU neural network: II. Ritz approximation to elliptic PDEs, Computers & Mathematics with Applications 113 (2022) 103–116.

[14] S. Wojtowytsch, W. E, Can shallow neural networks beat the curse of dimensionality? A mean field training perspective, IEEE Transactions on Artificial Intelligence 1 (2) (2020) 121–129.

[15] W. E, S. Wojtowytsch, Some observations on high-dimensional partial differential equations with Barron data, in: Mathematical and Scientific Machine Learning, PMLR, 2022, pp. 253–269.

[16] Z. Cai, R. Lazarov, T. Manteuffel, S. McCormick, First-order system least squares for second-order partial differential equations: Part I, SIAM Journal on Numerical Analysis 31 (6) (1994) 1785–1799.

[17] Z. Cai, J. Chen, M. Liu, X. Liu, Deep least-squares methods: An unsupervised learning-based numerical method for solving elliptic PDEs, Journal of Computational Physics 420 (2020) 109707.

[18] L. Lyu, Z. Zhang, M. Chen, J. Chen, Mim: A deep mixed residual method for solving high-order partial differential equations, arXiv preprint arXiv:2006.04146.

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

[20] H. Sheng, C. Yang, PFNN: a penalty-free neural network method for solving a class of second-order boundary-value problems on complex geometries, Journal of Computational Physics 428 (2021) 110085.

[21] J. Siegel, Q. Hong, X. Jin, W. Hao, J. Xu, A priori analysis of stable neural network solutions to numerical PDEs, arXiv preprint arXiv:2107.04466.

[22] U. Zerbinati, Pinns and gals: An priori error estimates for shallow physically informed neural network applied to elliptic problems, arXiv preprint arXiv:2202.01059.

[23] J. He, L. Li, J. Xu, Relu deep neural networks from the hierarchical basis perspective, Computers & Mathematics with Applications 120 (2022) 105–114.

[24] J. W. Siegel, J. Xu, High-order approximation rates for neural networks with ReLU$^k$ activation functions, arXiv preprint arXiv:2012.07205.

[25] J. W. Siegel, J. Xu, Sharp lower bounds on the approximation rate of shallow neural networks, arXiv preprint arXiv:2106.14997.

[26] D. Kingma, J. Ba, Adam: A method for stochastic optimization., in: In Proceedings of the 3rd International-Conference for Learning Representations—ICLR, San Diego, CA, 2015, pp. 7–9.

[27] G. Cybenko, Approximation by superpositions of a sigmoidal function, Mathematics of control, signals and systems 2 (4) (1989) 303–314.

[28] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural networks 4 (2) (1991) 251–257.

[29] A. Barron, Universal approximation bounds for superpositions of a sigmoidal function, IEEE Transactions on Information theory 39 (3) (1993) 930–945.

[30] D. Yarotsky, Error bounds for approximations with deep relu networks, Neural Networks 94 (2017) 103–114.

[31] J. He, L. Li, J. Xu, C. Zheng, Relu deep neural networks and linear finite elements, J. Comput. Math. 38 (3) (2020) 502–527. `doi:10.4208/jcm.1901-m2018-0160`.
URL `https://doi.org/10.4208/jcm.1901-m2018-0160`

[32] R. Arora, A. Basu, P. Mianjy, A. Mukherjee, Understanding deep neural networks with rectified linear units, in: International Conference on Learning Representations, 2018.

[33] A. Braides, A handbook of Γ-convergence, in: Handbook of Differential Equations: stationary partial differential equations, Vol. 3, Elsevier, 2006, pp. 101–213.

Departamento de Matemática, Universidad de Buenos Aires, Buenos Aires, Argentina
*Email address*: `fbersetche@dm.uba.ar`

Centro de Matemática, Universidad de la República, Montevideo, Uruguay
*Email address*: `jpb@cmat.edu.uy`