Journal of Machine Learning Research 27 (2026) 1-75 Submitted 11/24; Revised 1/26; Published 5/26

A Natural Primal-Dual Hybrid Gradient Method for Adversarial Neural Network Training on Solving Partial Differential Equations

Shu Liu SLIU11@FSU.EDU
Department of Mathematics
Florida State University,
Tallahassee, FL 32306, USA

Stanley Osher SJO@MATH.UCLA.EDU
Department of Mathematics
University of California, Los Angeles
Los Angeles, CA 90095, USA

Wuchen Li WUCHEN@MAILBOX.SC.EDU
Department of Mathematics
University of South Carolina
Columbia, SC 29208, USA

Editor: Weijie Su

Abstract

We propose a scalable preconditioned primal-dual hybrid gradient algorithm for solving partial differential equations (PDEs). We multiply the PDE with a dual test function to obtain an inf-sup problem whose loss functional involves lower-order differential operators. The Primal-Dual Hybrid Gradient (PDHG) algorithm is then leveraged for this saddle point problem. By introducing suitable precondition operators to the proximal steps in the PDHG algorithm, we obtain an alternative natural gradient ascent-descent optimization scheme for updating the neural network parameters. We apply the Krylov subspace method (MINRES) to evaluate the natural gradients efficiently. Such treatment readily handles the inversion of precondition matrices via matrix-vector multiplication. An *a posteriori* convergence analysis is established for the time-continuous version of the proposed algorithm for general linear PDEs. By incorporating appropriate boundary loss terms, we further obtain a refined *a priori* convergence result for elliptic equations in divergence form. The algorithm is tested on various types of PDEs with dimensions ranging from 1 to 50, including linear and nonlinear elliptic equations, reaction-diffusion equations, and Monge-Ampère equations stemming from the L^2 optimal transport problems. We compare the performance of the proposed method with several commonly used deep learning algorithms such as physics-informed neural networks (PINNs), the DeepRitz method and weak adversarial networks (WANs) using either the Adam or the L-BFGS optimizer. The numerical results suggest that the proposed method performs efficiently and robustly and converges more stably with higher accuracy.

Keywords: Deep learning for solving PDEs; Neural Networks; Inf-sup problem; Primal-Dual Hybrid Gradient (PDHG) algorithm; Natural Gradient; Convergence analysis; Monge-Ampère equation.

1. Introduction

Machine learning, particularly deep learning, is a fast-developing direction with modern computational technologies (Amari, 1998) and applications (Goodfellow et al., 2020; Arjovsky et al., 2017). Typical examples of applications often come from computer science, including creating new images, videos, and voices and generating languages. During the development of modern applications, machine learning has introduced a variety of nonlinear methodologies, including computational nonlinear models such as neural networks, as well as variational frameworks such as generative adversarial networks (Goodfellow et al., 2020). The impact of machine learning on scientific computing has therefore been profound and cannot be overstated.

In recent years, deep learning algorithms have been developed to solve partial differential equations (PDEs). The physics-informed neural networks (PINN) method (Raissi et al., 2019) employs neural networks to approximate PDE solutions by minimizing the discrepancy between observed data and the equation's residual. The DeepRitz method (Yu et al., 2018) computes neural network surrogate solutions for PDEs using a variational approach, minimizing the associated energy functional. The Forward-Backward Stochastic Differential Equation (FBSDE) method (Han et al., 2017) makes use of the nonlinear Feynman-Kac formula for semi-linear parabolic equations to derive numerical solutions at specific time-space points. Additionally, the Weak Adversarial Network (WAN) approach (Zang et al., 2020; Cai et al., 2024) leverages the weak formulation of PDEs by multiplying the original equation with a test function, resulting in an inf-sup saddle point problem for computing the equation. This approach is applicable to various types of equations and is scalable to PDEs in high dimensions.

While these methods demonstrate the potential of applying machine learning techniques in solving PDEs, challenges such as hyperparameter tuning, loss function designing, and convergence guarantees remain unresolved. More critically, due to the nonlinearity of neural networks, conventional optimizers such as Adam (Kingma, 2014) or RMSProp (Tieleman and Hinton, 2012) suffer from strong fluctuations and do not achieve stable convergence, which complicates the implementation of current algorithms.

In this research, we aim to tackle these challenges by adopting the adversarial training strategy and propose a crucial preconditioned optimizer that takes advantage of the primal-dual hybrid gradient (PDHG) algorithm (Zhu and Chan, 2008; Chambolle and Pock, 2011). We utilize suitable preconditioned gradients known as the natural gradients (Müller and Zeinhofer, 2023) to update the parameters of the neural networks. The proposed algorithm, named the Natural Primal-Dual Hybrid Gradient (NPDG) method, performs efficiently and converges more stably than classical machine learning-based PDE solvers. In addition, we also provide a theoretical convergence guarantee for the proposed algorithm.

To illustrate the main idea, we consider the following linear equation posed with suitable boundary condition,

$$\mathcal{L}u = f \text{ on } \Omega, \quad \mathcal{B}u = g \text{ on } \partial\Omega. \tag{1}$$

Here $\Omega \subset \mathbb{R}^d$ is a bounded open region, $\partial\Omega$ denotes the boundary of Ω, $f\colon \Omega \to \mathbb{R}$, $g\colon \partial\Omega \to \mathbb{R}$ are L^2 functions, and $u\colon \Omega \to \mathbb{R}$ belongs to $H^2(\Omega)$. We assume $\mathcal{L}$ being a second-order elliptic operator and $\mathcal{B}$ as a linear boundary operator, which indicates the Dirichlet or Neumann or more general boundary conditions. Here we assume that $\mathcal{L}$ can be split as

$\mathcal{L} = \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p$ with $\mathcal{M}_p, \mathcal{M}_d$ being first-order differential operators, $\widetilde{\mathcal{L}}$ is a well-conditioned bounded linear operator, and $\mathcal{M}_d^*$ denotes the L^2 adjoint of $\mathcal{M}_d$. Suppose this equation admits a unique classical solution $u_* \in C^2(\Omega) \cap C(\bar{\Omega})$. The goal is to efficiently compute u_*.

By introducing the test functions (dual variables) $\varphi \in H_0^1(\Omega)$ and $\psi \in L^2(\partial\Omega)$ into the equation (1), we consider the following inf-sup problem with quadratic regularization terms. Here, $\nu > 0$ denotes the regularization coefficient,

$$\begin{aligned} \inf_u \sup_{\varphi,\psi} \mathscr{E}(u,\varphi,\psi) :=& \langle \widetilde{\mathcal{L}} \mathcal{M}_p u, \mathcal{M}_d \varphi \rangle_{L^2(\Omega)} - \langle f, \varphi \rangle_{L^2(\Omega)} - \frac{\nu}{2} \|\mathcal{M}_d \varphi\|_{L^2(\Omega)}^2 \\ &+ \langle \mathcal{B}u - g, \psi \rangle_{L^2(\partial\Omega)} - \frac{\nu}{2} \|\psi\|_{L^2(\partial\Omega)}^2. \end{aligned} \tag{2}$$

Note that $u = u_*, \varphi = 0, \psi = 0$ form the saddle point of $\mathscr{E}$. In order to approach this saddle point and hence solve for u_*, we apply a preconditioned version of the PDHG algorithm to the inf-sup problem (2). The algorithm utilizes alternative proximal point steps and an intermediate extrapolation to solve the inf-sup problem with selected preconditioning operators $\mathcal{M}_p, \widetilde{\mathcal{L}}, \mathcal{M}_d$. More specifically, the algorithm repeats the following three-line iteration

$$\begin{aligned} &(\varphi_{n+1}, \psi_{n+1}) = \operatorname*{argmin}_{\varphi,\psi} \left\{ \frac{1}{2\tau_\varphi} (\|\mathcal{M}_d \varphi - \mathcal{M}_d \varphi_n\|_{L^2(\Omega)}^2 + \|\psi - \psi_n\|_{L^2(\partial\Omega)}^2) - \mathscr{E}(u_n, (\varphi, \psi)) \right\}, \\ &\widetilde{\varphi}_{n+1} = \varphi_{n+1} + \omega(\varphi_{n+1} - \varphi_n), \quad \widetilde{\psi}_{n+1} = \psi_{n+1} + \omega(\psi_{n+1} - \psi_n) \\ &u_{n+1} = \operatorname*{argmin}_{u} \left\{ \frac{1}{2\tau_u} (\|\mathcal{M}_p u - \mathcal{M}_p u_n\|_{L^2(\Omega)}^2 + \|\mathcal{B}u - \mathcal{B}u_n\|_{L^2(\partial\Omega)}^2) + \mathscr{E}(u, (\widetilde{\varphi}_{n+1}, \widetilde{\psi}_{n+1})) \right\}. \end{aligned} \tag{3}$$

Here $\tau_u, \tau_\varphi > 0$ are the step sizes of the algorithm and $\omega > 0$ denotes the extrapolation coefficient. We briefly illustrate the motivation of preconditioning steps in (3). In general, the differential operator $\mathcal{L}$ is usually ill-conditioned. As shown in (Liu et al., 2024a), the convergence rate of the un-preconditioned dynamic equals $1 - \mathcal{O}(\frac{1}{\kappa^2})$ with κ denoting the condition number of the spatial discretization of $\mathcal{L}$. The convergence speed decreases fast as κ gets larger. This pronounced slowdown motivates us to introduce appropriate preconditioning in the proximal steps (i.e., the first and third lines) of (3) in order to mitigate the resulting inefficiency.

So far, the algorithm we have developed remains at the functional level, which is generally intractable for practical implementation. To realize the proposed PDHG algorithm, we parameterize $u(\cdot)$, $\varphi(\cdot)$ and $\psi(\cdot)$ as $u_\theta(\cdot), \varphi_\eta(\cdot)$ and $\psi_\xi(\cdot)$ with the tunable parameters $\theta \in \Theta_\theta \subseteq \mathbb{R}^{m_\theta}, \eta \in \Theta_\eta \subseteq \mathbb{R}^{m_\eta}$ and $\xi \in \Theta_\xi \subseteq \mathbb{R}^{m_\xi}$. A straightforward parameterization approach involves expressing these functions as linear combinations of predefined basis functions—a method traditionally employed in finite element methods. However, as the problem's dimensionality increases, such parameterization becomes computationally prohibitive due to the curse of dimensionality, since it requires a significant number of basis functions to maintain accuracy (Hu et al., 2024). Recent advances in deep learning have highlighted the potential of neural networks as computational tools to solve PDEs. Given their flexibility and expressive power, we adopt three neural network functions, such as Multilayer Perceptrons (MLPs, see Appendix A for detailed definition), to represent u_θ, φ_η, and ψ_ξ. Therefore, we reduce the original algorithm in functional spaces to a time-discrete dynamic in which the parameters θ^n, η^n, ξ^n evolve together.

We replace the implicit proximal step for updating η, ξ, θ with an explicit scheme known as the linearized PDHG algorithm. We come up with the following algorithm:

$$\begin{aligned}
\begin{bmatrix} \eta^{n+1} \\ \xi^{n+1} \end{bmatrix} &= \begin{bmatrix} \eta^{n} \\ \xi^{n} \end{bmatrix} + \tau_\varphi \begin{bmatrix} M_d(\eta^n)^\dagger \nabla_\eta \mathscr{E}(u_{\theta^n}, \varphi_{\eta^n}, \psi_{\xi^n}) \\ M_{bdd}(\xi^n)^\dagger \nabla_\xi \mathscr{E}(u_{\theta^n}, \varphi_{\eta^n}, \psi_{\xi^n}) \end{bmatrix}, \\
\begin{bmatrix} \widetilde{\varphi}_{n+1} \\ \widetilde{\psi}_{n+1} \end{bmatrix} &= \begin{bmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{bmatrix} + \omega \left(\begin{bmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{bmatrix} - \begin{bmatrix} \varphi_{\eta^{n}} \\ \psi_{\xi^{n}} \end{bmatrix} \right), \\
\theta^{n+1} &= \theta^n - \tau_u M_p(\theta^n)^\dagger \nabla_\theta \mathscr{E}(u_{\theta^n}, \widetilde{\varphi}_{n+1}, \widetilde{\psi}_{n+1}).
\end{aligned} \tag{4}$$

Here $M_d(\eta) \in \mathbb{R}^{m_\eta \times m_\eta}$, $M_{bdd}(\xi) \in \mathbb{R}^{m_\xi \times m_\xi}$, $M_p(\theta) \in \mathbb{R}^{m_\theta \times m_\theta}$ are Gram type matrices. They are derived from the bilinear form approximation of the proximal steps in the PDHG algorithm (3). Here, we denote "$\dagger$" as the Moore–Penrose inverse of a matrix. The precondition matrix $M_d(\eta^n)$ incorporates the information of the precondition operator $\mathcal{M}_p$, which is built in the original operator $\mathcal{L}$; we call $M_d(\eta^n)^\dagger \nabla_\eta \mathscr{E}(u_{\theta^n}, \varphi_{\eta^n}, \psi_{\xi^n})$ the *natural gradient* of $\mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$ with respect to η. Similarly, we can define the natural gradient ascent and descent directions for variables ξ and θ. The algorithm alternatively updates the primal and dual parameters along the natural gradient directions. An additional extrapolation step in the functional space is introduced to enhance the convergence of the method. We denote the above updates as the **Natural Primal-Dual Hybrid Gradient** algorithm. For simplicity, we refer to this method as the **NPDG** algorithm in the following discussion. We refer the readers to Section 2 for a detailed derivation of the algorithm.

While the NPDG algorithm is designed around linear PDEs, it effectively accommodates equations with nonlinear terms. Additionally, it can be extended to address certain fully nonlinear equations, such as the Monge-Ampère equation, which emerges in the context of the L^2 optimal transport (OT) problem (Villani, 2021; De Philippis and Figalli, 2014). Since the OT problem can be formulated as a constrained optimization problem, introducing the Lagrange multiplier method leads to a saddle point scheme. This scheme involves adversarial training with the pushforward map and the dual potential function to solve the Monge-Ampère equation, substituting both the map and potential function with neural network approximations and applying the NPDG algorithm with precondition matrices. The L^2 Gram type matrices lead to stable and efficient numerical results. Further analysis around the saddle point of the loss function suggests a more canonical preconditioning approach, where the mapping still uses the L^2 Gram type matrix while the potential uses the H^1 Gram type matrix. For a detailed discussion, readers are referred to Section 2.5.

In this research, we provide an *a posteriori* convergence analysis for the time-continuous version of the NPDG algorithm when applied to the general linear PDE (1). Let $(\theta_t, \eta_t, \xi_t)$ be the solution obtained from the time-continuous algorithm for $0 \le t \le T$. Under specific conditions regarding the approximation capabilities of the tangent spaces spanned by $\{\partial_{\theta_k} u_{\theta_t}\}, \{\partial_{\eta_k} \varphi_{\eta_t}\}$, and $\{\partial_{\xi_k} \psi_{\xi_t}\}$, we establish the linear convergence of the numerical solution u_{θ_t}

$$\|\mathcal{M}_p(u_{\theta_t} - u_*)\|^2_{L^2(\Omega)} + \lambda \|\mathcal{B}(u_{\theta_t} - u_*)\|^2_{L^2(\partial\Omega)} \le C_0 \cdot \exp(-rt) \quad \text{for } 0 \le t \le T.$$

Here $C_0 > 0$ denotes the initial error, $r > 0$ is the convergence rate depending on the preconditioned operator $\widetilde{\mathcal{L}}$, the hyperparameters of the NPDG algorithm, and the neural network parameters. An explicit lower bound for r in the case where u_θ, φ_η, and ψ_ξ are

linear combinations of basis functions is provided in (42). A fast convergence rate of the time-continuous algorithm can be expected when the operator $\widetilde{\mathcal{L}}$ is well-conditioned and the hyperparameters are appropriately chosen.

We further refine the above convergence analysis and remove the restriction on finite time horizon $[0, T]$ for an important class of elliptic equations in divergence form. By incorporating a boundary loss term stronger than the standard $L^2(\partial\Omega)$ norm, we establish an *a priori* convergence bound in the setting where u_θ, φ_η, and ψ_ξ are linear combinations of prescribed basis functions $\{u_k\}$, $\{\varphi_k\}$, and $\{\psi_k\}$, respectively. More precisely, we have

$$\Big(\|\nabla u_{\theta_t} - \nabla u_*\|_{L^2}^2 + \lambda\|u_{\theta_t} - u_*\|_{\mathcal{X}}^2\Big)^{\frac{1}{2}} \;\le\; C_0 e^{-rt} + \frac{1 - e^{-rt}}{r}\Big(C_1\sqrt{\mathcal{E}_u} + C_2\sqrt{\mathcal{E}_{\nabla\varphi}} + C_3\sqrt{\mathcal{E}_\psi}\Big).$$

Here, $\|\cdot\|_{\mathcal{X}}$ denotes a boundary norm stronger than $L^2(\partial\Omega)$, such as $H^{1/2}(\partial\Omega)$ or $H^1(\partial\Omega)$. The constant C_0 depends on the initial error, while C_1, C_2, and C_3 are coefficients determined by the elliptic operator and the choice of hyperparameters. The quantities $\mathcal{E}_u$, $\mathcal{E}_{\nabla\varphi}$, and $\mathcal{E}_\psi$ represent the approximation errors of the chosen basis functions in approximating the exact solution u_* and the boundary data g. Moreover, the convergence rate r admits a uniform lower bound $r \ge \frac{2}{3\sqrt{3}}$ under a suitable selection of hyperparameters. This result reveals a clear two-phase behavior of the dynamics: at early stages, the numerical error is dominated by the decay of the initial error, while in the long-time regime, the error is governed by the intrinsic approximation of the chosen basis functions. The readers are referred to Section 3 for detailed discussions on this series of results.

In implementation, we apply the Monte-Carlo algorithm to approximate $\mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$; we use automatic differentiation to compute the derivatives of $\mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$ with respect to the parameters θ, η, ξ. It is usually prohibitively expensive to explicitly form the precondition matrices $M_p(\theta), M_d(\eta), M_{bdd}(\xi)$ given that m_θ, m_η, m_ξ might be very large. To cope with this, we evaluate the pseudo-inverse in (4) via the iterative solver such as the Minimal residual method (MINRES) (Paige and Saunders, 1975), which, instead of forming entire matrices, only requires matrix-vector multiplication. Further details of the implementation can be found in Section 4.

Numerical examples of linear PDEs (1), nonlinear PDEs (26), and Monge-Ampère equations (27) in Section 5 illustrate the accuracy, efficiency, and robustness of the NPDG method compared to classical methods, including the Physics-Informed Neural Network (PINN), the DeepRitz method, and the Weak Adversarial Network (WAN). Based on these numerical results, the algorithm demonstrates linear convergence for the high-dimensional PDEs tested in this section. Additionally, the proposed method converges more efficiently and achieves higher accuracy in both L^2 and H^1 norms compared to the other tested methods.

1.1 Related references

In recent years, machine learning algorithms have attracted increasing attention from the scientific computing community due to their flexibility and scalability. A considerable amount of these investigations are based on the Physics-Informed Neural Network (PINN) algorithm (Raissi et al., 2019; Lu et al., 2021a); further approaches that address the pathologies during PINN training include calibration of interior-boundary loss coefficients (Wang et al., 2022a), and variable splitting techniques (Basir, 2022; Park et al., 2024). The adaptive sampling

methods (Tang et al., 2022, 2023) are introduced to gain better accuracy of the neural network approximation. In addition to PINNs, a range of deep learning-based algorithms is introduced for solving various types of PDEs, demonstrating scalability to high-dimensional problems. These include the Deep Galerkin Method (Sirignano and Spiliopoulos, 2018), Deep Ritz method (Yu et al., 2018; Lu et al., 2021b; Liu et al., 2023a), Forward-Backward Stochastic Differential Equation (FBSDE) approaches (Han et al., 2017, 2018; Hutzenthaler et al., 2021), Extreme Learning Machines (Dong and Li, 2021; Ni and Dong, 2023; Wang and Dong, 2024), Tensor Neural Networks (Wang et al., 2022b, 2024), etc.

Recent research trends leverage adversarial training strategies (Goodfellow et al., 2020; Arjovsky et al., 2017) to improve algorithm performance. In the Weak Adversarial Network (WAN) algorithm, discriminator neural networks are used to enhance training efficiency by employing the weak formulation of PDEs (Zang et al., 2020; Bao et al., 2020). The weak formulation is further employed in (Cai et al., 2024) to train a generative model for generating samples from the invariant measure of stochastic dynamics. Additionally, a residual-attention-based approach has been introduced in (McClenny and Braga-Neto, 2020, 2023; Anagnostopoulos et al., 2023; Zeng et al., 2022) for seeking numerical solutions with higher precision.

The Primal-Dual Hybrid Gradient (PDHG) method, which is widely used in image processing problems (Zhu and Chan, 2008; Chambolle and Pock, 2011), has been introduced to handle nonlinear PDEs on classical numerical schemes (Liu et al., 2023b, 2024a; Meng et al., 2023). Suitable preconditioning is introduced to improve the convergence of the algorithm significantly. The method is shown to converge linearly in (Liu et al., 2025).

Large-scale optimization algorithms play a crucial role in machine learning research. Stochastic gradient descent (SGD) is a widely used first-order optimization method (Robbins and Monro, 1951; Saad, 1998; Bottou and Bousquet, 2008). One can improve the SGD's performance by incorporating momentum terms (Rumelhart et al., 1986; Nesterov, 1983; Su et al., 2016). Various modified versions of SGD with per-parameter learning rates—such as AdaGrad (Duchi et al., 2011), Adadelta (Zeiler, 2012), RMSProp (Tieleman and Hinton, 2012), and Adam (Kingma, 2014)—are popular optimizers in deep learning (Paszke et al., 2019). Additionally, second-order algorithms like the BFGS method (Fletcher, 2000), LBFGS method (Liu and Nocedal, 1989), and inexact-Newton methods (Dembo et al., 1982; Brown and Saad, 1990, 1994; Eisenstat and Walker, 1994; Martens, 2010; Roosta et al., 2022; Rathore et al., 2024) are also widely explored in machine learning research.

The natural gradient method is another critical category of second-order optimizers, initially introduced in (Amari, 1998) with further developments in (Amari, 2016; Thomas et al., 2016; Song et al., 2018). An efficient, scalable variant known as the K-FAC (Kronecker-factored Approximate Curvature) method was proposed in (Martens and Grosse, 2015). The natural gradient method finds its application under different scenarios, including optimization involving combined loss functionals (Ying, 2021), PDE-constrained optimization (Nurbekyan et al., 2023), simulation and acceleration of Wasserstein gradient flows (Li and Montufar, 2018; Chen and Li, 2018; Wang and Li, 2020; Shen et al., 2020; Liu et al., 2022). A series of research that utilizes the concept of the natural gradient to solve general time-dependent PDEs have been conducted, as detailed in (Du and Zaki, 2021; Bruna et al., 2024; Gaby et al., 2023; Chen et al., 2024) and the references therein. Recently, a natural gradient primal-dual algorithm for decentralized learning problems is proposed in (Niwa et al., 2024).

The natural gradient algorithm has recently been applied to training PINNs, achieving highly accurate solutions (Müller and Zeinhofer, 2023). The K-FAC method is exploited in the follow-up work (Dangel et al., 2024) to enable scalability in high-dimensional settings. Beyond natural gradients, the Gauss-Newton method has been introduced in (Hao et al., 2024) for computing variational PDEs. Additional preconditioning techniques for solving PDEs include the multigrid-augmented method (Azulay and Treister, 2022), domain decomposition strategies (Kopaničáková et al., 2024), and incomplete LU preconditioning (Liu et al., 2024b). However, these methods typically need to scale more effectively to compute high-dimensional problems.

Compared to these methods, we summarize the advantages of the proposed approach in two key aspects: the primal-dual hybrid gradient algorithmic framework and the application of natural gradients in neural network functions.

- On the primal-dual framework:
  - By applying integration by parts, we reduce the order of the differential operator $\mathcal{L}$ in the primal-dual formulation, lowering computational complexity when performing automatic differentiation on the neural networks.
  - The proposed primal–dual training scheme is versatile and adaptable, rendering the algorithm applicable to a broad class of partial differential equations, including linear elliptic problems, semi-linear equations with dominant viscosity terms, fully nonlinear PDEs such as the Monge–Ampère equation, etc.
- On the primal-dual hybrid natural gradients:
  - Unlike other second-order optimization algorithms, such as L-BFGS, which are unable to handle the training involving random batches, the proposed algorithm is well-suited to data stochasticity, performing robustly under stochastic approximation.
  - To address the computation of large-scale linear systems (specifically, the pseudo-inverse of preconditioning matrices), we introduce the iterative method (MINRES). Consequently, our approach readily accommodates high-dimensional PDEs requiring neural networks with a large number of parameters. In experiments, we handle neural networks with parameter counts ranging from 20,000 to 300,000.

Generally, the proposed algorithm converges smoothly, avoiding the intense fluctuations and spikes commonly observed in the loss decay curves of classical momentum-based optimizers such as Adam and RMSProp. With appropriate preconditioning, theoretical analysis (Theorem 7, Theorem 8) indicates linear convergence of the method. In practice, the approach performs more efficiently than classical machine learning methods and achieves higher precision in the norms L^2 and H^1. Furthermore, as reflected in later Table 4, the method demonstrates robustness with respect to its hyperparameters, including the regularization coefficient ν, step sizes τ_φ, τ_u, and the extrapolation coefficient ω. Typically, a standard configuration of $\nu = 0.1, \tau_\varphi = 0.095$, $\tau_u = 0.05$, and $\omega = 1$ yields satisfactory performance.

This paper is organized as follows. In Section 2, we provide a detailed derivation of the algorithm. Supplementary discussions on treating the semi-linear PDEs and the Monge-Ampère equations are provided in Section 2.5. Then, in Section 3, we establish a series

of convergence analysis results for the time-continuous version of the algorithm. Implementation details are demonstrated in Section 4. We demonstrate the numerical examples in Section 5. We provide further materials related to the algorithm, proof, and numerical examples in the Appendix.

2. Derivation of Natural Primal-Dual Hybrid Gradient (NPDG) method

In this section, we provide a detailed derivation of the proposed method by first introducing the Primal-Dual Hybrid Gradient algorithm for root-finding problems. We then apply this algorithm to solving PDEs in the functional space. We improve the algorithm's performance by introducing suitable preconditioning. Finally, we discuss how we realize the algorithm by substituting the functions with neural networks and introduce the Natural Primal-Dual Hybrid Gradient (NPDG) algorithm for adversarial training of the neural networks for solving PDEs.

2.1 Primal-Dual algorithm for root-finding problem

We first consider a root-finding problem defined on Hilbert space $\mathbb{X}$,

$$\mathcal{F}(x) = 0.$$

Here, we assume that $\mathcal{F} : \mathbb{X} \to \mathbb{Y}$ is a function from $\mathbb{X}$ to another Hilbert space $\mathbb{Y}$. The goal is to find a solution $x \in \mathbb{X}$. For a certain convex functional $\iota : \mathbb{Y} \to \mathbb{R}$ that satisfies $\iota(y) > 0$ iff $y \neq 0$ and $\iota(y) = 0$ whenever $y = 0$, the root-finding problem is equivalent to the following minimization problem

$$\inf_{x\in\mathbb{X}} \ \iota(\mathcal{F}(x)). \tag{5}$$

We denote the Legendre dual of $\iota(\cdot)$ as $\iota^*(\cdot)$ which is defined as $\iota^*(y) = \sup_{w\in\mathbb{Y}} \langle y, w\rangle_{\mathbb{Y}} - \iota(w)$. Here, we denote $\langle\cdot,\cdot\rangle_{\mathbb{Y}}$ as the inner product defined in the space $\mathbb{Y}$. Then

$$\iota(z) = \iota^{**}(z) = \sup_{y\in\mathbb{Y}} \ \langle z, y\rangle_{\mathbb{Y}} - \iota^*(y). \tag{6}$$

Substituting (6) into (5) yields the following saddle point problem

$$\inf_{x\in\mathbb{X}} \ \sup_{y\in\mathbb{Y}} \ \mathscr{E}(x, y) := \langle \mathcal{F}(x), y\rangle_{\mathbb{Y}} - \iota^*(y). \tag{7}$$

We now apply the PDHG algorithm to deal with the inf-sup problem (7), yielding

$$y_{n+1} = \operatorname*{argmin}_{y\in\mathbb{Y}} \ \frac{\|y - y_n\|_{\mathbb{Y}}^2}{2\tau_y} - \mathscr{E}(x_n, y) = \left(\mathrm{Id} - \tau_y D_y\mathscr{E}(x_n, \cdot)\right)^{-1} \ y_n, \tag{8}$$

$$\widetilde{y}_{n+1} = y_{n+1} + \omega(y_{n+1} - y_n),$$

$$x_{n+1} = \operatorname*{argmin}_{x\in\mathbb{X}} \ \frac{\|x - x_n\|_{\mathbb{X}}^2}{2\tau_x} + \mathscr{E}(x, \widetilde{y}_{n+1}) = \left(\mathrm{Id} + \tau_x D_x\mathscr{E}(\cdot, \widetilde{y}_{n+1})\right)^{-1} \ x_n. \tag{9}$$

Here $\tau_x, \tau_y > 0$ are the step sizes of the PDHG algorithm, $D_x\mathscr{E} \in \mathbb{X}, D_y\mathscr{E} \in \mathbb{Y}$ are the Fréchet derivatives and $\omega > 0$ denotes the extrapolation coefficient. The proximal steps (8), (9) can

be interpreted as the implicit update of the gradient ascent/descent algorithm of functional $\mathscr{E}$ as τ_x, τ_y are small enough. In practice, one can choose $\iota(\cdot) = \chi(\cdot)$, where χ is the indicator function defined as $\chi(y) = +\infty$ for $y \neq 0$ and $\chi(0) = 0$. In this case, the Legendre dual satisfies $\iota^*(\cdot) \equiv 0$. Another popular choice is $\iota(\cdot) = \frac{1}{2\nu}\|\cdot\|_{\mathbb{Y}}^2$ with $\iota^*(\cdot) = \frac{\nu}{2}\|\cdot\|_{\mathbb{Y}}^2$. Here, $\nu > 0$ is a tunable hyperparameter. We will mainly focus on the latter throughout the subsequent discussion of the paper.

2.2 Primal-Dual Hybrid Gradient algorithm for solving PDEs

From now on, we assume that $\Omega \subset \mathbb{R}^d$ is a bounded open set. Let us start by considering a linear equation defined on a Hilbert space $\mathbb{H}$,

$$\mathcal{L}u = f \text{ on } \Omega, \quad \text{with boundary condition } \mathcal{B}u = g \text{ on } \partial\Omega. \tag{10}$$

We denote $\mathbb{K} \subseteq L^2(\Omega)$, $\mathbb{K}_{\partial\Omega} \subseteq L^2(\partial\Omega)$ as two Hilbert spaces. Then, $\mathcal{L} : \mathbb{H} \to \mathbb{K} \subseteq L^2(\Omega)$ is a linear differential operator, $\mathcal{B} : \mathbb{H} \to \mathbb{K}_{\partial\Omega} \subseteq L^2(\partial\Omega)$ is a linear boundary operator. We assume $u_* \in C^2(\Omega) \cap C(\bar{\Omega}) \subset \mathbb{H}$ to be the classical solution to (10).

We now set $\mathcal{F} : \mathbb{H} \to \mathbb{K} \times \mathbb{K}_{\partial\Omega}, u \mapsto (\mathcal{L}u - f, \mathcal{B}u - g)$. By introducing the test variables $\varphi \in \mathbb{K}^{test} \subseteq L^2(\Omega)$ and $\psi \in \mathbb{K}_{\partial\Omega}^{test} \subseteq L^2(\partial\Omega)$, and by defining $\mathbb{L}^2 := L^2(\Omega) \times L^2(\partial\Omega)$, we arrive at the following saddle-point problem:

$$\begin{aligned}\inf_{u\in\mathbb{H}} \sup_{\substack{\varphi\in\mathbb{K}^{test}\\ \psi\in\mathbb{K}_{\partial\Omega}^{test}}} \mathscr{E}_0(u,\varphi,\psi) :=& \langle \mathcal{F}(u), (\varphi,\psi)\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\|(\varphi,\psi)\|_{\mathbb{L}^2}^2. \\ =& \langle \mathcal{L}u - f, \varphi\rangle_{L^2(\Omega)} - \frac{\nu}{2}\|\varphi\|_{L^2(\Omega)}^2 + \langle \mathcal{B}u - g, \psi\rangle_{L^2(\partial\Omega)} - \frac{\nu}{2}\|\psi\|_{L^2(\partial\Omega)}^2.\end{aligned} \tag{11}$$

It is not hard to verify that $u = u_*, \varphi = 0, \psi = 0$ form the saddle point of the inf-sup problem (11). We refer to (Huo and Liu, 2024) for further discussion of the saddle point structure of related inf-sup formulations. In practice, it is usually convenient to introduce a boundary loss coefficient $\lambda > 0$ and consider[1],

$$\mathscr{E}_0(u,\varphi,\psi) = \langle \mathcal{L}u - f, \varphi\rangle_{L^2(\Omega)} - \frac{\nu}{2}\|\varphi\|_{L^2(\Omega)}^2 + \lambda(\langle \mathcal{B}u - g, \psi\rangle_{L^2(\partial\Omega)} - \frac{\nu}{2}\|\psi\|_{L^2(\partial\Omega)}^2).$$

In this work, we propose the following PDHG algorithm to deal with inf-sup problem (11),

$$\begin{aligned}\begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix} &= \operatorname*{argmin}_{(\varphi,\psi)\in\mathbb{K}^{test}\times\mathbb{K}_{\partial\Omega}^{test}} \left\{\frac{1}{2\tau_\varphi}(\|\varphi-\varphi_n\|_{L^2(\Omega)}^2 + \|\psi-\psi_n\|_{L^2(\partial\Omega)}^2) - \mathscr{E}_0(u_n,\varphi,\psi)\right\},\\ \begin{bmatrix}\widetilde{\varphi}_{n+1}\\ \widetilde{\psi}_{n+1}\end{bmatrix} &= \begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix} + \omega\left(\begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix} - \begin{bmatrix}\varphi_n\\ \psi_n\end{bmatrix}\right),\\ u_{n+1} &= \operatorname*{argmin}_{u\in\mathbb{H}} \left\{\frac{1}{2\tau_u}(\|u-u_n\|_{L^2(\Omega)}^2 + \|\mathcal{B}u-\mathcal{B}u_n\|_{L^2(\partial\Omega)}^2) + \mathscr{E}_0(u,\widetilde{\varphi}_{n+1},\widetilde{\psi}_{n+1})\right\}.\end{aligned} \tag{12}$$

To develop an intuitive understanding of why the algorithm (12) has difficulties in approaching the PDE solution, we consider the square region Ω and discretize it into N_x^d lattices.

1. The new functional is obtained by considering root-finding problem $\mathcal{F}_\lambda(u) = 0$ with $\mathcal{F}_\lambda : u \mapsto (\mathcal{L}u - f, \sqrt{\lambda}(\mathcal{B}u - g))$, and setting $\mathscr{E}_0(u,\varphi,\psi) := \langle \mathcal{F}_\lambda(u), (\varphi, \sqrt{\lambda}\psi)\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\|(\varphi, \sqrt{\lambda}\psi)\|_{\mathbb{L}^2}^2$.

We apply the finite difference scheme to discretize (10) into grids. Solving the PDE yields a linear equation $Ax - b = 0$. Here, A is the matrix obtained upon discretizing $\mathcal{L}$. Roughly speaking, $A \in \mathbb{R}^{N_x^d \times N_x^d}$ is self-adjoint and non-singular, $x \in \mathbb{R}^{N_x^d}$ denotes the numerical solution of the PDE on the grid points, $b \in \mathbb{R}^{N_x^d}$ is the vector encoding f and its boundary condition. The proposed PDHG algorithm yields

$$\begin{aligned} y_{n+1} =& \operatorname*{argmin}_y \frac{\|y - y_n\|^2}{2\tau_y} - (Ax_n - b)^\top y, \\ \widetilde{y}_{n+1} =& 2y_{n+1} - y_n, \\ x_{n+1} =& \operatorname*{argmin}_x \frac{\|x - x_n\|^2}{2\tau_x} + (Ax - b)^\top \widetilde{y}_{n+1}. \end{aligned}$$

Here, we set $\nu = 0$ and $\omega = 1$ to simplify the discussion. And $\|\cdot\|$ denotes the ℓ_2 norm of $\mathbb{R}^N$. It is not hard to verify that the above algorithm is equivalent to the following update:

$$\begin{bmatrix} \widehat{x}_{n+1} \\ y_{n+1} \end{bmatrix} = \underbrace{\begin{bmatrix} I - 2\tau_x\tau_y A^\top A & -\tau_x A^\top \\ \tau_y A & I \end{bmatrix}}_{\text{denote as } \Gamma} \begin{bmatrix} \widehat{x}_n \\ y_n \end{bmatrix}.$$

Here, we denote x_* as the solution to $Ax - b = 0$ and $\widehat{x}_n = x_n - x_*$. The convergence rate of the PDHG algorithm depends on the spectrum radius $\rho(\Gamma)$ of Γ. The value of $\rho(\Gamma)$ equals $\sqrt{1 - \frac{c}{\kappa^2}}$, where $c \in [1, \frac{4}{3})$ and κ denotes the condition number of A (we refer readers to Theorem 1 in (Liu et al., 2024a) for a detailed discussion). For Laplace operator $\mathcal{L} = \Delta$, the matrix A obtained via central difference scheme takes condition number $\kappa = \mathcal{O}(N_x^2)$ (Kulkarni et al., 1999). This indicates that the convergence rate of the PDHG method is $\sqrt{1 - \frac{c}{\kappa^2}} = 1 - \mathcal{O}(\frac{1}{N_x^4})$, which is very inefficient as N_x increases.

2.3 Preconditioning of the primal-dual algorithm

The discussion in Section 1 suggests that we should introduce preconditioning to the original algorithm (12). As mentioned previously, we assume that $\mathcal{L}$ admits the splitting $\mathcal{L} = \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p$, where $\mathcal{M}_d^*$, $\widetilde{\mathcal{L}}$, and $\mathcal{M}_p$ are linear differential operators acting between the functional spaces specified below.

$$\mathbb{H} \xrightarrow{\mathcal{M}_p} \widetilde{\mathbb{H}} \xrightarrow{\widetilde{\mathcal{L}}} \widetilde{\mathbb{K}} \xrightarrow{\mathcal{M}_d^*} \mathbb{K} \subseteq L^2(\Omega)$$

$$L^2(\Omega; \mathbb{R}^r) \supseteq \widetilde{\mathbb{K}}^{test} \xleftarrow{\mathcal{M}_d} \mathbb{K}^{test} \subseteq L^2(\Omega)$$

Here we assume $\widetilde{\mathbb{H}}, \widetilde{\mathbb{K}} \subseteq L^2(\Omega; \mathbb{R}^r)$ are Hilbert spaces. Moreover, $\mathcal{M}_d : \mathbb{K}^{test} \to \widetilde{\mathbb{K}}^{test}$ is a linear operator. $\mathcal{M}_d^*$ is treated as the "adjoint" of $\mathcal{M}_d$ in the sense of

$$\langle \mathcal{M}_d^* \mathbf{u}, \varphi \rangle_{L^2(\Omega)} = \langle \mathbf{u}, \mathcal{M}_d \varphi \rangle_{L^2(\Omega; \mathbb{R}^r)}, \quad \forall\ \mathbf{u} \in \widetilde{\mathbb{K}},\ \varphi \in \mathbb{K}^{test}.$$

Now recall that $u_* \in \mathbb{H}$ is the solution to (10). For any $u \in \mathbb{H}, \varphi \in \mathbb{K}^{test}$, we have

$$\begin{aligned} \langle \mathcal{L}u - f, \varphi \rangle_{L^2(\Omega)} = \langle \mathcal{L}(u - u_*), \varphi \rangle_{L^2(\Omega)} &= \langle \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p (u - u_*), \varphi \rangle_{L^2(\Omega)} \\ &= \langle \widetilde{\mathcal{L}}\ \mathcal{M}_p (u - u_*), \mathcal{M}_d \varphi \rangle_{L^2(\Omega; \mathbb{R}^r)}. \end{aligned} \tag{13}$$

Example 1 *Taking the negative Laplace operator $\mathcal{L} = -\Delta$ as an example, by setting $\mathbb{H} = H^2(\Omega), \widetilde{\mathbb{H}} = \widetilde{\mathbb{K}} = H^1(\Omega, \mathbb{R}^d), \mathbb{K} = L^2(\Omega)$, and $\mathbb{K}^{test} = H_0^1(\Omega), \widetilde{\mathbb{K}}^{test} = L^2(\Omega; \mathbb{R}^d)$, and choosing $\mathcal{M}_d = \mathcal{M}_p = \nabla$ and $\widetilde{\mathcal{L}} = \mathrm{Id}$, we obtain*

$$\langle -\Delta u - f, \varphi\rangle_{L^2(\Omega)} = \langle -\Delta(u - u_*), \varphi\rangle_{L^2(\Omega)} = \langle \nabla(u - u_*), \nabla\varphi\rangle_{L^2(\Omega;\mathbb{R}^d)},$$

for any $u \in \mathbb{H} = H^2(\Omega), \varphi \in \mathbb{K}^{test} = H_0^1(\Omega)$.

Example 2 *Consider the elliptic operator $\mathcal{L} = \mathrm{Id} - \Delta$, where* Id *is an identity operator. By setting $\mathbb{H} = H^2(\Omega)$, $\widetilde{\mathbb{H}} = \widetilde{\mathbb{K}} = H^2(\Omega) \times H^1(\Omega; \mathbb{R}^d)$, $\mathbb{K} = L^2(\Omega)$, and $\mathbb{K}^{test} = H_0^1(\Omega)$, $\widetilde{\mathbb{K}}^{test} = H_0^1(\Omega) \times L^2(\Omega; \mathbb{R}^d)$, we can split the elliptic operator as*

$$\mathrm{Id} - \Delta = \begin{bmatrix} \mathrm{Id} & -\nabla\cdot \end{bmatrix} \begin{bmatrix} \mathrm{Id} & \\ & \mathrm{Id} \end{bmatrix} \begin{bmatrix} \mathrm{Id} \\ \nabla \end{bmatrix} = \begin{bmatrix} \mathrm{Id} \\ \nabla \end{bmatrix}^* \begin{bmatrix} \mathrm{Id} & \\ & \mathrm{Id} \end{bmatrix} \begin{bmatrix} \mathrm{Id} \\ \nabla \end{bmatrix} = \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p.$$

*Denoting (by abuse of notation) "*Id*" as the identity map on its corresponding space, we have*

$$\langle u - \Delta u - f, \varphi\rangle_{L^2(\Omega)} = \langle (\mathrm{Id} - \Delta)(u - u_*), \varphi\rangle_{L^2(\Omega)} = \left\langle \begin{bmatrix} u - u_* \\ \nabla(u - u_*) \end{bmatrix}, \begin{bmatrix} \varphi \\ \nabla\varphi \end{bmatrix} \right\rangle_{L^2(\Omega;\mathbb{R}^{1+d})}$$

for any $u \in H^2(\Omega), \varphi \in H_0^1(\Omega)$.

Further examples of linear equations with elliptic operators $\mathcal{L}$ belonging to divergence form $\mathcal{L} = -\nabla \cdot (\kappa(x)\nabla\,)$ with $\kappa \in C^1(\Omega)$ are given in Section 5.

Remark 1 ($\mathcal{L}$ of non-divergence form) *Consider a general second-order differential operator $\mathcal{L} = \sum_{i,j=1}^d a_{ij}(x) \frac{\partial^2}{\partial x_i \partial x_j}$ with $a_{ij}(\cdot) \in L^2(\Omega)$. The operator shows up as the generator of diffusion processes and acts as a fundamental role in a wide range of applications. It is not always tractable to split $\mathcal{L}$ into differential operators with lower orders as illustrated in Example 1 and 2. In such a case, one possible treatment is to set $\mathcal{M}_p = \mathcal{L}, \widetilde{\mathcal{L}} = \mathrm{Id}, \mathcal{M}_d = \mathrm{Id}$ with $\mathbb{H} = H^2(\Omega), \widetilde{\mathbb{H}} = \widetilde{\mathbb{K}} = \mathbb{K} = L^2(\Omega)$ and apply the algorithm. However, in the current work, we will mainly focus on elliptic equations of the divergence form and leave the non-divergence cases for future investigation.*

Similar to (13), recall that $\mathcal{B}$ is a linear boundary operator, for any $u \in \mathbb{H}, \psi \in \mathbb{K}_{\partial\Omega}^{test}$, we have

$$\langle \mathcal{B}u - g, \psi\rangle_{L^2(\partial\Omega)} = \langle \mathcal{B}(u - u_*), \psi\rangle_{L^2(\partial\Omega)}.$$

As mentioned in Section 1, we shall substitute u, φ, ψ in the proximal steps of (12) with $(\mathcal{M}_p(u - u_*), \sqrt{\lambda}\mathcal{B}(u - u_*))$ and $\mathcal{M}_d\varphi$, $\sqrt{\lambda}\psi$. Correspondingly, we use the following modified functional $\mathscr{E} : \mathbb{H} \times \mathbb{K}^{test} \times \mathbb{K}_{\partial\Omega}^{test} \to \mathbb{R}$ (we denote $L^2 := L^2(\Omega)$, $L^2_{\partial\Omega} := L^2(\partial\Omega)$ for simplicity),

$$\begin{aligned}
\mathscr{E}(u, \varphi, \psi) =& \langle \widetilde{\mathcal{L}}\mathcal{M}_p(u - u_*), \mathcal{M}_d\varphi\rangle_{L^2} + \lambda\langle \mathcal{B}(u - u_*), \psi\rangle_{L^2_{\partial\Omega}} - \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|_{L^2}^2 + \lambda\|\psi\|_{L^2}^2) \\
=& \langle \widetilde{\mathcal{L}}\mathcal{M}_p u, \mathcal{M}_d\varphi\rangle_{L^2} - \langle \mathcal{L}u_*, \varphi\rangle_{L^2} + \lambda\langle \mathcal{B}(u - u_*), \psi\rangle_{L^2_{\partial\Omega}} - \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|_{L^2}^2 + \lambda\|\psi\|_{L^2}^2) \\
=& \langle \widetilde{\mathcal{L}}\mathcal{M}_p u, \mathcal{M}_d\varphi\rangle_{L^2} - \langle f, \varphi\rangle_{L^2} + \lambda\langle \mathcal{B}u - g, \psi\rangle_{L^2_{\partial\Omega}} - \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|_{L^2}^2 + \lambda\|\psi\|_{L^2}^2).
\end{aligned} \tag{14}$$

We then consider the inf-sup problem

$$\inf_{u\in\mathbb{H}} \sup_{\varphi\in\mathbb{K}^{test},\, \psi\in\mathbb{K}^{test}_{\partial\Omega}} \mathscr{E}(u,\varphi,\psi). \tag{15}$$

The following theorem proves the consistency between the solution to this inf-sup problem and the solution to the PDE (10).

Theorem 2 (Consistency) *Assume the test spaces $\mathbb{K}^{test}, \mathbb{K}^{test}_{\partial\Omega}$ are dense in the spaces $L^2(\Omega,\mu)$, $L^2(\partial\Omega,\mu_{\partial\Omega})$, respectively. Suppose that $(\widehat{u},\widehat{\varphi},\widehat{\psi})\in\mathbb{H}\times\mathbb{K}^{test}\times\mathbb{K}^{test}_{\partial\Omega}$ is a solution to the inf-sup problem* (15)*. Then $\widehat{u}$ is a strong solution to* (10) *in the sense that $\mathcal{L}u-f=0$, almost everywhere (a.e.) on Ω and $\mathcal{B}u=g$, a.e. on $\partial\Omega$.*

The proof of the theorem is provided in Appendix B. As long as (10) admits a unique strong solution, the function $\widehat{u}$ must coincide with the classical solution u_*.

To seek for the solution of the inf-sup problem (15), we treat $(\mathcal{M}_p(u-u_*),\sqrt{\lambda}\mathcal{B}(u-u_*))$, together with $(\mathcal{M}_d\varphi,\sqrt{\lambda}\psi)$, as the *new* primal and dual variables of the algorithm. By doing so, we substitute $(u,\mathcal{B}u)$, (φ,ψ) in the proximal steps (the 1st and the 3rd line) of (12) with $(\mathcal{M}_p(u-u_*),\sqrt{\lambda}\mathcal{B}(u-u_*)),(\mathcal{M}_d\varphi,\sqrt{\lambda}\psi)$. Therefore, we come up with the following preconditioned version of the PDHG algorithm. This treatment is also known as the G-prox PDHG algorithm introduced in (Jacobs et al., 2019).

$$\begin{aligned}
&\begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix} = \operatorname*{argmin}_{\substack{\varphi\in\mathbb{K}^{test}\\ \psi\in\mathbb{K}^{test}_{\partial\Omega}}} \left\{\frac{1}{2\tau_\varphi}(\|\mathcal{M}_d\varphi-\mathcal{M}_d\varphi_n\|^2_{L^2(\Omega)}+\lambda\|\psi-\psi_n\|^2_{L^2(\partial\Omega)})-\mathscr{E}(u_n,\varphi,\psi)\right\},\\
&\begin{bmatrix}\widetilde{\varphi}_{n+1}\\ \widetilde{\psi}_{n+1}\end{bmatrix} = \begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix} + \omega\left(\begin{bmatrix}\varphi_{n+1}\\ \psi_{n+1}\end{bmatrix}-\begin{bmatrix}\varphi_{n}\\ \psi_{n}\end{bmatrix}\right),\\
&u_{n+1} = \operatorname*{argmin}_{u\in\mathbb{H}} \left\{\frac{1}{2\tau_u}(\|\mathcal{M}_p u-\mathcal{M}_p u_n\|^2_{L^2(\Omega)}+\lambda\|\mathcal{B}u-\mathcal{B}u_n\|^2_{L^2(\partial\Omega)})+\mathscr{E}(u,\widetilde{\varphi}_{n+1},\widetilde{\psi}_{n+1})\right\},
\end{aligned} \tag{16}$$

2.4 Natural Primal-Dual Hybrid Gradient (NPDG) algorithm for neural networks

At the beginning of this section, we briefly introduce the idea of the Natural Gradient method (Amari, 1998, 2016; Martens, 2020).

2.4.1 Natural Gradient method

For a wide range of machine learning problems, we assume that the loss function $J(\theta)=\mathscr{J}(u_\theta)$ where $\mathscr{J}:\mathbb{U}\to\mathbb{R}$ denotes the loss functional and u_θ is the parametrized function on the metric space $\mathbb{U}$ with the parameter $\theta\in\Theta\subset\mathbb{R}^m$ to be determined. The essential idea of the natural gradient algorithm is to conduct gradient descent on u_θ as an entity in the functional space rather than on the parameter θ. This can be realized by considering the proximal algorithm

$$\inf_{u_\theta\in\mathbb{U}} \frac{d^2(u_\theta,u_{\theta^n})}{2\tau}+J(\theta). \tag{17}$$

The preconditioning matrix $G(\theta)$ can thus be obtained by investigating the infinitesimal distance $d^2(u_\theta, u_{\theta^n}) \approx (\theta-\theta^n)^\top G(\theta)(\theta-\theta^n)$, where $d(\cdot,\cdot)$ is a distance function enriches the Hessian information of the loss functional $\mathscr{J}$. By sending $\tau \to 0$, the implicit scheme (17) reduces to the natural gradient flow

$$\dot{\theta}_t = -G(\theta)^{-1}\nabla_\theta J(\theta).$$

As a result, viewing from the parameter space, the natural gradient algorithm can be realized by applying $G(\theta)$-preconditioned gradient descent steps to loss function $J(\theta)$. We refer the interested readers to (Amari, 2016) for a comprehensive illustration of the Natural Gradient methods.

Let us continue the discussion on the derivation of NPDG algorithm. We substitute $u(\cdot), \varphi(\cdot), \psi(\cdot)$ with neural networks $u_\theta(\cdot), \varphi_\eta(\cdot)$ and $\psi_\xi(\cdot)$ with tunable parameters $\theta \in \Theta_\theta \subseteq \mathbb{R}^{m_\theta}, \eta \in \Theta_\eta \subseteq \mathbb{R}^{m_\eta}, \xi \in \Theta_\xi \subseteq \mathbb{R}^{m_\xi}$. Here, we assume that the parameter spaces $\Theta_\theta, \Theta_\eta, \Theta_\xi$ are open sets of the Euclidean space. Then, algorithm (16) becomes

$$\begin{aligned}
&\begin{bmatrix} \eta^{n+1} \\ \xi^{n+1} \end{bmatrix} = \underset{\substack{\eta\in\mathbb{R}^{m_\eta}\\ \xi\in\mathbb{R}^{m_\xi}}}{\operatorname{argmin}} \left\{ \frac{1}{2\tau_\varphi}(\|\mathcal{M}_d\varphi_\eta - \mathcal{M}_d\varphi_{\eta^n}\|^2_{L^2(\Omega)} + \lambda\|\psi_\xi - \psi_{\xi^n}\|^2_{L^2(\partial\Omega)}) - \mathscr{E}(u_{\theta^n}, \varphi_\eta, \psi_\xi) \right\},\\
&\begin{bmatrix} \widetilde{\varphi}_{n+1} \\ \widetilde{\psi}_{n+1} \end{bmatrix} = \begin{bmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{bmatrix} + \omega \left(\begin{bmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{bmatrix} - \begin{bmatrix} \varphi_{\eta^n} \\ \psi_{\xi^n} \end{bmatrix} \right),\\
&\theta^{n+1} = \underset{\theta\in\mathbb{R}^{m_\theta}}{\operatorname{argmin}} \left\{ \frac{1}{2\tau_u}(\|\mathcal{M}_p u_\theta - \mathcal{M}_p u_{\theta^n}\|^2_{L^2(\Omega)} + \lambda\|\mathcal{B}u_\theta - \mathcal{B}u_{\theta^n}\|^2_{L^2(\partial\Omega)}) + \mathscr{E}(u_\theta, \widetilde{\varphi}_{n+1}, \widetilde{\psi}_{n+1}) \right\}.
\end{aligned} \tag{18}$$

Let us take a closer look at the first line of (18). Since φ and ψ are separable in $\mathscr{E}$, it is not hard to verify that the updating rule of η^{n+1} can be formulated as

$$\eta^{n+1} = \underset{\eta\in\mathbb{R}^{m_\eta}}{\operatorname{argmin}} \left\{ \frac{1}{2\tau_\varphi}\|\mathcal{M}_d\varphi_\eta - \mathcal{M}_d\varphi_{\eta^n}\|^2_{L^2(\Omega)} - \mathscr{E}(u_{\theta^n}, \varphi_\eta, \psi_\xi) \right\}. \tag{19}$$

As suggested in Section 1, we approximate $\mathcal{M}_d\varphi_\eta(x) - \mathcal{M}_d\varphi_{\eta^n}(x)$ with $\frac{\partial}{\partial\eta}\mathcal{M}_d\varphi_{\eta^n}(x)(\eta-\eta^n)$, where we denote $\frac{\partial}{\partial\eta}\mathcal{M}_d\varphi_{\eta_n}(x) \in \mathbb{R}^{r\times m_\eta}$ as the Jacobian matrix of $\mathcal{M}_d\varphi_\eta(\cdot)$ with respect to the parameter η at η^n. Therefore,

$$\begin{aligned}
\|\mathcal{M}_d\varphi_\eta - \mathcal{M}_d\varphi_{\eta^n}\|^2_{L^2(\Omega;\mathbb{R}^r)} &\approx \int_\Omega \|\frac{\partial}{\partial\eta}\mathcal{M}_d\varphi_{\eta^n}(x)(\eta-\eta^n)\|^2 \,\mathrm{d}\mu(x)\\
&= \sum_{i=1}^{m_\eta}\sum_{j=1}^{m_\eta} \left\langle \frac{\partial}{\partial\eta_i}\mathcal{M}_d\varphi_{\eta^n}, \frac{\partial}{\partial\eta_j}\mathcal{M}_d\varphi_{\eta^n} \right\rangle_{L^2(\Omega;\mathbb{R}^r)} (\eta_i-\eta_i^n)(\eta_j-\eta_j^n)\\
&= (\eta-\eta^n)^\top M_d(\eta^n)(\eta-\eta^n),
\end{aligned} \tag{20}$$

where we denote

$$M_d(\eta^n) = \int_\Omega \frac{\partial}{\partial\eta}\mathcal{M}_d\varphi_{\eta^n}(x)^\top \frac{\partial}{\partial\eta}\mathcal{M}_d\varphi_{\eta^n}(x) \,\mathrm{d}\mu(x), \tag{21}$$

as an $m_\eta \times m_\eta$ symmetric, positive semidefinite Gram matrix that encodes the information of $\mathcal{M}_d$.

Replacing the proximal term in (19) with the quadratic term (20) yields

$$\frac{1}{2\tau_\varphi}\Delta\eta^\top M_d(\eta^n)\Delta\eta - \widehat{E}(\theta^n, \eta^n + \Delta\eta, \xi^n). \tag{22}$$

Here we denote $\Delta\eta = \eta - \eta^n$ and $\widehat{E}(\theta, \eta, \xi) = \mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$ for shorthand. By linearizing $\widehat{E}(\theta^n, \eta, \xi^n)$ at $\eta = \eta^n$, the quantity (22) yields

$$\begin{aligned}
&\frac{1}{2}\Delta\eta^\top M_d(\eta^n)\Delta\eta - \tau_\varphi(\widehat{E}(\theta^n, \eta^n + \Delta\eta, \xi^n) - \widehat{E}(\theta^n, \eta^n, \xi^n)) \\
\approx &\frac{1}{2}\Delta\eta^\top M(\eta^n)\Delta\eta - \tau_\varphi \nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n)^\top \Delta\eta + \mathcal{O}(\tau_\varphi \|\Delta\eta\|^2).
\end{aligned}$$

We further omit the term $\mathcal{O}(\tau_\varphi\|\Delta\eta\|^2)$ to obtain the linearized version of (19)

$$\min_{\Delta\eta\in\mathbb{R}^{m_\eta}} \left\{ \frac{1}{2}\Delta\eta^\top M_d(\eta^n)\Delta\eta - \tau_\varphi \nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n)^\top \Delta\eta \right\}. \tag{23}$$

According to Lemma 6 from the next Section 3, we have $\nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n) \in \mathrm{Ran}(M_d(\eta^n))$. Therefore, the minimum value of (23) is finite. Recall that $M_d(\eta^n)^\dagger$ denotes the Moore-Penrose inverse of $M_d(\eta^n)$, an optimal solution to this least squares problem (23) can be denoted as

$$\Delta\eta = \tau_\varphi \cdot M_d(\eta^n)^\dagger \nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n).$$

The resulting formula suggests that we explicitly update η along the gradient ascent direction preconditioned by the Gram matrix $M_d(\eta^n)$,

$$\eta^{n+1} = \eta^n + \tau_\varphi \cdot M_d(\eta^n)^\dagger \nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n).$$

By doing so, we exchange some of the numerical stability enjoyed by the proximal step for computational feasibility and efficacy.

Remark 3 *It is worth mentioning that the Moore-Penrose inverse used here is generally unnecessary. In order to determine a solution* $\mathbf{v}$ *to* (23)*, one only needs to guarantee that* $M_d(\eta^n)\mathbf{v} = \nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n)$*. Consider any pseudo-inverse matrix* $M_d^+(\eta^n)$ *such that* $M_d(\eta^n)M_d^+(\eta^n)$ *preserves the column vectors of* $M_d(\eta^n)$*, i.e.,* $M_d(\eta^n)M_d^+(\eta^n)M_d(\eta^n) = M_d(\eta^n)$*. Then,* $\mathbf{v} = M_d^+(\eta^n)\nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n)$ *will be a solution to* (23)*. Thus, we can set*

$$\Delta\eta = M_d^+(\eta^n)\nabla_\eta \widehat{E}(\theta^n, \eta^n, \xi^n).$$

In this research, we pick $M_d^+(\eta^n)$ *as the Moore-Penrose inverse for convenience in discussion.*

Moreover, we utilize the same idea to update the parameters ξ and θ, such that

$$\begin{aligned}
\xi^{n+1} =& \xi^n + \tau_\varphi \cdot M_{bdd}(\xi^n)^\dagger \nabla_\xi \widehat{E}(\theta^n, \eta^n, \xi^n), \\
\theta^{n+1} =& \theta^n - \tau_u \cdot M_p(\theta^n)^\dagger \nabla_\theta \mathscr{E}(u_{\theta^n}, \widetilde{\varphi}_{n+1}, \widetilde{\psi}_{n+1}),
\end{aligned}$$

where $\widetilde{\varphi}_{n+1} = \varphi_{\eta^{n+1}} + \omega(\varphi_{\eta^{n+1}} - \varphi_{\eta^n}), \widetilde{\psi}_{n+1} = \psi_{\xi^{n+1}} + \omega(\psi_{\xi^{n+1}} - \psi_{\xi^n})$ are obtained via the extrapolation. The Gram type matrices $M_p(\theta), M_{bdd}(\xi)$ are computed as

$$M_p(\theta) = \int_\Omega \frac{\partial}{\partial\theta}\mathcal{M}_p u_\theta(x)^\top \frac{\partial}{\partial\theta}\mathcal{M}_p u_\theta(x)\, \mathrm{d}\mu + \lambda \int_{\partial\Omega} \frac{\partial}{\partial\theta}\mathcal{B}u_\theta(y)^\top \frac{\partial}{\partial\theta}\mathcal{B}u_\theta(y)\, \mathrm{d}\mu_{\partial\Omega}. \quad (24)$$

$$M_{bdd}(\xi) = \lambda \int_{\partial\Omega} \frac{\partial\psi_\xi(y)}{\partial\xi}^\top \frac{\partial\psi_\xi(y)}{\partial\xi}\, \mathrm{d}\mu_{\partial\Omega}, \quad (25)$$

This yields our NPDG algorithm (4).

In practice, choosing the stepsize τ_φ ranging from 10^{-2} to 10^{-1} usually yields stable and efficient performance of this explicit scheme. The study on the optimal choice of the stepsizes, as well as the application of more meticulous line search strategies will serve as the future research directions.

Remark 4 (Adoption of stronger boundary norm) *For Dirichlet problem, $\mathcal{B}$ is treated as the trace operator. Then $\mathcal{B} : H^1(\Omega) \to H^{1/2}(\partial\Omega) \subsetneq L^2(\partial\Omega)$ is a continuous and surjective mapping (Grisvard, 2011), it is thus more natural to employ the $H^{1/2}(\partial\Omega)$ norm—or even stronger boundary norms—rather than $L^2(\partial\Omega)$ for the boundary loss in* (19)*. A detailed treatment is deferred to Section 3.2.2.*

We conclude this subsection by briefly discussing the advantages and limitations of different ways to split the operator $\mathcal{L}$ associated with the linear PDE (10). In general, $\mathcal{L}$ may admit multiple splittings, each leading to a distinct preconditioning strategy. For example, the operator $-\Delta$ can be decomposed as in Example 1, or alternatively as described in Remark 1.

Comparing these two choices, the former is typically more computationally efficient, as it avoids the evaluation of second-order derivatives. Moreover, the theoretical analysis presented in Theorem 7 in the following section shows that, under the former splitting, the error term $u_\theta - u_*$ can be bounded using a more effective H^1 seminorm, rather than the $(-\Delta)$-weighted L^2 norm that arises in the latter case.

As a trade-off, as discussed in Remark 14 of Section 4, adaptive sampling strategies cannot be readily incorporated into the former framework, whereas such techniques can be naturally integrated under the latter choice.

In practice, we focus on the first splitting for the elliptic equations considered in this work. A comprehensive investigation and systematic comparison of different splitting and preconditioning strategies will be pursued in future investigations.

2.5 Nonlinear Equations

A similar treatment can be applied to the semi-linear equations taking the form of

$$\mathcal{L}u + \mathcal{N}u = f, \quad \text{on } \Omega, \quad \mathcal{B}u = g \quad \text{on } \partial\Omega, \quad (26)$$

where $\mathcal{L}$, $\mathcal{N}$ denote the linear and nonlinear operators, respectively. $\mathcal{B}$ is the boundary operator. $f : \Omega \to \mathbb{R}$, $g : \partial\Omega \to \mathbb{R}$. Suppose that $\mathcal{L}$ splits as $\mathcal{L} = \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p$. We then

multiply the equation and its boundary condition with the dual variables φ, ψ and derive the functional

$$\mathscr{E}(u,\varphi,\psi) = \langle \widetilde{\mathcal{L}}\mathcal{M}_p u, \mathcal{M}_d\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)} + \langle \mathcal{N}(u),\varphi\rangle_{L^2(\Omega)} - \langle f,\varphi\rangle_{L^2(\Omega)} + \lambda\langle \mathcal{B}u,\psi\rangle_{L^2(\partial\Omega)} - \frac{\nu}{2}\left(\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)} + \lambda\|\psi\|^2_{L^2(\partial\Omega)}\right).$$

We can now apply algorithm (4) with preconditioning matrices M_d, M_p, M_{bdd} mentioned above in (21), (24), (25) to solve the equation. Related numerical examples and more detailed descriptions of our treatment can be found in Section 5.3 and 5.4.

2.5.1 Monge-Ampère equation

The algorithm can readily handle some fully nonlinear equation that possesses a saddle point formulation, such as the Monge-Ampère equation.

$$|\det(D^2u(x))| = \frac{\rho_0(x)}{\rho_1(\nabla u(x))},\ \rho_0 \mathrm{d}x - a.e., \quad u \text{ is convex on } \mathbb{R}^d. \tag{27}$$

Here, ρ_0, ρ_1 are probability density functions. D^2u denotes the Hessian matrix of the potential function u. This equation takes an equivalent form of

$$\nabla u_\sharp \mu_0 = \mu_1, \quad u \text{ is convex on } \mathbb{R}^d,$$

where μ_0, μ_1 are probability distributions. We assume μ_0, μ_1 are absolutely continuous with respect to the Lebesgue measure on $\mathbb{R}^d$, with density functions ρ_0, ρ_1. And $\sharp$ denotes the "pushforward" of probability distribution μ_0 by the map ∇u in the sense of

$$\int_{\mathbb{R}^d} h(\nabla u(x))\,\mathrm{d}\mu_0(x) = \int_{\mathbb{R}^d} h(y)\,\mathrm{d}\mu_1(y), \quad \text{for any measurable function } h \text{ defined on } \mathbb{R}^d.$$

There is already adequate research on the classical numerical methods for the Monge-Ampère equation. We refer the readers to (Benamou et al., 2010; Froese and Oberman, 2011; Benamou et al., 2014; Neilan et al., 2020) and the references therein for further discussion.

In this research, we aim to propose a mesh-free algorithm based on the data samples drawn from ρ_0 and ρ_1 to evaluate $\nabla u(\cdot)$ of the equation. We should first point out that the Monge-Ampère equation is closely related to the following Optimal Transport (OT) problem (also known as the Monge problem) (Villani et al., 2009; De Philippis and Figalli, 2014),

$$\min_{\substack{T\in\mathcal{M}(\mathbb{R}^d,\mathbb{R}^d)\\ T_\sharp\mu_0=\mu_1}} \int_{\mathbb{R}^d} \frac{1}{2}\|x - T(x)\|^2\,\mathrm{d}\mu_0(x). \tag{28}$$

Here $\mathcal{M}(\mathbb{R}^d,\mathbb{R}^d)$ denotes the space of measurable maps from $\mathbb{R}^d$ to $\mathbb{R}^d$. We aim at computing for the optimal map T that transports the probability distribution ρ_0 to ρ_1 by minimizing the L^2 transportation cost. One can show that (c.f. Section 3 of (De Philippis and Figalli, 2014)) the optimal map T_* of (28) exists uniquely as long as μ_0, μ_1 possess densities ρ_0, ρ_1, and there exists a convex function $u : \mathbb{R}^d \to \mathbb{R}$ such that $T_*(x) = \nabla u(x)$ for μ_0-a.e. $x \in \mathbb{R}^d$. Furthermore, if μ_0, μ_1 are supported on bounded smooth open sets $X, Y \subset \mathbb{R}^d$, and ρ_0, ρ_1

are bounded away from zero and infinity on X and Y, then the potential u solves the Monge-Ampère equation (27).

Given the connection between the Monge-Ampère equation and the OT problem, we mainly focus on computing (28) instead of (27). The goal is to compute the OT map T_* (or ∇u). Notice that (28) is a constrained optimization problem. By denoting $\mathcal{C}_b(\mathbb{R}^d)$ as the space of bounded continuous functions, it is natural to introduce the Lagrange multiplier $\varphi \in \mathcal{C}_b(\mathbb{R}^d)$ (also known as the Kantorovich potential of the OT problem) to the constraint $T_\sharp \rho_0 = \rho_1$, and obtain

$$\mathscr{E}(T, \varphi) = \int_{\mathbb{R}^d} \frac{1}{2}\|x - T(x)\|^2 \,\mathrm{d}\mu_0(x) + \int_{\mathbb{R}^d} \varphi(T(x)) \,\mathrm{d}\mu_0(x) - \int_{\mathbb{R}^d} \varphi(y) \,\mathrm{d}\mu_1(y). \tag{29}$$

Upon solving (28), we consider the sup-inf saddle point problem

$$\sup_{\varphi \in \mathcal{C}_b(\mathbb{R}^d)} \inf_{T \in \mathcal{M}(\mathbb{R}^d, \mathbb{R}^d)} \mathscr{E}(T, \varphi). \tag{30}$$

It is shown in (Fan et al., 2023) that, as long as μ_0, μ_1 are compactly supported, and μ_0 is absolutely continuous w.r.t. Lebesgue measure, the saddle point $(T_\star, \varphi_\star)$ of (30) exists, and the map $T_\star(\cdot)$ equals the OT map $T_*(\cdot)$ μ_0-almost surely.

In the computation, we substitute T, φ with neural networks T_θ, φ_η. A natural way of preconditioning this problem is to set $\mathcal{M}_p = \mathrm{Id}$ and $\mathcal{M}_d = \mathrm{Id}$ for $M_p(\theta), M_d(\eta)$, i.e.,

$$\begin{aligned} M_p(\theta) &= \int_{\mathbb{R}^d} \frac{\partial T_\theta(x)}{\partial \theta}^\top \frac{\partial T_\theta(x)}{\partial \theta} \, \rho_0(x) \,\mathrm{d}x, \\ M_d(\eta; \theta) &= \int_{\mathbb{R}^d} \frac{\partial \varphi_\eta(T_\theta(x))}{\partial \eta} \frac{\partial \varphi_\eta(T_\theta(x))}{\partial \eta}^\top \rho_0(x) \,\mathrm{d}x. \end{aligned} \tag{31}$$

However, a more canonical choice is to set $\mathcal{M}_p = \mathrm{Id}$ and $\mathcal{M}_d = \nabla$. To motivate this preconditioning technique, we carry out the following calculation. Suppose $(T_\star, \varphi_\star)$ is the saddle point of the above problem (30). As $T_\star(\cdot) = T_*(\cdot)$ μ_0-almost surely, one can show that $T_{\star\sharp}\mu_0 = \mu_1$, and

$$T_\star(x) - x + \nabla \varphi_\star(T_\star(x)) = 0, \quad \mu_0 - a.s. \tag{32}$$

Here we denote $\nabla \varphi_\star(T_\star(x))$ as $\nabla \varphi_\star(y)|_{y = T_\star(x)}$.

We have

$$
\begin{aligned}
\mathscr{E}(T,\varphi) &= \int_{\mathbb{R}^d} \frac{1}{2}\|T(x) - T_\star(x) + T_\star(x) - x\|^2 \rho_0(x)\,\mathrm{d}x + \int_{\mathbb{R}^d} (\varphi(T(x)) - \varphi(T_\star(x)))\rho_0(x)\mathrm{d}x \\
&= \int_{\mathbb{R}^d} \left(\frac{1}{2}\|T_\star(x) - x\|^2 + (T(x) - T_\star(x))^\top (T_\star(x) - x) + \frac{1}{2}\|T(x) - T_\star(x)\|^2\right)\rho_0(x)\mathrm{d}x \\
&\quad + \int_{\mathbb{R}^d} \nabla\varphi(T(x))^\top (T(x) - T_\star(x))\rho_0(x)\mathrm{d}x \\
&\quad + \int_{\mathbb{R}^d} \frac{1}{2}(T(x) - T_\star(x))^\top \nabla^2\varphi(\xi(x))(T(x) - T_\star(x))\rho_0(x)\mathrm{d}x \\
&= \int_{\mathbb{R}^d} \left((T(x) - T_\star(x))^\top (T_\star(x) - x) + \nabla\varphi(T(x))^\top (T(x) - T_\star(x))\right)\rho_0(x)\mathrm{d}x \\
&\quad + \int_{\mathbb{R}^d} \frac{1}{2}(T(x) - T_\star(x))^\top (I - \nabla^2\varphi(\xi(x)))(T(x) - T_\star(x))\rho_0(x)\mathrm{d}x \\
&\quad + \int_{\mathbb{R}^d} \frac{1}{2}\|T_\star(x) - x\|^2 \rho_0(x)\mathrm{d}x \\
&= \int_{\mathbb{R}^d} (T_\star(x) - x + \nabla\varphi(T(x)))^\top (T(x) - T_\star(x))\rho_0(x)\mathrm{d}x \qquad (33) \\
&\quad + \int_{\mathbb{R}^d} \frac{1}{2}(T(x) - T_\star(x))^\top (I - \nabla^2\varphi(\xi(x)))(T(x) - T_\star(x))\rho_0(x)\mathrm{d}x \\
&\quad + \text{Const.}
\end{aligned}
$$

Here we denote $\xi(x) = (1-s)T(x) + sT_\star(x)$ with $0 \le s \le 1$.

Now recall the optimal relation (32), we can reformulate the term (33) as

$$
\begin{aligned}
&\int_{\mathbb{R}^d} (x - \nabla\varphi_\star(T_\star(x)) - x + \nabla\varphi(T(x)))^\top (T(x) - T_\star(x))\rho_0(x)\,\mathrm{d}x \\
&= \int_{\mathbb{R}^d} (\nabla\varphi(T(x)) - \nabla\varphi_\star(T_\star(x)))^\top (T(x) - T_\star(x))\rho_0(x)\,\mathrm{d}x. \qquad (34)
\end{aligned}
$$

Now (33) and (34) suggest that as (T,φ) approaches the optimal solution $(T_\star,\varphi_\star)$, the loss functional $\mathscr{E}(T,\varphi)$ is roughly the sum of $\left\langle \nabla\varphi(T(\cdot)) - \nabla\varphi_\star(T_\star(\cdot)),\ T(\cdot) - T_\star(\cdot)\right\rangle_{L^2(\rho_0)}$ and a quadratic term of $T(\cdot)$. This suggests setting $T(\cdot) - T_*(\cdot)$ and $\nabla\varphi(T(\cdot))$ as new entities of primal and dual variables in optimization. This yields the preconditioning matrices:

$$
\begin{aligned}
M_p(\theta) &= \int_{\mathbb{R}^d} \frac{\partial T_\theta(x)}{\partial\theta}^\top \frac{\partial T_\theta(x)}{\partial\theta}\ \rho_0(x)\,\mathrm{d}x, \\
M_d(\eta;\theta) &= \int_{\mathbb{R}^d} \frac{\partial}{\partial\eta}(\nabla\varphi_\eta(T_\theta(x)))^\top \frac{\partial}{\partial\eta}(\nabla\varphi_\eta(T_\theta(x)))\rho_0(x)\,\mathrm{d}x.
\end{aligned} \qquad (35)
$$

Applying (4) to the adversarial training of T_θ, φ_η leads to a faster, more robust algorithm for computing the saddle point $(T_\star,\varphi_\star)$ of (30), where $T_\star$ yields the desired OT map T_*, or equivalently, the solution ∇u to the Monge-Ampère equation. We refer the readers to Section 5.5 for details on implementation and numerical examples.

Remark 5 *It is worth mentioning that the idea of optimizing φ with respect to the H^1 metric has also been discussed in (Jacobs and Léger, 2020), in which the authors introduce a back-and-forth algorithm with H^1-preconditioned optimization to deal with the Kantorovich dual problem of* (28).

3. Convergence Analysis of the NPDG flow

In this section, we provide an *a posteriori* convergence analysis on the time-continuous version of the NPDG algorithm (4) as $\tau_u, \tau_\varphi \to 0$ and $\omega\tau_\varphi \to \gamma > 0$.

Recall that (4) can be reformulated as

$$\begin{aligned}
&\left(\begin{pmatrix} \eta^{n+1} \\ \xi^{n+1} \end{pmatrix} - \begin{pmatrix} \eta^{n} \\ \xi^{n} \end{pmatrix}\right) \Big/ \tau_\varphi = \begin{pmatrix} M_d(\eta^n)^\dagger \nabla_\eta \mathscr{E}(u_{\theta^n}, \varphi_{\eta^n}, \psi_{\xi^n}) \\ M_{bdd}(\xi^n)^\dagger \nabla_\xi \mathscr{E}(u_{\theta^n}, \varphi_{\eta^n}, \psi_{\xi^n}) \end{pmatrix}, \\
&\begin{pmatrix} \widetilde{\varphi}_{n+1} \\ \widetilde{\psi}_{n+1} \end{pmatrix} = \begin{pmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{pmatrix} + \omega\tau_\varphi \left(\begin{pmatrix} \varphi_{\eta^{n+1}} \\ \psi_{\xi^{n+1}} \end{pmatrix} - \begin{pmatrix} \varphi_{\eta^{n}} \\ \psi_{\xi^{n}} \end{pmatrix}\right) \Big/ \tau_\varphi, \\
&\frac{\theta^{n+1} - \theta^n}{\tau_u} = -M_p(\theta^n)^\dagger \nabla_\theta \mathscr{E}(u_{\theta^n}, \widetilde{\varphi}_{n+1}, \widetilde{\psi}_{n+1}).
\end{aligned}$$

By replacing the finite differences by the time derivatives, as $\tau_u, \tau_\varphi \to 0$ and $\omega\tau_\varphi \to \gamma$, we verify that (4) converges to

$$\begin{aligned}
\dot{\eta}_t =& M_d(\eta_t)^\dagger \nabla_\eta \mathscr{E}(u_{\theta_t}, \varphi_{\eta_t}, \psi_{\xi_t}), \\
\dot{\xi}_t =& M_{bdd}(\xi_t)^\dagger \nabla_\xi \mathscr{E}(u_{\theta_t}, \varphi_{\eta_t}, \psi_{\xi_t}), \\
\dot{\theta}_t =& -M_p(\theta_t)^\dagger \nabla_\theta \mathscr{E}(u_{\theta_t}, \widetilde{\varphi}_t, \widetilde{\psi}_t),
\end{aligned} \tag{36}$$

where we denote

$$\begin{pmatrix} \widetilde{\varphi}_t \\ \widetilde{\psi}_t \end{pmatrix} = \begin{pmatrix} \varphi_{\eta_t} \\ \psi_{\xi_t} \end{pmatrix} + \gamma \begin{pmatrix} \dot{\varphi}_{\eta_t} \\ \dot{\psi}_{\xi_t} \end{pmatrix}. \tag{37}$$

We call the above time-continuous dynamic (36) the **NPDG flow**. In this section, we analyze the convergence of the numerical solution $\{u_{\theta_t}\}$ along (36).

3.1 Natural gradient induces orthogonal projections of Fréchet derivatives

Before our discussion, we need the following lemma. Similar results have already been proved in several references, including (Liu et al., 2022; Nurbekyan et al., 2023; Wu et al., 2023; Müller and Zeinhofer, 2023; Zuo et al., 2024). We restate the lemma here for the sake of completeness.

Lemma 6 *Given a certain Hilbert space $\mathbb{X}$, we consider a Fréchet differentiable functional $\mathscr{F} : \mathbb{X} \to \mathbb{R}$. Suppose $\Theta \subseteq \mathbb{R}^m$ denotes the parameter space, we consider a parametrized family of functions $\{u_\theta\}_{\theta\in\Theta}$ which belong to $\mathbb{X}$. We denote $D_u\mathscr{F}(u) \in (\mathbb{X})^* = \mathbb{X}$ as the Fréchet derivative at u. Assume that u_θ is differentiable with respect to θ and $\frac{\partial u_\theta}{\partial \theta_i} \in \mathbb{X}$ for arbitrary $1 \le i \le m$, $\theta \in \Theta$. We define the $m \times m$ Gram matrix $M(\theta)$ as*

$$(M(\theta))_{ij} = \left\langle \frac{\partial u_\theta}{\partial \theta_i}, \frac{\partial u_\theta}{\partial \theta_j} \right\rangle_{\mathbb{X}}, \quad 1 \le i, j \le m.$$

Furthermore, we denote $F(\theta) = \mathscr{F}(u_\theta)$. Then one can show that

- $\nabla_\theta F(\theta) \in \mathrm{Ran}(M(\theta))$,
- *For any* $\mathbf{v} \in \mathbb{R}^m$ *such that* $M(\theta)\mathbf{v} = \nabla_\theta F(\theta)$, *we can show that* $\mathbf{v}$ *is the solution to the following least squares problem*[2].

$$\mathbf{v} \in \operatorname*{argmin}_{\zeta \in \mathbb{R}^m} \left\{ \left\| D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta} \zeta \right\|_{\mathbb{X}}^2 \right\} = \operatorname*{argmin}_{\zeta \in \mathbb{R}^m, \zeta_1, \ldots, \zeta_m \in \mathbb{R}} \left\{ \left\| D_u \mathscr{F}(u_\theta) - \sum_{i=1}^m \zeta_i \frac{\partial u_\theta}{\partial \theta_i} \right\|_{\mathbb{X}}^2 \right\}.$$

One can also verify that

$$D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta} \mathbf{v}$$

as a vector in $\mathbb{X}$, *is orthogonal (w.r.t. inner product defined on* $\mathbb{X}$*) to the subspace spanned by* $\{\frac{\partial u_\theta}{\partial \theta_1}, \ldots, \frac{\partial u_\theta}{\partial \theta_m}\}$. *Or equivalently,* $\frac{\partial u_\theta}{\partial \theta}\mathbf{v}$ *is the orthogonal projection of* $D_u \mathscr{F}(u_\theta)$ *on* $\mathrm{span}\{\frac{\partial u_\theta}{\partial \theta_1}, \ldots, \frac{\partial u_\theta}{\partial \theta_m}\}$.

We defer the proof of this lemma to Appendix C.1.

We should mention that the Moore-Penrose inverse $M(\theta)^\dagger \nabla_\theta F(\theta)$ yields a solution to the least square problem mentioned above. For the convenience of our future discussion, we denote the orthogonal projection (w.r.t. inner product on $\mathbb{X}$) onto $\mathrm{span}\{\frac{\partial u_\theta}{\partial \theta_1}, \ldots, \frac{\partial u_\theta}{\partial \theta_m}\}$ as $\Pi_{\partial_\theta u_\theta} : \mathbb{X} \to \mathbb{X}$, we thus have

$$\frac{\partial u_\theta}{\partial \theta} M(\theta)^\dagger \nabla_\theta F(\theta) = \Pi_{\partial_\theta u_\theta} \left[D_u \mathscr{F}(u_\theta) \right].$$

Correspondingly, we denote the orthogonal projection onto the orthogonal complement of $\mathrm{span}\{\frac{\partial u_\theta}{\partial \theta_1}, \ldots, \frac{\partial u_\theta}{\partial \theta_m}\}$ as $\Pi_{\partial_\theta u_\theta^\perp} : \mathbb{X} \to \mathbb{X}$, we have,

$$D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta} M(\theta)^\dagger \nabla_\theta F(\theta) = \Pi_{\partial_\theta u_\theta^\perp} \left[D_u \mathscr{F}(u_\theta) \right].$$

3.2 Convergence analysis of the NPDG flow

Throughout this section, we assume that $\Omega \subset \mathbb{R}^d$ is a bounded open domain with Lipschitz boundary $\partial\Omega$ (Grisvard, 2011). That is, $\partial\Omega$ can be locally represented as the graph of a Lipschitz function.

3.2.1 *A posteriori* convergence result for general linear PDEs

Recall that we consider the linear equation (10) defined on $\mathbb{H}$. We assume u_* as a real solution to (10). We will adopt the notations used in previous Section 2.2 and 2.3. In our discussion, we always assume that the operator $\widetilde{\mathcal{L}}$ is bounded from above and below in the sense of

$$0 < L_0 \le \inf_{\mathbf{u} \in \widetilde{\mathbb{H}}} \frac{\|\widetilde{\mathcal{L}}\mathbf{u}\|_{L^2(\Omega)}}{\|\mathbf{u}\|_{L^2(\Omega)}} \le \sup_{\mathbf{u} \in \widetilde{\mathbb{H}}} \frac{\|\widetilde{\mathcal{L}}\mathbf{u}\|_{L^2(\Omega)}}{\|\mathbf{u}\|_{L^2(\Omega)}} \le L_1 < \infty. \tag{38}$$

We denote $L_1 \vee 1 = \max\{L_1, 1\}$ and $L_0 \wedge 1 = \min\{L_0, 1\}$, and

$$\widetilde{\kappa} = \frac{L_1 \vee 1}{L_0 \wedge 1} \tag{39}$$

2. It is worth mentioning that for fixed x, $\frac{\partial u_\theta(x)}{\partial \theta}$ is a $k \times m$ matrix.

for shorthand.

Suppose that we perform the NPDG flow up to a time T. We denote $\alpha, \beta_1, \beta_2 \in [0,1]$ as coefficients quantifying the approximation power of the subspaces spanned by $\left\{(\frac{\partial \mathcal{M}_p u_{\theta_t}}{\partial \theta_k}, \sqrt{\lambda}\frac{\partial \mathcal{B} u_{\theta_t}}{\partial \theta_k})\right\}_{1\le k\le m_\theta}$, $\left\{\frac{\partial \mathcal{M}_d \varphi_{\eta_t}}{\partial \eta_k}\right\}_{1\le k\le m_\eta}$, and $\left\{\frac{\partial \psi_\xi}{\partial \xi_k}\right\}_{1\le k\le m_\xi}$ for $t\in[0,T]$. To be more specific, α is a constant satisfying

$$\min_{\substack{\zeta\in\mathbb{R}^{m_\theta}\\ \zeta_1,\dots,\zeta_{m_\theta}\in\mathbb{R}}} \left\{ \Big\| \sum_{k=1}^{m_\theta} \zeta_k \frac{\partial \mathcal{M}_p u_{\theta_t}}{\partial \theta_k} - \mathcal{M}_p(u_{\theta_t}-u_*)\Big\|^2_{L^2(\Omega)} + \Big\| \sum_{k=1}^{m_\theta} \zeta_k \sqrt{\lambda}\frac{\partial \mathcal{B} u_{\theta_t}}{\partial \theta} - \sqrt{\lambda}\mathcal{B}(u_{\theta_t}-u_*)\Big\|^2_{L^2(\partial\Omega)} \right\}$$
$$\le \alpha^2(\|\mathcal{M}_p(u_{\theta_t}-u_*)\|^2_{L^2(\Omega)} + \|\sqrt{\lambda}\mathcal{B}(u_{\theta_t}-u_*)\|^2_{L^2(\partial\Omega)}), \quad \text{for all } t\in[0,T].$$

Recall that we define $\mathbb{L}^2 = L^2(\Omega)\times L^2(\partial\Omega)$, we denote the subspace

$$\partial_\theta \mathbf{U}_\theta = \operatorname{span}\left\{ \left(\frac{\partial \mathcal{M}_p u_\theta}{\partial \theta_1}, \sqrt{\lambda}\frac{\partial \mathcal{B} u_\theta}{\partial \theta_1}\right), \dots, \left(\frac{\partial \mathcal{M}_p u_\theta}{\partial \theta_{m_\theta}}, \sqrt{\lambda}\frac{\partial \mathcal{B} u_\theta}{\partial \theta_{m_\theta}}\right)\right\} \subset \mathbb{L}^2.$$

Then, α quantifies the upper bound of the relative $\mathbb{L}^2$ norm of $\partial_\theta \mathbf{U}^\perp_{\theta_t}$ component of $(\mathcal{M}_p(u_{\theta_t}-u_*), \mathcal{B}(u_{\theta_t}-u_*))$ for $t\in[0,T]$. Here we denote $\partial_\theta \mathbf{U}^\perp_{\theta_t}$ as the subspace of $\mathbb{L}^2$ that is orthogonal to $\partial_\theta \mathbf{U}_{\theta_t}$ w.r.t. $\mathbb{L}^2$ inner product. Similarly, we denote the subspace

$$\partial_{\eta,\xi}\mathbf{\Phi}_{\eta,\xi} = \operatorname{span}\left\{\frac{\partial \mathcal{M}_d \varphi_\eta}{\partial \eta_k}\right\}_{k=1}^{m_\eta} \times \operatorname{span}\left\{\sqrt{\lambda}\frac{\partial \psi_\xi}{\partial \xi_k}\right\}_{k=1}^{m_\xi} \subset \mathbb{L}^2.$$

β_1, β_2 denote the upper bounds of the relative $\mathbb{L}^2$ norms of $\partial_{\eta,\xi}\mathbf{\Phi}^\perp_{\eta_t,\xi_t}$ components of $(\widetilde{\mathcal{L}}\mathcal{M}_p(u_{\theta_t}-u_*), \sqrt{\lambda}\mathcal{B}(u_{\theta_t}-u_*))$ and $(\mathcal{M}_d\varphi_{\eta_t}, \sqrt{\lambda}\psi_{\xi_t})$. The detailed definitions of α, β_1, β_2 can be found later in (69), (70) and (71).

The following Theorem analyzes the convergence of the numerical solution u_{θ_t} solved from (36) on $[0,T]$.

Theorem 7 (*A posteriori* convergence analysis of NPDG flow) *Suppose $\{(\theta_t, \eta_t, \xi_t)\}$ solves the NPDG flow (36) on $[0,T]$. Recall that α, β_1, β_2 quantify the approximation quality of neural networks $u_{\theta_t}, \varphi_{\eta_t}, \psi_{\xi_t}$ through $[0,T]$, and $\widetilde{\kappa}$ denotes the condition number (39). Suppose $\alpha+\beta_1 < \frac{1}{\widetilde{\kappa}^2}$, $\beta_2<1$, if we further assume that the hyperparameters of the NPDG flow $\gamma, \nu > 0$ satisfy*

$$\left(\frac{1}{\widetilde{\kappa}^2} - (\alpha+\beta_1)\right)\cdot(1-\beta_2) > \frac{((1+\beta_1)\gamma\nu + \beta_2 + \alpha|1-\gamma\nu|)^2}{4\gamma\nu}. \tag{40}$$

Then there exists a constant $r>0$, such that

$$\|\mathcal{M}_p(u_{\theta_t}-u_*)\|^2_{L^2(\Omega;\mathbb{R}^r)} + \lambda\|\mathcal{B}(u_{\theta_t}-u_*)\|^2_{L^2(\partial\Omega)} \le 2\exp(-rt)\cdot C_0 \quad \text{for } 0\le t\le T.$$

Here $C_0\ge 0$ is a constant depending on the initial value (θ_0,η_0,ξ_0) of the NPDG flow. We note that $r>0$ is the convergence rate depending on $\widetilde{\mathcal{L}}$, the hyperparameters γ,ν, and the relative errors α,β_1,β_2. The explicit form of r is provided in (72).

The proof of Theorem 7 is provided in Appendix C.2.

The dependence of the convergence rate r on $\gamma, \nu, \alpha, \beta_1, \beta_2$ can be significantly simplified as α and β_2 approach 0. This assumption is reasonable provided that the tangent spaces associated with the primal and test networks, $\partial \mathbf{U}_\theta$ and $\partial \Phi_{\eta,\xi}$, possess sufficiently strong approximation power for $\mathbf{U}_\theta$ and $\Phi_{\eta,\xi}$, respectively. Actually, if we assume that $u_\theta(\cdot), \varphi_\eta(\cdot), \psi_\xi(\cdot)$ are linear combinations of basis functions, i.e.,

$$u_\theta(x) = \sum_{k=1}^{m_\theta} \theta_k u_k(x), \quad \varphi_\eta(x) = \sum_{k=1}^{m_\eta} \eta_k \varphi_k(x), \quad \psi_\xi(x) = \sum_{k=1}^{m_\xi} \xi_k \psi_k(x), \tag{41}$$

with θ, η, ξ serving as the coefficients of the basis functions and $u_k \in H^2(\Omega)$, $\varphi_k \in H_0^1(\Omega)$, $\psi_k \in L^2(\partial\Omega)$. If u_* can further be represented by linear combination of $\{u_k\}_{k=1}^{m_\theta}$, then it holds exactly that $\alpha = \beta_2 = 0$. And the sufficient condition (40) for the convergence of NPDG flow reduces to $\gamma\nu \leq \frac{4(\widetilde{\kappa}^{-2}-\beta_1)}{(1+\beta_1)^2}$. Furthermore, we have the explicit lower bound of r

$$r \geq \frac{\left(4\left(\frac{1}{\widetilde{\kappa}^2} - \beta_1\right) - (1+\beta_1)^2\gamma\nu\right) \cdot \gamma\nu}{8\left(\left(\frac{1}{\widetilde{\kappa}^2} - \beta_1\right)\gamma + \frac{\nu}{(L_1 \vee 1)^2}\right)}. \tag{42}$$

Theorem 7 is *a posteriori* analysis as one may verify whether the estimate in the theorem is valid *after* obtaining the solution $(\theta_t, \eta_t, \xi_t)$, by checking whether condition (40) holds on a finite time interval $[0, T]$. It is worth mentioning that as t increases, u_{θ_t} will approach the real solution u_*; however, as the approximation gets better, the error term $(\mathcal{M}_p(u_{\theta_t} - u_*), \mathcal{B}(u_{\theta_t} - u_*))$ will erect orthogonally away from the exploration space $\partial_\theta \mathbf{U}_{\theta_t}$. Consequently, the quantity α will approach 1, and so will $\beta_1 \to 1$. That may prevent condition $\alpha + \beta_1 < \frac{1}{\widetilde{\kappa}^2}$ at a certain time t along the NPDG flow (recall that $\widetilde{\kappa} > 1$), thus yielding the analysis only applicable on a finite time interval.

3.2.2 A refined convergence analysis for elliptic PDEs of divergence form

The main challenge in establishing an *a priori* convergence result that extends to the infinite time horizon stems from the imbalance between the primal function space $\mathbb{H}$ and the test function space $\mathbb{K}^{\text{test}}$. To be more specific, let us consider the Dirichlet boundary problem associated with Example 1

$$-\Delta u = f, \quad \mathcal{B}u = g,$$

where $\mathcal{B}$ denotes the trace operator. Then we set $\mathbb{H} = H^2(\Omega), \mathbb{K}^{test} = H_0^1(\Omega)$. Suppose we compute the equation using $u_\theta, \varphi_\eta, \psi_\xi$ as defined in (41). Then, the convergence of the NPDG flow is anticipated as long as the tangent space $\partial_\theta \mathbf{U}_\theta = \text{span}\{\nabla u_k, \mathcal{B}u_k\}$ and $\partial_{\eta,\xi}\mathbf{\Phi}_{\eta,\xi} = \text{span}\{\nabla\varphi_k\}, \times\text{span}\{\psi_k\}$ along which the primal function $(\nabla u_\theta, \mathcal{B}u_\theta)$ and the test functions (φ_η, ψ_ξ) move can effectively approximate the gradients of $\mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$. The main issue arises when using $\text{span}\{\nabla\varphi_k\}$ to approximate the term $\nabla(u_\theta - u_*)$. Since each $\varphi_k \in \mathbb{K}^{test} = H_0^1(\Omega)$, however, $u_\theta - u_* \in \mathbb{H} = H^2(\Omega)$ is not guaranteed to lie in $H_0^1(\Omega)$. Therefore, a significant approximation error

$$\|\Pi_{\text{span}\{\nabla\varphi_1,\ldots,\nabla\varphi_{m_\eta}\}^\perp}[\nabla(u_\theta - u_*)]\|_{L^2(\Omega)}^2 = \min_{\varphi \in \text{span}\{\varphi_1,\ldots,\varphi_{m_\eta}\}} \|\nabla(u_\theta - u_*) - \nabla\varphi\|_{L^2(\Omega)}^2, \tag{43}$$

will arise and it will bound β_1 away from 0, and hence impeding the convergence of the NPDG flow. Intuitively, if we assume that $\text{span}\{\varphi_1, \ldots, \varphi_{m_\eta}\}$ constitutes a basis of $H_0^1(\Omega)$ as $m_\eta \to \infty$, the term (43) can exactly be bounded from above by the fractional Sobolev norm $\|u_\theta - u_*\|^2_{H^{1/2}(\partial\Omega)}$ (Gagliardo, 1957; Grisvard, 2011). This key observation yields a remedy to substitute the original boundary loss term $L^2(\partial\Omega)$ with a stronger norm such as $H^{1/2}(\partial\Omega)$ in order to offset the approximation error caused by the imbalance between the primal and the test functional spaces during natural gradient optimization.

In the following discussion, we establish a modified version of Theorem 7 by incorporating suitable boundary loss. We primarily focus on an important class of linear elliptic equations in divergence form,

$$-\nabla \cdot (A(x)\nabla u(x)) = f(x) \text{ on } \Omega, \quad u(x) = g(x) \text{ on } \partial\Omega,$$

where $f \in L^2(\Omega), g \in H^1(\partial\Omega) \subset H^{1/2}(\partial\Omega)$, $A(\cdot) : \Omega \to \mathbb{R}^{d\times d}$ with each entry $A_{ij}(\cdot) \in L^2(\Omega) \cap C^1(\Omega)$. We further assume $A(\cdot)$ is bounded from both above and below, i.e., for any $x \in \mathbb{R}^d$, we always have $\underline{A} \cdot \|\xi\|^2 \le \xi^\top A(x)\xi \le \overline{A}\|\xi\|^2$, with the constants $\overline{A} \ge \underline{A} > 0$.

Let us denote the Hilbert space $\mathcal{X}$ such that $H^{3/2}(\partial\Omega) \subseteq \mathcal{X} \subseteq H^{1/2}(\partial\Omega)$ with continuous inclusion map $\mathcal{X} \hookrightarrow H^{1/2}(\partial\Omega)$. Then, there exists a constant $C_{\mathcal{X}} > 0$ such that

$$\|\mathcal{B}w\|_{\mathcal{X}} \ge C_{\mathcal{X}}\|\mathcal{B}w\|_{H^{1/2}(\partial\Omega)}, \quad \text{for any } w \in \mathcal{X}.$$

Denote $\langle\cdot\,,\cdot\rangle_{\mathcal{X}}$ as the inner product on $\mathcal{X}$, we slightly modify the original functional $\mathscr{E}$ by using the $\mathcal{X}-$boundary norm,

$$\mathscr{E}_{\mathcal{X}}(u,\varphi,\psi) = \langle\widetilde{\mathcal{L}}\mathcal{M}_p u, \mathcal{M}_d\varphi\rangle_{L^2} - \langle f,\varphi\rangle_{L^2} + \lambda\langle\mathcal{B}u - g, \psi\rangle_{\mathcal{X}} - \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|^2_{L^2} + \lambda\|\psi\|^2_{\mathcal{X}}). \quad (44)$$

Suppose we conduct the NPDG flow (36) associated with $\mathscr{E}_{\mathcal{X}}(u_\theta, \varphi_\eta, \psi_\xi)$. We further introduce a useful seminorm $|\cdot|_{H^1(\Omega,A)}$ on $H^1(\Omega)$:

$$|u|^2_{H^1(\Omega,A)} = \int_\Omega \nabla u(x)^\top A(x)\nabla u(x)\,\mathrm{d}x, \quad \text{for } u \in H^1(\Omega). \quad (45)$$

We have the following theorem.

Theorem 8 (A refined convergence analysis of NPDG flow) *Suppose we pick the hyperparameters $\gamma, \nu > 0$ such that $\gamma\nu < 2$. Assume the coefficient λ associated with the boundary loss is sufficiently large, i.e., $\lambda \ge 8\overline{A}\left(\frac{\gamma(1+\nu)}{C_{\mathcal{X}}\cdot c_\Omega}\right)^2$, where $c_\Omega > 0$ is a constant whose value is discussed in detail in Appendix C.3. Then we have*

$$|u_{\theta_t} - u_*|^2_{H_1(\Omega,A)} + \lambda\|u_{\theta_t} - u_*\|^2_{\mathcal{X}} \le \left(e^{-rt}\sqrt{E_0} + \int_0^t \frac{1}{2}e^{-r(t-\tau)}\mathrm{Err}(\theta_\tau, \eta_\tau, \xi_\tau, \gamma, \nu, \lambda)\mathrm{d}\tau\right)^2. \quad (46)$$

Here, r denotes the convergence rate. It can be shown that the convergence rate

$$r \ge \frac{1}{2}\cdot\frac{\gamma\nu(2-\gamma\nu)}{\gamma + 2\nu}. \quad (47)$$

We set $E_0 = |u_{\theta_0} - u_*|^2_{H_1(\Omega,A)} + \lambda\|u_{\theta_0} - u_*\|^2_{\mathcal{X}}$ as the initial error. $\mathrm{Err}(\theta_t, \eta_t, \xi_t, \gamma, \nu, \lambda)$ represents the summation of the approximation errors associated with the tangential spaces of primal and test networks. The detailed formulation is provided in (91).

We prove Theorem 8 in Appendix C.4.

As a corollary of Theorem 8, an *a priori* convergence result can be obtained under the case in which $u_\theta, \varphi_\eta, \psi_\xi$ are linear combinations of basis functions.

Corollary 9 (*A priori* convergence of NPDG flow) *Suppose that $u_\theta, \varphi_\eta, \psi_\xi$ are linear combinations (41) of suitable basis functions $u_k \in H^2(\Omega)$, $\varphi_k \in H_0^1(\Omega)$, $\psi_k \in \mathcal{X}$. We keep all the notations and the conditions on γ, ν, λ mentioned in Theorem 8. We denote*

$$\mathcal{E}_u := \min_{\zeta\in\mathbb{R}^{m_\theta}} \int_\Omega \|\sum_{k=1}^{m_\theta} \zeta_k \nabla u_k - \nabla u_*\|^2 \mathrm{d}x + \frac{\lambda}{\overline{A}} \|\sum_{k=1}^{m_\theta} \zeta_k \mathcal{B} u_k - g\|_{\mathcal{X}}^2.$$

$$\mathcal{E}_{\nabla\varphi} := \min_{\zeta\in\mathbb{R}^{m_\eta}} \int_\Omega \|\sum_{k=1}^{m_\eta} \zeta_k \nabla \varphi_k - \nabla \varphi_*\|^2 \mathrm{d}x, \quad \mathcal{E}_\psi := \min_{\zeta\in\mathbb{R}^{m_\xi}} \|\sum_{k=1}^{m_\xi} \zeta_k \psi_k - g\|_{\mathcal{X}}^2.$$

Here we denote $\varphi_* \in H_0^1(\Omega)$ as the (weak) solution φ to

$$-\nabla\cdot(A(x)\nabla\varphi(x)) = -\nabla\cdot(A(x)\nabla u_*(x)), \text{ on } \Omega, \quad \varphi = 0 \text{ on } \partial\Omega.$$

Suppose the basis $\{u_k\}$ are picked so that $\mathrm{span}\{\mathcal{T}u_k\} \subseteq \mathrm{span}\{\varphi_k\}, \mathrm{span}\{\mathcal{B}u_k\} \subseteq \mathrm{span}\{\psi_k\}$, we then have

$$\begin{aligned}\left(|u_{\theta_t} - u_*|_{H_1(\Omega,A)}^2 + \lambda\|u_{\theta_t} - u_*\|_{\mathcal{X}}^2\right)^{\frac{1}{2}} &\le e^{-rt}\sqrt{E_0} \\ &\quad + \frac{1-e^{-rt}}{r}\left[(3\vee\gamma)\sqrt{\overline{A}\,\mathcal{E}_u} + \frac{\gamma+3}{\sqrt{2}}\sqrt{\overline{A}\,\mathcal{E}_{\nabla\varphi} + \mathcal{E}_\psi}\right].\end{aligned}$$

The convergence rate $r > 0$ satisfies the same lower bound as described in (47).

The corollary is proved in Appendix C.5.

Remark 10 *It is straightforward to verify the following bound*

$$\begin{aligned}\left(\|\nabla u_{\theta_t} - \nabla u_*\|_{L^2}^2 + \lambda\|u_{\theta_t} - u_*\|_{\mathcal{X}}^2\right)^{\frac{1}{2}} &\le e^{-rt}\sqrt{\frac{E_0}{\underline{A}}} \\ &\quad + \frac{1-e^{-rt}}{r}\left[(3\vee\gamma)\sqrt{\kappa(A)\,\mathcal{E}_u} + \frac{\gamma+3}{\sqrt{2}}\sqrt{\kappa(A)\,\mathcal{E}_{\nabla\varphi} + \frac{\mathcal{E}_\psi}{\underline{A}}}\right].\end{aligned}$$

provided $\xi^\top A(\cdot)\xi \ge \underline{A}\|\xi\|^2$ for $\xi \in \mathbb{R}^d$. Here we denote $\kappa(A) := \overline{A}/\underline{A} \ge 1$.

A natural choice of the boundary norm $\mathcal{X}$ is the fractional Sobolev norm $H^{1/2}(\partial\Omega)$. However, the definition (73) might not be convenient to evaluate. A more practical choice could be the $H^1(\partial\Omega)$ norm, which has been considered in recent reference such as (Shao et al., 2025) for nonlinear equations. In Section 5.2, we present numerical examples that incorporate the $H^1(\partial\Omega)$ boundary norm computed using Monte-Carlo approximations. The numerical evidence provided in Figure 4 suggests improved convergence and accuracy of the algorithm.

Remark 11 (Hyperparameters γ, ν, λ) *Notice that in the a posteriori analysis presented in Theorem 7, the convergence rate r depends not only on the hyperparameters γ and ν, but also on the relative approximation errors α, β_1, β_2, which themselves depend on γ and ν. This interdependence makes it infeasible to determine suitable values of the hyperparameters γ and ν in practice.*

In contrast, under the current framework, the convergence rate r depends solely on γ, ν. As a result, we can explicitly maximize the lower bound $\frac{1}{2} \cdot \frac{\gamma\nu\left(2-\gamma\nu\right)}{\gamma+2\nu}$, which is attained at $\gamma_* = \frac{2}{\sqrt{3}}$ and $\nu_* = \frac{1}{\sqrt{3}}$. Consequently, the convergence rate satisfies $r \geq \frac{2}{3\sqrt{3}}$.

In both Theorem 8 and Corollary 9, the choice of $\lambda > 0$ relies on the constant c_Ω associated with the $H^{1/2}(\partial\Omega)$ norm. It is usually intractable to determine c_Ω for general Ω. In practice, we pick a rather large $\lambda = 10$ to ensure better performance of the method.

Remark 12 *The Lyapunov-based proof framework for the NPDG flow developed in this section is inspired by earlier studies in (Liu et al., 2023b), (Liu et al., 2025) which focus on fully time-implicit schemes for conservation laws and reaction–diffusion equations. We clarify several fundamental differences that distinguish the present theoretical analysis from these previous works in Appendix C.6.*

4. Algorithm

In this section, we provide a detailed description on implementing the NPDG algorithm (4). We take the linear PDE (10) as an illustrative example.

4.1 Loss functional and the precondition matrices

Recall our discussions in Section 2.3. We introduce the pair of dual neural networks (φ_η, ψ_ξ) to equation (10) and consider the loss functional

$$\begin{aligned}\mathscr{E}(u, \varphi, \psi) = &\left(\int_\Omega \widetilde{\mathcal{L}}\mathcal{M}_p u(x) \cdot \mathcal{M}_d\varphi(x) - f(x)\varphi(x)\,\mathrm{d}\mu - \frac{\nu}{2}\int_\Omega \mathcal{M}_d\varphi(x) \cdot \mathcal{M}_d\varphi(x)\,\mathrm{d}\mu\right) \\ &+ \lambda\left(\int_{\partial\Omega} (\mathcal{B}u(y) - g(y)) \cdot \psi(y)\,\mathrm{d}\mu_{\partial\Omega} - \frac{\nu}{2}\int_{\partial\Omega} \psi(y) \cdot \psi(y)\,\mathrm{d}\mu_{\partial\Omega}\right).\end{aligned}$$

It is worth noting that the loss functional $\mathscr{E}(u, \varphi, \psi)$ considered here differs slightly from that introduced in (14), as we incorporate probability measures μ and $\mu_{\partial\Omega}$ in the evaluation of the integrals so that the loss can be efficiently approximated using Monte Carlo methods. Moreover, we assume that both measures are absolutely continuous with respect to the Lebesgue measures on Ω and $\partial\Omega$, respectively, with strictly positive densities, ensuring that the algorithm adequately explores the entire domain and boundary. A convenient choice is to take both μ and $\mu_{\partial\Omega}$ to be uniform distributions over Ω and $\partial\Omega$.

Sometimes, it also helps if we add the L^2 boundary loss

$$\|\mathcal{B}u - g\|^2_{L^2(\partial\Omega, \mu_{\partial\Omega})} = \int_{\partial\Omega} (\mathcal{B}u - g)^2\,\mathrm{d}\mu_{\partial\Omega}$$

into $\mathscr{E}(u,\varphi,\psi)$, and consider

$$\widetilde{\mathscr{E}}(u,\varphi,\psi)=\mathscr{E}(u,\varphi,\psi)+\lambda\|\mathcal{B}u-g\|^2_{L^2(\partial\Omega,\mu_{\partial\Omega})}.$$

We also recall that the precondition matrices $M_p(\theta)\in\mathbb{R}^{m_\theta\times m_\theta}$, $M_d(\eta)\in\mathbb{R}^{m_\eta\times m_\eta}$, $M_{bdd}(\xi)\in\mathbb{R}^{m_\xi\times m_\xi}$ are defined in (24), (21) and (25).

4.2 Monte-Carlo approximation

We apply the Monte Carlo algorithm to approximate the loss function $\mathscr{E}(u_\theta;\varphi_\eta,\psi_\xi)$ throughout our computation. Assume that $\{\boldsymbol{X}_i\}_{1\le i\le N_{in}}$, $\{\boldsymbol{Y}_j\}_{1\le j\le N_{bdd}}$ are samples uniformly drawn from the domain Ω and its boundary $\partial\Omega$, respectively. We compute

$$\begin{aligned}\mathscr{E}(u_\theta;\varphi_\eta,\psi_\xi)\approx&\frac{1}{N_{in}}\sum_{i=1}^{N_{in}}\widetilde{\mathcal{L}}\mathcal{M}_pu(\boldsymbol{X}_i)\cdot\mathcal{M}_d\varphi(\boldsymbol{X}_i)-f(\boldsymbol{X}_i)\varphi(\boldsymbol{X}_i)-\frac{\nu}{2}\mathcal{M}_d\varphi_\eta(\boldsymbol{X}_i)\cdot\mathcal{M}_d\varphi_\eta(\boldsymbol{X}_i)\\&+\lambda\left(\frac{1}{N_{bdd}}\sum_{j=1}^{N_{bdd}}(\mathcal{B}u(\boldsymbol{Y}_j)-g(\boldsymbol{Y}_j))\psi_\xi(\boldsymbol{Y}_j)-\frac{\nu}{2}\psi_\xi(\boldsymbol{Y}_j)\psi_\xi(\boldsymbol{Y}_j)\right).\end{aligned}$$

Here, "$\cdot$" denotes the inner product of vectors. For example, if $\mathcal{M}_p=\mathcal{M}_d=\nabla$, $\widetilde{\mathcal{L}}=\mathrm{Id}$, then

$$\widetilde{\mathcal{L}}\mathcal{M}_pu(\boldsymbol{X}_i)\cdot\mathcal{M}_d\varphi(\boldsymbol{X}_i)=\nabla u(\boldsymbol{X}_i)\cdot\nabla\varphi(\boldsymbol{X}_i),\quad \mathcal{M}_d\varphi_\eta(\boldsymbol{X}_i)\cdot\mathcal{M}_d\varphi_\eta(\boldsymbol{X}_i)=\|\nabla\varphi(\boldsymbol{X}_i)\|^2.$$

For general nonlinear PDE, the loss function $\mathscr{E}(u_\theta;\varphi_\eta,\psi_\xi)$ can also be approximated via the Monte-Carlo algorithm. Its gradient $\nabla_\theta\mathscr{E}(u_\theta;\varphi_\eta,\psi_\xi)$ can be computed using auto-differentiation (Baydin et al., 2018).

Furthermore, it is also straightforward to evaluate the preconditioning matrices $M_p(\theta)$ via Monte Carlo method, for example $M_p(\theta)$ can be computed as [3]

$$M_p(\theta)\approx\frac{1}{N_{in}}\sum_{i=1}^{N_{in}}\frac{\partial}{\partial\theta}(\mathcal{M}_pu_\theta(\boldsymbol{X}_i))^\top\frac{\partial}{\partial\theta}(\mathcal{M}_pu_\theta(\boldsymbol{X}_i))+\frac{\lambda}{N_{in}}\sum_{j=1}^{N_{bdd}}\frac{\partial}{\partial\theta}u_\theta(\boldsymbol{Y}_j)^\top\frac{\partial}{\partial\theta}u_\theta(\boldsymbol{Y}_j),$$

$$\text{i.e., }(M_p(\theta))_{ij}=\frac{1}{N_{in}}\sum_{i=1}^{N_{in}}\frac{\partial\mathcal{M}_pu_\theta(\boldsymbol{X}_i)}{\partial\theta_i}\cdot\frac{\partial\mathcal{M}_pu_\theta(\boldsymbol{X}_i)}{\partial\theta_j}+\frac{\lambda}{N_{bdd}}\sum_{j=1}^{N_{bdd}}\frac{\partial u_\theta(\boldsymbol{Y}_j)}{\partial\theta_i}\frac{\partial u_\theta(\boldsymbol{Y}_j)}{\partial\theta_j}.$$

It is worth mentioning that we use the *same* set of samples for computation of both the loss function and the preconditioning matrices. A new set of samples is generated at each iteration of the NPDG algorithm.

4.3 Inverting the preconditioning matrices via Krylov iterative solver

We then solve the least square problem (23) for $\mathbf{v}$. As mentioned in Remark 3, this is equivalent to solving the linear equation

$$M_p(\theta_k)\mathbf{v}=\nabla_\theta\mathscr{E}(u_{\theta_k};\varphi_{\eta_k},\psi_{\xi_k}).\tag{48}$$

3. Recall that we denote $\frac{\partial}{\partial\theta}\mathcal{M}_pu_\theta(\boldsymbol{X}_i)$ as the Jacobian of $\mathcal{M}_pu_\theta$ at $\boldsymbol{X}_i$. For example, if one sets $\mathcal{M}_p=\nabla$, then $\frac{\partial}{\partial\theta}\mathcal{M}_pu_\theta(\boldsymbol{X}_i)$ is a $d\times m$ matrix. Similarly, we assume $\frac{\partial u_\theta(\boldsymbol{Y}_j)}{\partial\theta}$ is $1\times d$.

However, this may suffer from the limitation on scalability: The method always computes and records the entire preconditioning matrix $M_p(\theta_k)$ at each optimization step. For neural networks such as Multilayer Perceptron, $M_p(\theta)$ is generally non-sparse, which suggests that forming this $m \times m$ matrix will occupy immense memory space of the computing resources as the number of parameters of the neural networks increases. For example, in numerical experiment 5.2, we deal with MLP $u_\theta(\cdot)$ with $d_{in} = 50, d_h = 256, d_{out} = 1$ and $n_l = 6$, this neural network contains $m_\theta = 279090$ parameters. Forming such $m_\theta \times m_\theta$ matrix is generally infeasible.

As a mitigation, instead of the direct evaluation of the preconditioning matrices, we apply the MINRES algorithm, which is an iterative solver, to solve (48). The MINRES iterative solver only requires matrix-vector multiplications that can readily avoid the direct formation of the preconditioning matrices. Similar treatment is also utilized in (Dembo et al., 1982; Martens, 2010; Roosta et al., 2022; Rathore et al., 2024) and the references therein in optimization problems. The same technique is also used in (Wu et al., 2023; Jin et al., 2024) to handle the computation of Wasserstein geometric flows.

We briefly describe how we evaluate $M_p(\theta)\mathbf{v}$ for arbitrary vector $\mathbf{v} \in \mathbb{R}^m$ under the deep learning framework. Given neural network $u_\theta(\cdot) : \mathbb{R}^d \to \mathbb{R}$ with parameter $\theta \in \mathbb{R}^m$, we make a copy $u_{\theta'}^{\text{copy}}(\cdot)$ by inheriting the architecture of $u_\theta(\cdot)$ and by setting $\theta' = \theta$. We apply auto-differentiation to evaluate $\{\mathcal{M}_p u_\theta(\boldsymbol{X}_i)\}_{i=1}^{N_{in}}$ and $\{\mathcal{M}_p u_{\theta'}^{\text{copy}}(\boldsymbol{X}_i)\}_{i=1}^{N_{in}}$, we also evaluate $\{u_\theta(\boldsymbol{Y}_j)\}_{j=1}^{N_{bdd}}, \{u_{\theta'}^{\text{copy}}(\boldsymbol{Y}_j)\}_{j=1}^{N_{bdd}}$. Then we compute the scalar

$$\Gamma(\theta', \theta) = \frac{1}{N_{in}} \sum_{i=1}^{N_{in}} \mathcal{M}_p u_{\theta'}^{\text{copy}}(\boldsymbol{X}_i) \cdot \mathcal{M}_p u_\theta(\boldsymbol{X}_i) + \frac{\lambda}{N_{bdd}} \sum_{j=1}^{N_{bdd}} u_{\theta'}^{\text{copy}}(\boldsymbol{Y}_j) u_\theta(\boldsymbol{Y}_j). \tag{49}$$

Now, by applying auto-differentiation again, we take the partial derivative of $\Gamma_{in}(\theta', \theta)$ w.r.t. θ, and making an inner product with $\mathbf{v}$, this yields $\partial_\theta \Gamma_{in}(\theta', \theta)^\top \mathbf{v}$. Finally, taking the partial derivative w.r.t. θ' yields $\partial_{\theta'}(\partial_\theta \Gamma(\theta', \theta)\mathbf{v})|_{\theta'=\theta} \approx M_p(\theta)\mathbf{v}$. This suggests an effective way of evaluating $M_p(\theta)\mathbf{v}$ without forming $M_p(\theta)$ explicitly. We summarize this procedure in the following Algorithm 1. Similarly, the matrix-vector multiplication involving $M_d(\eta), M_{bdd}(\xi)$

Algorithm 1 Evaluating $M_p(\theta)\mathbf{v}$

Input: Preconditioning operators $\mathcal{M}_p, \mathcal{M}_d$. Neural network $u_\theta(\cdot)$, samples $\{\boldsymbol{X}_i\}_{i=1}^{N_{in}} \subset \Omega$, $\{\boldsymbol{Y}_j\}_{j=1}^{N_{bdd}} \subset \partial\Omega$, vector $\mathbf{v} \in \mathbb{R}^m$.

1: Make a copy $u_{\theta'}^{\text{copy}}(\cdot)$ of the given $u_\theta(\cdot)$ with $\theta' = \theta$.
2: Evaluate $\Gamma(\theta', \theta)$ as defined in (49).
3: Apply auto-differentiation to evaluate $\partial_\theta \Gamma(\theta', \theta)\mathbf{v}$.
4: Apply auto-differentiation to evaluate $\mathbf{u} = \partial_{\theta'}(\partial_\theta \Gamma(\theta', \theta)\mathbf{v})$.

Return: u

can be computed by using the same technique.

Remark 13 *Calculating $M_p(\theta)\mathbf{v}$ can be further simplified by using the finite-difference approximation, which may lead to faster speed and lower memory cost. This technique has been conducted in several Hessian-free optimization algorithms (Martens, 2010; Knoll and Keyes,*

2004; Rathore et al., 2024). This possible improvement will serve as the future research directions.

We adopt the `scipy.sparse.linalg.minres` solver in SciPy (Virtanen et al., 2020) throughout our implementation. This algorithm involves two key hyperparameters: the maximum number of iterations n_{MINRES}, and the tolerance value tol_{MINRES}, which determines the acceptable relative residual. For more details on selecting these parameters, please refer to Section 5.

4.4 Sketch of main algorithm

We summarize the proposed method in Algorithm 2.

Remark 14 *The probability measures μ and $\mu_{\partial\Omega}$ used in our implementation need not be uniform. In fact, it is often more appealing to choose these measures adaptively according to the residuals $\left|\mathcal{L}u_\theta(\cdot) - f(\cdot)\right|$, $\left|\mathcal{B}u_\theta(\cdot) - g(\cdot)\right|$. Samples from such adaptive distributions can be generated using Markov chain Monte Carlo methods or deep generative models (Tang et al., 2023). However, when the density of μ is non-constant, applying integration by parts introduces additional score terms associated with μ, which may significantly complicate the construction of the corresponding preconditioning matrices. One possible alternative is to choose $\mathcal{M}_p = \mathcal{L}$, $\widetilde{\mathcal{L}} = \mathrm{Id}$, and $\mathcal{M}_d = \mathrm{Id}$, thereby avoiding integration by parts altogether. The integration of the NPDG framework with adaptive sampling strategies will remain as an important direction for future research.*

5. Numerical Examples

In this section, we apply the proposed Natural Primal-Dual Hybrid Gradient (NPDG) algorithm to various types of PDEs, including linear and nonlinear, static, and time-dependent equations. We denote our method as the NPDG algorithm for simplicity.

Throughout numerical experiments, we set neural networks as Multilayer Perceptron (MLP). That is, the fully connected neural network with the input dimension d_{in}, the hidden dimension d_{hidden}, the output dimension d_{out}, and the number of layers n_{MLP}. We denote such MLP with activation function f as $\texttt{MLP}_f(d_{\text{in}}, d_{\text{hidden}}, d_{\text{out}}, n_{\text{MLP}})$. Readers are referred to Appendix A for further details on MLP.

Throughout the experiments in this section, we fix the hyperparameters $\nu = 1$ and $\omega = 1$ for the NPDG algorithm. When applicable, unless otherwise specified, the boundary loss coefficient is always set to $\lambda = 10$. We always choose the maximum iteration number for the MINRES algorithm as $n_{\text{MINRES}} = 1000$.

We compare the proposed algorithm with a series of commonly used deep-learning solvers, namely, Physics-Informed Neural Network (PINN) (Raissi et al., 2019), Deep Ritz method (Yu et al., 2018), and primal-dual-type algorithms for PDEs/optimal transport (Zang et al., 2020) (Fan et al., 2023). We apply Adam (Kingma, 2014; Paszke et al., 2019) and (or) L-BFGS (Liu and Nocedal, 1989; Paszke et al., 2019) algorithms to PINN. When we use the L-BFGS method, we choose $lr = 1.0$ as the default. The L-BFGS method does not perform stably with the Deep Ritz and primal-dual type methods. We will only apply

Algorithm 2 Natural Primal-Dual Hybrid Gradient method (NPDG)

Input: The equation $F(u, \nabla u, \nabla^2 u, \dots) = 0$ on Ω with (if any) boundary condition $\mathcal{B}u = g$ on $\partial\Omega$. Preconditioning operators $\mathcal{M}_p, \mathcal{M}_d$. The functional $\mathscr{E}(u, \varphi, \psi)$. Stepsizes τ_u, τ_φ, τ_ψ of the NPDG algorithm; extrapolation coefficient ω; Total iteration number of the NPDG algorithm N_{iter}. Number of samples drawn from Ω and $\partial\Omega$: N_{in}, N_{bdd}. Max iteration number n_{MINRES} and tolerance of relative residual tol_{MINRES} of the MINRES algorithm.

Initialize the primal neural network $u_\theta(\cdot)$, dual neural network(s) $\varphi_\eta(\cdot)$ and $\psi_\xi(\cdot)$ if the equation is equipped with boundary condition(s).

for $iter = 1$ to N_{iter} **do**

  Set $\eta_0 = \eta, \xi_0 = \xi$.

  Resample points for the Monte-Carlo estimations in each iteration.

  Apply Monte-Carlo algorithm and auto-differentiation to evaluate

$$(\mathbf{w}_\varphi^\top, \mathbf{w}_\psi^\top)^\top = \nabla_{(\eta,\xi)} \mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi).$$

  Apply MINRES algorithm $(n_{\text{MINRES}}, tol_{\text{MINRES}})$ with Algorithm 1 to solve

$$M_d(\eta)\mathbf{v}_\varphi = \mathbf{w}_\varphi, M_{bdd}(\xi)\mathbf{v}_\psi = \mathbf{w}_\psi.$$

  Update $\eta = \eta + \tau_\varphi \mathbf{v}_\varphi,\ \xi = \xi + \tau_\psi \mathbf{v}_\psi$. ▷ Natural gradient ascent

  Set $\widetilde{\varphi} = \varphi_\eta + \omega(\varphi_\eta - \varphi_{\eta_0})$, $\widetilde{\psi} = \psi_\xi + \omega(\psi_\xi - \psi_{\xi_0})$. ▷ Extrapolation in *functional space*

  Apply Monte-Carlo algorithm and auto-differentiation to evaluate

$$\mathbf{w}_u = \nabla_\theta \mathscr{E}(u_\theta, \widetilde{\varphi}, \widetilde{\psi}).$$

  Apply MINRES algorithm (n_{MINRES}, tol_{MINRES}) with Algorithm 1 to solve

$$M_p(\theta)\mathbf{v}_u = \mathbf{w}_u.$$

  Update $\theta = \theta - \tau_u \mathbf{v}_u$ ▷ Natural gradient descent

end for

Return: $u_\theta(\cdot)$

the Adam algorithm to these two methods. To remain consistent with NPDG Algorithm 2, we resample at every iteration for all tested algorithms, except for the L-BFGS method, which exhibits severe instability when trained with randomly resampled batches.

To keep the comparison fair, we keep the same neural network architecture for all the methods tested. We justify the computational efficiency of the proposed methods by summarizing the GPU-time costs of each method for different PDEs with various dimensions in Table 4. The robustness of the proposed method is reflected in the semi-log plots of the relative L^2-loss for different equations. Necessary plots are also provided to visualize the numerical results produced by the proposed method.

The Python codes associated with the examples tested in this section can be accessed from the GitHub repository https://github.com/LSLSliushu/NPDG.

5.1 Poisson's equation (10D, 50D)

We consider the following Poisson's equation defined on the region $\Omega = [0,1]^d$.

$$-\Delta u = f, \text{ on } \Omega, \quad u = g, \text{ on } \partial\Omega, \tag{50}$$

where we define $f(x) = \sum_{k=1}^{d} \frac{\pi^2}{4} \sin(\frac{\pi}{2}x_k)$, and $g(x) = \sum_{k=1}^{d} \sin(\frac{\pi}{2}x_k)$ on $\partial\Omega$. The exact solution of this equation is

$$u_*(x) = \sum_{k=1}^{d} \sin(\frac{\pi}{2}x_k).$$

In this example, by multiplying the dual functions φ and ψ to the equation $-\Delta u = f$, and its boundary condition $u|_{\partial\Omega} = g$, we introduce the loss functional $\mathscr{E} : H^2(\Omega) \times H_0^1(\Omega) \times L^2(\partial\Omega) \to \mathbb{R}$ as

$$\begin{aligned}\mathscr{E}(u;\varphi,\psi) &= \int_\Omega (-\Delta u - f)\varphi \mathrm{d}\mu - \frac{\nu}{2}\int_\Omega \|\nabla\varphi\|^2 \mathrm{d}\mu + \lambda\left(\int_{\partial\Omega}(u-g)\psi \mathrm{d}\mu_{\partial\Omega} - \frac{\nu}{2}\int_{\partial\Omega}\psi^2 \mathrm{d}\mu_{\partial\Omega}\right)\\ &= \int_\Omega \nabla u\cdot\nabla\varphi - f\varphi\,\mathrm{d}\mu - \frac{\nu}{2}\int_\Omega \|\nabla\varphi\|^2 \mathrm{d}\mu + \lambda\left(\int_{\partial\Omega}(u-g)\psi \mathrm{d}\mu_{\partial\Omega} - \frac{\nu}{2}\int_{\partial\Omega}\psi^2 \mathrm{d}\mu_{\partial\Omega}\right).\end{aligned}$$

In practice, we discover that it is helpful to add the $L^2(\partial\Omega, \mu_{\partial\Omega})$ loss functional to $\mathscr{E}(u,\varphi,\psi)$. Thus, we obtain

$$\widetilde{\mathscr{E}}(u,\varphi,\psi) = \mathscr{E}(u,\varphi,\psi) + \lambda\|\mathcal{B}u - g\|^2_{L^2(\partial\Omega,\mu_{\partial\Omega})}.$$

In short, we use the functional $\widetilde{\mathscr{E}}$. We substitute u,φ,ψ with MLPs with number of layers n_l, and tanh as activation,

$$u_\theta = \mathtt{MLP}_{\tanh}(d,256,1,n_l), \quad \varphi_\eta = \mathtt{MLP}_{\tanh}(d,256,1,n_l)\cdot\zeta, \quad \psi_\xi = \mathtt{MLP}_{\tanh}(d,64,1,n_l).$$

Here, we multiply the MLP with the truncation function

$$\zeta(x) = \min_{1\le k\le d}\{x_k, 1-x_k\},$$

in order to enforce $\varphi_\eta \in H_0^1(\Omega)$. Furthermore, based on the definition of $\mathscr{E}$, we set

$$\mathcal{M}_p = \mathcal{M}_d = \nabla$$

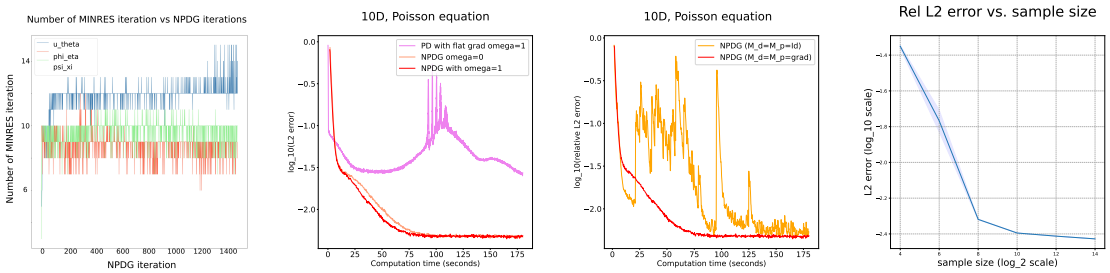
as discussed in Section 3. And recall the definition (24), (21) and (25), we define the preconditioning matrices in the proposed NPDG algorithm as

$$M_p(\theta) = \int_\Omega \frac{\partial}{\partial\theta}(\nabla u_\theta(x))^\top \frac{\partial}{\partial\theta}(\nabla u_\theta(x))\,\mathrm{d}\mu + \lambda\int_{\partial\Omega}\frac{\partial u_\theta(y)}{\partial\theta}^\top \frac{\partial u_\theta(y)}{\partial\theta}\,\mathrm{d}\mu_{\partial\Omega}$$

$$M_d(\eta) = \int_\Omega \frac{\partial}{\partial\eta}(\nabla\varphi_\eta(x))^\top \frac{\partial}{\partial\eta}(\nabla\varphi_\eta(x))\,\mathrm{d}\mu, \quad M_{bdd}(\xi) = \lambda\int_{\partial\Omega}\frac{\partial}{\partial\xi}\psi_\xi(y)^\top\frac{\partial}{\partial\xi}\psi_\xi(y)\mathrm{d}\mu_{\partial\Omega}.$$

In this example, we pick $N_{in} = 2000$ for $d = 10$, $N_{in} = 4000$ for $d = 50$, and $N_{bdd} = 80d$. We set the stepsizes $\tau_u = 0.5 \cdot 10^{-1}$, $\tau_\varphi = \tau_\psi = 0.95 \cdot 10^{-1}$. We test the thresholds $tol_{\text{MINRES}} = 10^{-3}, 10^{-4}$ in the algorithm. Throughout this research, we consider the relative L^2 error of u_θ and ∇u_θ.

In the 10D case, we set $n_l = 4$. We plot MINRES iteration numbers at each NPDG step in Figure 1a. We investigate the effectiveness of our natural(preconditioned)-gradient method by comparing it with the same algorithm using flat gradients. That is, we replace line 7, line 11 in Algorithm 2 by $\eta = \eta + \tau_\varphi \mathbf{w}_\varphi$, $\xi = \xi + \tau_\psi \mathbf{w}_\psi$, and $\theta = \theta - \tau_u \mathbf{w}_u$. This is demonstrated in Figure 1b. In the same plot, it is also observed that the extrapolation step (line 7 of Algorithm 2) will slightly enhance the convergence of the proposed algorithm. Furthermore, choosing suitable preconditioning matrices compatible with the mathematical nature of the PDE is crucial for the proposed method. In Figure 1c, we compare our treatment with the NPDG algorithm with $M_p(\theta), M_d(\eta)$ obtained by setting $\mathcal{M}_p = \mathcal{M}_d = \text{Id}$. As reflected in the plot, naïve preconditioning may lead to instabilities in the optimization procedure. We tested the NPDG algorithms using exponentially growing sample sizes $N_r = 2^k$ and $N_r = 10 \cdot 2^{k-3}$ with $k = 4, 6, 8, 10, 12, 14$. Figure 1d illustrates how the relative L^2 error of u_θ, computed by the NPDG algorithm, varies with the sample complexity. The relative error gradually plateaus as the sample size increases beyond $2^{10} = 1024$. Based on this observation, throughout the remaining examples in this section, we consistently choose sample sizes in the range of $10^3 \sim 10^4$.



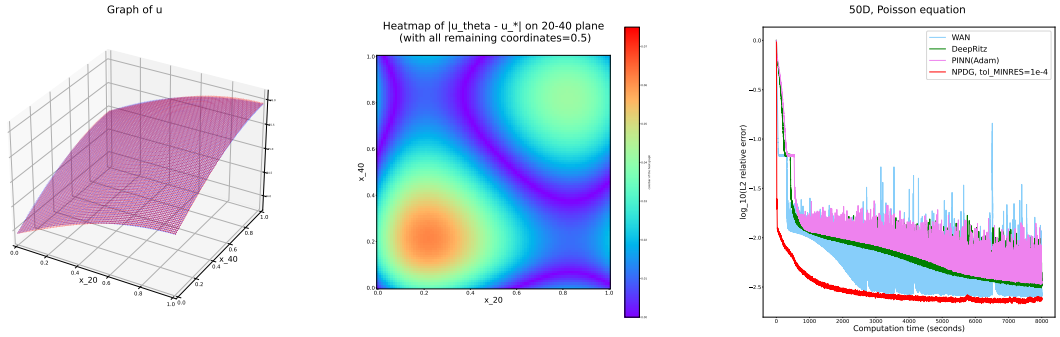
(a) (b) (c) (d)

Figure 1: (10D Poisson equation) 1a: Numbers of iterations required by the MINRES algorithm for updating θ, η, ξ at each NPDG step vs. NPDG iteration. 1b: Comparison with the same algorithm using flat gradients instead (pink), and with the same algorithm without extrapolation ($\omega = 0$) (light red); 1c: Comparison with our NPDG method, but using $M_p(\theta), M_d(\eta)$ obtained by $\mathcal{M}_p = \mathcal{M}_d = \text{Id}$ as our preconditioning (orange). All the plots in these two figures are relative L2 error vs. computational time (seconds). 1d: Log-log plot of relative L^2 error vs. different sample sizes.

In addition, we also test the same example with $d = 50$. We set MLP depth $n_l = 6$. We choose the tolerance $tol_{\text{MINRES}} = 10^{-4}$ to ensure higher accuracy in computing the natural gradient. We compare the algorithms with the PINN, DeepRitz, and WAN methods. The detailed settings for these three methods are provided in Table 2. We run each method up to 8000 seconds and make semi-log plots of relative error vs. computational time for all the methods tested. Figure 2 presents the associated numerical results. The loss plot 2c

suggests that our proposed NPDG algorithm converges faster and more stably compared with the algorithms based on the Adam optimizer.

Furthermore, we record the GPU time spent by each method to achieve a certain accuracy for various dimensions $d = 5, 10, 20, 50$. Details are provided in Table 4 of Appendix E. The proposed method performs more efficiently than the other methods as the dimension d increases.



(a) (b) (c)

Figure 2: (50D Poisson equation) **Left**: Graph of learned u_θ and real solution u_* on the 20-40 coordinate plane (with remaining coordinates equal to $\frac{1}{2}$) in $\mathbb{R}^{50}$; **Middle**: Heatmap of $|u_\theta(x) - u_*(x)|$ on the same plane. **Right**: Semi-log plot of relative L2 error vs. computational time (seconds). The values of $\|u_*\|_{L^2(\Omega,\mu)}$ and $\|\nabla u_*\|_{L^2(\Omega,\mu)}$ are provided in Table 3.

5.2 Elliptic equation with variable coefficients (10D, 20D, 50D)

We consider the following elliptic equation with a variable coefficient

$$-\nabla \cdot (\kappa(x)\nabla u(x)) = f(x), \quad u(y) = g(y) \text{ on } \partial\Omega. \tag{51}$$

Here we assume $\Omega = [-1, 1]^d$ with even dimension d. We set

$$\kappa(x) = \frac{x^\top \Lambda x + 1}{2}, \quad \text{with } \Lambda = \mathrm{diag}(\lambda_0, \lambda_1, \ldots, \lambda_0, \lambda_1),$$

where $\lambda_0 = 1, \lambda_1 = 4$ appear alternately for $\frac{d}{2}$ times, and choose

$$f(x) = -\frac{\mathrm{Tr}(\Lambda^{-1})}{2}(x^\top \Lambda x + 1) - \|x\|^2, \quad \text{and } g(y) = \frac{1}{2} y^\top \Lambda^{-1} y, \ y \in \partial\Omega.$$

The solution to this equation is $u_*(x) = \frac{1}{2} x^\top \Lambda^{-1} x$.

Similar to the previous example, we introduce φ, ψ to the equation and its boundary condition. Integration by parts yields the functional

$$\begin{aligned}\mathscr{E}(u, \varphi, \psi) = &\int_\Omega \kappa(x)\nabla\varphi(x) \cdot \nabla u(x) - f(x)\varphi(x) \, \mathrm{d}\mu - \frac{\nu}{2}\int_\Omega \|\nabla\varphi(x)\|^2 \, \mathrm{d}\mu \\ &+ \lambda\left(\int_{\partial\Omega} (u - g)\psi \, \mathrm{d}\mu_{\partial\Omega} - \frac{\nu}{2}\int_{\partial\Omega} \psi^2 \, \mathrm{d}\mu_{\partial\Omega}\right).\end{aligned}$$

Similarly, we add the boundary loss function to $\mathscr{E}(u,\varphi,\psi)$ to obtain

$$\widetilde{\mathscr{E}}(u,\varphi,\psi)=\mathscr{E}(u,\varphi,\psi)+\lambda\|\mathcal{B}u-g\|^2_{L^2(\partial\Omega,\mu_{\partial\Omega})}.$$

We use $\widetilde{\mathscr{E}}$ in the computation. We set

$$\mathcal{M}_p=\mathcal{M}_d=\nabla$$

for the preconditioning matrices $M_p(\theta), M_d(\eta), M_{bdd}(\xi)$ as defined in (24), (21) and (25).

In this example, we also employ the stronger $H^1(\partial\Omega,\mu_{\partial\Omega})$ boundary loss discussed in Section 3.2.2. In our implementation, the boundary integral of the functional $\mathscr{E}(u,\varphi,\psi)$ is now replaced by

$$\sum_{j=1}^{2d}\int_{S_j}(u-g)\psi+\nabla^{S_j}(u-g)\cdot\nabla^{S_j}\psi\,\mathrm{d}\mu_{\partial\Omega}-\frac{\nu}{2}\int_{S_j}\psi^2+\|\nabla^{S_j}\psi\|^2\,\mathrm{d}\mu_{\partial\Omega}.$$

Here, for each $j=1,\dots,d-1$, we denote

$$S_j^{\pm}:=\Big\{(\hat{x}_1,\dots,\hat{x}_{j-1},\pm1,\hat{x}_j,\dots,\hat{x}_{d-1})\ :\ \hat{x}\in[-1,1]^{d-1}\Big\}$$

as each face of the cubic region Ω. Then $\partial\Omega=\bigcup_{i=1}^d(S_i^+\cup S_i^-)$. ∇^{S_j} denotes the gradient of a function restricted to the face $S_j^{\pm}$. The functional is modified correspondingly as

$$\widetilde{\mathscr{E}}(u,\varphi,\psi)=\mathscr{E}(u,\varphi,\psi)+\lambda\|\mathcal{B}u-g\|^2_{H^1(\partial\Omega,\mu_{\partial\Omega})}.$$

Furthermore, to ensure consistency with the strengthened boundary norm, we modify the preconditioning matrices accordingly. In particular, the matrices $M_p(\theta)$, $M_d(\eta)$ are obtained from (24) and (21) by setting

$$\mathcal{M}_p=\mathcal{M}_d=\sqrt{\kappa(\cdot)}\,\nabla.$$

Moreover, in the definition (24) of the primal preconditioning matrix $\big(M_p(\theta)\big)_{ij}$, the boundary integration is replaced by

$$\lambda\sum_{j=1}^{2d}\int_{S_j}\frac{\partial u_\theta(y)}{\partial\theta_i}\frac{\partial u_\theta(y)}{\partial\theta_j}+\frac{\partial}{\partial\theta_i}\big(\nabla^{S_j}u_\theta(y)\big)\cdot\frac{\partial}{\partial\theta_j}\big(\nabla^{S_j}u_\theta(y)\big)\,\mathrm{d}\mu_{\partial\Omega}. \tag{52}$$

Matrix $(M_{bdd}(\xi))_{ij}$ is reformulated analogously to (52), with the partial derivatives $\frac{\partial}{\partial\theta}$ and the function u_θ replaced by $\frac{\partial}{\partial\xi}$ and ψ_ξ, respectively.

We test this example with $d=10,20,50$. We substitute u,φ,ψ with MLPs with softplus$(\cdot)$ as activation functions. Here, softplus$(\cdot)$ is a smooth approximation of the ReLU function defined as[4]

$$\mathrm{softplus}(x)=\frac{1}{\beta}\log(1+\exp(\beta x))$$

with $\beta=\frac{1}{4}$. We summarize the neural net architecture of our experiments in Table 1. Similar to our treatment for the Poisson's equation, we multiply φ_η by the truncation function $\zeta(\cdot)$ to enforce $\varphi_\eta\in H_0^1(\Omega)$.

4. In PyTorch, for numerical stability, the implementation of softplus$(\cdot)$ reverts to the linear function when $x>\frac{\text{threshold}}{\beta}$. The default value for the threshold equals 20.

	Primal & Dual Neural Networks			N_{in}, N_{bdd}	$\tau_u, \tau_\varphi, \tau_\psi$	MINRES tol
	u_θ	φ_η	ψ_ξ			
$d = 10$	$(d, 256, 1, 4)$	$(d, 256, 1, 4)$	$(d, 128, 1, 4)$	$4000, 80d$	$0.1, 0.19, 0.19$	$0.5 \cdot 10^{-3}$
$d = 20$					$0.05, 0.095, 0.095$	
$d = 50$	$(d, 256, 1, 6)$	$(d, 256, 1, 6)$	$(d, 128, 1, 6)$	$6000, 80d$		10^{-4}

Table 1: Basic setting of our experiments on computing (51).

In this example, for all dimensions $d = 10, 20, 50$, the stepsizes $\tau_u, \tau_\varphi, \tau_\psi$, the number of samples N_{in}, N_{bdd}, as well as the tolerance of MINRES, are summarized in Table 1. We improve the tolerance of the MINRES algorithm from $0.5 \cdot 10^{-3}$ to 10^{-4} as the dimension d increases to 50. We run the proposed method for 500 and 1000 seconds for 10D and 20D problems. For $d = 50$, we perform the proposed method using $L^2(\partial\Omega, \mu_{\partial\Omega})$ boundary loss for 36000 iterations and the method using $H^1(\partial\Omega, \mu_{\partial\Omega})$ loss for 3000 iterations. For all $d = 10, 20, 50$, we compare the algorithm with the PINN, DeepRitz, and WAN methods. The detailed settings for these three methods are provided in Table 2. We make semi-log/log-log plots of relative error vs. computational time for all methods. The error plots are presented in Figure 3. The plots justify the linear convergence of the proposed method. The experimental results further demonstrate improved convergence rates and accuracy for the NPDG algorithms, both with and without the use of a stronger boundary norm. Compared with the other algorithms based on Adam optimizers, the proposed method performs more stably and achieves higher accuracy in this example. We also record the GPU time spent by each method to achieve a certain accuracy. One can find the details in Table 4 of Appendix E. It turns out that only the proposed method can achieve an accuracy such that $\frac{\|u_\theta - u_*\|_{L^2(\Omega,\mu)}}{\|u_*\|_{L^2(\Omega,\mu)}} \leq 0.005$.

For $d = 20$, we visualize the solution u_θ learned by the NPDG algorithm by plotting the graph of u_θ on the $9 - 10$ plane while fixing the remaining coordinates to 0 and 0.5 for $d = 20$ in Figure 4. The associated heatmaps of $|u_\theta(x) - u_*(x)|$ on the $9 - 10$ plane are also provided in Figure 4. To investigate the accuracy of u_θ over the entire space of Ω, we separate $\Omega = \bigcup_{l=1}^{50} \Omega_l$ into 50 square shells with gradually increasing sizes,

$$\Omega_l := \{x = (x_1, \ldots, x_d)^\top \in \mathbb{R}^d | (l-1)/50 \leq |x_k| < l/50,\ 1 \leq k \leq d\}.$$

We plot the average L^2 error of u_θ computed via different methods on Ω_l with respect to the size $l/50$ of each square shell Ω_l in Figure 4e.

Different MINRES tolerances: Slightly improving (i.e., decreasing) the tolerance tol_{MINRES} of the MINRES algorithm yields more accurate directions of the natural gradients and enhances the convergence of the NPDG algorithm. However, selecting tol_{MINRES} too small makes the algorithm sensitive with respect to data stochasticity and thus may introduce instability to the method. This is reflected in Figure 5a and 5b.

Comparing with L-BFGS optimizer: We apply the L-BFGS optimizer to PINN and compare its convergence speed with the proposed method. L-BFGS utilizes the second-order information from the loss function in optimization. However, L-BFGS is known to be unstable in stochastic settings–using random batches is not a feasible strategy for L-BFGS method. In this example, we fix the Monte-Carlo samples in the algorithm and optimize the

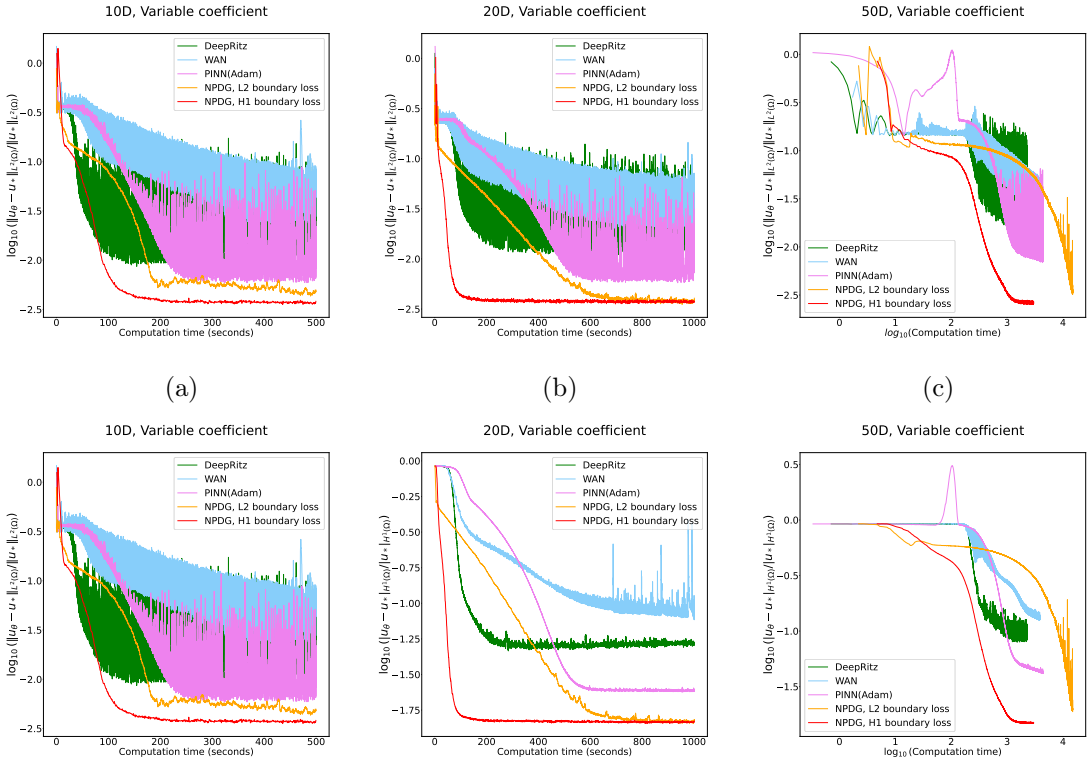


Figure 3: **Left** column (3a) (3d): Semi-log plot (up) of relative L2 error vs. computational time(seconds) and semi-log plot (down) of relative H^1 seminorm error $(\frac{\|\nabla u_\theta - \nabla u_*\|_{L^2(\Omega,\mu)}}{\|\nabla u_*\|_{L^2(\Omega,\mu)}})$ vs. computational time. Dimension $d = 10$; **Middle** column (3b) (3e): The same plots for $d = 20$; **Right** column (3c) (3f): The same plots (but in Log-log form) for $d = 50$. The values of $\|u_*\|_{L^2(\Omega,\mu)}$ and $\|\nabla u_*\|_{L^2(\Omega,\mu)}$ are provided in Table 3.

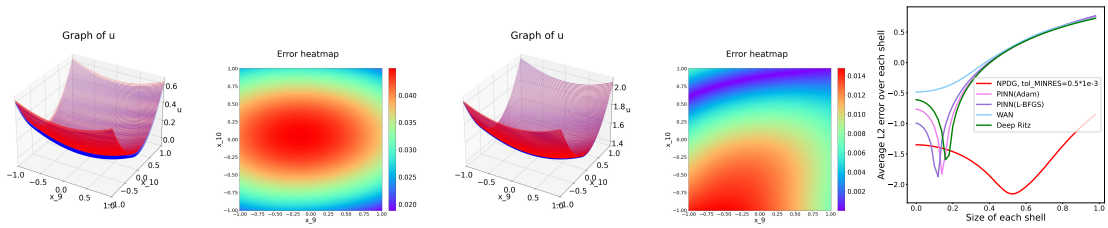
PINN loss function with L-BFGS method. For $d = 20$, as shown in Figure 5c, our NPDG algorithm with $tol_{\text{MINRES}} = 10^{-4}$ converges faster than the L-BFGS method. Moreover, the L-BFGS method faces instability even without data stochasticity. As demonstrated in Figure 5d, the L-BFGS method always blows up given a long enough running time for dimensions $d = 20$ and $d = 50$.

5.3 Nonlinear elliptic equation (5D)

We consider the following nonlinear elliptic equation equipped with Dirichlet boundary condition on a d-dimensional ball with radius $R = 3$

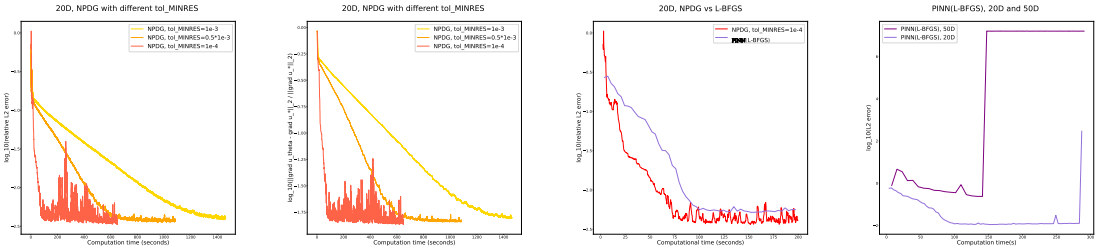
$$B_{d,R} = \{x \in \mathbb{R}^d \mid \|x\| \leq R\}.$$

$$\frac{1}{2}\|\nabla u(x)\|^2 + V(x) = \Delta u(x), \quad u|_{\partial B_{d,R}} = 0. \tag{53}$$



(a) (b) (c) (d) (e)

Figure 4: Plots for $d = 20$. 4a: Graph of u_θ obtained by NPDG method (blue) with real solution (red) plotted on $9-10$ plane with remaining coordinates fixed to 0; 4b Heatmap of error $|u_\theta(x) - u_*(x)|$ plotted on $9-10$ plane with remaining coordinates fixed to 0. 4c, 4d: Same plots plotted on $9-10$ plane with remaining coordinates fixed to 0.5; 4e: Semi-log plot of $\log_{10}\left(\frac{1}{|\Omega_l|}\|u_\theta - u_*\|_{L^2(\Omega_l)}\right)$ vs. size of each square shell Ω_l.



(a) (b) (c) (d)

Figure 5: Figures 5a, 5b: Plots of relative error vs. computation time(seconds) with different $tol_{\text{MINRES}} = 10^{-4}, 0.5\cdot 10^{-3}, 10^{-3}$. Figure 5c: Plot of relative L^2 error vs. computation time (seconds), we compare NPDG with $tol_{\text{MINRES}} = 10^{-4}$ to PINN using L-BFGS optimizer. Figure 5d: Long-time behavior of L-BFGS optimizer when applied to 20D and 50D problems.

Here we set

$$V(x) = -\frac{\pi^2}{8}\sin^2(\frac{\pi}{2}r) - \frac{\pi^2}{4}\cos(\frac{\pi}{2}r) - \frac{\pi(d-1)}{2r}\sin(\frac{\pi}{2}r)$$

with $r = \|x\|$. The solution to this equation is the radial function

$$u_*(x) = \cos(\frac{\pi}{2}r).$$

Similar to the previous examples, we introduce φ, ψ to the equation and its boundary condition. We consider solving $\inf_u \sup_{\varphi,\psi} \widetilde{\mathscr{E}}(u,\varphi,\psi) := \mathscr{E}(u,\varphi,\psi) + \lambda\|\mathcal{B}u\|^2_{L^2(\mu_{\partial\Omega})}$ with

$$\mathscr{E}(u,\varphi,\psi) = \left(\int_\Omega \nabla\varphi(x)\cdot\nabla u(x) + \frac{1}{2}\|\nabla u(x)\|^2\varphi(x) + V(x)\varphi(x)\,\mathrm{d}\mu - \frac{\nu}{2}\int_\Omega \|\nabla\varphi(x)\|^2\,\mathrm{d}\mu\right)$$
$$+\lambda\left(\int_{\partial\Omega} u\psi\,\mathrm{d}\mu_{\partial\Omega} - \frac{\nu}{2}\int_{\partial\Omega}\psi^2\,\mathrm{d}\mu_{\partial\Omega}\right).$$

It is still unclear what is the optimal way to precondition the nonlinear term in this equation. In our treatment, we only focus on the linear part Δu and set

$$\mathcal{M}_p = \mathcal{M}_d = \nabla$$

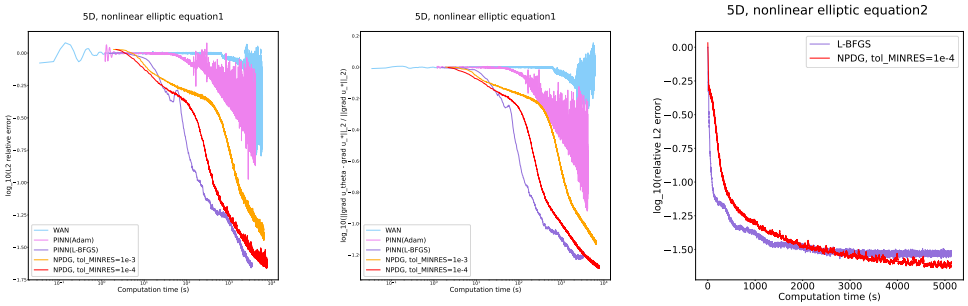
for the preconditioning matrices $M_p(\theta), M_d(\eta), M_{bdd}(\xi)$.

We test this example with $d = 5$, we set

$$u_\theta = \texttt{MLP}_{\tanh}(d, 256, 1, 4),\ \varphi_\eta = \texttt{MLP}_{\tanh}(d, 256, 1, 4), \quad \psi_\xi = \texttt{MLP}_{\tanh}(d, 128, 1, 4).$$

The stepsizes are chosen as $\tau_u = 0.05, \tau_\varphi = 0.095, \tau_\psi = 0.095$. We apply Monte-Carlo method to evaluate the loss function, in order to sample uniformly from $B_{d,R}$, we first randomly sample $N_{in} = 4000$ points $\rho_1, \ldots, \rho_{N_{in}}$ from the interval $[0, R]$ following the density function $p(\rho) = \frac{d+1}{R}\left(\frac{\rho}{R}\right)^d, \rho \in [0, R]$[5]. Then we sample N_{in} points $\mathbf{w}_1, \ldots, \mathbf{w}_{N_{in}}$ from the standard Gaussian distribution $\mathcal{N}(0, I_d)$. Thus, we obtain N_{in} sample points in $B_{d,R}$ by forming $x_i = \rho_i \frac{\mathbf{w}_i}{\|\mathbf{w}_i\| + e_0}$, $1 \le i \le N_{in}$. We add $e_0 = 10^{-8}$ to prevent division by zero. We run the proposed method for $N_{iter} = 10000$ iterations.

In this example, we also test the PINN(Adam/L-BFGS) and WAN methods. The hyperparameters for these methods are provided in Table 2. Log-log plots of the relative error vs. the computation time among the methods are provided in Figure 6. We plot the graph



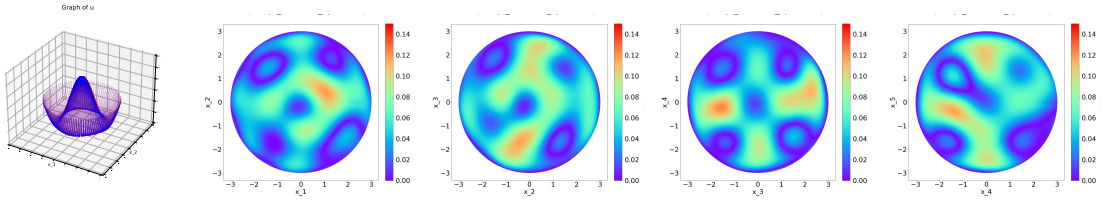
(a) (b) (c)

Figure 6: Equation (53): **Left**: Log-log plot of relative L^2 error vs. computational time (seconds); **Middle**: Log-log plot of relative H^1 seminorm error vs. computational time (seconds). The values of $\|u_*\|_{L^2(\Omega,\mu)}$ and $\|\nabla u_*\|_{L^2(\Omega,\mu)}$ are provided in Table 3. Equation (54): **Right**: Semi-log plot of relative L2 error vs. computational time.

of u_θ obtained by the algorithm on the $1-2$ coordinate plane in Figure 7a. We also plot the heat maps of the error function $|u_\theta(\cdot) - u_*(\cdot)|$ on various coordinate planes in Figures 7b-7e. Similar to previous examples, we record the GPU times spent by different methods for achieving certain accuracy in Table 4 of Appendix E. Furthermore, we also consider the following equation on the same region $B_{d,R}$ ($d = 5$, $R = 3$) with a weaker nonlinear term,

$$\frac{\epsilon_0}{2}\|\nabla u(x)\|^2 + \Delta u(x) = V(x), \quad u|_{\partial B_{d,R}} = 0. \tag{54}$$

5. This can be done by first sampling n_ρ points $r_1, \ldots, r_{n_\rho}$ uniformly from $[0, 1]$ and then transforming each r_i to $\rho_i = r_i^{\frac{1}{d}} \cdot R^{1-\frac{1}{d}}$ for $1 \le i \le n_\rho$.



(a) (b) (c) (d) (e)

Figure 7: Figure 7a: Graph of u_θ on the $1-2$ coordinate plane (that is, the plane spanned by the first and second components with the remaining coordinates fixed to 0). The parameter θ is obtained by the NPDG method after 10000 iterations; Figures 7b-7e: Heatmaps of $|u_\theta(\cdot) - u_*(\cdot)|$ plotted on $1-2$, $2-3$, $3-4$, $4-5$ coordinate planes.

Here we set $\epsilon_0 = \frac{1}{10}$ and

$$V(x) = \frac{\epsilon_0 \pi^2}{8} \sin^2(\frac{\pi}{2} r) - \frac{\pi^2}{4} \cos(\frac{\pi}{2} r) - \frac{\pi(d-1)}{2r} \sin(\frac{\pi}{2} r).$$

The solution to this equation is still $u_*(x) = \cos(\frac{\pi}{2} r)$. We apply the NPDG algorithm with exactly the same neural network architecture and hyperparameters as in (53) to solve equation (54). We also test the L-BFGS optimizer to minimize the PINN loss of equation (54). Figure 6c indicates that the proposed method achieves performance that is comparable with L-BFGS in this example.

5.4 Allen-Cahn equation

We have discussed several examples of time-independent PDEs. We now briefly show how the proposed method is applied to resolve the time-implicit, semi-discrete schemes of the time-dependent equations. In this section, we primarily focus on the 1D and 2D Allen-Cahn equations to illustrate the main idea. Future research will explore additional approaches, such as adaptive sampling techniques (Wight and Zhao, 2020) and extensions to higher dimensions.

We consider the Allen-Cahn equation on a bounded domain Ω posed with the homogeneous Neumann boundary condition on time interval $[0, T]$.

$$\frac{\partial u(x,t)}{\partial t} = \epsilon_0 \Delta u(x,t) - \frac{1}{\epsilon_0} W'(u), \quad \frac{\partial u}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \quad u(\cdot, 0) = u_0(\cdot).$$

Here we define the double-well potential function $W(u) = \frac{1}{4}(1-u^2)^2$, with $W'(u) = u^3 - u$. It is well-known that the Allen-Cahn equation can be viewed as the L^2-gradient flow of the energy functional $E(u) = \int_\Omega \frac{\epsilon_0}{2} \|\nabla u\|^2 + \frac{1}{2\epsilon_0} W(u) \, \mathrm{d}x$.

In this research, we focus on resolving the time-implicit, semi-discrete numerical scheme of this equation. We divide the time interval into N_t subintervals and consider

$$\frac{u^t(x) - u^{t-1}(x)}{h_t} = \epsilon_0 \Delta u^t(x) - \frac{1}{\epsilon_0} W'(u^t(x)), \ \frac{\partial u^t}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega,$$

sequentially for $1 \leq t \leq N_t$ with $u^0(\cdot)$ set as $u_0(\cdot)$. One motivation for considering this implicit scheme is its energy stability, in the sense that $E(u^t) \leq E(u^{t-1})$, which respects the gradient-flow nature of the equation.

The problem boils down to solving N_t consecutive elliptic equations with a cubic term as shown below,

$$u^t(x) - \epsilon_0 h_t \Delta u^t(x) + \frac{h_t}{\epsilon_0}((u^t(x))^3 - u^t(x)) = u^{t-1}(x), \ \frac{\partial u^t}{\partial \mathbf{n}} = 0 \text{ on } \partial\Omega, \quad 1 \leq t \leq N_t. \ (55)$$

In the implementation, we substitute the primal function u, and the test functions φ, ψ with neural networks with tanh activations,

$$u_\theta = \texttt{MLP}_{\tanh}(d, 128, 1, 5), \quad \varphi_\eta = \texttt{MLP}_{\tanh}(d, 128, 1, 5) \cdot \zeta, \quad \psi_\xi = \texttt{MLP}_{\tanh}(d, 64, 1, 5).$$

We adopt different preconditioning strategies based on the magnitude of ϵ_0. A detailed discussion is provided in Appendix F.

1D example We first test the algorithm on the 1D example with $\Omega = [0, 2]$, initial data $u_0(x) = (1 - \cos(\pi(x-1)))\cos(\pi(x-1))$. We work on two cases in which $\epsilon_0 = 0.1, T = 1, N_t = 10$; and $\epsilon_0 = 0.01, T = 0.08, N_t = 80$. We treat the distribution $\mu_{\partial\Omega} = \frac{1}{2}(\delta_0 + \delta_2)$ where δ_x denotes the Dirac measure[6] concentrated on the point $x \in \mathbb{R}$.

For the algorithm, we set $N_{in} = 2000$ and $N_{bdd} = 2$. Since $\partial\Omega = \{0, 2\}$, one boundary sample is assigned to each endpoint. The boundary loss coefficient is chosen as $\lambda = 1$. We keep $\tau_u = 0.05$, $\tau_\varphi = 0.095$, and $\tau_\psi = 0.095$ for $\epsilon_0 = 0.1$, and choose smaller stepsizes $\tau_u = 0.01$, $\tau_\varphi = 0.02$, and $\tau_\psi = 0.02$ for $\epsilon_0 = 0.01$.

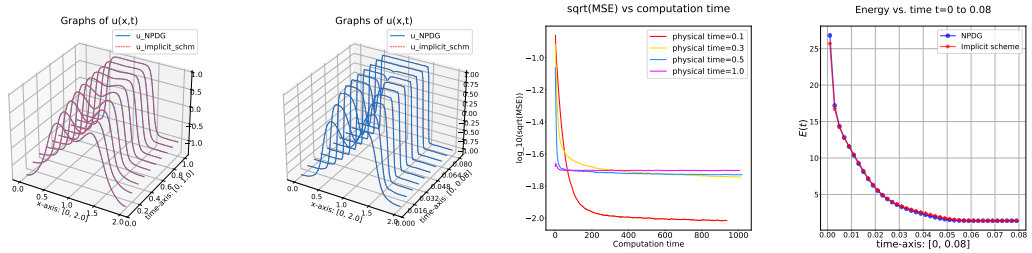
In Figure 8, we plot the graphs of our numerical solution u_{θ_k} obtained at different time nodes $t_k = \frac{k}{N_t}$ $(1 \leq k \leq N_t)$ with the benchmark solution $\{U^k\}_{k=1}^{N_t}$ solved from time-implicit, finite difference scheme (94) provided in Appendix F.2. The semi-log curve of $\sqrt{\frac{1}{N_x}\sum_{i=1}^{N_x}(u_{\theta_k}(x_i) - U_i^k)^2}$ vs. the computation time ($\epsilon_0 = 0.1$) is presented in Figure 8c. The energy decay plot of $E(u_{\theta_k})$ versus t_k for $\epsilon_0 = 0.01$ is included in Figure 8d. The numerical solution u_{θ_k} exhibits monotonic decay of its energy and shows agreement with the benchmark solution.

2D example We further consider a 2D Allen-Cahn equation with $\Omega = [0, 2]^2$, $\epsilon_0 = 0.1$ and the initial condition $u_0(x) = \tanh\left(-\frac{\|x - x_0\| - R}{s}\right)$ with $x_0 = (1, 1)^\top$, $R = 0.5$ and $s = 0.1$. We set $T = 1.5$ and $N_t = 15$. We keep the hyperparameters of the NPDG algorithm the same as the previous example except that we set $N_{iter} = 1000$. The numerical results are provided in Appendix F.2.

5.5 Monge-Ampère equation for the L^2-Optimal Transport problem

In this section, we focus on the computation of the Monge-Ampère equation (27). A PINN solver for this equation is proposed in (Singh et al., 2021). Deep learning algorithms from

6. That is, $\delta_x(E) = 1$ for any measurable set $E \subset \mathbb{R}$ that contains x, and $\delta_x(E) = 0$ for measurable sets that do not contain x.



(a) $\epsilon_0 = 0.1$ (b) $\epsilon_0 = 0.01$ (c) MSE vs. walltime (d) Energy decay

Figure 8: 8a&8b: The graph of $u_{\theta_k}(\cdot)$ (blue) obtained from the NPDG algorithm for $t_k = \frac{k}{N_t}$, $1 \le k \le N_t$ together with the benchmark solution (red, dashed line). 8c: Semi-log plots of the $\sqrt{\text{MSE loss}}$ vs. computation time (seconds) at physical time $0.1, 0.3, 0.5, 1.0$. 8d: Plot of energy $E(u_{\theta_k})$ versus time t_k.

the optimal transport perspective are discussed in (Korotin et al., 2019; Makkuva et al., 2020; Fan et al., 2023), among other references.

As discussed in Section 2.5.1, solving the equation is equivalent to solving the L^2-optimal transport problem. This can be further reduced to a sup-inf saddle point problem (30). In this research, we assume that the samples of μ_0, μ_1 are available. In order to evaluate the functional $\mathscr{E}(T_\theta, \varphi_\eta)$, we generate samples $\{\boldsymbol{X}_i\}_{i=1}^N \sim \mu_0 = \rho_0 \mathrm{d}x$ and $\{\boldsymbol{Y}_i\}_{i=1}^N \sim \mu_1 = \rho_1 \mathrm{d}y$ and apply the Monte-Carlo algorithm,

$$\mathscr{E}(T_\theta, \varphi_\eta) \approx \frac{1}{N} \sum_{i=1}^{N} \frac{1}{2} \|\boldsymbol{X}_i - T_\theta(\boldsymbol{X}_i)\|^2 + \varphi_\eta(T_\theta(\boldsymbol{X}_i)) - \varphi_\eta(\boldsymbol{Y}_i).$$

By applying Algorithm 1, we calculate the natural (preconditioned) gradients of $\mathscr{E}(T_\theta, \varphi_\eta)$ with respect to θ, η. We then apply the NPDG algorithm 2 to solve the saddle point problem (30) for $T_*(\cdot)$ $(\nabla u(\cdot))$.

In experiments, we use the Primal-Dual algorithm with the Adam optimizer (PD-Adam) proposed in (Fan et al., 2023) as a benchmark for the proposed method. A brief description of this method, as well as its hyperparameters used in all tests, are provided in Appendix G. We test three numerical examples as a demonstration. The first two examples possess explicit formulas for the OT maps. In the third example, we compute the OT map from standard Gaussian to mixed Gaussian distributions embedded in 10D and 50D spaces. In the implementation, we set $T_\theta(\cdot)$, $\varphi_\eta(\cdot)$ as MLP with PReLU activation function

$$\mathrm{PReLU}(x) = \begin{cases} x, & \text{if } x \ge 0 \\ ax, & \text{otherwise,} \end{cases}$$

where $a \in \mathbb{R}$ is a learnable parameter. The Input Convex Neural Networks (ICNN) architecture (Amos et al., 2017) advocated in (Makkuva et al., 2020) will be considered in future research.

5.5.1 1D Gaussian to mixed Gaussian

We set $\rho_0 = \mathcal{N}(0,1)$, $\rho_1 = \sum_{k=1}^m \lambda_k \mathcal{N}(\mu_k, \sigma_k^2)$ with $\lambda_k > 0$, $\sum_{k=1}^m \lambda_k = 1$, $\mu_k \in \mathbb{R}$, $\sigma_k > 0$. The optimal transport map takes the explicit form,

$$T_*(x) = F_1^{-1}(F_0(x)), \quad F_0(x) = \sum_{k=1}^m \frac{\lambda_k}{2}(1 + \operatorname{erf}\left(\frac{x-\mu_k}{\sqrt{2}\sigma_k}\right)), \quad F_1^{-1}(y) = \operatorname{erf}^{-1}(2y-1).$$

In the example, we consider $m = 2$, $\lambda_1 = \frac{2}{3}, \mu_1 = -1, \sigma_1 = 0.5$; $\lambda_2 = \frac{1}{3}, \mu_2 = 1, \sigma_2 = 0.5$. We set $T_\theta(\cdot)$ and φ_η as

$$T_\theta = \texttt{MLP}_{\text{PReLU}}(1, 50, 1, 3), \ \varphi_\eta = \texttt{MLP}_{\text{PReLU}}(1, 50, 1, 3).$$

We set the sample size $N = 800$, $\omega = 1$, and $\tau_u = \tau_\varphi = 1.5 \cdot 10^{-1}$. We perform the NPDG algorithm for 6000 iterations. Figure 9a demonstrates the semi-log plots of the $L^2(\rho_0)$ error $\|T_\theta - T_*\|_{L^2(\rho_0)}$ versus the computation time. We make comparisons among the NPDG algorithms with different preconditioners ((31) and (35)), as well as the PD-Adam method.

5.5.2 5D Gaussian to Gaussian

For $\mu_0, \mu_1 \in \mathbb{R}^5$ and positive-definite symmetric matrices $\Sigma_0, \Sigma_1 \in \mathbb{R}^{5\times 5}$, we set $\rho_0 = \mathcal{N}(\mu_0, \Sigma_0)$, $\rho_1 = \mathcal{N}(\mu_1, \Sigma_1)$. One can verify that the OT map takes the affine form $T_*(\mathbf{x}) = A\mathbf{x} + b$ with

$$A = \sqrt{\Sigma_0}^{-1}(\sqrt{\Sigma_0}\Sigma_1\sqrt{\Sigma_0})^{1/2}\sqrt{\Sigma_0}^{-1}, \quad b = \mu_1 - A\mu_0.$$

For simplicity, we set $\mu_0 = \mu_1 = 0$ in the test example. The cases in which $\mu_0 \neq \mu_1$ can be readily handled by the pre-translating technique introduced in (Kuang and Tabak, 2017), which reduces the problem to the case in which $\mu_0 = \mu_1$. We define

$$\Sigma_0 = \operatorname{diag}(1/4, 1, 1, 1), \quad \Sigma_1 = \operatorname{diag}(1, 1/4, 1) \oplus \begin{bmatrix} 5/8 & 3/8 \\ 3/8 & 5/8 \end{bmatrix}.$$

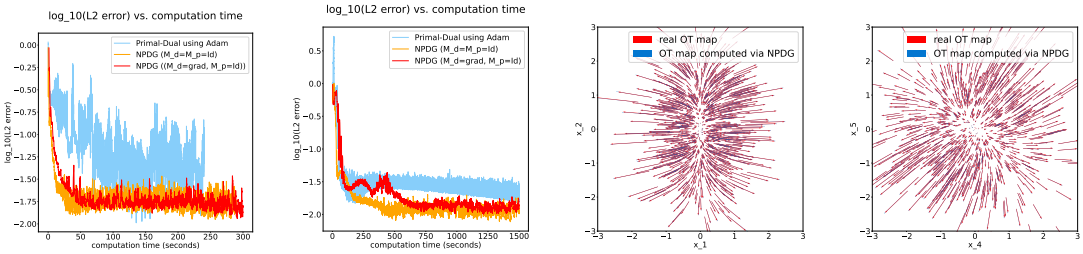
Then the OT map is given by $T_*(x) = \sqrt{\Sigma_0^{-1}\Sigma_1}x$, with

$$\sqrt{\Sigma_0^{-1}\Sigma_1} = \operatorname{diag}(2, 1/2, 1) \oplus \begin{bmatrix} 3/4 & 1/4 \\ 1/4 & 3/4 \end{bmatrix}.$$

We set $T_\theta(\cdot)$ and φ_η as

$$T_\theta = \texttt{MLP}_{\text{PReLU}}(5, 80, 5, 4), \ \varphi_\eta = \texttt{MLP}_{\text{PReLU}}(5, 80, 1, 4).$$

We set the sample size $N = 2000$, $\omega = 1$, and $\tau_u = 0.5 \cdot 10^{-1}, \tau_\varphi = 0.95 \cdot 10^{-1}$. We perform the NPDG algorithm for 20000 iterations. Similar to the previous example, we present the semi-log plots of $L^2(\rho_0)$ error vs computation time in Figure (9b). The plots of the computed transportation map $T_\theta(\cdot)$ together with $T_*(\cdot)$ are provided in Figure (9c) and (9d).



(a) (b) (c) (d)

Figure 9: **OT problem (1D, 5D)**: 9a: Semi-log plots of $\|T_\theta - T_*\|_{L^2(\rho_0)}$ vs computation time (seconds) for the 1D problem discussed in Section 5.5.1; 9b: Semi-log plots of $\|T_\theta - T_*\|_{L^2(\rho_0)}$ vs computation time (seconds) for the 5D problem discussed in Section 5.5.2; 9c: Plot of the computed transport map $T_\theta(\cdot)$ (blue) with real OT map $T_*(\cdot)$ (red) on 1-2 plane; 9d: Plot of the computed transport map (blue) with real OT map (red) on 4-5 plane.

5.5.3 High dimensional Gaussian to mixed Gaussian (10D, 50D)

We consider the mixed-Gaussian distribution $\sum_{k=1}^{8} \lambda_k \mathcal{N}(\mu_k, \sigma_k^2 I)$ defined on $\mathbb{R}^d$, where

$$\mu_k = \left(0, \ldots, R\cos\left(\frac{k}{4}\pi\right), \ldots, R\sin\left(\frac{k}{4}\pi\right), \ldots, 0\right)^\top \text{ with } R = 3, \quad \sigma_k = \frac{4}{25}.$$

We assume that the two nonzero entries of μ_k are located in the i_0 and i_1 entries. We denote ρ_a as equal mixed-Gaussian

$$\rho_a = \sum_{k=1}^{8} \lambda_k \mathcal{N}(\mu_k, \sigma_k^2 I), \quad \lambda_k = \frac{1}{8},\ 1 \le k \le 8;$$

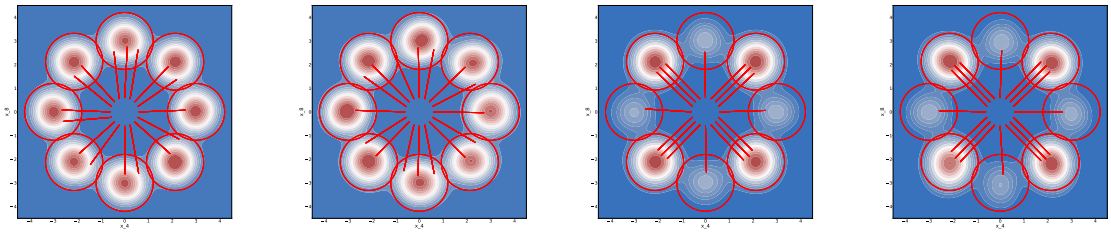
we denote ρ_b as a non-equally distributed mixed-Gaussian distribution with

$$\rho_b = \sum_{k=1}^{8} \lambda_k \mathcal{N}(\mu_k, \sigma_k^2 I), \quad \lambda_k = \begin{cases} \frac{1}{5} & k \text{ is even}, \\ \frac{1}{20} & k \text{ is odd}. \end{cases},\ 1 \le k \le 8.$$

Consider $\rho_0 = \mathcal{N}(0, I)$. We compute the optimal transport from ρ_0 to ρ_a, as well as ρ_0 to ρ_b, by solving the sup-inf problem (30) using the NPDG algorithm. In the implementation, we always set

$$u_\theta(\cdot) = \texttt{MLP}_{\text{PReLU}}(d, 120, d, 6),\ \varphi_\eta(\cdot) = \texttt{MLP}_{\text{PReLU}}(d, 120, 1, 6).$$

We first test the algorithm by setting $d = 10$, and $i_0 = 4, i_1 = 8$. We choose (31) as preconditioners for NPDG algorithm. we choose the sample size $N = 2000$, $\omega = 1$, and $\tau_u = 0.5 \cdot 10^{-2}, \tau_\varphi = 0.95 \cdot 10^{-2}$; we perform the NPDG algorithm for 15000 iterations. We compute the optimal transport maps from ρ_0 to ρ_a and ρ_0 to ρ_b by applying the NPDG algorithm and the PD-Adam method. We compare the computational results in Figure 10. The pushforwarded distribution $T_{\theta\sharp}\rho_0$ of the proposed method outperforms PD-Adam in terms of homogeneity and shape of the mixed Gaussians.



(a) NPDG (b) PD-Adam (c) NPDG (d) PD-Adam

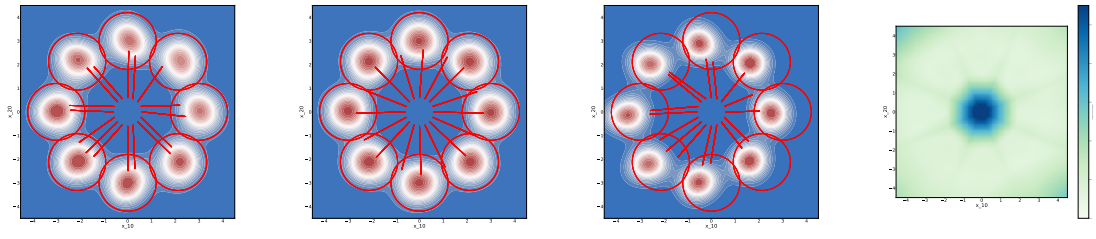
Figure 10: **OT problem (10D)**: Plots of the pushforwarded density $T_{\theta\sharp}\rho_0$ by using Kernel Density Estimation (KDE), together with the optimal transport map (red segments). **Left two figures**: OT from ρ_0 to ρ_a, 10a: Numerical result obtained by NPDG, 10b Numerical result obtained by PD-Adam; **Right two figures**: OT from ρ_0 to ρ_b, 10c: Numerical result obtained by NPDG, 10d: Numerical result obtained by PD-Adam. All figures are plotted on the $4-8$ plane.

We further consider the OT problem with dimension $d = 50$ with $i_0 = 10, i_1 = 20$ in which the NPDG algorithm performs more robustly and achieves more accurate solutions compared to the PD-Adam algorithm. We set $tol_{\text{MINRES}} = 10^{-4}$. We choose the sample size $N = 2000$, the extrapolation coefficient $\omega = 5$ and stepsizes $\tau_u = \tau_\varphi = 0.5 \cdot 10^{-2}$. We perform the NPDG algorithm for 20000 iterations.

We first test the case of transporting ρ_0 to equally distributed mixed-Gaussian distribution ρ_a. We test the NPDG algorithm with various preconditioning (31), (35), as well as the PD-Adam method. The results are presented in Figure 11. It is worth mentioning that upon comparing the transport maps shown in Figure 11a and 11b, the more canonical precondition (35) yields a solution with higher accuracy. We then test the case of transporting ρ_0 to non-equal mixed-Gaussian distribution ρ_b. The results are presented in Figure 12. Again, our NPDG algorithm with precondition (35) produces the transport map with better quality. Further plots on the numerical solutions can be found in Appendix H. The PD-Adam method does not behave as robustly as the NPDG algorithm in this 50D example.

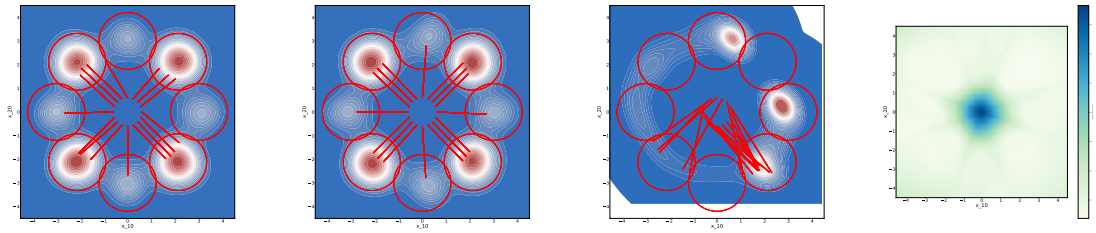
6. Discussions

In this paper, we design a preconditioned adversarial training algorithm called Natural Primal-Dual Hybrid Gradient (NPDG) for solving various PDEs. We incorporate the precondition operators $\mathcal{M}_p, \mathcal{M}_d$ in the precondition matrices $M_p(\theta), M_d(\eta)$ for computing the natural gradients. Alternative gradient descent and ascent algorithms, together with suitable extrapolation, are utilized to update the primal and dual neural network parameters. Linear convergence guarantees are established for the time-continuous version of the NPDG algorithm. In practice, we apply the MINRES iterative solver to handle natural gradients efficiently. The proposed algorithm outperforms classical machine learning–based approaches—including PINNs (Adam/LBFGS), the Deep Ritz method, and the Weak Adver-



(a) NPDG with (31) (b) NPDG with (35) (c) PD-Adam (d) Heat graph of $\varphi_\eta(\cdot)$

Figure 11: **OT problem from ρ_0 to ρ_a (50D)**: Plots of the pushforwarded density $T_{\theta\sharp}\rho_0$ by using Kernel Density Estimation (KDE). 11a-11c: Numerical results produced by NPDG method and PD-Adam method. 11d: heat graph of the Kantorovich dual function $\varphi_\eta(\cdot)$ learned from NPDG algorithm with precondition (35). All figures are plotted on the $10-20$ coordinate plane.



(a) NPDG with (31) (b) NPDG with (35) (c) PD-Adam (d) Heat graph of $\varphi_\eta(\cdot)$

Figure 12: **OT problem from ρ_0 to ρ_b (50D)**: Plots of the pushforwarded density $T_{\theta\sharp}\rho_0$ by using Kernel Density Estimation (KDE). 12a-12c: Numerical results produced by NPDG method and PD-Adam method. 12d: heat graph of the Kantorovich dual function $\varphi_\eta(\cdot)$ learned from NPDG algorithm with precondition (35). All figures are plotted on the $10-20$ coordinate plane.

sarial Network / Primal–Dual Adam algorithm—in terms of convergence speed, robustness, and accuracy across various classes of PDEs, particularly in high-dimensional settings.

Based on the numerical experiments, several critical questions about the proposed algorithm have arisen. The first concerns the convergence analysis of the time-discrete NPDG algorithm—namely, what are the optimal step sizes $\tau_u, \tau_\varphi, \tau_\psi$? Is it possible to adopt adaptive step sizes? Another crucial aspect that warrants more investigation is reducing the computational burden and improving the accuracy of the method by adopting refined strategies for evaluating natural gradients, such as Kronecker-factored Approximate Curvature (KFAC) (Martens and Grosse, 2015; George et al., 2018; Dangel et al., 2024) and randomized Nyström methods (Martinsson and Tropp, 2020; Bioli et al., 2025).

A primary motivation for developing the preconditioned primal–dual algorithm stems from the weak formulation obtained via integration by parts. However, this approach relies critically on the assumption that the dominant elliptic operator in the equation is of divergence form. Extensions to more general linear PDEs—such as elliptic equations in non-

divergence form, as noted in Remark 1, are not addressed in the present work and constitute an important direction for future research.

Beyond time-dependent reaction–diffusion equations, extending the NPDG method to equation systems involving first-order convection terms, which commonly arise in the modeling of complex fluids, remains a challenging and largely unexplored direction. Other important time-dependent physical equations that are worthy of further investigation include Navier–Stokes equations and Maxwell's equations.

Although the NPDG algorithm has demonstrated satisfactory performance on several classes of nonlinear PDEs, rigorous theoretical guarantees, particularly concerning the convergence of the method, remain open problems and will be pursued in the future work.

In addition to the task of handling various types of PDEs, the proposed research also paves the way for the future application of natural gradient algorithms in adversarial training of neural networks, including Generative Adversarial Networks (GANs) (Goodfellow et al., 2020; Arjovsky et al., 2017) and large-scale optimal transport problems (Fan et al., 2023; Korotin et al., 2022).

Acknowledgement: S. Liu is partially supported by AFOSR YIP award No. FA9550-23-1-0087. S. Liu and S. Osher are partially funded by STROBE NSF STC DMR 1548924, AFOSR MURI FA9550-18-502, and ONR N00014-20-1-2787. W. Li is partially supported by AFOSR YIP award No. FA9550-23-1-0087, NSF DMS-2245097, and NSF RTG: 2038080. The authors would like to thank Prof. Xiaochuan Tian for constructive discussion. They would also appreciate the feedback from the anonymous reviewers that help improve the paper.

Appendix A. Multiple Layer Perceptron (MLP)

In this research, we denote a Multiple Layer Perceptron (MLP) with activation function f, input dimension d_{in}, hidden dimension d_h, output dimension d_{out}, and number of layers n_l as $\texttt{MLP}_f(d_{in}, d_h, d_{out}, n_l)$. Such MLP takes the form

$$\texttt{MLP}_f(d_{in}, d_h, d_{out}, n_l)(x) = h_{n_l} \circ \cdots \circ h_2 \circ h_1(x),$$

where each $h_k(\cdot)$ is defined as

$$h_k(x) = \begin{cases} f(W_1 x + b_1) & \text{here } W_1 \in \mathbb{R}^{d_h \times d_{in}}, b_1 \in \mathbb{R}^{d_h} \quad \text{if } k = 1 \\ f(W_k x + b_k) & \text{here } W_k \in \mathbb{R}^{d_h \times d_h}, b_k \in \mathbb{R}^{d_h} \quad \text{if } 2 \le k \le n_l - 1 \\ W_{n_l} x + b_{n_l} & \text{here } W_{n_l} \in \mathbb{R}^{d_{out} \times d_h}, b_{n_l} \in \mathbb{R}^{d_{out}} \quad \text{if } k = n_l \end{cases}.$$

The parameters of the MLP are $(W_{n_l}, b_{n_l}, \ldots, W_1, b_1)$. The number of the parameters equals $d_{out}(d_h + 1) + (n_l - 2) \cdot d_h(d_h + 1) + d_h(d_{in} + 1)$. The activation function f of the MLP is usually chosen as a nonlinear function such as $\mathrm{ReLU}(\cdot)$, $\tanh(\cdot)$, etc[7].

Appendix B. Proof of the consistency Theorem 2

Proof Without loss of generality, we always assume $\lambda = 1$ for brevity in this proof. We first show

$$\sup_{\varphi \in \mathbb{K}^{test}, \psi \in \mathbb{K}^{test}_{\partial\Omega}} \mathscr{E}(\widehat{u}, \varphi, \psi) = 0. \tag{56}$$

Notice that for arbitrary $u \in \mathbb{H}$, we pick $\varphi = 0$, $\psi = 0$. Then we obtain $\mathscr{E}(u, 0, 0) = 0$. This yields that $\sup_{\varphi,\psi} \mathscr{E}(u, \varphi, \psi) \ge 0$. For brevity, we omit the name of functional spaces for φ, ψ, and u in this proof. This leads to

$$\inf_u \sup_{\varphi,\psi} \mathscr{E}(u, \varphi, \psi) \ge 0. \tag{57}$$

On the other hand, since u_* is the solution to (10), we have

$$\begin{aligned} \mathscr{E}(u_*, \varphi, \psi) &= \langle \mathcal{L}u_* - f, \varphi \rangle_{L^2(\Omega)} + \langle \mathcal{B}u_* - g, \psi \rangle_{L^2(\partial\Omega)} - \frac{\nu}{2}(\|\mathcal{M}_d \varphi\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|\psi\|^2_{L^2(\partial\Omega)}) \\ &= -\frac{\nu}{2}(\|\mathcal{M}_d \varphi\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|\psi\|^2_{L^2(\partial\Omega)}), \end{aligned}$$

with the supremum value

$$\sup_{\varphi,\psi} \mathscr{E}(u_*, \varphi, \psi) = 0.$$

Combining this with (57), we obtain

$$\inf_u \sup_{\varphi,\psi} \mathscr{E}(u, \varphi, \psi) = 0.$$

Since $(\widehat{u}, \widehat{\varphi}, \widehat{\psi})$ is the solution to the inf-sup problem (15), we obtain (56).

7. $\mathrm{ReLU}(x) = \max\{x, 0\}, \tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}$.

Notice that equation (56) further yields

$$\langle\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\mathcal{M}_d\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)}-\frac{\nu}{2}\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)}$$
$$+\langle\mathcal{B}(\widehat{u}-u_*),\psi\rangle_{L^2(\partial\Omega)}-\frac{\nu}{2}\|\psi\|^2_{L^2(\partial\Omega)}\leq 0$$

for arbitrary $\varphi\in\mathbb{K}^{test}$, $\psi\in\mathbb{K}^{test}_{\partial\Omega}$. By setting $\psi=0$ and then $\varphi=0$ in the above inequality, we obtain

$$\langle\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\mathcal{M}_d\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)}-\frac{\nu}{2}\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)}\leq 0,\quad\forall\ \varphi\in\mathbb{K}^{test} \tag{58}$$

and

$$\langle\mathcal{B}(\widehat{u}-u_*),\psi\rangle_{L^2(\partial\Omega)}-\frac{\nu}{2}\|\psi\|^2_{L^2(\partial\Omega)}\leq 0,\quad\forall\ \psi\in\mathbb{K}^{test}_{\partial\Omega}.$$

We first prove that (58) leads to

$$\langle\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\mathcal{M}_d\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)}=0,\quad\forall\ \varphi\in\mathbb{K}^{test}. \tag{59}$$

Let us suppose (59) does not hold, then, there exists $\widetilde{\varphi}\in\mathbb{K}^{test}$ such that

$$\langle\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\mathcal{M}_d\widetilde{\varphi}\rangle_{L^2(\Omega;\mathbb{R}^r)}=\alpha\neq 0.$$

This also yields $\mathcal{M}_d\widetilde{\varphi}\neq 0$, otherwise, $\langle\widetilde{\mathcal{L}}\mathcal{M}_p(u-u_*),\mathcal{M}_d\widetilde{\varphi}\rangle_{L^2(\Omega;\mathbb{R}^r)}=0$ leads to contradiction.

Now, substituting $\varphi=s\widetilde{\varphi}$ in (58) leads to

$$\langle\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\mathcal{M}_d\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)}-\frac{\nu}{2}\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)}=\alpha s-\frac{\nu}{2}\|\mathcal{M}_d\widetilde{\varphi}\|^2_{L^2(\Omega;\mathbb{R}^r)}\cdot s^2.$$

Since $\mathcal{M}_d\widetilde{\varphi}\neq 0$, we have $\|\mathcal{M}_d\varphi\|_{L^2(\Omega;\mathbb{R}^r)}>0$. By setting $s=\frac{\alpha}{\nu\|\mathcal{M}_d\widetilde{\varphi}\|^2_{L^2(\Omega;\mathbb{R}^r)}}$, the above inner product equals

$$\frac{\alpha^2}{2\nu\|\mathcal{M}_d\widetilde{\varphi}\|^2_{L^2(\Omega;\mathbb{R}^r)}}>0.$$

This is in contradiction to (58). We thus prove (59).

Using the similar argument, we prove

$$\langle\mathcal{B}(\widehat{u}-u_*),\psi\rangle_{L^2(\partial\Omega)}=0,\quad\forall\ \psi\in\mathbb{K}^{test}_{\partial\Omega}.$$

Now (59) further leads to $\langle\mathcal{M}_d^*\widetilde{\mathcal{L}}\mathcal{M}_p(\widehat{u}-u_*),\varphi\rangle_{L^2(\Omega;\mathbb{R}^r)}=0$, for arbitrary $\varphi\in\mathbb{K}^{test}$. That is,

$$\langle\mathcal{L}(\widehat{u}-u_*),\varphi\rangle_{L^2(\Omega)}=0,\quad\forall\ \varphi\in\mathbb{K}^{test}.$$

Since $\mathbb{K}^{test}$ is dense in $L^2(\Omega)$, this further leads to $\|\mathcal{L}(\widehat{u}-u_*)\|_{L^2(\Omega)}=0$.

Recall that $\mathcal{L}u_*=f\in\mathbb{K}\subset L^2(\Omega)$, we thus have $\|\mathcal{L}\widehat{u}-f\|_{L^2(\Omega)}=0$. We deduce that $\mathcal{L}\widehat{u}=f$, a.e. on Ω. Similarly, we also prove that $\mathcal{B}\widehat{u}=g$, a.e. on $\partial\Omega$. ■

Example 3 *Consider the Poisson equation $-\Delta u=f,\quad u|_{\partial\Omega}=g$, or the linear elliptic equation $-\Delta u+u=f,\quad u|_{\partial\Omega}=g$, as mentioned in Example 1 or 2, we set the test functional spaces $\mathbb{K}^{test}=H^1_0(\Omega)$ and $\mathbb{K}^{test}_{\partial\Omega}=L^2(\partial\Omega)$. Since $H^1_0(\Omega)$ is dense in $L^2(\Omega)$, Theorem 2 justifies the consistency between the solution to the inf-sup scheme* (15) *and the solutions to the equations.*

Appendix C. Supplementary proofs and discussions regarding Section 3

In this section, we present the proof to Lemma 6, Theorem 7 and provide further discussions regarding the theoretical work. We first prove Lemma 6.

C.1 Proof of Lemma 6

Proof We first prove that $\nabla_\theta F(\theta) \in \mathrm{Ran}(M(\theta))$. We can first calculate

$$\nabla_\theta F(\theta) = \Big\langle D_u \mathscr{F}(u_\theta), \frac{\partial u_\theta}{\partial \theta} \Big\rangle_{\mathbb{X}}.$$

By decomposing $D_u \mathscr{F}(u_\theta)$ as

$$D_u \mathscr{F}(u_\theta) = \Pi_{\partial u_\theta}[D_u \mathscr{F}(u_\theta)] + \Pi_{\partial u_\theta^\perp}[D_u \mathscr{F}(u_\theta)].$$

The first term can be written as the linear combination of $\{\frac{\partial u_\theta}{\partial \theta_k}\}_{k=1}^m$, i.e. $\Pi_{\partial u_\theta}[\mathscr{F}(u_\theta)] = \frac{\partial u_\theta}{\partial \theta}\mathbf{u}$ for certain $\mathbf{u} \in \mathbb{R}^m$. The inner product between $\Pi_{\partial u_\theta^\perp}[D_u \mathscr{F}(u_\theta)]$ and $\frac{\partial u_\theta}{\partial \theta}$ equals 0. As a result, we have

$$\nabla_\theta F(\theta) = \Big\langle \frac{\partial u_\theta}{\partial \theta}\mathbf{u}, \frac{\partial u_\theta}{\partial \theta} \Big\rangle_{\mathbb{X}} = M(\theta)\mathbf{u} \in \mathrm{Ran}(M(\theta)).$$

On the other hand, we write

$$f(\zeta) = \Big\| D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta}\zeta \Big\|_{\mathbb{X}}^2 = \zeta^\top M(\theta)\zeta - 2\zeta^\top \nabla_\theta F(\theta) + \mathrm{Const.}$$

Recall that $M(\theta)$ is a Gram matrix, it is positive semi-definite, thus $f(\zeta)$ is a convex function. Thus, $\mathbf{v}$ is a minimum of $f(\zeta)$ iff $\nabla f(\zeta) = 0$, which is equivalent to $M(\theta)\mathbf{v} = \nabla_\theta F(\theta)$.

To show the orthogonality, consider arbitrary $\mathbf{w} \in \mathbb{R}^m$, for any $s \in \mathbb{R}$, $f(\mathbf{v}+s\mathbf{w}) \geq f(\mathbf{v})$. This yields

$$0 = \frac{d}{ds} f(\mathbf{v}+s\mathbf{w})\Big|_{s=0} = \Big\langle D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta}\mathbf{v}, \frac{\partial u_\theta}{\partial \theta}\mathbf{w} \Big\rangle_{\mathbb{X}} \quad \text{for any } \mathbf{w} \in \mathbb{R}^m.$$

This verifies the fact that $D_u \mathscr{F}(u_\theta) - \frac{\partial u_\theta}{\partial \theta}\mathbf{v}$ is orthogonal to the subspace $\mathrm{span}\{\frac{\partial u_\theta}{\partial \theta_1}, \ldots, \frac{\partial u_\theta}{\partial \theta_m}\}$. ■

C.2 Proof of Theorem 7

Proof We first recall the functional $\mathscr{E}: \mathbb{H} \times \mathbb{K}^{test} \times \mathbb{K}^{test}_{\partial\Omega} \to \mathbb{R}$ defined in (14),

$$\begin{aligned}
\mathscr{E}(u,\varphi,\psi) =& \langle \mathcal{L}u - f, \varphi\rangle_{L^2(\Omega)} + \lambda\langle \mathcal{B}u - g, \psi\rangle_{L^2(\partial\Omega)} \\
&- \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|\psi\|^2_{L^2(\partial\Omega)}) \\
=& \Big\langle \mathcal{M}_d^*\widetilde{\mathcal{L}}\mathcal{M}_p(u-u_*), \varphi\Big\rangle_{L^2(\Omega)} + \lambda\Big\langle \mathcal{B}(u-u_*), \psi\Big\rangle_{L^2(\partial\Omega)} \\
&- \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega;\mathbb{R}^r)} + \lambda\|\psi\|^2_{L^2(\partial\Omega)}) \\
=& \Big\langle \widetilde{\mathcal{L}}\mathcal{M}_p(u-u_*),\ \mathcal{M}_d\varphi\Big\rangle_{L^2(\Omega;\mathbb{R}^r)} + \Big\langle \sqrt{\lambda}\mathcal{B}(u-u_*), \sqrt{\lambda}\psi\Big\rangle_{L^2(\partial\Omega)} \\
&- \frac{\nu}{2}(\|\mathcal{M}_d\varphi\|^2_{L^2(\Omega,\mathbb{R}^r)} + \lambda\|\psi\|^2_{L^2(\partial\Omega)}) \\
=& \Big\langle \begin{pmatrix} \widetilde{\mathcal{L}} & \\ & \mathrm{Id} \end{pmatrix} \begin{pmatrix} \mathcal{M}_p(u-u_*) \\ \sqrt{\lambda}\mathcal{B}(u-u_*) \end{pmatrix}, \begin{pmatrix} \mathcal{M}_d\varphi \\ \sqrt{\lambda}\psi \end{pmatrix} \Big\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\Big\| \begin{pmatrix} \mathcal{M}_d\varphi \\ \sqrt{\lambda}\psi \end{pmatrix} \Big\|^2_{\mathbb{L}^2}.
\end{aligned}$$

We now substitute u, φ, ψ with parametrized functions $u_\theta, \varphi_\eta, \psi_\xi$, with $\theta \in \Theta_\theta \subseteq \mathbb{R}^{m_\theta}, \eta \in \Theta_\eta \subseteq \mathbb{R}^{m_\eta}, \xi \in \Theta_\xi \subseteq \mathbb{R}^{m_\xi}$. Recall that we define as $\widehat{E}(\theta;\eta,\xi) = \mathscr{E}(u_\theta;\varphi_\eta,\psi_\xi)$. In our discussion, we assume that $\mathcal{M}_p(u_\theta - u_*)$, $\mathcal{B}(u_\theta - u_*)$, $\mathcal{M}_d\varphi_\eta$ and ψ_ξ are differentiable w.r.t. parameters θ, η, ξ; and $\frac{\partial}{\partial\theta}(\mathcal{M}_p(u_\theta - u_*)) \in \widetilde{\mathbb{H}}$, $\frac{\partial}{\partial\eta}(\mathcal{M}_d\varphi_\eta) \in \widetilde{\mathbb{K}}^{test}$, and $\frac{\partial}{\partial\xi}(\sqrt{\lambda}\psi_\xi) \in \mathbb{K}^{test}_{\partial\Omega}$ for arbitrary $\theta \in \Theta_\theta, \eta \in \Theta_\eta, \xi \in \Theta_\xi$.

Now recall the preconditioning matrices introduced in (21), (25) and (24), they can be formulated as:

$$\begin{aligned}
(M_p(\theta))_{ij} &= \Big\langle \frac{\partial}{\partial\theta_i} \begin{pmatrix} \mathcal{M}_p(u_\theta - u_*) \\ \sqrt{\lambda}\mathcal{B}(u_\theta - u_*) \end{pmatrix},\ \frac{\partial}{\partial\theta_j} \begin{pmatrix} \mathcal{M}_p(u_\theta - u_*) \\ \sqrt{\lambda}\mathcal{B}(u_\theta - u_*) \end{pmatrix} \Big\rangle_{\mathbb{L}^2} \\
(M_d(\eta))_{ij} &= \Big\langle \frac{\partial}{\partial\eta_i}(\mathcal{M}_d\varphi_\eta),\ \frac{\partial}{\partial\eta_j}(\mathcal{M}_d\varphi_\eta) \Big\rangle_{L^2(\Omega;\mathbb{R}^r)} \\
(M_{bdd}(\xi))_{ij} &= \Big\langle \frac{\partial}{\partial\xi_i}(\sqrt{\lambda}\psi_\xi), \frac{\partial}{\partial\xi_j}(\sqrt{\lambda}\psi_\xi) \Big\rangle_{L^2(\partial\Omega)}.
\end{aligned}$$

To alleviate our notation, we denote $M_{d,bdd}(\eta,\xi) = M_d(\eta) \oplus M_{bdd}(\xi)$. We further denote

$$\mathbf{U}_\theta = \begin{pmatrix} \mathcal{M}_p(u_\theta - u_*) \\ \sqrt{\lambda}\mathcal{B}(u_\theta - u_*) \end{pmatrix} \in \widetilde{\mathbb{H}} \times \mathbb{K}_{\partial\Omega} \subseteq \mathbb{L}^2, \quad \mathbf{\Phi}_{\eta,\xi} = \begin{pmatrix} \mathcal{M}_d\varphi_\eta \\ \sqrt{\lambda}\psi_\xi \end{pmatrix} \in \widetilde{\mathbb{K}}^{test} \times \mathbb{K}^{test}_{\partial\Omega} \subseteq \mathbb{L}^2.$$

By slightly abusing the notation, we denote $\widetilde{\mathscr{E}}: \mathbb{L}^2 \times \mathbb{L}^2 \to \mathbb{R}$ as

$$\mathscr{E}(\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi}) = \Big\langle (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi} \Big\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\|\mathbf{\Phi}_{\eta,\xi}\|^2_{\mathbb{L}^2},$$

which is equal to the previous functional $\mathscr{E}(u_\theta, \varphi_\eta, \psi_\xi)$.

Notice that (37) is denoted as $\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t}$ by using our new notation, the NPDG flow (36) can be formulated as

$$\begin{aligned}
(\dot\eta_t^\top, \dot\xi_t^\top)^\top =& M_{d,bdd}(\eta_t,\xi_t)^\dagger \nabla_{\eta,\xi}\mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) \\
\dot\theta_t =& -M_p(\theta_t)^\dagger \nabla_\theta \mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t}).
\end{aligned} \tag{60}$$

Now suppose $(\theta_t, \eta_t, \xi_t)$ solves (60); we compute

$$\dot{\mathbf{\Phi}}_{\eta_t,\xi_t} = \frac{\partial \mathbf{\Phi}_{\eta_t\xi_t}}{\partial(\eta,\xi)} M_{d,bdd}(\eta_t,\xi_t)^\dagger \nabla_{\eta,\xi} \mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}). \tag{61}$$

By treating $\mathbb{X} = \mathbb{L}^2$ and $\mathscr{F}(\cdot)$ as $\mathscr{E}(\mathbf{U}_\theta, \cdot)$ in Lemma 6, the right-hand side of (61) is nothing but the orthogonal projection of $D_{\mathbf{\Phi}}\mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) = (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}$ onto the tangent space $\partial_{\eta,\xi}\mathbf{\Phi}_{\eta_t,\xi_t}$, that is,

$$\begin{aligned}\frac{\partial \mathbf{\Phi}_{\eta_t,\xi_t}}{\partial(\eta,\xi)} M_{d,bdd}(\eta_t,\xi_t)^\dagger \nabla_{\eta,\xi} \mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) =& \Pi_{\partial_{\eta,\xi}\mathbf{\Phi}_{\eta_t,\xi_t}}[D_{\mathbf{\Phi}}\mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t})] \\ =& \Pi_{\partial_{\eta,\xi}\mathbf{\Phi}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}].\end{aligned}$$

Similarly

$$\dot{\mathbf{U}}_{\theta_t} = -\frac{\partial \mathbf{U}_{\theta_t}}{\partial \theta} M_p(\theta_t)^\dagger \nabla_\theta \mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t} + \gamma \dot{\mathbf{\Phi}}_{\eta_t,\xi_t}). \tag{62}$$

By denoting $\widetilde{\mathcal{L}}^*$ as the adjoint operator[8] of $\widetilde{\mathcal{L}}$, we have

$$\mathscr{E}(\mathbf{U}, \mathbf{\Phi}) = \left\langle (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}, \mathbf{\Phi} \right\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\|\mathbf{\Phi}\|_{\mathbb{L}^2}^2 = \left\langle \mathbf{U}, (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})\mathbf{\Phi} \right\rangle_{\mathbb{L}^2} - \frac{\nu}{2}\|\mathbf{\Phi}\|_{\mathbb{L}^2}^2.$$

Then, the right-hand side of (62) equals

$$-\Pi_{\partial_\theta \mathbf{U}_{\theta_t}}[D_{\mathbf{U}}\mathscr{E}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t})] = -\Pi_{\partial_\theta \mathbf{U}_{\theta_t}}[(\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t})],$$

Thus the corresponding dynamic of (60) in the functional space can be formulated as

$$\begin{aligned}\dot{\mathbf{\Phi}}_{\eta_t,\xi_t} &= \Pi_{\partial_{\eta,\xi}\mathbf{\Phi}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}], \\ \dot{\mathbf{U}}_{\theta_t} &= -\Pi_{\partial_\theta \mathbf{U}_\theta}[(\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t})].\end{aligned}$$

We now consider the Lyapunov functional

$$\begin{aligned}\mathcal{I}(\mathbf{U}, \mathbf{\Phi}) =& \frac{1}{2}(\|\mathcal{M}_p(u - u_*)\|_{L^2(\Omega;\mathbb{R}^r)}^2 + \lambda\|\mathcal{B}(u - u_*)\|_{L^2(\partial\Omega)}^2 + \|\mathcal{M}_d\varphi\|_{L^2(\Omega;\mathbb{R}^r)}^2 + \lambda\|\psi\|_{L^2(\partial\Omega)}^2) \\ =& \frac{1}{2}\|\mathbf{U}\|_{\mathbb{L}^2}^2 + \frac{1}{2}\|\mathbf{\Phi}\|_{\mathbb{L}^2}^2.\end{aligned} \tag{63}$$

We shall study the decay of this Lyapunov functional along $\{(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t})\}$. We calculate

$$\begin{aligned}\frac{d}{dt}\mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) &= \left\langle \mathbf{U}_{\theta_t}, \dot{\mathbf{U}}_{\theta_t} \right\rangle_{\mathbb{L}^2} + \left\langle \mathbf{\Phi}_{\eta_t,\xi_t}, \dot{\mathbf{\Phi}}_{\eta_t,\xi_t} \right\rangle_{\mathbb{L}^2} \\ &= \underbrace{\left\langle \mathbf{U}_{\theta_t}, -\Pi_{\partial_\theta \mathbf{U}_{\theta_t}}[(\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma\dot{\mathbf{\Phi}}_{\eta_t,\xi_t})] \right\rangle_{\mathbb{L}^2}}_{(1)} \\ &\quad + \underbrace{\left\langle \mathbf{\Phi}_{\eta_t,\xi_t},\ \Pi_{\partial_{\eta,\xi}\mathbf{\Phi}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu\mathbf{\Phi}_{\eta_t,\xi_t}] \right\rangle_{\mathbb{L}^2}}_{(2)}\end{aligned} \tag{64}$$

8. In the sense that

$$\left\langle \widetilde{\mathcal{L}}v, w \right\rangle_{L^2(\Omega;\mathbb{R}^r)} = \left\langle v, \widetilde{\mathcal{L}}^* w \right\rangle_{L^2(\Omega;\mathbb{R}^r)}, \quad \forall\ v \in \widetilde{\mathbb{H}},\ w \in \widetilde{\mathbb{K}}^{test}.$$

We further compute (1) as:

$$\begin{aligned}
(1) =& -\Big\langle \mathbf{U}_{\theta_t}, \quad \Pi_{\partial \mathbf{U}_{\theta_t}}[(\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma \Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}])] \Big\rangle_{\mathbb{L}^2} \\
=& -\Big\langle \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \quad (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \gamma\nu \mathbf{\Phi}_{\eta_t,\xi_t}) \Big\rangle_{\mathbb{L}^2} \\
&+ \Big\langle \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \quad \gamma(\widetilde{\mathcal{L}}^* \oplus \mathrm{Id}) \Pi_{\partial \mathbf{\Phi}^{\perp}_{\eta_t,\xi_t}}((\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}) \Big\rangle_{\mathbb{L}^2} \\
=& \underbrace{-\Big\langle \mathbf{U}_{\theta_t}, \quad (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})((1-\gamma\nu)\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}) \Big\rangle_{\mathbb{L}^2}}_{(A)} \\
&+ \underbrace{\Big\langle \Pi_{\partial \mathbf{U}^{\perp}_{\theta_t}}[\mathbf{U}_{\theta_t}], \quad (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})((1-\gamma\nu)\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}) \Big\rangle_{\mathbb{L}^2}}_{(R1)} \\
&+ \underbrace{\gamma \Big\langle (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\ \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \quad \Pi_{\partial \mathbf{\Phi}^{\perp}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}] \Big\rangle_{\mathbb{L}^2}}_{(R2)}.
\end{aligned} \tag{65}$$

For the second equality, we use the fact that the orthogonal projection $\Pi_{\partial \mathbf{U}_{\theta_t}}$ is self-adjoint on $\mathbb{L}^2$.

Furthermore, the term (2) equals

$$\begin{aligned}
(2) &= \Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, \quad (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t} \Big\rangle_{\mathbb{L}^2} + \Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, \quad \Pi_{\partial \mathbf{\Phi}^{\perp}_{\eta_t,\xi_t}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}] \Big\rangle_{\mathbb{L}^2} \\
&= \underbrace{\Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, \quad (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t} \Big\rangle_{\mathbb{L}^2}}_{(B)} + \underbrace{\Big\langle \Pi_{\partial \mathbf{\Phi}^{\perp}_{\eta_t,\xi_t}}[\mathbf{\Phi}_{\eta_t,\xi_t}], \quad (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t} \Big\rangle_{\mathbb{L}^2}}_{(R3)}
\end{aligned} \tag{66}$$

Then one can calculate

$$\begin{aligned}
&(A) + (B) \\
&= -\Big\langle \mathbf{U}_{\theta_t}, (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})((1-\gamma\nu)\mathbf{\Phi}_{\eta_t,\xi_t} + \gamma(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t})) \Big\rangle_{\mathbb{L}^2} + \Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t} \Big\rangle_{\mathbb{L}^2} \\
&= -\gamma \Big\langle \mathbf{U}_{\theta_t}, (\widetilde{\mathcal{L}}^* \oplus \mathrm{Id})(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} \Big\rangle_{\mathbb{L}^2} + \gamma\nu \Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, (\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} \Big\rangle_{\mathbb{L}^2} - \nu \Big\langle \mathbf{\Phi}_{\eta_t,\xi_t}, \mathbf{\Phi}_{\eta_t,\xi_t} \Big\rangle_{\mathbb{L}^2}
\end{aligned} \tag{67}$$

Recall the assumption (38), we have:

$$\begin{aligned}
\|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}\|^2_{\mathbb{L}^2} &= \|\widetilde{\mathcal{L}}\mathbf{u}\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|w\|^2_{L^2(\partial\Omega)} \\
&\le L_1^2 \|\mathbf{u}\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|w\|^2_{L^2(\partial\Omega)} \\
&\le (L_1^2 \vee 1) \cdot (\|\mathbf{u}\|^2_{L^2(\Omega;\mathbb{R}^r)} + \|w\|^2_{L^2(\partial\Omega)}) = (L_1^2 \vee 1) \cdot \|\mathbf{U}\|^2_{\mathbb{L}^2}.
\end{aligned}$$

That is, $\|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}\|_{\mathbb{L}^2} \le (L_1 \vee 1) \cdot \|\mathbf{U}\|_{\mathbb{L}^2}$. Similarly, we have $\|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}\|_{\mathbb{L}^2} \ge (L_0 \wedge 1)\|\mathbf{U}\|_{\mathbb{L}^2}$.

We can verify that (67) yields

$$(A) + (B) \le -\gamma(L_0 \wedge 1)^2 \|\mathbf{U}_{\theta_t}\|^2_{\mathbb{L}^2} + \gamma\nu(L_1 \vee 1)\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} - \nu \|\mathbf{\Phi}_{\eta_t,\xi_t}\|^2_{\mathbb{L}^2}. \tag{68}$$

Moreover, by Cauchy-Schwarz inequality, we estimate the remainder terms $(R1), (R2), (R3)$ as

$$\begin{aligned}(R1) &\le \|\Pi_{\partial \mathbf{U}_{\theta_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2} \cdot ((L_1 \vee 1)|1-\gamma\nu|\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} + \gamma(L_1 \vee 1)^2\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}) \\ &\le \alpha\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot ((L_1 \vee 1)|1-\gamma\nu|\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} + \gamma(L_1 \vee 1)^2\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}) \\ &= \alpha \cdot (L_1 \vee 1) \cdot |1-\gamma\nu| \cdot \|\mathbf{U}_{\theta_t}\| \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} + \alpha \cdot \gamma \cdot (L_1 \vee 1)^2 \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2.\end{aligned}$$

$$\begin{aligned}(R2) &\le \gamma \cdot (L_1 \vee 1)\|\Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2} \cdot \|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}^{\perp}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu\mathbf{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2} \\ &\le \gamma \cdot (L_1 \vee 1) \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot (\|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}^{\perp}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2} + \nu\|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}}[\mathbf{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2}) \\ &\le \gamma \cdot (L_1 \vee 1) \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot (\beta_1\|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} + \nu\beta_2\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}) \\ &\le \gamma \cdot (L_1 \vee 1)^2 \cdot \beta_1\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 + \gamma\nu \cdot (L_1 \vee 1) \cdot \beta_2 \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}.\end{aligned}$$

$$\begin{aligned}(R3) &\le \|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2} \cdot \|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t} - \nu\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \\ &\le \beta_2 \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \cdot ((L_1 \vee 1)\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} + \nu\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}) \\ &= \beta_2 \cdot (L_1 \vee 1) \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} + \beta_2 \cdot \nu \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}^2.\end{aligned}$$

Here, we denote

$$\alpha = \max_{t\in[0,T]} \frac{\|\Pi_{\partial \mathbf{U}_{\theta_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}}{\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}}; \tag{69}$$

$$\beta_1 = \max_{t\in[0,T]} \frac{\|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}^{\perp}}[(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}}{\|(\widetilde{\mathcal{L}} \oplus \mathrm{Id})\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}}; \tag{70}$$

$$\beta_2 = \max_{t\in[0,T]} \frac{\|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2}}{\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}}. \tag{71}$$

It is not hard to tell that $0 \le \alpha, \beta_1, \beta_2 \le 1$.

Now, recall (64) and (67), together with the estimates on the remainder terms $(R1), (R2), (R3)$ we obtain

$$\begin{aligned}\frac{d}{dt}\mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) \le &- \gamma \cdot ((L_0 \wedge 1)^2 - (L_1 \vee 1)^2(\alpha+\beta_1)) \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 \\ &+ (L_1 \vee 1) \cdot ((1+\beta_1)\gamma\nu + \beta_2 + \alpha|1-\gamma\nu|) \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \\ &- \nu \cdot (1-\beta_2) \cdot \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}^2. \\ \le &- [\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}, \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}] \underbrace{\begin{bmatrix} \Gamma_{\mathbf{UU}} & \Gamma_{\mathbf{\Phi U}}/2 \\ \Gamma_{\mathbf{\Phi U}}/2 & \Gamma_{\mathbf{\Phi\Phi}} \end{bmatrix}}_{\Gamma} \begin{bmatrix} \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \\ \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \end{bmatrix}.\end{aligned}$$

Here we denote

$$\begin{aligned}&\Gamma_{\mathbf{UU}} = \gamma \cdot ((L_0 \wedge 1)^2 - (L_1 \vee 1)^2(\alpha+\beta_1)), \quad \Gamma_{\mathbf{\Phi\Phi}} = \nu(1-\beta_2), \\ &\Gamma_{\mathbf{\Phi U}} = -(L_1 \vee 1) \cdot ((1+\beta_1)\gamma\nu + \beta_2 + \alpha|1-\gamma\nu|).\end{aligned}$$

Since we assumed that $\frac{1}{\tilde{\kappa}^2} > \alpha + \beta_1$, this yields $\Gamma_{\mathbf{UU}} > 0$; and $\beta_2 < 1$ yields $\Gamma_{\mathbf{\Phi\Phi}} > 0$; moreover, (40) is equivalent to $\det(\Gamma) = \Gamma_{\mathbf{UU}}\Gamma_{\mathbf{\Phi\Phi}} - \frac{1}{4}\Gamma_{\mathbf{\Phi U}}^2 > 0$. In conclusion, these lead to the fact that Γ is positive definite. Further, we denote the smaller eigenvalue of Γ as

$$r = \frac{1}{2}\left(\Gamma_{\mathbf{UU}} + \Gamma_{\mathbf{\Phi\Phi}} - \sqrt{(\Gamma_{\mathbf{UU}} - \Gamma_{\mathbf{\Phi\Phi}})^2 + \Gamma_{\mathbf{\Phi U}}^2}\right). \tag{72}$$

Thus, $r > 0$, and we obtain

$$\frac{d}{dt}\mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) \leq -r \cdot \mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}), \quad t \in [0,T].$$

Applying the Grönwall's inequality yields

$$\mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}) \leq \exp(-rt) \cdot \mathcal{I}(\mathbf{U}_{\theta_0}, \mathbf{\Phi}_{\eta_0,\xi_0}),$$

for $t \in [0,T]$. Recall definition (63), we have proven the theorem

$$\|\mathcal{M}_p(u_{\theta_t} - u_*)\|_{L^2(\Omega;\mathbb{R}^r)}^2 + \lambda\|\mathcal{B}(u_{\theta_t} - u_*)\|_{L^2(\partial\Omega)}^2 \leq 2\exp(-rt) \cdot \mathcal{I}(\mathbf{U}_{\theta_0}, \mathbf{\Phi}_{\eta_0,\xi_0}), \quad 0 \leq t \leq T.$$

■

C.3 Some definitions related to the fractional Sobolev space

We give a brief definition of the fractional Sobolev space $H^{1/2}(\partial\Omega)$ and state a useful result to be used in the next section characterizing its norm.

Definition 15 ($H^{1/2}(\partial\Omega)$ space) *For open bounded domain $\Omega \subset \mathbb{R}^d$ with Lipschitz boundary $\partial\Omega$, we define*

$$H^{1/2}(\partial\Omega) = \left\{u \in L^2(\partial\Omega) \,:\, \|u\|_{H^{1/2}(\partial\Omega)} < \infty\right\},$$

where we define the fractional Sobolev norm (also known as the Sobolev-Slobodeckii norm using general L^p norm, see (Gagliardo, 1957) and the references therein for more details)

$$\|u\|_{H^{1/2}(\partial\Omega)}^2 := \|u\|_{L^2(\partial\Omega)}^2 + |u|_{H^{1/2}(\partial\Omega)}^2, \tag{73}$$

with seminorm

$$|u|_{H^{1/2}(\partial\Omega)}^2 := \int_{\partial\Omega}\int_{\partial\Omega} \frac{|u(x) - u(y)|^2}{\|x - y\|^d} \,\mathrm{d}s_x \,\mathrm{d}s_y.$$

One can then define the bounded, surjective trace operator $\mathcal{B} : H^1(\Omega) \to H^{1/2}(\partial\Omega)$ by extending it from the smooth function space (McLean, 2000; Evans, 2022). We have the following theorem:

Theorem 16 *The $H^{1/2}(\partial\Omega)$ norm is equivalent to the following norm $\|\cdot\|_\star$ up to a constant:*

$$\|g\|_\star := \inf_{u \in H^1(\Omega), \mathcal{B}u = g} \|u\|_{H^1(\Omega)}, \quad \text{for any } g \in \mathcal{B}(H^1(\Omega)) = H^{1/2}(\partial\Omega). \tag{74}$$

That is, there exist constants $C_\Omega > c_\Omega > 0$ only depending on Ω s.t. for any $g \in H^{1/2}(\partial\Omega)$,

$$C_\Omega\|g\|_\star \geq \|g\|_{H^{1/2}(\partial\Omega)} \geq c_\Omega\|g\|_\star.$$

We refer the readers to Theorem 1.I in (Gagliardo, 1957) as a proof. More comprehensive discussions regarding this result can be found in (McLean, 2000; Pechstein, 2013).

C.4 Proof of Theorem 8.

As we focus on the Dirichlet boundary problem, we always treat $\mathcal{B}$ as the trace operator throughout this section. Before working on this proof, we slightly modify the definitions of several notations that are used in the previous proof of Theorem 7 for the current result. In this case, we have $\mathbb{H} = H^2(\Omega), \widetilde{\mathbb{H}} = H^1(\Omega;\mathbb{R}^d), \widetilde{\mathbb{K}} = H^1(\Omega;\mathbb{R}^d), \mathbb{K} = L^2(\Omega)$, $\mathbb{K}_{\partial\Omega} = \mathcal{X}$ and $\widetilde{\mathbb{K}}^{test} = L^2(\Omega;\mathbb{R}^d), \mathbb{K}^{test} = H_0^1(\Omega), \mathbb{K}_{\partial\Omega}^{test} = \mathcal{X}$. The operators $\mathcal{M}_p, \widetilde{\mathcal{L}}, \mathcal{M}_d$ are defined as

$$\mathcal{M}_p : \mathbb{H} \to \widetilde{\mathbb{H}}, u \mapsto \sqrt{A(\cdot)}\nabla u(\cdot), \quad \widetilde{\mathcal{L}} = \mathrm{Id} : \widetilde{\mathbb{H}} \to \widetilde{\mathbb{K}}, \quad \mathcal{M}_d : \widetilde{\mathbb{K}}^{test} \to \mathbb{K}^{test}, \varphi \mapsto \sqrt{A(\cdot)}\nabla\varphi(\cdot).$$

In this proof, we define $\mathbb{L}^2 := L^2(\Omega;\mathbb{R}^d) \times \mathcal{X}$. We keep the notations of $\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi}$ as

$$\mathbf{U}_\theta = \begin{pmatrix} \mathcal{M}_p(u_\theta - u_*) \\ \sqrt{\lambda}\mathcal{B}(u_\theta - u_*) \end{pmatrix} \in \mathbb{L}^2, \quad \mathbf{\Phi}_{\eta,\xi} = \begin{pmatrix} \mathcal{M}_d\varphi_\eta \\ \sqrt{\lambda}\psi_\xi \end{pmatrix} \in \mathbb{L}^2,$$

and define the preconditioning matrices

$$(M_p(\theta))_{ij} = \Big\langle \frac{\partial \mathbf{U}_\theta}{\partial\theta_i}, \frac{\partial \mathbf{U}_\theta}{\partial\theta_j} \Big\rangle_{\mathbb{L}^2}, \quad (M_d(\eta))_{ij} = \Big\langle \frac{\partial \mathcal{M}_d\varphi_\eta}{\partial\eta_i}, \frac{\partial \mathcal{M}_d\varphi_\eta}{\partial\eta_j} \Big\rangle_{L^2(\Omega;\mathbb{R}^d)},$$
$$(M_{bdd}(\xi))_{ij} = \Big\langle \frac{\partial(\sqrt{\lambda}\psi_\xi)}{\partial\xi_i}, \frac{\partial(\sqrt{\lambda}\psi_\xi)}{\partial\xi_j} \Big\rangle_{\mathcal{X}}.$$

Again, we assume the differentiability of $\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi}$ w.r.t. parameters θ, η, ξ; and $\frac{\partial}{\partial\theta}(\mathcal{M}_p(u_\theta - u_*)) \in \widetilde{\mathbb{H}}$, $\frac{\partial}{\partial\eta}(\mathcal{M}_d\varphi_\eta) \in \widetilde{\mathbb{K}}^{test}$, and $\frac{\partial}{\partial\xi}(\sqrt{\lambda}\psi_\xi) \in \mathbb{K}_{\partial\Omega}^{test}$ for arbitrary θ, η, and ξ.

Recall that $\Pi_{\partial\mathbf{U}_\theta} : \mathbb{L}^2 \to \mathbb{L}^2$ denotes the orthogonal projection onto $\mathrm{span}\{\partial_{\theta_k}\mathbf{U}_\theta\}$ w.r.t. the $\mathbb{L}^2$ inner product, while similarly, $\Pi_{\partial\mathbf{\Phi}_{\eta,\xi}} = \Pi_{\partial\boldsymbol{\varphi}_\eta} \oplus \Pi_{\partial\boldsymbol{\psi}_\xi}$ with $\Pi_{\partial\boldsymbol{\varphi}_\eta}, \Pi_{\partial\boldsymbol{\psi}_\xi}$ denote the orthogonal projections onto $\partial\boldsymbol{\varphi}_\eta = \mathrm{span}\{\partial_{\eta_k}\mathcal{M}_d\varphi_\eta\}$ and $\mathrm{span}\{\partial_{\xi_k}\psi_\xi\}$ w.r.t. $L^2(\Omega;\mathbb{R}^d)$ and $\mathcal{X}$ inner products, respectively.

Before we present the proof, we need the following two lemmas.

Lemma 17 *For a given $w \in H^2(\Omega)$, the following variational problem admits a unique minimizer $\widehat{\varphi} \in H_0^1(\Omega)$,*

$$\min_{\varphi\in H_0^1(\Omega)} \ |\varphi - w|^2_{H^1(\Omega,A)}. \tag{75}$$

Denote $\mathcal{T} : H^2(\Omega) \to H_0^1(\Omega), w \mapsto \widehat{\varphi}$, then $\mathcal{T}$ is a linear operator. Furthermore, $\sqrt{A(\cdot)}(\nabla\widehat{\varphi} - \nabla w)$ is orthogonal to all $\sqrt{A(\cdot)}\nabla\phi$ with $\phi \in H_0^1(\Omega)$ w.r.t. $L^2(\Omega;\mathbb{R}^d)$ inner product.

Proof The existence and uniqueness of the minimizer of (75) is a standard result in calculus of variations. Readers are referred to (Evans, 2022) (c.f. Theorem 2 and Theorem 3 in Chap. 8) for a proof. It can be verified that $\mathcal{T}w = \widehat{\varphi}$ is the weak solution to the Euler-Lagrange equation, which is a linear elliptic equation:

$$-\nabla\cdot(A(x)\nabla\varphi(x)) = -\nabla\cdot(A(x)\nabla w(x)), \text{ on } \Omega, \quad \varphi = 0 \text{ on } \partial\Omega. \tag{76}$$

This yields that $\int_\Omega \nabla(\widehat{\varphi}(x) - w(x))^\top A(x)\nabla\phi(x)\mathrm{d}x = 0$ for arbitrary $\phi \in H_0^1(\Omega)$. This verifies the orthogonality assertion of the Lemma. Conversely, any weak solution φ of (76) is a minimizer of (75). See Section 8.2.3 of (Evans, 2022) for a detailed discussion. The

equivalence between the minimizer of the variational problem and the solution to Euler-Lagrange equation verifies the linearity of $\mathcal{T}$.

■

Lemma 18 *For arbitrary $w \in H^1(\Omega)$, denote $\widehat{\varphi} = \mathcal{T}w = \underset{\varphi \in H_0^1(\Omega)}{\operatorname{argmin}} |w - \varphi|_{H_1(\Omega,A)}$, we have the inequality*

$$|w - \widehat{\varphi}|_{H_1(\Omega,A)} \leq \sqrt{\overline{A}} \|\mathcal{B}w\|_\star.$$

Proof Let us first consider the variational problem

$$\min_{\phi \in H_0^1(\Omega)} \|\phi + w\|^2_{H^1(\Omega)}. \tag{77}$$

By using the similar arguments for proving Lemma 17, one can show that there exists unique $\phi_* \in H_0^1(\Omega)$ that minimizes (77).

On the other hand, it is straightforward to verify the equivalence between (77) and (74). Therefore, the optimal value for (77) yields $\|\phi_* + w\|_{H^1(\Omega)} = \|\mathcal{B}w\|_\star$.

Furthermore, we have

$$|w - \widehat{\varphi}|^2_{H^1(\Omega,A)} = \min_{\varphi \in H_0^1(\Omega)} |w - \varphi|^2_{H_1(\Omega,A)} \leq |w + \phi_*|^2_{H_1(\Omega,A)},$$

and

$$|w + \phi_*|^2_{H_1(\Omega,A)} = \int_\Omega \nabla(w + \phi_*)^\top A(x) \nabla(w + \phi_*) \mathrm{d}x \leq \overline{A} \|w + \phi_*\|^2_{H^1(\Omega)} = \overline{A} \|\mathcal{B}w\|^2_\star.$$

Combining the two inequalities proves the assertion.

■

We are now ready to prove the result:

Proof Recall that $\{(\theta_t, \eta_t, \xi_t)\}_{t \geq 0}$ denotes the solution to the NPDG flow associated with the functional $\mathscr{E}_{\mathcal{X}}(u_\theta, \varphi_\eta, \psi_\xi)$ defined in (44). We again consider the Lyapunov functional

$$\mathcal{I}(\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi}) = \frac{1}{2}(\|\mathbf{U}_\theta\|^2_{\mathbb{L}^2} + \|\mathbf{\Phi}_{\eta,\xi}\|^2_{\mathbb{L}^2}),$$

defined in (63). Using almost the identical derivation demonstrated in the proof of Theorem 7, we obtain the time derivative of $\mathcal{I}(\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi})$ as

$$\begin{aligned} \frac{d}{dt} I(\mathbf{U}_\theta, \mathbf{\Phi}_{\eta,\xi}) =& (1) + (2) \\ =& [(\mathrm{A}) + (\mathrm{R1}) + (\mathrm{R2})] + [(\mathrm{B}) + (\mathrm{R3})], \end{aligned} \tag{78}$$

where the terms (1) and (2) take the same forms as in (64). They are further decomposed as (A)+(R1)+(R2) as in (65) and (B)+(R3) as in (66) respectively.

We first estimate the remainder terms (R1) and (R2) in (65). Instead of introducing the relative errors α, β_1, β_2, we keep the approximation errors in the present estimation. For (R1), we have

$$\begin{aligned}(R1) \leq & \|\Pi_{\partial \mathbf{U}_{\theta_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2} \cdot \|\boldsymbol{\Phi}_{\eta_t,\xi_t} + \gamma \Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}}[\mathbf{U}_{\theta_t} - \nu \boldsymbol{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2} \\ \leq & \mathrm{err}(\mathbf{U}_{\theta_t} | \, \partial \mathbf{U}_{\theta_t}) \cdot ((1+\gamma\nu)\|\boldsymbol{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} + \gamma \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}) \\ \leq & 2\left((1+\gamma\nu) \vee \gamma\right) \cdot \mathrm{err}(\mathbf{U}_{\theta_t} | \, \partial \mathbf{U}_{\theta_t}) \cdot \sqrt{\mathcal{I}(\mathbf{U}_{\theta_t}, \boldsymbol{\Phi}_{\eta_t,\xi_t})}.\end{aligned} \tag{79}$$

The first inequality is due to the Cauchy-Schwarz inequality and $\widetilde{\mathcal{L}} = \mathrm{Id}$. Here, we denote the approximation error $\mathrm{err}(\mathbf{U}_\theta \mid \partial \mathbf{U}_\theta) := \|\Pi_{\partial \mathbf{U}_\theta^{\perp}}[\mathbf{U}_\theta]\|_{\mathbb{L}^2}$, which can be formulated as[9]

$$\mathrm{err}(\mathbf{U}_\theta \mid \partial \mathbf{U}_\theta)^2 = \min_{\zeta \in \mathbb{R}^{m_\theta}} \int_\Omega \|\nabla \langle \partial_\theta u_\theta(x), \zeta \rangle - \nabla(u_\theta - u_*)\|_{A(x)}^2 \, \mathrm{d}x + \lambda \|\langle \partial_\theta u_\theta, \zeta \rangle - (u_\theta - u_*)\|_{\mathcal{X}}^2, \tag{80}$$

where we denote $\langle \partial_\theta u_\theta(x), \zeta \rangle = \sum_{k=1}^{m_\theta} \zeta_k \partial_{\theta_k} u_\theta(x) \in \partial \mathbf{U}_{\theta_t}$. For the term (R2) in (65), we have

$$\begin{aligned}(R2) &= \gamma \cdot \langle \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \, \Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t} - \nu \boldsymbol{\Phi}_{\eta_t,\xi_t}] \rangle_{\mathbb{L}^2} \\ &= \gamma \cdot \langle \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \, \Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}] \rangle_{\mathbb{L}^2} - \gamma\nu \cdot \langle \Pi_{\partial \mathbf{U}_{\theta_t}}[\mathbf{U}_{\theta_t}], \, \Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\boldsymbol{\Phi}_{\eta_t,\xi_t}] \rangle_{\mathbb{L}^2} \\ &\leq \gamma \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot \|\Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2} + \gamma\nu \cdot \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \cdot \|\Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}}[\boldsymbol{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2}.\end{aligned} \tag{81}$$

To estimate $\|\Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}$ in the first term above, we have

$$\|\Pi_{\partial \boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}^2 = \|\Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\mathcal{M}_p(u_{\theta_t} - u_*)]\|_{L^2(\Omega;\mathbb{R}^d)}^2 + \|\Pi_{\partial \psi_{\xi_t}^{\perp}}[u_\theta - u_*]\|_{\mathcal{X}}^2.$$

The estimation of $\|\Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\mathcal{M}_p(u_{\theta_t} - u_*)]\|_{L^2(\Omega;\mathbb{R}^d)}^2$ requires more effort as it accounts for the approximation of using elements in $H_0^1(\Omega)$ to approximate the vector in $H^1(\Omega)$, thus yielding non-negligible discrepancy. To deal with this term, we can decompose

$$\mathcal{M}_p(u_{\theta_t} - u_*) = \sqrt{A(\cdot)}\nabla(u_\theta - u_*) = \sqrt{A(\cdot)}(\nabla(u_\theta - u_*) - \nabla\widehat{\varphi}) + \sqrt{A(\cdot)}\nabla\widehat{\varphi}(\cdot).$$

Here we denote $\widehat{\varphi} = \mathcal{T}(u_\theta - u_*) \in H_0^1(\Omega)$, with the operator $\mathcal{T} : H^1(\Omega) \to H_0^1(\Omega)$ defined in Lemma 17.

Then we have

$$\begin{aligned}\Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\mathcal{M}_p(u_{\theta_t} - u_*)] =& \Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}(\nabla(u_\theta - u_*) - \nabla\widehat{\varphi})] + \Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}] \\ =& \sqrt{A(\cdot)}(\nabla(u_\theta - u_*) - \nabla\widehat{\varphi}) + \Pi_{\partial \boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}].\end{aligned} \tag{82}$$

The second equality is due to Lemma 17, which asserts that $\sqrt{A(\cdot)}(\nabla(u_\theta - u_*) - \nabla\widehat{\varphi})$ is orthogonal to each $\partial_{\eta_k}\mathcal{M}_p\varphi_{\eta_t} \in \partial \boldsymbol{\varphi}_{\eta_t}$ for $k = 1, \ldots, m_\eta$.

9. For matrix $A \in \mathbb{R}^{d\times d}$, we denote the vector norm $\|\mathbf{x}\|_A^2 := \mathbf{x}^\top A \mathbf{x}$ for $\mathbf{x} \in \mathbb{R}^d$.

Furthermore, we have

$$\begin{aligned}\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}] =&(\mathrm{Id}-\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}})[\sqrt{A(\cdot)}\nabla\widehat{\varphi}]\\ =&\sqrt{A(\cdot)}\nabla\widehat{\varphi}-\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}]\in\{\sqrt{A(\cdot)}\nabla\phi\,|\,\phi\in H_0^1(\Omega)\}.\end{aligned}$$

Thus, the two vectors in (82) are orthogonal to each other in $L^2(\Omega;\mathbb{R}^d)$ thanks to Lemma 17. We can then compute

$$\|\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}^{\perp}}[\mathcal{M}_p(u_{\theta_t}-u_*)]\|_{L^2(\Omega;\mathbb{R}^d)}^2=|(u_{\theta_t}-u_*)-\widehat{\varphi}|_{H^1(\Omega,A)}^2+\|\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}]\|_{L^2(\Omega;\mathbb{R}^d)}^2.$$

Here we denote the seminorm $|\cdot|_{H^1(\Omega,A)}$ as defined in (45). Lemma 18 leads to the estimation

$$|(u_{\theta_t}-u_*)-\widehat{\varphi}|_{H^1(\Omega,A)}\le\sqrt{\overline{A}}\|\mathcal{B}[u_{\theta_t}-u_*]\|_\star\le\frac{\sqrt{\overline{A}}}{c_\Omega}\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)}.$$

The second inequality is due to Theorem 16.

On the other hand, we denote

$$\mathrm{err}(u_{\theta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t}):=\|\Pi_{\partial\boldsymbol{\varphi}_{\eta_t}^{\perp}}[\sqrt{A(\cdot)}\nabla\widehat{\varphi}]\|_{L^2(\Omega;\mathbb{R}^d)}=\inf_{\zeta\in\mathbb{R}^{m_\eta}}|\langle\partial_\eta\varphi_{\eta_t},\zeta\rangle-\widehat{\varphi}|_{H^1(\Omega,A)},\tag{83}$$

with $\langle\partial_\eta\varphi_\eta(x),\zeta\rangle=\sum_{k=1}^{m_\eta}\zeta_k\partial_{\eta_k}\varphi_\eta(x)\in\partial\boldsymbol{\varphi}_{\eta_t}$. And analogously,

$$\mathrm{err}(\mathcal{B}u_{\theta_t}|\,\partial\boldsymbol{\psi}_{\xi_t}):=\|\Pi_{\partial\boldsymbol{\psi}_{\xi_t}^{\perp}}[u_{\theta_t}-u_*]\|_{\mathcal{X}}=\inf_{\zeta\in\mathbb{R}^{m_\xi}}\|\langle\partial_\xi\psi_{\xi_t},\zeta\rangle-(u_{\theta_t}-u_*)\|_{\mathcal{X}},\tag{84}$$

with $\langle\partial_\xi\psi_\xi(x),\zeta\rangle=\sum_{k=1}^{m_\xi}\zeta_k\partial_{\xi_k}\psi_\xi(x)\in\partial\boldsymbol{\psi}_{\xi_t}$.

As a result, we can bound $\|\Pi_{\partial\boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}^2$ as

$$\|\Pi_{\partial\boldsymbol{\Phi}_{\eta_t,\xi_t}^{\perp}}[\mathbf{U}_{\theta_t}]\|_{\mathbb{L}^2}^2\le\left(\frac{\sqrt{\overline{A}}}{c_\Omega}\right)^2\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)}^2+\mathrm{err}(u_{\theta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2+\lambda\,\mathrm{err}(\mathcal{B}u_{\theta_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2.$$

We define

$$\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})=\min_{\zeta\in\mathbb{R}^{m_\eta}}|\langle\partial_\eta\varphi_{\eta_t},\zeta\rangle-\varphi_{\eta_t}|_{H^1(\Omega,A)}.\tag{85}$$

$$\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})=\min_{\zeta\in\mathbb{R}^{m_\xi}}\|\langle\partial_\xi\psi_{\xi_t},\zeta\rangle-\psi_{\xi_t}\|_{\mathcal{X}}.\tag{86}$$

Then, $\|\Pi_{\partial\boldsymbol{\Phi}_{\eta_t,\xi_t}}[\boldsymbol{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2}^2=\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2+\lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2$.

As a result, the remainder term (R2) in (81) can be bounded by

$$\begin{aligned}(R2)\le\sqrt{2}\gamma(1+\nu)\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}\cdot\Bigg(\left(\frac{\sqrt{\overline{A}}}{c_\Omega}\right)^2\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)}^2+\mathrm{err}(u_{\theta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2\\ +\lambda\,\mathrm{err}(\mathcal{B}u_{\theta_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2+\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2+\lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2\Bigg)^{\frac{1}{2}}.\end{aligned}\tag{87}$$

The remainder term (R3) in (66) can be bounded using

$$\begin{aligned}(R3) \leq & \|\Pi_{\partial \mathbf{\Phi}_{\eta_t,\xi_t}}[\mathbf{\Phi}_{\eta_t,\xi_t}]\|_{\mathbb{L}^2} \cdot \|\mathbf{U}_{\theta_t} - \nu \mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \\ \leq & (\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2 + \lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2)^{\frac{1}{2}} \cdot (\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} + \nu\|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}) \\ \leq & \frac{1\vee\nu}{2}(\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2 + \lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2)^{\frac{1}{2}} \cdot \sqrt{\mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t})}.\end{aligned} \tag{88}$$

We have now established estimations for the remainder terms (R1), (R2), (R3) in (78). The term (A)+(B) can be estimated using the same technique presented in (68). However, before estimating (A)+(B), recall that (A) $= -\gamma\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 - (1-\gamma\nu)\langle \mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}\rangle_{\mathbb{L}^2}$, we shall separate (A) as

$$(\mathrm{A}) = \underbrace{-\frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 - (1-\gamma\nu)\langle \mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}\rangle_{\mathbb{L}^2}}_{(\mathrm{A'})} - \frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2,$$

where the $-\frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2$ term will be used to offset the boundary term $\|u_{\theta_t} - u_*\|_{H^{1/2}(\partial\Omega)}$ arising in the estimation (87) of (R2).

Recall that $\widetilde{\mathcal{L}} = \mathrm{Id}$, hence $L_1 = L_0 = 1$, we have

$$\begin{aligned}(\mathrm{A'}) + (\mathrm{B}) = [\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}, \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}] \begin{bmatrix} -\frac{\gamma}{2} & \frac{\gamma\nu}{2} \\ \frac{\gamma\nu}{2} & -\nu \end{bmatrix} \begin{bmatrix} \|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \\ \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2} \end{bmatrix} &\leq -r \cdot (\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 + \|\mathbf{\Phi}_{\eta_t,\xi_t}\|_{\mathbb{L}^2}^2) \\ &= -2r \cdot \mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t}).\end{aligned} \tag{89}$$

Here we denote the larger eigenvalue of the above 2×2 matrix as $-r$ with

$$r = \frac{\gamma+2\nu}{4} - \sqrt{\left(\frac{\gamma+2\nu}{4}\right)^2 - \left(\frac{\gamma\nu}{2} - \frac{(\gamma\nu)^2}{4}\right)}.$$

One can verify that $r > 0$ as long as $\gamma\nu < 2$. Furthermore, $r > \frac{\frac{\gamma\nu}{2} - \frac{(\gamma\nu)^2}{4}}{2(\frac{\gamma+2\nu}{4})} = \frac{1}{2}\cdot\frac{\gamma\nu(2-\gamma\nu)}{\gamma+2\nu}$.

Finally, we combine our estimations (89), (79), (81), and (88) together to obtain

$$\begin{aligned}\frac{d}{dt}\mathcal{I}_t =& [(\mathrm{A'}) + (\mathrm{B})] - \frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 + (\mathrm{R1}) + (\mathrm{R2}) + (\mathrm{R3}) \\ \leq & -2r\cdot\mathcal{I}_t - \frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 \\ & + 2\left((1+\gamma\nu)\vee\gamma\right)\cdot\mathrm{err}(\mathbf{U}_{\theta_t}|\,\partial\mathbf{U}_{\theta_t})\cdot\sqrt{\mathcal{I}_t} \\ & + \sqrt{2}\gamma(1+\nu)\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}\cdot\Bigg(\left(\frac{\sqrt{\overline{A}}}{c_\Omega}\right)^2\|u_{\theta_t} - u_*\|_{H^{1/2}(\partial\Omega)}^2 + \mathrm{err}(u_{\theta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2 \\ & \qquad + \lambda\,\mathrm{err}(\mathcal{B}u_{\theta_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2 + \mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2 + \lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2\Bigg)^{\frac{1}{2}} \\ & + \frac{1\vee\nu}{2}(\mathrm{err}(\boldsymbol{\varphi}_{\eta_t}|\,\partial\boldsymbol{\varphi}_{\eta_t})^2 + \lambda\,\mathrm{err}(\boldsymbol{\psi}_{\xi_t}|\,\partial\boldsymbol{\psi}_{\xi_t})^2)^{\frac{1}{2}}\cdot\sqrt{\mathcal{I}_t} \\ \leq & -2r\cdot\mathcal{I}_t + \mathrm{Err}(\theta_t,\eta_t,\xi_t,\gamma,\nu,\lambda)\cdot\sqrt{\mathcal{I}_t} \\ & - \frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 + \sqrt{2}\gamma(1+\nu)\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}\cdot\frac{\sqrt{\overline{A}}}{c_\Omega}\|u_{\theta_t} - u_*\|_{H^{1/2}(\partial\Omega)},\end{aligned} \tag{90}$$

where we denote $\mathcal{I}_t = \mathcal{I}(\mathbf{U}_{\theta_t}, \mathbf{\Phi}_{\eta_t,\xi_t})$ for brevity. For the second inequality, we use the fact that $\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} \le \sqrt{\mathcal{I}_t}$, and $(a^2+b^2)^{\frac{1}{2}} \le |a|+|b|$ for any $a, b \in \mathbb{R}$. The quantity $\mathrm{Err}(\theta_t, \eta_t, \xi_t, \lambda)$ collects all the approximation errors

$$\begin{aligned}\mathrm{Err}(\theta_t, \eta_t, \xi_t, \gamma, \nu, \lambda) = 2\big((1+\gamma\nu)\vee\gamma\big)\cdot \mathrm{err}\big(\mathbf{U}_{\theta_t}\,|\,\partial\mathbf{U}_{\theta_t}\big) \\ + \sqrt{2}\,\gamma(1+\nu)\Big(\mathrm{err}\big(u_{\theta_t}\,|\,\partial\boldsymbol{\varphi}_{\eta_t}\big)^2 + \lambda\,\mathrm{err}\big(\mathcal{B}u_{\theta_t}\,|\,\partial\boldsymbol{\psi}_{\xi_t}\big)^2 \\ + \mathrm{err}\big(\boldsymbol{\varphi}_{\eta_t}\,|\,\partial\boldsymbol{\varphi}_{\eta_t}\big)^2 + \lambda\,\mathrm{err}\big(\boldsymbol{\psi}_{\xi_t}\,|\,\partial\boldsymbol{\psi}_{\xi_t}\big)^2\Big)^{\frac{1}{2}} \\ + \frac{1\vee\nu}{2}\Big(\mathrm{err}\big(\boldsymbol{\varphi}_{\eta_t}\,|\,\partial\boldsymbol{\varphi}_{\eta_t}\big)^2 + \lambda\,\mathrm{err}\big(\boldsymbol{\psi}_{\xi_t}\,|\,\partial\boldsymbol{\psi}_{\xi_t}\big)^2\Big)^{\frac{1}{2}}.\end{aligned} \tag{91}$$

Recall that $\mathrm{err}(\mathbf{U}_{\theta_t}\,|\,\partial\mathbf{U}_{\theta_t}), \mathrm{err}(u_{\theta_t}\,|\,\partial\boldsymbol{\varphi}_{\eta_t}), \mathrm{err}(u_{\theta_t}\,|\,\partial\boldsymbol{\psi}_{\xi_t}), \mathrm{err}(\boldsymbol{\varphi}_{\eta_t}\,|\,\partial\boldsymbol{\varphi}_{\eta_t}), \mathrm{err}(\boldsymbol{\psi}_{\xi_t}\,|\,\partial\boldsymbol{\psi}_{\xi_t})$ are defined in (80), (83), (84), (85), (86) respectively. In the following discussion, we denote $\mathrm{Err}(\theta_t, \eta_t, \xi_t, \gamma, \nu, \lambda)$ as $\mathrm{Err}(\theta_t, \eta_t, \xi_t)$ for simplicity.

The last terms in (90) yield

$$\begin{aligned}&-\frac{\gamma}{2}\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}^2 + \sqrt{2}\gamma(1+\nu)\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}\cdot\frac{\sqrt{\overline{A}}}{c_\Omega}\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)} \\ =&-\frac{\gamma\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}}{2}\left(\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2} - 2\sqrt{2}\gamma(1+\nu)\frac{\sqrt{\overline{A}}}{c_\Omega}\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)}\right) \\ \le&-\frac{\gamma\|\mathbf{U}_{\theta_t}\|_{\mathbb{L}^2}}{2}\left(\sqrt{\lambda}\|u_{\theta_t}-u_*\|_{\mathcal{X}} - 2\sqrt{2}\gamma(1+\nu)\frac{\sqrt{\overline{A}}}{c_\Omega}\|u_{\theta_t}-u_*\|_{H^{1/2}(\partial\Omega)}\right).\end{aligned} \tag{92}$$

Now recall that the boundary norm $\|\cdot\|_{\mathcal{X}}$ satisfies that for arbitrary $w \in H^1(\Omega)$,

$$\|\mathcal{B}w\|_{\mathcal{X}} \ge C_{\mathcal{X}}\|\mathcal{B}w\|_{H^{1/2}(\partial\Omega)},$$

together with fact that

$$\sqrt{\lambda} \ge \frac{2\sqrt{2}\gamma(1+\nu)\sqrt{\overline{A}}}{C_{\mathcal{X}}\cdot c_\Omega},$$

we know the quantity (92) ≤ 0.

In conclusion, (90) yields

$$\frac{d\mathcal{I}_t}{dt} \le -2r\cdot\mathcal{I}_t + \mathrm{Err}(\theta_t, \eta_t, \xi_t)\cdot\sqrt{\mathcal{I}_t},$$

which further leads to $\frac{d}{dt}(e^{2rt}\mathcal{I}_t) \le e^{2rt}\cdot\mathrm{Err}(\theta_t, \eta_t, \xi_t)\cdot\sqrt{\mathcal{I}_t}$.

Now, denoting $\mathcal{J}_t = e^{2rt}\mathcal{I}_t$ yields

$$\frac{d\mathcal{J}_t}{dt} \le e^{rt}\cdot\mathrm{Err}(\theta_t, \eta_t, \xi_t)\cdot\sqrt{\mathcal{J}_t}.$$

As long as $\mathcal{J}_t > 0$, we have

$$\frac{d}{dt}\sqrt{\mathcal{J}_t} \le \frac{e^{rt}}{2}\mathrm{Err}(\theta_t, \eta_t, \xi_t).$$

Integration on the interval $[0,t]$ yields

$$\mathcal{I}_t \leq \left(e^{-rt}\sqrt{\mathcal{I}_0} + \int_0^t \frac{1}{2} e^{-r(t-\tau)} \mathrm{Err}(\theta_\tau, \eta_\tau, \xi_\tau) \mathrm{d}\tau \right)^2.$$

This proves the result. ■

C.5 Proof of Corollary 9

Proof Notice that u_θ is the linear combination of basis functions $\{u_k\}_{k=1}^{m_\theta}$, we have

$$\begin{aligned}
\overline{A}\,\mathcal{E}_u &\geq \inf_{\zeta \in \mathbb{R}^{m_\theta}} \int_\Omega \| \sum_{k=1}^{m_\theta} \zeta_k \nabla u_k - \nabla u_* \|_{A(x)}^2 \mathrm{d}x + \lambda \| \sum_{k=1}^{m_\theta} \zeta_k \mathcal{B} u_k - g \|_{\mathcal{X}}^2 \\
&= \inf_{\zeta \in \mathbb{R}^{m_\theta}} \int_\Omega \| \sum_{k=1}^{m_\theta} \zeta_k \nabla u_k - \nabla (u_\theta - u_*) \|_{A(x)}^2 \mathrm{d}x + \lambda \| \sum_{k=1}^{m_\theta} \zeta_k \mathcal{B} u_k - (\mathcal{B} u_\theta - g) \|_{\mathcal{X}}^2 \\
&\overset{(80)}{=} \mathrm{err}(\mathbf{U}_{\theta_t} | \, \partial \mathbf{U}_{\theta_t})^2.
\end{aligned}$$

This is

$$\mathrm{err}(\mathbf{U}_{\theta_t} | \, \partial \mathbf{U}_{\theta_t}) \leq \sqrt{\overline{A}\,\mathcal{E}_u}.$$

Recall the approximation error $\mathrm{err}(u_{\theta_t} | \, \partial \boldsymbol{\varphi}_{\eta_t})$ defined in (83). Notice that $\widehat{\varphi} = \mathcal{T}(u_{\theta_t} - u_*)$, as $\mathcal{T}$ is linear, this yields $\widehat{\varphi} = \sum_{k=1}^{m_\theta} \theta_k \mathcal{T} u_k - \mathcal{T} u_*$. We thus have

$$\begin{aligned}
\mathrm{err}(u_{\theta_t} | \, \partial \boldsymbol{\varphi}_{\eta_t})^2 &= \inf_{\zeta \in \mathbb{R}^{m_\eta}} |\langle \partial_\eta \varphi_{\eta_t}, \zeta \rangle - \widehat{\varphi}|_{H^1(\Omega, A)}^2 \\
&= \inf_{\zeta \in \mathbb{R}^{m_\eta}} \int_\Omega \| \sum_{k=1}^{m_\eta} \zeta_k \nabla \varphi_k(x) - \left(\sum_{k=1}^{m_\theta} \theta_k \nabla \mathcal{T}(u_k)(x) - \nabla \mathcal{T}(u_*)(x) \right) \|_{A(x)}^2 \mathrm{d}x.
\end{aligned}$$

Now, as we have assumed that $\mathrm{span}\{\mathcal{T} u_k\} \subseteq \mathrm{span}\{\varphi_k\}$, we obtain

$$\overline{A}\,\mathcal{E}_{\nabla\varphi} \geq \inf_{\zeta \in \mathbb{R}^{m_\eta}} \int_\Omega \| \sum_{k=1}^{m_\eta} \zeta_k \nabla \varphi_k(x) - \nabla \mathcal{T}(u_*)(x) \|_{A(x)}^2 \mathrm{d}x = \mathrm{err}(u_{\theta_t} | \, \partial \boldsymbol{\varphi}_{\eta_t})^2,$$

which leads to

$$\mathrm{err}(u_{\theta_t} | \, \partial \boldsymbol{\varphi}_{\eta_t}) \leq \sqrt{\overline{A}\,\mathcal{E}_{\nabla\varphi}}.$$

Moreover, as $\mathrm{span}\{\mathcal{B} u_k\} \subseteq \mathrm{span}\{\psi_k\}$, we have

$$\mathrm{err}(u_{\theta_t} | \, \partial \boldsymbol{\psi}_{\xi_t}) = \sqrt{\mathcal{E}_\psi}.$$

Furthermore, as $\varphi_\theta \in \mathrm{span}\{\varphi_k\}, \psi_\xi \in \mathrm{span}\{\psi_k\}$, we obtain

$$\mathrm{err}(\boldsymbol{\varphi}_{\eta_t} | \, \partial \boldsymbol{\varphi}_{\eta_t}) = 0, \ \mathrm{err}(\boldsymbol{\psi}_{\xi_t} | \, \partial \boldsymbol{\psi}_{\xi_t}) = 0.$$

Plugging these estimations into (46) of Theorem 8 yields the result. ■

C.6 Comparison with previous works

The Lyapunov-based proof framework for the NPDG flow developed in this work is inspired by earlier studies in (Liu et al., 2023b), (Liu et al., 2025). Nevertheless, we shall clarify that there are several fundamental differences that distinguish the present theoretical analysis from these previous works.

First, the methods in (Liu et al., 2023b, 2025) are formulated within a finite-difference setting, where the primal and dual variables u and φ are approximated by their values on discrete grid points. In contrast, our approach employs intact parametrized functions as computational models. This leads to substantially different convergence analyses: the former relies primarily on spectral estimations of the preconditioned matrices arising from spatial discretization, whereas the latter exploits the projection properties between primal and test functional spaces induced by natural gradient structures.

Moreover, in (Liu et al., 2023b, 2025), the discretized differential operator is incorporated into the preconditioner without splitting, namely, $\mathcal{L} = \mathcal{M}_d^* \widetilde{\mathcal{L}} \mathcal{M}_p$ with $\mathcal{M}_d = \mathrm{Id}, \widetilde{\mathcal{L}} = \mathrm{Id}, \mathcal{M}_p = \mathcal{L}$, and the dual variable is allowed to vary freely without boundary constraints. By contrast, our framework accommodates general operator-splitting strategies based on integration by parts. This naturally enforces the choice of the test space $\mathbb{K} = H_0^1(\Omega)$, leading to a more canonical formulation but also necessitating a more delicate treatment of boundary error term and a more refined convergence analysis as demonstrated in Theorem 8.

Appendix D. Basic settings for the methods tested in Section 5

We provide the loss function, as well as the hyperparameters of the three methods PINN, Deep Ritz, and WAN tested in experiments in the following Table 2. In the following Table 3, we summarize the real solutions and their norms for equation (50), (51) and (53) tested in our experiments.

Appendix E. Comparison among different methods

In the following Table 4, we test four different methods with various step sizes on different equations. The step sizes used for each method are summarized below.

- **NPDG** $(\tau_u, \tau_\varphi, \tau_\psi)$: A.$(1.5 \cdot 10^{-1}, 1.5 \cdot 10^{-1}, 1.5 \cdot 10^{-1})$, B.$(10^{-1}, 10^{-1}, 10^{-1})$, C.$(0.5 \cdot 10^{-1}, 0.95 \cdot 10^{-1}, 0.95 \cdot 10^{-1})$, D.$(0.5 \cdot 10^{-1}, 0.5 \cdot 10^{-1}, 0.5 \cdot 10^{-1})$;
  We fix $tol_{\mathrm{MINRES}} = 10^{-3}$ for $d = 5, 10, 20$, and $tol_{\mathrm{MINRES}} = 10^{-4}$ for $d = 50$.
- **PINN(Adam)** (lr): A.$(0.5 \cdot 10^{-2})$ B.(10^{-3}) C.$(0.5 \cdot 10^{-3})$ D.(10^{-4}) E.$(0.5 \cdot 10^{-4})$;
- **DeepRitz** (lr): A.$(0.5 \cdot 10^{-2})$ B.(10^{-3}) C.$(0.5 \cdot 10^{-3})$ D.(10^{-4}) E.$(0.5 \cdot 10^{-4})$;
- **WAN** (τ_θ, τ_η): A.$(0.5 \cdot 10^{-2}, 0.5 \cdot 10^{-1})$, B.$(10^{-3}, 10^{-2})$, C.$(0.5 \cdot 10^{-3}, 0.5 \cdot 10^{-2})$, D.$(10^{-4}, 10^{-3})$, E.$(0.5 \cdot 10^{-4}, 0.5 \cdot 10^{-3})$.

We record the computation time (seconds) spent by each method to achieve accuracy δ in Table 4, we only present the time for the most efficient step size(s). When applying NPDG to VarCoeff, we adopt the $H^1(\partial\Omega, \mu_{\partial\Omega})$ boundary loss as described in Section 5.2.

		PINN	Deep Ritz	WAN/Primal-Dual using Adam
Poisson (50) ($d = 50$)	loss function	$\int_\Omega \|-\Delta u_\theta - f\|^2 \mathrm{d}x$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$	$\int_\Omega \frac{1}{2}\|\nabla u_\theta\|^2 - f u_\theta \mathrm{d}x$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$	$\log\left(\|\int_\Omega \nabla u_\theta \cdot \nabla \varphi_\eta - f\varphi_\eta \mathrm{d}x\|^2\right)$ $-\log\left(\int_\Omega \varphi_\eta^2 \mathrm{d}x\right)$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$
	λ	10^4	10^4	10^4
	lr	$lr = 10^{-4}$	$lr = 10^{-4}$	$\tau_\theta = 0.5 \cdot 10^{-3}$ $\tau_\eta = 0.5 \cdot 10^{-2}$
	N_{iter}	Iterate till GPU time reaches 200s ($d = 10$)/8000s ($d = 50$)		
	(N_{in}, N_{bdd})	$(4000, 80d)$	$(4000, 80d)$	$(10000, 60d)$
	NN	$u_\theta = \mathtt{MLP}_{\tanh}(d, 256, 1, 6)$, $\varphi_\eta = \mathtt{MLP}_{\tanh}(d, 256, 1, 6) \cdot \zeta$		
VarCoeff (51) ($d = 10, 20, 50$)	loss function	$\int_\Omega \|-\nabla \cdot (\kappa \nabla u_\theta) - f\|^2 \mathrm{d}x$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$	$\int_\Omega \kappa \|\nabla u_\theta\|^2 \mathrm{d}x$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$	$\log\left(\|\int_\Omega \kappa \nabla u_\theta \cdot \nabla \varphi_\eta \mathrm{d}x\|^2\right)$ $-\log\left(\int_\Omega \varphi_\eta^2 \mathrm{d}x\right)$ $+\lambda \int_{\partial\Omega} \|u_\theta - g\|^2 \mathrm{d}\sigma$
	λ	10^4	10^3	10^4
	lr	$lr = 10^{-4}$	$lr = 0.5 \cdot 10^{-3}$	$(d = 10)$ $\tau_\theta = 0.5 \cdot 10^{-2}$ $\tau_\eta = 0.5 \cdot 10^{-1}$ $(d = 20, 50)$ $\tau_\theta = 0.5 \cdot 10^{-3}$ $\tau_\eta = 0.5 \cdot 10^{-2}$
	N_{iter}	Iterate till GPU time reaches 500s ($d = 10$)/1500s ($d = 20$)		
		14000 ($d = 50$)	10000 ($d = 50$)	12000 ($d = 50$)
	(N_{in}, N_{bdd})	$(4000, 80d)$	$(4000, 80d)$	$(4000, 80d)$
	NN	$u_\theta = \mathtt{MLP}_{\mathrm{softplus}}(d, 256, 1, 4)$, $\varphi_\eta = \mathtt{MLP}_{\mathrm{softplus}}(d, 256, 1, 4) \cdot \zeta$ for $d = 10, 20$ $u_\theta = \mathtt{MLP}_{\mathrm{softplus}}(d, 256, 1, 6)$, $\varphi_\eta = \mathtt{MLP}_{\mathrm{softplus}}(d, 256, 1, 6) \cdot \zeta$ for $d = 50$		
Nonlinear Elliptic (53) $d = 5$	loss function	$\int_\Omega \|\frac{1}{2}\|\nabla u_\theta\|^2 + V - \Delta u_\theta\|^2 \mathrm{d}x$ $+\lambda \int_{\partial\Omega} u_\theta^2 \mathrm{d}\sigma$	N.A.	$\log(\|\int_\Omega \nabla u_\theta \cdot \nabla \varphi_\eta$ $+\frac{1}{2}\|\nabla u_\theta\|^2 \varphi_\eta + V\varphi_\eta \mathrm{d}x\|^2)$ $-\log\left(\int_\Omega \varphi_\eta^2 \mathrm{d}x\right)$ $+\lambda \int_{\partial\Omega} u_\theta^2 \mathrm{d}\sigma$
	λ	10^4	N.A.	10^3
	lr	10^{-4}	N.A.	$0.5 \cdot 10^{-3}, 0.5 \cdot 10^{-2}$
	N_{iter}	20000	N.A.	20000
	(N_{in}, N_{bdd})	$(4000, 40d)$	N.A.	$(4000, 40d)$
	NN	$u_\theta = \mathtt{MLP}_{\tanh}(d, 256, 1, 4)$, $\varphi_\eta = \mathtt{MLP}_{\tanh}(d, 256, 1, 4) \cdot \zeta$		

Table 2: Loss functions and hyperparameters of the different methods tested in our experiments.

	Domain Ω	Solution u_*	$\|u_*\|_{L^2(\Omega,\mu)}$	$\|\nabla u_*\|_{L^2(\Omega,\mu)}$
Poisson (50)	$[0,1]^d$ $\|\Omega\| = 1$	$\sum_{k=1}^d \sin\left(\frac{\pi}{2} x_k\right)$	$5d:\ 3.2566$ $10d:\ 6.4402$ $20d:\ 12.8066$ $50d:\ 31.9052$	$5d:\ 2.4836$ $10d:\ 3.5124$ $20d:\ 4.9673$ $50d:\ 7.8539$
VarCoeff (51)	$[-1,1]^d$ $\|\Omega\| = 2^d$	$\frac{1}{2} x^\top \Lambda^{-1} x$ $\lambda_0 = 1,\ \lambda_1 = 4$	$10d:\ 1.0969$ $20d:\ 2.1392$ $50d:\ 5.2647$	$10d:\ 1.4434$ $20d:\ 2.0412$ $50d:\ 3.2275$
Nonlinear Elliptic (53)	$B_{d,3}$ $\|\Omega\| = \frac{\pi^{d/2} 3^d}{\Gamma(\frac{d}{2}+1)}$	$\cos\left(\frac{\pi}{2}\|x\|\right)$	$5d:\ 0.6285$	$5d:\ 1.2218$

Table 3: Solutions and their $L^2(\Omega, \mu)$ norms used for benchmarking.

Appendix F. Algorithmic details for the Allen-Cahn equation

In this section, we provide more details on the NPDG algorithm for the Allen-Cahn equation discussed in Section 5.4. Given the time implicit scheme of the equation, our goal is to solve the semi-linear elliptic equation (55) at each time step.

Equ	δ^*	d	PINN(Adam)	Deep Ritz	WAN	NPDG
Poisson	0.005	5D	26.22 (A)	**25.11** (A)	51.14 (B)	68.87 (A)
		10D	44.83 (A)	43.45 (B)	51.65 (C)	**40.98** (B)
		20D	160.82 (C)	183.49 (B)	460.12 (D)	**110.42** (B)
		50D	1989.06 (C)	1452.29 (B)	2117.24 (D)	**821.24** (C)
VarCoeff	0.01	10D	–	**105.2** (C)	–	151.80 (C)
		20D	–	**228.55** (C)	–	309.05 (C)
		50D	774.70 (D)	–	–	**419.15** (C)
	0.005	10D	–	–	–	**231.49** (C)
		20D	–	–	–	**382.59** (C)
		50D	–	–	–	**674.18** (C)
Nonlinear Elliptic	0.1	5D	2805.92 (B)	N.A.	1130.76 (C)	**1086.35** (C)
	0.05	5D	–	N.A.	–	**1894.89** (C)

Table 4: GPU time (seconds) spent by different methods upon achieving the designated accuracy δ. The uppercase letters inside each parenthesis indicate the optimal learning rate(s) used in the algorithm. We apply the Monte-Carlo method with sample size 10^5 to evaluate the relative L^2 error of u_θ. "–" denotes that the method does not achieve the designated accuracy in a given time.

F.1 Handling Different Regimes of ϵ_0

When the positive diffusion coefficient ϵ_0 is bounded away from 0, equation (55) is dominated by the linear Laplacian operator. We can further tame the nonlinear term $W'(u) = u^3 - u$ by subtracting its linear approximation at the equilibrium state $\bar{u} = \pm 1$, i.e., we consider $R(u) = W'(u) - (W'(\bar{u}) + W''(\bar{u})(u - \bar{u}))$. One can verify $W''(\bar{u}) = W''(\pm 1) = 2$. We then absorb the linear term $W''(\bar{u})u$ of $W'(\bar{u}) + W''(\bar{u})(u - \bar{u})$ to the linear portion of (55) to obtain

$$\underbrace{((1 + \frac{h_t W''(\bar{u})}{\epsilon_0})\mathrm{Id} - h_t\epsilon_0\Delta)}_{\mathcal{D}}\, u + \frac{h_t}{\epsilon_0}R(u) = u^{t-1} - \underbrace{\frac{h_t}{\epsilon_0}(W'(\bar{u}) - W''(\bar{u})\bar{u})}_{\text{Const}}.$$

It is reasonable to precondition on the linear differential operator $\mathcal{D}$ for this equation. We introduce the operators

$$\mathcal{M}_p = \mathcal{M}_d : u \mapsto \begin{pmatrix} \sqrt{1 + h_t W''(\bar{u})/\epsilon_0}\, u \\ \sqrt{\epsilon_0 h_t}\nabla u \end{pmatrix}.$$

It is not difficult to verify that $\langle \mathcal{M}_p u, \mathcal{M}_d \varphi\rangle_{L^2} = \langle \mathcal{D}u, \varphi\rangle_{L^2}$ for arbitrary $\varphi \in H_0^1(\Omega)$. Thus, we introduce $\varphi \in H_0^1(\Omega), \psi \in L^2(\partial\Omega)$ for the equation and its boundary condition and design the loss functional

$$\begin{aligned}\mathscr{E}(u, \varphi, \psi|\ u^{t-1}) = &\int_\Omega (u - u^{t-1} + \frac{h_t}{\epsilon_0}W'(u))\varphi + \epsilon_0 h_t \nabla u \cdot \nabla\varphi \,\mathrm{d}\mu(x) \\ &- \frac{\nu}{2}\left(\left(1 - \frac{h_t}{\epsilon_0}W''(\bar{u})\right)\int_\Omega \varphi^2 \,\mathrm{d}\mu(x) - \epsilon_0 h_t \int_\Omega \|\nabla\varphi\|^2 \,\mathrm{d}\mu(x)\right) \\ &+ \lambda\left(\int_{\partial\Omega}\frac{\partial u}{\partial \mathbf{n}}\psi \,\mathrm{d}\mu_{\partial\Omega}(y) - \frac{\nu}{2}\int_{\partial\Omega}\psi^2 \,\mathrm{d}\mu_{\partial\Omega}\right).\end{aligned}$$

In practice, we found that it makes the optimization more stable if we add the residual loss

$$\mathscr{E}_{\mathrm{Res}}(u|u^{t-1}) = \int_\Omega \left| u - u^{t-1} - \epsilon_0 h_t \Delta u + \frac{h_t}{\epsilon_0} W'(u) \right|^2 \mathrm{d}\mu(x) + \lambda \int_{\partial\Omega} \left| \frac{\partial u}{\partial \mathbf{n}} \right|^2 \mathrm{d}\mu_{\partial\Omega},$$

as a regularization term to $\mathscr{E}(u, \varphi, \psi)$, and consider

$$\widetilde{\mathscr{E}}(u, \varphi, \psi \,|\, u^{t-1}) = \mathscr{E}(u, \varphi, \psi \,|\, u^{t-1}) + \mathscr{E}_{\mathrm{Res}}(u, \varphi, \psi \,|\, u^{t-1}).$$

Correspondingly, the precondition matrices are set as

$$\begin{aligned} M_p(\theta) = &\left(1 + \frac{h_t W''(\bar{u})}{\epsilon_0}\right) \int_\Omega \frac{\partial u_\theta}{\partial \theta}^\top \frac{\partial u_\theta}{\partial \theta} \,\mathrm{d}\mu(x) + h_t \epsilon_0 \int_\Omega \frac{\partial}{\partial \theta} (\nabla u_\theta)^\top \frac{\partial}{\partial \theta} (\nabla u_\theta) \,\mathrm{d}\mu(x) \\ &+ \lambda \int_{\partial\Omega} \frac{\partial}{\partial \theta} (\partial_{\mathbf{n}} u_\theta)^\top \frac{\partial}{\partial \theta} (\partial_{\mathbf{n}} u_\theta) \,\mathrm{d}\mu_{\partial\Omega}(y), \end{aligned} \tag{93}$$

$$M_d(\eta) = \left(1 + \frac{h_t W''(\bar{u})}{\epsilon_0}\right) \int_\Omega \frac{\partial \varphi_\eta}{\partial \eta}^\top \frac{\partial \varphi_\eta}{\partial \eta} \,\mathrm{d}\mu(x) + h_t \epsilon_0 \int_\Omega \frac{\partial}{\partial \eta} (\nabla \varphi_\eta)^\top \frac{\partial}{\partial \eta} (\nabla \varphi_\eta) \,\mathrm{d}\mu(x),$$

$$M_{bdd}(\xi) = \lambda \int_{\partial\Omega} \frac{\partial}{\partial \xi} \psi_\xi^\top \frac{\partial}{\partial \xi} \psi_\xi \,\mathrm{d}\mu_{\partial\Omega}(y).$$

We apply this treatment to both the 1D and 2D examples in Section 5.4 with $\epsilon_0 = 0.1$.

In the presence of strong reaction and weak diffusion, the parameter ϵ_0 approaches 0. Equation (55) is dominated by the nonlinear term $\frac{1}{\epsilon_0} W'(u)$. Under such regime, we change our strategy and consider the test functions $\varphi \in L^2(\Omega), \psi \in L^2(\partial\Omega)$, together with the functional

$$\begin{aligned} \mathscr{E}(u, \varphi, \psi \,|\, u^{t-1}) = &\int_\Omega (u - u^{t-1} - \epsilon_0 h_t \Delta u + \frac{h_t}{\epsilon_0} W'(u)) \varphi \,\mathrm{d}\mu_\Omega(x) - \frac{\nu}{2} \int_\Omega \varphi^2 \mathrm{d}\mu_\Omega(x) \\ &+ \lambda \left(\int_{\partial\Omega} \frac{\partial u}{\partial \mathbf{n}} \psi \,\mathrm{d}\mu_{\partial\Omega}(y) - \frac{\nu}{2} \int_{\partial\Omega} \psi^2 \,\mathrm{d}\mu_{\partial\Omega} \right) \end{aligned}$$

By dropping the Laplacian term in the Jacobian of $u - u^{t-1} - \epsilon_0 h_t \Delta u + \frac{h_t}{\epsilon_0} W'(u)$, we approximate the Jacobian operator using $\mathcal{G} := \mathrm{Id} + \frac{h_t}{\epsilon_0} W''(u) : u \mapsto u + \frac{h_t}{\epsilon_0} W''(u) u$. By incorporating $\mathcal{G}$ as the preconditioning operator, we obtain

$$M_p(\theta) = \int_\Omega (1 + \frac{h_t}{\epsilon_0} W''(u_\theta))^2 \frac{\partial u_\theta}{\partial \theta}^\top \frac{\partial u_\theta}{\partial \theta} \,\mathrm{d}\mu_\Omega(x) + \lambda \int_{\partial\Omega} \frac{\partial}{\partial \theta} (\partial_{\mathbf{n}} u_\theta)^\top \frac{\partial}{\partial \theta} (\partial_{\mathbf{n}} u_\theta) \,\mathrm{d}\mu_{\partial\Omega}(y).$$

Meanwhile, the preconditioning matrices $M_d(\eta)$ and $M_{\mathrm{bdd}}(\xi)$ are derived by considering the identity operators on $L^2(\Omega)$ and $L^2(\partial\Omega)$. Consequently, $M_d(\eta) = \int_\Omega \frac{\partial \varphi_\eta}{\partial \eta}^\top \frac{\partial \varphi_\eta}{\partial \eta} \,\mathrm{d}\mu_\Omega$, while $M_{\mathrm{bdd}}(\xi)$ remains the same as defined in (93).

This treatment is adopted in the 1D example presented in Section 5.4 with $\epsilon_0 = 0.01$. An advantage of this treatment is that it avoids integration by parts, thereby allowing greater flexibility in the choice of the error-adaptive measure μ_Ω. In our implementation, we take $\mu_\Omega = \frac{1}{2}\mathcal{U}[0, 2] + \frac{1}{2}\mathcal{U}[0.8, 1.2]$, which places additional sampling weight near $x = 1$ and thus enables a more accurate resolution of the solution profile in that region. Here, $\mathcal{U}[a, b]$ denotes the uniform distribution on the interval $[a, b]$.

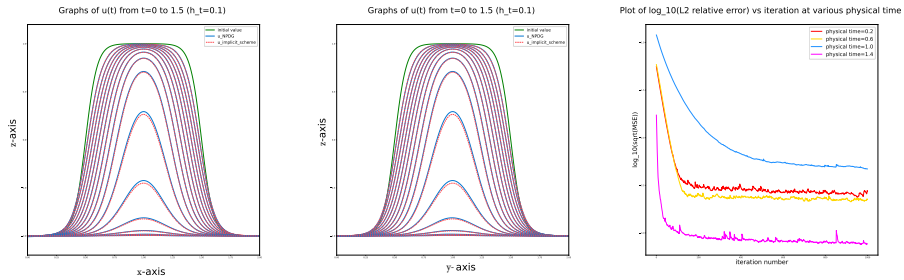
F.2 Benchmark solution & Comparison

We use the numerical solution $\{U^k\}_{k=1}^{N_t}$ solved from the following time-implicit, finite difference scheme

$$\frac{U_i^k - U_i^{k-1}}{h_t} = \epsilon_0 \frac{U_{i+1}^k - 2U_i^k + U_{i-1}^k}{h_x^2} - \frac{1}{\epsilon_0}({U_i^k}^3 - U_i^k), \tag{94}$$
$$U_{-1}^k = U_0^k,\ U_{N_x+1}^k = U_{N_x}^k, \quad \forall\ 0 \le i \le N_x, \quad \text{for}\ \ 1 \le k \le N_t,$$

as the benchmark for u_θ computed from the NPDG algorithm. In our computation, we set $N_x = 400$, $h_x = 2/N_x$, $U_i^0 = u_0(\frac{2i}{N_x})$.

For the 2D example, in Figure 13, we plot the graphs of the neural network solution u_{θ_k} together with the numerical solution $\{U_{ij}^k\}$ obtained via the time-implicit finite difference scheme. The semi-log curves for $\sqrt{\text{MSE loss}}$ versus training time are provided in Figure 13. Further comparison plots and the heatmaps of the pointwise error $|u_{\theta_k}(\cdot) - U^k|$ are presented in Figure 14.



(a) (b) (c)

Figure 13: 13a & 13b: Comparison of neural network solution $u_\theta(\cdot)$ (blue) and finite difference solution U^k (red) along the x and y axis at time t_k, $1 \le k \le 15$. 13c: Semi-log plots of $\sqrt{\text{MSE loss}}$ vs computation time (seconds) at physical time $0.2, 0.6, 1.0, 1.4$.

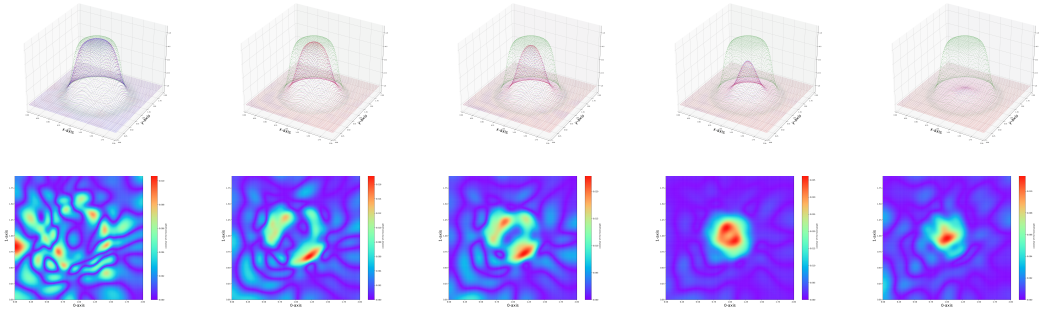


Figure 14: **Up row**: plots of u_{θ_k} (blue) together with the numerical solution $\{U_{ij}^k\}$ solved from implicit finite-difference scheme (red) on Ω at physical time $0.2, 0.6, 0.8, 1.0, 1.2$. The initial function u_0 is marked with green color; **Down row**: heatmaps of the error term $|u_{\theta_k}(\cdot) - U^k|$ at physical time $0.2, 0.6, 0.8, 1.0, 1.2$.

Remark 19 *Recall the semi-log plots of* $\sqrt{\text{MSE loss}}$ *vs. training time for both 1D and 2D Allen Cahn equations presented in Figure 8c and 13c, respectively. It is worth noting that in the 1D case, the accuracy of the numerical solution deteriorates as the physical time* t_k *increases. This behavior can be attributed to the stationary profile of the equation: its highly localized and strongly curved geometry makes it increasingly difficult for the MLP to accurately approximate. In contrast, the stationary profile associated with the 2D equation collapses to a flat plane, making it fairly easy for the MLP to approximate. Thus, we observe increasing accuracy as* t_k *increases.*

Appendix G. Primal-Dual algorithm using Adam optimizer for Optimal Transport problem

In this section, we briefly describe the PD-Adam algorithm tested in Section 5.5. Recall the loss functional $\mathcal{L}(T,\varphi)$ defined in (29), we parametrize both the map T and the dual function φ by neural networks T_θ, φ_η. We aim at solving the following saddle point problem

$$\begin{aligned}\max_{\eta}\min_{\theta}\ \mathcal{L}(T_\theta,\varphi_\eta) :=& \int_{\mathbb{R}^d}\frac{1}{2}\|x-T(x)\|^2\ \rho_0\,\mathrm{d}x+\int_{\mathbb{R}^d}\varphi(T(x))\ \rho_0\,\mathrm{d}x-\int_{\mathbb{R}^d}\varphi(y)\ \rho_1\,\mathrm{d}y\\ \approx&\frac{1}{N}\sum_{i=1}^{N}\frac{1}{2}\|\boldsymbol{X}_i-T_\theta(\boldsymbol{X}_i)\|^2-\varphi_\eta(T_\theta(\boldsymbol{X}_i))+\varphi_\eta(\boldsymbol{Y}_i),\end{aligned}\tag{95}$$

where N is the size of the datasets, $\{\boldsymbol{X}_i\}_{i=1}^N, \{\boldsymbol{Y}_i\}_{i=1}^N$ are samples drawn by ρ_0 and ρ_1. The PD-Adam algorithm is summarized in Algorithm 3.

Algorithm 3 Computing optimal Monge map from ρ_a to ρ_b

Input: Marginal distributions ρ_0 and ρ_1, learning rate lr_u, lr_φ of the Adam algorithm; Batch size N, total iteration number N_{iter}.
Initialize T_θ, φ_η.
for $iter = 1$ to N_{iter} **do**
  Sample $\{\boldsymbol{X}_i\}_{i=1}^N \sim \rho_a$. Sample $\{\boldsymbol{Y}_i\}_{i=1}^N \sim \rho_b$.
  Update θ to decrease (95) by Adam algorithm with learning rate lr_u for K_1 steps.
  Update η to increase (95) by Adam algorithm with learning rate lr_φ for K_2 steps.
end for
Output: The transport map T_θ.

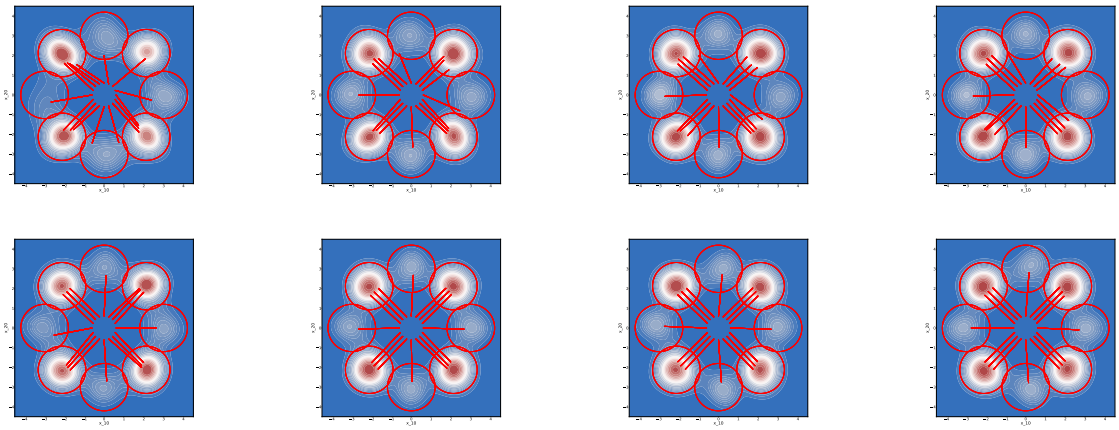
In all tests, we always set $K_1 = K_2 = 1$. We summarize all the other hyperparameters of the PD-Adam algorithm in Section 5.5 in Table 5.

Appendix H. Further numerical results regarding Section 5.5.3

For the OT problem from ρ_0 to ρ_b, we provide the intermediate results obtained by the NPDG algorithms as well as the PD-Adam algorithms in Figure 15. PD-Adam behaves unstably in this example, while NPDG method performs robustly for both preconditions.

		lr_u, lr_φ	N_{iter}	N	NN architecture
5.5.1 (1D)		$0.5 \cdot 10^{-3}, 0.5 \cdot 10^{-3}$	40000	800	$\texttt{MLP}_{\text{PReLU}}(1, 50, 1, 3)$
5.5.2 (5D)		$0.5 \cdot 10^{-4}, 0.5 \cdot 10^{-4}$	200000	2000	$\texttt{MLP}_{\text{PReLU}}(5, 80, 5, 4)$
5.5.3	(10D)	$0.5 \cdot 10^{-4}, 0.5 \cdot 10^{-4}$	100000	2000	$\texttt{MLP}_{\text{PReLU}}(10, 120, 10, 6)$
	(50D)	$10^{-5}, 10^{-5}$	300000	2000	$\texttt{MLP}_{\text{PReLU}}(50, 120, 50, 6)$

Table 5: Some hyperparameters used in the PD-Adam algorithm tested in Section 5.5.



(a) NPDG iter 5000 (b) NPDG iter 10000 (c) NPDG iter 15000 (d) NPDG iter 20000

(e) PD-Adam iter 150000 (f) PD-Adam iter 200000 (g) PD-Adam iter 250000 (h) PD-Adam iter 300000

Figure 15: OT problem from ρ_0 to ρ_b: Plots of the pushforwarded densities $T_{\theta\sharp}\rho_0$ of the computed T_θ obtained by NPDG method (1st row ((31) as preconditioning) & 2nd row ((35) as preconditioning)) and PD-Adam method (3rd row). All figures are plotted on the $10-20$ coordinate plane.

References

Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2): 251–276, 1998.

Shun-ichi Amari. *Information geometry and its applications*, volume 194. Springer, 2016.

Brandon Amos, Lei Xu, and J Zico Kolter. Input convex neural networks. In *International Conference on Machine Learning*, pages 146–155. PMLR, 2017.

Sokratis J Anagnostopoulos, Juan Diego Toscano, Nikolaos Stergiopulos, and George Em Karniadakis. Residual-based attention and connection to information bottleneck theory in PINNs. *arXiv preprint arXiv:2307.00379*, 2023.

Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein Generative Adversarial Networks. In *International Conference on Machine Learning*, pages 214–223. PMLR, 2017.

Yael Azulay and Eran Treister. Multigrid-augmented deep learning preconditioners for the Helmholtz equation. *SIAM Journal on Scientific Computing*, 45(3):S127–S151, 2022.

Gang Bao, Xiaojing Ye, Yaohua Zang, and Haomin Zhou. Numerical solution of inverse problems by weak adversarial networks. *Inverse Problems*, 36(11):115003, 2020.

Shamsulhaq Basir. Investigating and mitigating failure modes in physics-informed neural networks (PINNs). *arXiv preprint arXiv:2209.09988*, 2022.

Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of machine learning research*, 18(153):1–43, 2018.

Jean-David Benamou, Brittany D Froese, and Adam M Oberman. Two numerical methods for the elliptic Monge-Ampère equation. *ESAIM: Mathematical Modelling and Numerical Analysis*, 44(4):737–758, 2010.

Jean-David Benamou, Brittany D Froese, and Adam M Oberman. Numerical solution of the Optimal Transportation problem using the Monge–Ampère equation. *Journal of Computational Physics*, 260:107–126, 2014.

Ivan Bioli, Carlo Marcati, and Giancarlo Sangalli. Accelerating natural gradient descent for pinns with randomized numerical linear algebra. *arXiv preprint arXiv:2505.11638*, 2025.

Léon Bottou and Olivier Bousquet. The Tradeoffs of Large Scale Learning. In J.C. Platt, D. Koller, Y. Singer, and S. Roweis, editors, *Advances in Neural Information Processing Systems 20 (NIPS 2007)*, pages 161–168. NIPS Foundation (http://books.nips.cc), 2008. URL `http://leon.bottou.org/papers/bottou-bousquet-2008`.

Peter N Brown and Youcef Saad. Hybrid Krylov methods for nonlinear systems of equations. *SIAM Journal on Scientific and Statistical Computing*, 11(3):450–481, 1990.

Peter N Brown and Youcef Saad. Convergence theory of nonlinear Newton–Krylov algorithms. *SIAM Journal on Optimization*, 4(2):297–330, 1994.

Joan Bruna, Benjamin Peherstorfer, and Eric Vanden-Eijnden. Neural Galerkin schemes with active learning for high-dimensional evolution equations. *Journal of Computational Physics*, 496:112588, 2024.

Zhiqiang Cai, Yu Cao, Yuanfei Huang, and Xiang Zhou. Weak generative sampler to efficiently sample invariant distribution of stochastic differential equation. *arXiv preprint arXiv:2405.19256*, 2024.

Antonin Chambolle and Thomas Pock. A first-order primal-dual algorithm for convex problems with applications to imaging. *Journal of Mathematical Imaging and Vision*, 40: 120–145, 2011.

Yifan Chen and Wuchen Li. Natural Gradient in Wasserstein Statistical Manifold. *arXiv:1805.08380 [cs, math]*, 2018.

Zhuo Chen, Jacob McCarran, Esteban Vizcaino, Marin Soljacic, and Di Luo. Teng: Time-evolving natural gradient for solving pdes with deep neural nets toward machine precision. In *Forty-first International Conference on Machine Learning*, 2024.

Felix Dangel, Johannes Müller, and Marius Zeinhofer. Kronecker-Factored Approximate Curvature for Physics-Informed Neural Networks. *arXiv preprint arXiv:2405.15603*, 2024.

Guido De Philippis and Alessio Figalli. The Monge–Ampère equation and its link to Optimal Transportation. *Bulletin of the American Mathematical Society*, 51(4):527–580, 2014.

Ron S Dembo, Stanley C Eisenstat, and Trond Steihaug. Inexact Newton methods. *SIAM Journal on Numerical analysis*, 19(2):400–408, 1982.

Suchuan Dong and Zongwei Li. Local extreme learning machines and domain decomposition for solving linear and nonlinear partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 387:114129, 2021.

Yifan Du and Tamer A Zaki. Evolutional deep neural network. *Physical Review E*, 104(4): 045303, 2021.

John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12(7), 2011.

Stanley C Eisenstat and Homer F Walker. Globally convergent inexact Newton methods. *SIAM Journal on Optimization*, 4(2):393–422, 1994.

Lawrence C Evans. *Partial differential equations*, volume 19. American mathematical society, 2022.

Jiaojiao Fan, Shu Liu, Shaojun Ma, Haomin Zhou, and Yongxin Chen. Neural Monge Map estimation and its applications. *Transactions on machine learning research*, 2023.

Roger Fletcher. *Practical methods of optimization*. John Wiley & Sons, 2000.

Brittany D Froese and Adam M Oberman. Convergent finite difference solvers for viscosity solutions of the elliptic Monge–Ampère equation in dimensions two and higher. *SIAM Journal on Numerical Analysis*, 49(4):1692–1714, 2011.

Nathan Gaby, Xiaojing Ye, and Haomin Zhou. Neural control of parametric solutions for high-dimensional evolution PDEs. *arXiv preprint arXiv:2302.00045*, 2023.

Emilio Gagliardo. Caratterizzazioni delle tracce sulla frontiera relative ad alcune classi di funzioni in n variabili. *Rendiconti del seminario matematico della universita di Padova*, 27:284–305, 1957.

Thomas George, César Laurent, Xavier Bouthillier, Nicolas Ballas, and Pascal Vincent. Fast approximate natural gradient descent in a Kronecker factored eigenbasis. *Advances in Neural Information Processing Systems*, 31, 2018.

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *Communications of the ACM*, 63(11):139–144, 2020.

Pierre Grisvard. *Elliptic problems in nonsmooth domains*. SIAM, 2011.

Jiequn Han, Arnulf Jentzen, et al. Deep learning-based numerical methods for high-dimensional parabolic partial differential equations and backward stochastic differential equations. *Communications in Mathematics and Statistics*, 5(4):349–380, 2017.

Jiequn Han, Arnulf Jentzen, and Weinan E. Solving high-dimensional partial differential equations using deep learning. *Proceedings of the National Academy of Sciences*, 115(34): 8505–8510, 2018.

Wenrui Hao, Qingguo Hong, and Xianlin Jin. Gauss Newton method for solving variational problems of PDEs with neural network discretizaitons. *Journal of Scientific Computing*, 100(1):17, 2024.

Zheyuan Hu, Khemraj Shukla, George Em Karniadakis, and Kenji Kawaguchi. Tackling the curse of dimensionality with physics-informed neural networks. *Neural Networks*, 176: 106369, 2024.

Xiaokai Huo and Hailiang Liu. Inf-Sup neural networks for high-dimensional elliptic PDE problems. *Journal of Computational Physics*, page 113188, 2024.

Martin Hutzenthaler, Arnulf Jentzen, Thomas Kruse, et al. Multilevel Picard iterations for solving smooth semilinear parabolic heat equations. *Partial Differential Equations and Applications*, 2(6):1–31, 2021.

Matt Jacobs and Flavien Léger. A fast approach to optimal transport: The back-and-forth method. *Numerische Mathematik*, 146(3):513–544, 2020.

Matt Jacobs, Flavien Léger, Wuchen Li, and Stanley Osher. Solving large-scale optimization problems with a convergence rate independent of grid size. *SIAM Journal on Numerical Analysis*, 57(3):1100–1123, 2019.

Yijie Jin, Shu Liu, Hao Wu, Xiaojing Ye, and Haomin Zhou. Parameterized Wasserstein Gradient Flow. *arXiv preprint arXiv:2404.19133*, 2024.

Diederik P Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Dana A Knoll and David E Keyes. Jacobian-free Newton–Krylov methods: a survey of approaches and applications. *Journal of Computational Physics*, 193(2):357–397, 2004.

Alena Kopaničáková, Hardik Kothari, George E Karniadakis, and Rolf Krause. Enhancing training of physics-informed neural networks using domain decomposition–based preconditioning strategies. *SIAM Journal on Scientific Computing*, pages S46–S67, 2024.

Alexander Korotin, Vage Egiazarian, Arip Asadulaev, Alexander Safin, and Evgeny Burnaev. Wasserstein-2 Generative Networks. *arXiv preprint arXiv:1909.13082*, 2019.

Alexander Korotin, Daniil Selikhanovych, and Evgeny Burnaev. Neural optimal transport. *arXiv preprint arXiv:2201.12220*, 2022.

Max Kuang and Esteban G Tabak. Preconditioning of optimal transport. *SIAM Journal on Scientific Computing*, 39(4):A1793–A1810, 2017.

Devadatta Kulkarni, Darrell Schmidt, and Sze-Kai Tsui. Eigenvalues of tridiagonal pseudo-Toeplitz matrices. *Linear Algebra and its Applications*, 297:63–80, 1999.

Wuchen Li and Guido Montufar. Natural Gradient via Optimal Transport. *arXiv:1803.07033 [cs, math]*, 2018.

Dong C Liu and Jorge Nocedal. On the limited memory BFGS method for large scale optimization. *Mathematical programming*, 45(1):503–528, 1989.

Min Liu, Zhiqiang Cai, and Karthik Ramani. Deep Ritz method with adaptive quadrature for linear elasticity. *Computer Methods in Applied Mechanics and Engineering*, 415:116229, 2023a.

Shu Liu, Wuchen Li, Hongyuan Zha, and Haomin Zhou. Neural Parametric Fokker–Planck Equation. *SIAM Journal on Numerical Analysis*, 60(3):1385–1449, 2022. doi: 10.1137/20M1344986. URL `https://doi.org/10.1137/20M1344986`.

Shu Liu, Siting Liu, Stanley Osher, and Wuchen Li. A first-order computational algorithm for reaction-diffusion type equations via primal-dual hybrid gradient method. *Journal of Computational Physics*, 500:112753, 2024a. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2024.112753. URL `https://www.sciencedirect.com/science/article/pii/S0021999124000020`.

Shu Liu, Xinzhe Zuo, Stanley Osher, and Wuchen Li. Numerical analysis of a first-order computational algorithm for reaction-diffusion equations via the primal-dual hybrid gradient method. *Mathematics of Computation*, 2025.

Siting Liu, Stanley Osher, Wuchen Li, and Chi-Wang Shu. A primal-dual approach for solving conservation laws with implicit in time approximations. *Journal of Computational Physics*, 472, 2023b.

Songming Liu, Chang Su, Jiachen Yao, Zhongkai Hao, Hang Su, Youjia Wu, and Jun Zhu. Preconditioning for physics-informed neural networks. *arXiv preprint arXiv:2402.00531*, 2024b.

Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A deep learning library for solving differential equations. *SIAM review*, 63(1):208–228, 2021a.

Yulong Lu, Jianfeng Lu, and Min Wang. A Priori Generalization Analysis of the Deep Ritz Method for Solving High Dimensional Elliptic Partial Differential Equations. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 3196–3241. PMLR, 15–19 Aug 2021b. URL `https://proceedings.mlr.press/v134/lu21a.html`.

Ashok Makkuva, Amirhossein Taghvaei, Sewoong Oh, and Jason Lee. Optimal transport mapping via input convex neural networks. In *International Conference on Machine Learning*, pages 6672–6681. PMLR, 2020.

James Martens. Deep learning via Hessian-free optimization . In *ICML*, volume 27, pages 735–742, 2010.

James Martens. New insights and perspectives on the natural gradient method. *Journal of Machine Learning Research*, 21(146):1–76, 2020.

James Martens and Roger Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *International Conference on Machine Learning*, pages 2408–2417. PMLR, 2015.

Per-Gunnar Martinsson and Joel Tropp. Randomized numerical linear algebra: foundations & algorithms (2020). *arXiv preprint arXiv:2002.01387*, 2020.

Levi McClenny and Ulisses Braga-Neto. Self-adaptive physics-informed neural networks using a soft attention mechanism. *arXiv preprint arXiv:2009.04544*, 2020.

Levi D McClenny and Ulisses M Braga-Neto. Self-adaptive physics-informed neural networks. *Journal of Computational Physics*, 474:111722, 2023.

William Charles Hector McLean. *Strongly elliptic systems and boundary integral equations*. Cambridge university press, 2000.

Tingwei Meng, Wenbo Hao, Siting Liu, Stanley J Osher, and Wuchen Li. Primal-dual hybrid gradient algorithms for computing time-implicit Hamilton-Jacobi equations. *arXiv preprint arXiv:2310.01605*, 2023.

Johannes Müller and Marius Zeinhofer. Achieving high accuracy with PINNs via energy natural gradient descent. In *International Conference on Machine Learning*, pages 25471–25485. PMLR, 2023.

Michael Neilan, Abner J Salgado, and Wujun Zhang. The Monge–Ampère equation. In *Handbook of Numerical Analysis*, volume 21, pages 105–219. Elsevier, 2020.

Y. Nesterov. A method for solving the convex programming problem with convergence rate $O\left(\frac{1}{k^2}\right)$. *Doklady Akademii Nauk SSSR*, 269:543–547, 1983. URL `https://cir.nii.ac.jp/crid/1370862715914709505`.

Naxian Ni and Suchuan Dong. Numerical computation of partial differential equations by hidden-layer concatenated extreme learning machine. *Journal of Scientific Computing*, 95 (2):35, 2023.

Kenta Niwa, Hiro Ishii, Hiroshi Sawada, Akinori Fujino, Noboru Harada, and Rio Yokota. Natural gradient primal-dual method for decentralized learning. *IEEE Transactions on Signal and Information Processing over Networks*, 10:417–433, 2024. doi: 10.1109/TSIPN.2024.3388948.

Levon Nurbekyan, Wanzhou Lei, and Yunan Yang. Efficient natural gradient descent methods for large-scale PDE-based optimization problems. *SIAM Journal on Scientific Computing*, 45(4):A1621–A1655, 2023.

Christopher C Paige and Michael A Saunders. Solution of sparse indefinite systems of linear equations. *SIAM journal on numerical analysis*, 12(4):617–629, 1975.

Yesom Park, Changhoon Song, and Myungjoo Kang. Beyond Derivative Pathology of PINNs: Variable Splitting Strategy with Convergence Analysis. *arXiv preprint arXiv:2409.20383*, 2024.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019. URL `http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf`.

Clemens Pechstein. Boundary element methods. 2013.

Maziar Raissi, Paris Perdikaris, and George 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*, 378:686–707, 2019.

Pratik Rathore, Weimu Lei, Zachary Frangella, Lu Lu, and Madeleine Udell. Challenges in training PINNs: A loss landscape perspective. *arXiv preprint arXiv:2402.01868*, 2024.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951.

Fred Roosta, Yang Liu, Peng Xu, and Michael W Mahoney. Newton-MR: Inexact Newton method with minimum residual sub-problem solver. *EURO Journal on Computational Optimization*, 10:100035, 2022.

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning representations by back-propagating errors. *Nature*, 323(6088):533–536, 1986.

David Saad. Online algorithms and stochastic approximations. *Online Learning*, 5(3):6, 1998.

Zihan Shao, Konstantin Pieper, and Xiaochuan Tian. Solving nonlinear pdes with sparse radial basis function networks. *arXiv preprint arXiv:2505.07765*, 2025.

Zebang Shen, Zhenfu Wang, Alejandro Ribeiro, and Hamed Hassani. Sinkhorn natural gradient for generative models. *Advances in Neural Information Processing Systems*, 33: 1646–1656, 2020.

Amanpreet Singh, Martin Bauer, and Sarang Joshi. Physics informed convex artificial neural networks (PICANNs) for optimal transport based density estimation. *arXiv preprint arXiv:2104.01194*, 2021.

Justin Sirignano and Konstantinos Spiliopoulos. DGM: A deep learning algorithm for solving partial differential equations. *Journal of Computational Physics*, 375:1339–1364, 2018.

Yang Song, Jiaming Song, and Stefano Ermon. Accelerating natural gradient with higher-order invariance. In *International Conference on Machine Learning*, pages 4713–4722. PMLR, 2018.

Weijie Su, Stephen Boyd, and Emmanuel J Candes. A differential equation for modeling Nesterov's accelerated gradient method: Theory and insights. *Journal of Machine Learning Research*, 17(153):1–43, 2016.

Kejun Tang, Xiaoliang Wan, and Qifeng Liao. Adaptive deep density approximation for Fokker-Planck equations. *Journal of Computational Physics*, 457:111080, 2022.

Kejun Tang, Xiaoliang Wan, and Chao Yang. DAS-PINNs: A deep adaptive sampling method for solving high-dimensional partial differential equations. *Journal of Computational Physics*, 476:111868, 2023.

Philip Thomas, Bruno Castro Silva, Christoph Dann, and Emma Brunskill. Energetic natural gradient descent. In *International Conference on Machine Learning*, pages 2887–2895. PMLR, 2016.

Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop, coursera: Neural networks for machine learning. *University of Toronto, Technical Report*, 6, 2012.

Cédric Villani. *Topics in Optimal Transportation*, volume 58. American Mathematical Soc., 2021.

Cédric Villani et al. *Optimal Transport: Old and New*, volume 338. Springer, 2009.

Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

Sifan Wang, Xinling Yu, and Paris Perdikaris. When and why PINNs fail to train: A neural tangent kernel perspective. *Journal of Computational Physics*, 449:110768, 2022a.

Yifan Wang, Pengzhan Jin, and Hehu Xie. Tensor neural network and its numerical integration. *arXiv preprint arXiv:2207.02754*, 2022b.

Yifan Wang, Zhongshuo Lin, Yangfei Liao, Haochen Liu, and Hehu Xie. Solving High-Dimensional Partial Differential Equations Using Tensor Neural Network and A Posteriori Error Estimators. *Journal of Scientific Computing*, 101(3):1–29, 2024.

Yifei Wang and Wuchen Li. Information Newton's flow: second-order optimization method in probability space. *arXiv preprint arXiv:2001.04341*, 2020.

Yiran Wang and Suchuan Dong. An extreme learning machine-based method for computational PDEs in higher dimensions. *Computer Methods in Applied Mechanics and Engineering*, 418:116578, 2024.

Colby L Wight and Jia Zhao. Solving Allen-Cahn and Cahn-Hilliard equations using the adaptive physics informed neural networks. *arXiv preprint arXiv:2007.04542*, 2020.

Hao Wu, Shu Liu, Xiaojing Ye, and Haomin Zhou. Parameterized Wasserstein Hamiltonian flow. *arXiv preprint arXiv:2306.00191*, 2023.

Lexing Ying. Natural gradient for combined loss using wavelets. *Journal of Scientific Computing*, 86(2):26, 2021.

Bing Yu et al. The deep Ritz method: a deep learning-based numerical algorithm for solving variational problems. *Communications in Mathematics and Statistics*, 6(1):1–12, 2018.

Yaohua Zang, Gang Bao, Xiaojing Ye, and Haomin Zhou. Weak adversarial networks for high-dimensional partial differential equations. *Journal of Computational Physics*, 411: 109409, 2020.

Matthew D Zeiler. ADADELTA: an adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

Qi Zeng, Yash Kothari, Spencer H Bryngelson, and Florian Schäfer. Competitive physics informed networks. *arXiv preprint arXiv:2204.11144*, 2022.

Mingqiang Zhu and Tony Chan. An efficient primal-dual hybrid gradient algorithm for total variation image restoration. *UCLA CAM Report*, 34:8–34, 2008.

Xinzhe Zuo, Jiaxi Zhao, Shu Liu, Stanley Osher, and Wuchen Li. Numerical Analysis on Neural Network Projected Schemes for Approximating One Dimensional Wasserstein Gradient Flows. *arXiv preprint arXiv:2402.16821*, 2024.